computers

MDPI

Review

Artificial Intelligence for High-Availability Systems: A Comprehensive Review

Lidia Fotia [1], Rosario Gaeta [2], Fabrizio Messina [3,*], Domenico Rosaci [4] and Giuseppe M. L. Sarné [5]

[1] Department of Information and Electrical Engineering and Applied Mathematics (DIEM), University of Salerno, 84084 Fisciano, Italy; lfotia@unisa.it
[2] Department of Industrial Engineering (DIIN), University of Salerno, 84084 Fisciano, Italy; rgaeta@unisa.it
[3] Department of Mathematics and Informatics (DMI), University of Catania, 95124 Catania, Italy
[4] Department of Information Engineering, Infrastructure and Sustainable Energy (DIIES), University Mediterranea of Reggio Calabria, 89124 Reggio Calabria, Italy; domenico.rosaci@unirc.it
[5] Department of Psychology, University of Milan Bicocca, 20126 Milan, Italy; giuseppe.sarne@unimib.it
* Correspondence: fabrizio.messina@unict.it

Abstract

High-availability (HA) systems—essential in many contemporary contexts—are designed to guarantee the availability of processes and data for more than 99% of their operational time. These systems are typically implemented as Cloud/Edge infrastructures that are properly maintained by human operators and intelligent agents in order to guarantee the required level of availability. Moreover, we are witnessing the widespread adoption of AI-based automation across many industries. AI-based software agents are increasingly being adopted to introduce more automation in highly available systems, particularly for monitoring and fault detection, fault prediction, recovery, and optimization processes. In this review paper, we discuss the state of the art of AI-based solutions for HA systems. In particular, we focus on the use of AI for the core operational mechanisms of monitoring, failure detection, and recovery. Our discussion begins by reviewing a few key background concepts of HA architectures, then we review recent work on AI-based solutions for monitoring, fault detection and recovery in HA systems.

Keywords: high-availability systems; artificial intelligence; Cloud/Edge infrastructures

1. Introduction

High availability (HA) refers to the ability of a computing system to remain continuously operational and accessible for a very high percentage of the time, often quantified through well-established metrics such as the so-called five nines (99.999%) availability [1]. In contemporary digital infrastructures, HA is no longer a desirable property but a fundamental requirement. Enterprises operating in critical sectors such as finance, healthcare, manufacturing, and transportation increasingly depend on complex software-intensive systems where even short service disruptions may lead to significant economic losses, safety risks, or violations of regulatory constraints.

To meet these requirements, organizations deploy high-availability clusters (HACs), i.e., software-managed environments composed of interconnected servers, storage subsystems, and communication networks designed to ensure the uninterrupted execution of critical applications. HACs continuously monitor the operational state of system components and automatically trigger recovery actions (such as failover, restart, or migration) to minimize downtime and preserve data integrity. The primary objective of these systems

Academic Editor: Yan Zhang

Received: 25 February 2026
Revised: 26 March 2026
Accepted: 3 April 2026
Published: 8 April 2026

is to protect the components that constitute a single point of failure, including databases, coordination services, and load balancers.

The diversity of enterprise workloads has resulted in a wide range of HA designs. Transactional systems, such as enterprise resource planning platforms, require strict guarantees on atomicity, consistency, isolation, and durability during failure recovery; meanwhile, data analytics and stream-processing applications often prioritize scalability and throughput over strong consistency. As a consequence, HAC architectures have evolved into heterogeneous solutions tailored to different requirements at the application level, in addition to operational constraints [2,3].

In recent years, the design space for HA systems has expanded significantly due to the widespread adoption of virtualization technologies, container orchestration platforms, and Cloud/Edge integration. Cloud infrastructures rely on microservices and container-based deployments orchestrated by frameworks such as Kubernetes, which support redundancy, self-healing, and elastic scaling mechanisms [4–6]. At the same time, Edge and fog computing environments introduce additional challenges. For example, availability management deals with limited computational resources, intermittent connectivity, and strict latency constraints that make traditional replication or quorum-based techniques less effective [7–9].

Along with these changes in system architecture, there has been a growing interest in integrating artificial intelligence (AI) techniques into HA systems to enhance automation and operational intelligence. Traditional HA mechanisms are characterized by high reactivity, relying on predefined rules and thresholds to detect failures and initiate recovery. In contrast, AI-based approaches enable data-driven monitoring, anomaly detection, and fault prediction by learning patterns derived from large volumes of system telemetry data, logs, and performance metrics. These techniques have highlighted promising results in both improving the mean time between failures (MTBF) and reducing recovery time objectives (RTOs), particularly in large-scale and highly dynamic environments [10,11].

Recent research has explored the use of machine learning and autonomic computing principles to support proactive availability management. Intelligent software agents can anticipate failures before they manifest, dynamically selecting recovery strategies based on contextual information, and adapting the system behavior to change workloads and environmental conditions. This shift marks a transition from purely reactive failover mechanisms to more proactive and context-aware HA systems, where monitoring, fault detection, fault prediction, and recovery are tightly integrated within a closed control loop [12,13].

Despite substantial progress, the field remains fragmented. Existing studies often focus on specific techniques, platforms, or application domains, using heterogeneous terminology and evaluation methodologies [2,14]. Moreover, there is still a lack of comprehensive surveys that systematically analyze how AI-based solutions are applied to the core operational mechanisms of HA systems—namely monitoring, failure detection, fault prediction, and recovery—across Cloud and Edge infrastructures. This fragmentation makes it difficult to compare approaches, identify best practices, and assess their practical applicability [2,15,16].

To address this gap, this survey provides a systematic and structured review of AI-based solutions for high-availability systems. Building on a rigorous systematic literature review methodology, we analyze recent research with a particular focus on how AI techniques are employed to support operational HA functions. The contributions of this paper are threefold: (i) we review the architectural contexts in which AI-enhanced HA mechanisms are deployed, with emphasis on Cloud and Edge environments; (ii) we classify and compare AI-based approaches for monitoring, fault detection, fault prediction, and recovery; and (iii) we discuss open challenges and future research directions, including

model reliability, real-time constraints, and the integration of AI-driven decision making into production-grade HA platforms.

By consolidating and interpreting recent advances, this work aims to provide both researchers and practitioners with a coherent perspective on the evolving role of AI in HA systems and to outline promising directions toward more dependable, autonomous, and self-managing computing infrastructures.

2. Review Research Method

This survey adopts a systematic literature review (SLR) methodology conducted in accordance with the guidelines proposed by Kitchenham [17]. The objective of the review is to systematically analyze and classify state-of-the-art artificial intelligence (AI)-based solutions adopted in high-availability (HA) systems, with a specific focus on operational mechanisms such as monitoring, fault detection, fault prediction, and recovery.

High-availability systems are typically deployed in large-scale Cloud and Edge infrastructures and are designed to guarantee service and data availability exceeding 99% of operational time. Traditionally, such systems are based on human operators and rule-based automation. However, recent advances have led to the widespread adoption of AI-based software agents to increase automation, responsiveness, and robustness in HA management. This review aims to provide a structured overview of these AI-driven approaches.

The SLR protocol was defined a priori to ensure transparency, reproducibility, and methodological rigor. In the following, we detail the research questions, inclusion and exclusion criteria, and the search and selection process.

2.1. Research Questions

The review is guided by the following research questions:

- **RQ1:** What types of high-availability architecture adopt AI-based solutions for operational management?
- **RQ2:** Which AI techniques are employed for monitoring, fault detection, and fault prediction in HA systems?
- **RQ3:** How are AI-based methods used to support recovery processes and reduce service downtime in HA systems?
- **RQ4:** What are the main challenges and open research issues in applying AI to the operational management of HA systems?

RQ1 investigates the architectural contexts in which AI-based solutions are deployed, focusing on Cloud, Edge, and hybrid infrastructures that require continuous availability. The goal is to identify recurring architectural patterns in support of AI-driven operational control.

RQ2 focuses on AI techniques used for system monitoring and fault management, which include machine learning, deep learning, and statistical learning approaches applied to anomaly detection, failure classification, and fault prediction.

RQ3 examines AI-driven recovery mechanisms, such as automated remediation, resource reconfiguration, and adaptive control, evaluating how these techniques contribute to reducing the mean time to recovery (MTTR) and improving the general availability of the system.

Finally, **RQ4** addresses the open challenges that limit the adoption of AI in HA systems, including data availability, model reliability, real-time constraints, scalability, and integration with existing operational workflows.

2.2. Inclusion and Exclusion Criteria

Following Kitchenham's SLR guidelines [17], inclusion and exclusion criteria were defined prior to the review process.

Inclusion criteria:

- (I1) Studies explicitly addressing high availability, fault tolerance, or fault management in computing systems.
- (I2) Works proposing or evaluating AI-based techniques for monitoring, fault detection, fault prediction, or recovery.
- (I3) Peer-reviewed journal articles, conference papers, or book chapters.
- (I4) Publications over the time span of 2015–2025, reflecting the recent adoption of AI in HA systems.
- (I5) English-language publications.

Exclusion criteria:

- (E1) Studies lacking technical detail or experimental validation.
- (E2) Works not related to the operational management of HA systems.
- (E3) Non-peer-reviewed sources, such as white papers or technical reports.
- (E4) Duplicate studies.
- (E5) Publications without accessible full text.

All candidate studies were independently screened by two reviewers, and disagreements were resolved through discussion to reduce selection bias.

2.3. Search Process

The literature search was conducted using the major scientific databases, including IEEE Xplore, Scopus, and Web of Science. Google Scholar was also consulted to identify complementary studies. The search targeted publications from 2015 to 2025.

The following search string was used, with minor adaptations depending on the database:

((high availability OR fault tolerance) AND (monitoring OR fault detection OR fault prediction OR recovery) AND (artificial intelligence OR machine learning OR deep learning))

For each selected study, the following data were extracted:

- Publication venue and year.
- Target system architecture (Cloud, Edge, hybrid).
- AI techniques used.
- Operational focus (monitoring, fault detection, prediction, recovery).
- Evaluation metrics (availability, MTTR, MTTF, downtime).

The study-selection process was conducted following a PRISMA-inspired protocol to ensure transparency and reproducibility. The procedure consisted of four main phases: identification, screening, eligibility, and inclusion.

In the identification phase, a total of 5800 records were retrieved. After removing duplicates, 5000 unique records remained. During the screening phase, 4500 records were excluded based on title and abstract analysis, resulting in 500 papers. In the eligibility phase, full-text assessment led to the exclusion of 458 studies due to lack of relevance or insufficient methodological rigor. Finally, 42 studies were included in the qualitative synthesis. A detailed overview of the selection pipeline is provided in Figure 1.

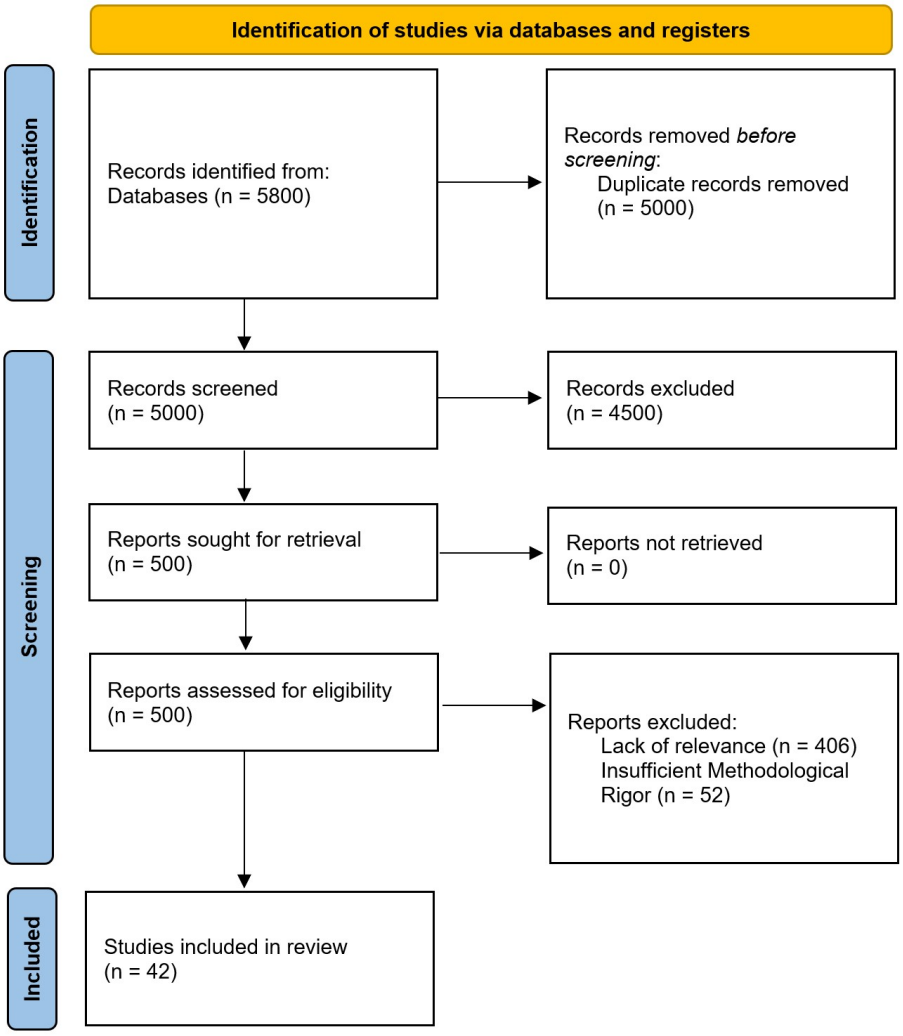

Figure 1. PRISMA-compliant flow diagram of the study-selection process.

2.4. Quality Assessment

To ensure the reliability of the selected studies, a quality assessment was conducted based on a set of predefined criteria. Each study was evaluated according to the following aspects:

- Clarity of the research objectives;
- Adequacy of the methodology;
- Reproducibility of the results;
- Relevance to the research questions;
- Empirical validation (if applicable).

Each criterion was scored using a three-point scale, defined as follows:

- 0 = low (criterion not satisfied or poorly addressed);
- 1 = medium (partially addressed);
- 2 = high (fully satisfied with clear methodological evidence).

The overall quality score was computed as the sum of individual criteria scores, resulting in a maximum score of 10. Studies with a total score lower than 6 were excluded from the final analysis. Table 1 represents the distribution of quality assessment scores across the selected studies.

Table 1. Distribution of quality assessment scores across the selected studies.

Score Range	Number of Studies	Interpretation
8–10	24	High quality
6–7	18	Medium quality
<6	0	Excluded

In general, the quality assessment indicates that the final pool of 42 studies is methodologically robust. More than half of the included papers (24 out of 42) achieved a high-quality score, while the remaining 18 studies were assessed as medium quality. No low-quality studies (score below 6) were retained, thus ensuring consistency and reliability in the evidence synthesis.

2.5. Classification Rationale

In the process of the full-text analyses, the selected studies were inductively classified into three main thematic categories:

- **AI-based Monitoring and Fault Detection**: studies focusing on real-time system observation, anomaly identification, and early fault detection.
- **AI-based Fault Prediction**: studies addressing the anticipation of failures to enable proactive intervention and preventive actions.
- **AI-driven Recovery Mechanisms in HA Systems**: studies covering automated remediation, system reconfiguration, and service restoration.

This classification reflects the core operational mechanisms of high-availability systems and directly mirrors the focus outlined in the research questions. The collected data were analyzed across architectural, methodological, and performance dimensions to assess the effectiveness and limitations of AI-based solutions for HA systems.

2.6. Cross-Study Synthesis and Research Trends

Beyond the individual analysis of the selected studies, a cross-study comparison reveals several recurring patterns and emerging trends in the literature. First, a set of dominant methodological paradigms can be identified. The majority of the reviewed works adopt data-driven approaches, leveraging machine learning and deep learning models trained on historical telemetry data. A smaller portion of the literature employs model-based approaches, typically relying on analytical or statistical representations of system behavior. Hybrid approaches, which combine data-driven learning with rule-based or statistical mechanisms, are also increasingly explored to improve robustness and interpretability. Overall, data-driven methods clearly represent the dominant paradigm, while purely model-based techniques are less frequently adopted.

Second, the analysis highlights several converging research trends. In recent years, there has been increasing attention toward predictive fault management, autonomous self-healing systems, and hybrid AI architectures that integrate multiple learning paradigms. These trends reflect the growing importance of managing the increasing complexity and dynamic behavior of modern distributed systems, such as Cloud-native and Edge infrastructures. Overall, this evolution suggests a shift from reactive, rule-based system management to proactive, data-driven and autonomous operational paradigms.

Third, despite the progress observed, several limitations and gaps remain. A significant portion of the literature lacks large-scale validation, real-world deployment, and standardized benchmarks, which limits the comparability and generalizability of the proposed approaches.

Overall, this synthesis provides a more integrated view of the field, highlighting not only what has been done but also what remains to be addressed.

Recent advances in artificial intelligence are driving a significant transformation in the design and management of High-availability (HA) systems. Three key trends can be identified. First, there is a growing adoption of predictive and proactive fault management. Traditional HA systems relied on reactive mechanisms, such as redundancy and failover, activated only after a failure occurred. In contrast, AI-based approaches leverage anomaly detection and time-series forecasting to anticipate failures before they manifest, enabling preventive actions and reducing downtime. Second, the integration of self-healing and autonomous systems is becoming increasingly prominent. Techniques such as reinforcement learning and automated orchestration allow systems to dynamically adapt and recover from failures without human intervention. This shift reduces operational complexity and improves system resilience in large-scale and distributed environments. Third, there is a trend toward data-driven and hybrid AI approaches. Modern HA solutions increasingly combine multiple AI techniques (e.g., machine learning, deep learning, and rule-based systems) to improve accuracy and robustness. These hybrid models enable more reliable fault detection and diagnosis by leveraging heterogeneous data sources. These trends reflect a larger technological shift from static and rule-based system management to adaptive, intelligent, and autonomous infrastructures. This shift is mainly driven by the increasing complexity of distributed systems, such as Cloud-native architectures and Edge computing, where traditional approaches are no longer sufficient to guarantee high availability. From a paradigm perspective, the field is transitioning from a reactive and infrastructure-centric paradigm, where availability is ensured through redundancy and manual intervention, to a proactive and AI-driven paradigm, where availability is maintained through continuous monitoring, prediction, and autonomous decision making.

2.7. Trade-Offs Between Classical Machine Learning and Deep Learning Approaches

A relevant aspect emerging from the literature is the trade-off between classical machine learning models and deep learning architectures, particularly in operational environments. Classical machine learning approaches (e.g., decision trees, support vector machines, and ensemble methods) are generally characterized by lower computational requirements, higher interpretability, and faster training times. These properties make them particularly suitable for scenarios where explainability, limited computational resources, or strict latency constraints are critical. In contrast, deep learning models offer superior performance in handling high-dimensional and unstructured data, enabling more accurate predictions in complex scenarios. However, they typically require significantly higher computational resources, longer training times, and are often perceived as less interpretable, which may limit their adoption in safety-critical or resource-constrained environments. From an operational perspective, the choice between these approaches often depends on the specific requirements of the system, including scalability, real-time constraints, and deployment complexity. As a result, several works explore hybrid solutions that aim to combine the efficiency of classical models with the expressive power of deep learning.

From a computational perspective, deep learning models (e.g., LSTM, transformers) generally require higher training time, memory consumption, and specialized hardware (e.g., GPUs), which may limit their deployment in resource-constrained environments such as Edge systems. In contrast, classical machine learning models (e.g., decision trees, random forests) are computationally more efficient, easier to deploy, and better suited for real-time inference with limited resources. Furthermore, deep learning approaches typically require larger labeled datasets, while classical methods can operate effectively with smaller datasets. This trade-off directly impacts their suitability depending on the

application scenario, particularly in high-availability systems where latency, interpretability, and resource constraints are critical.

2.8. Integration of AI Models in Production-Grade HA Systems

Beyond model performance, an important challenge lies in the integration of AI-based solutions into production-grade high-availability (HA) systems. Modern operational environments increasingly rely on containerized infrastructures and orchestration platforms (e.g., Kubernetes), which introduce additional constraints and opportunities for AI deployment. In such contexts, AI models must be designed not only for accuracy, but also for robustness, scalability, and fault tolerance. Key challenges include model deployment and life cycle management, real-time inference under strict latency requirements, and resilience to system failures. Moreover, the integration of AI components within distributed systems requires careful consideration of monitoring, retraining strategies, and interaction with orchestration mechanisms (e.g., auto-scaling and self-healing features). The literature shows that, while several approaches demonstrate promising results in controlled environments, fewer studies address the full integration of AI models into production-grade HA platforms. This highlights a gap between research prototypes and real-world deployment, suggesting an important direction for future work.

2.9. Linking AI Techniques to High-Availability Management

Although many of the reviewed studies focus on anomaly detection and monitoring in generic IoT or cyber–physical systems, their contribution can be more clearly understood when framed within high-availability (HA) management mechanisms. In particular, anomaly detection techniques play a key role in early fault identification, enabling proactive interventions that can reduce system downtime. This directly impacts availability-related metrics such as Mean Time To Repair (MTTR), by facilitating faster diagnosis, and Mean Time Between Failures (MTBF), by supporting predictive maintenance strategies. Similarly, machine learning models used for performance monitoring and prediction can contribute to maintaining service-level objectives (SLOs), by anticipating degradations and triggering adaptive resource management or fault mitigation actions. From a system perspective, the surveyed approaches can be mapped onto three core HA functions:

- **Fault detection:** identification of anomalies or deviations from normal behavior;
- **Fault diagnosis:** classification and root-cause analysis of detected issues;
- **Fault recovery and mitigation:** support for automated or semi-automated corrective actions.

This mapping (see Table 2) highlights that even when not explicitly designed for HA architectures, many AI-based techniques can be effectively integrated into availability-oriented system management frameworks. Representative studies are reported for each category to illustrate the mapping.

Table 2. Mapping of AI techniques to HA management functions with representative studies.

AI Technique	Fault Detection	Diagnosis	Recovery	Support
Anomaly Detection [18–20]	✓			✓
Classification Models [21–23]	✓	✓		✓
Time-series Forecasting [18,24–26]	✓			✓
Reinforcement Learning [27,28]			✓	✓
Hybrid Approaches [29]	✓	✓	✓	✓

From an availability perspective, the adoption of AI-based techniques can contribute to improving key metrics such as MTBF and MTTR, as well as ensuring compliance with

service-level objectives (SLOs). In particular, predictive models enable earlier detection of potential failures, while automated response mechanisms can reduce recovery time.

2.10. Limitations of Simulation-Based Validation

A relevant limitation observed across the reviewed studies is the widespread reliance on simulation-based validation. While simulation environments provide a controlled and reproducible setting for experimentation, they often fail to capture the full complexity of real-world operational systems. In particular, simulation-based evaluations may not adequately reflect issues such as noisy data, system heterogeneity, dynamic workloads, and unexpected failure modes, which are common in production environments. As a result, the performance of AI-based approaches reported in the literature may not directly translate to real-world scenarios. Moreover, only a limited number of studies validate their approaches in real-world or production-grade environments. This highlights a gap between research-oriented solutions and their practical deployment, especially in high-availability systems where robustness, scalability, and reliability are critical. Addressing this gap requires more extensive validation in operational settings, as well as the development of benchmarks and evaluation frameworks that better reflect real-world conditions.

2.11. Security Considerations and Vulnerabilities

The adoption of AI-based techniques in operational and high-availability systems also introduces relevant security challenges. One important concern is the vulnerability of machine learning and deep learning models to adversarial attacks. Carefully crafted inputs can lead to incorrect predictions or misclassification, potentially affecting critical functions such as anomaly detection and fault diagnosis. Another critical aspect is the reliability of the telemetry data used for monitoring and decision making. In distributed systems, telemetry streams may be incomplete, noisy, or even intentionally manipulated, which can compromise the effectiveness of AI-based models. Data poisoning or manipulation attacks can lead to degraded model performance or incorrect system behavior. These issues are particularly relevant in high-availability environments, where incorrect predictions may have direct consequences on system reliability and service continuity. Therefore, ensuring robustness against adversarial inputs and improving the trustworthiness of monitoring data represent key challenges for the practical adoption of AI-driven solutions.

2.12. Search Strategies

It is worth noting that the use of the search string "(high availability OR fault tolerance)" may retrieve studies that are not explicitly labeled as high-availability research. In particular, some works originate from related domains such as predictive maintenance in industrial or cyber–physical systems. These studies have been included when they address concepts that are directly relevant to high availability, such as fault prediction, anomaly detection, and reliability improvement. Their inclusion is justified by their contribution to reducing downtime and enhancing system resilience, which are core objectives of high-availability systems.

While the current comparative analysis focuses on the core characteristics of the reviewed approaches, additional dimensions such as dataset properties, evaluation settings, scalability, and deployment constraints represent important aspects that deserve further investigation. We consider this an interesting direction for future work. In particular, an important aspect not explicitly addressed in this work concerns the computational requirements of AI models, such as inference latency and memory consumption, which are particularly relevant in Edge and resource-constrained environments. We consider this a promising direction for future research and plan to investigate it in a dedicated study.

3. AI-Based Monitoring and Fault Detection

AI-based monitoring and detection refers to the activity related to the continuous analysis of operational telemetry, such as metrics, logs, and traces, to recognize abnormal behavior, performance degradation, and early signals of failure. Unlike traditional rule-based monitoring systems that rely on static thresholds, these models learn the normal behavior of the system under varying load conditions and across different time windows. As a result, they can detect statistically significant deviations and correlate related symptoms across multiple components. The goal is to reduce alert noise, shorten the mean time to detect incidents, and support faster recovery by identifying issues before they violate reliability targets [30].

3.1. Machine Learning Approaches

Kalaskar et al. [23] study supervised monitoring for Cloud infrastructure. The technique is a classical ML pipeline: preprocessing and labeling jobs as failed/successful, explicitly handling strong class imbalance with under-sampling and over-sampling, and then training and comparing multiple models. The evaluated learners include linear regression, decision trees and boosting-style methods, with C5.0 and XGBoost reported as the most accurate and reliable for fault/performance prediction. Limitations include heavy dependence on dataset preprocessing and labeling definitions, the risk of optimistic evaluation if train test splits are too similar to the training environment, and limited discussion of how to maintain accuracy under concept drift and changing schedulers in real-time production Clouds.

Odyurt et al. [31] propose a data-driven method to detect and classify performance anomalies in industrial cyber–physical systems (CPSs). It focuses on extra-functional behavior (EFB) metrics such as timing, CPU/communication activity, power, observed during repetitive workloads. The technique segments execution into phases and builds phase fingerprints as behavioral signatures and behavioral passports. Fingerprints are obtained by fitting regression models to phase time-series and measuring deviations via goodness-of-fit indicators (e.g., R^2, RMSD). Then these deviation features are fed to standard ML classifiers, notably referred to decision trees and random forests, to detect anomalies and distinguish anomaly types. Results report a very high anomaly classification accuracy in some settings (up to $\sim$99%), suggesting the fingerprints capture meaningful behavioral changes. Limitations include reliance on careful phase/metric selection and feature engineering, possible generalization issues across systems, and added monitoring/modeling overhead.

Li et al. [32] propose a fully data-driven, distributed integrated design of fault detection (FD) and fault-tolerant control (FTC) for large-scale interconnected systems. A key technical step is to handle unknown interconnection inputs (neighboring subsystem states) by representing them using only measurable local and neighboring input/output data. Using this predictor, each subsystem builds a residual generator via least-squares identification and applies a chi-square test statistic to decide fault-free vs. faulty operation. When faults are detected, a data-driven predictive control law computes an FTC input sequence to track a reference trajectory and recover performance after fault propagation. Limitations include the need to find a network cluster containing all interacting neighbors without external input for accurate unknown-input representation, which may be restrictive for general topologies.

Boubiche et al. [19] introduce ASOCIDA, an adaptive self-optimizing framework for anomaly detection and collaborative isolation in wireless sensor networks. It performs local real-time monitoring and flags anomalies using Mahalanobis distance dissimilarity with adaptive EWMA and dynamic thresholds. Thresholds and smoothing factors are adjusted on-the-fly (with Bayesian optimization) to track changing data variability while limiting

false alerts. Suspected anomalies are then validated, in a collaborative way, via a distributed reputation system combined with a Byzantine-fault-tolerant weighted consensus. Reported results show high accuracy and very low false alarms with energy savings and latency improvements, while limits include simulation-based validation and added overhead from optimization/consensus in dense networks.

Kaur et al. [27] propose an intelligent fault-tolerant data routing scheme for Industrial IoT supported by the wireless sensor network (WSN). It targets both node faults and link faults, which otherwise may reduce packet delivery, increase delay, and shorten lifetime of the network. The method forms clusters via K-means at the sink and periodically checks cluster-head fitness using lightweight HELLO/REPLY exchanges to limit the control overhead. Limitations include reliance on a powerful, always-available sink, added computational complexity for RLWOA, and validation mostly through simulation plus a limited-scale testbed.

Khomenko et al. [33] present a modular, local-first IoT architecture to monitor and control office environments using Home Assistant as the core automation engine. The technical backbone integrates heterogeneous protocols (MQTT, Zigbee, REST, WebSocket, Wi-Fi/Thread/Matter gateways). Time-series analytics is performed with simple predictive models such as ridge regression and bagged trees with periodic retraining to avoid drift. Limitations include infrastructure complexity, dependence on careful configuration/tuning of automations and statistical thresholds, and validation being limited to a single office-scale deployment rather than multi-building, long-term field studies.

3.2. Deep Learning Approaches

Guo et al. [18] propose OADS, an online log anomaly monitoring and fault-detection system that aims to improve service availability by moving from purely reactive alarms to predictive.The technique couples a long sequence forecaster, LSP-Informer, with a real-time detector, LADM, implemented as a temporal convolutional network with attention. The approach is evaluated on parsed HDFS logs then converted into multivariate time-series through template extractions and embeddings. The results report a strong detection performance (TCNA-based LADM F1 around 0.986) and an effective multi-step forecasting approach (LSP-Informer F1 around 0.980 for 5-step-ahead prediction), allowing anomalies to be anticipated 10 steps in advance. The system also maintains interpretability and robustness even in the event of masking, achieving a Jaccard index greater than 0.73 even in the presence of significant missing segments. The main limitations are the operational sensitivity to log parsing and preprocessing choices, the risk that performance may not transfer across log sources with different template vocabularies and rhythms, and the added deployment complexity and latency of cascading prediction plus detection when production conditions drift rapidly.

Bang et al. [24] present an AI-driven framework for real-time failure detection in high-performance computing (HPC) system logs to reduce mean time to detect (MTTD) and improve availability. Monitoring is performed on streaming system logs from many subsystems, and fault detection is learned with HABERT, a bidirectional encoder representations from transformers. It is a RoBERTa based classifier adapted through domain and task adaptive pretraining, then fine-tuned using imbalance-aware training. In a three-month AI training production run, near real-time detection reduced failure propagation time from roughly 15 min to about seconds and improved availability from 95.9% to 97.8%, translating to thousands of additional compute hours. The paper also reports scalability measurements showing feasible throughput and $\sim$1.4 s end-to-end latency for 1000-line batches under their tested setup. Limitations include difficulty separating intermittent from permanent issues using logs alone, limited fault localization when identical error

lines appear across many nodes, and challenges in quantifying the scope or impact of a detected issue at job, node, or system level, which matters for automated mitigation in high-availability environments.

Selvaraj et al. [20] propose an automatic fault-detection framework for self-healing networks with the aim of improving reliability and reducing downtime in modern mobile/optical network operations. Fault classification is performed with a multi-scale dilated bidirectional LSTM with an attention mechanism (MDBiLSTM-AM), designed to capture long-range temporal patterns while enlarging receptive fields via dilation and focusing on informative features via attention. The approach is validated on three benchmark datasets available from public repositories. The reported results show as the proposed model achieves an accuracy around 97.4% across the three datasets and improves over CNN, VGG19, ResNet, and standard LSTM, with lower false positive/negative rates. Furthermore, ablation studies show that combining multi-scale dilation and attention to BiLSTM yields better performance than simpler variants. Main limitations include reliance on curated benchmark datasets rather than live operator traces, added complexity from oversampling and deep sequence modeling, and the lack of explicit hyperparameter/architecture optimization for MDBiLSTM-AM, which the authors note as the subject for future work.

Wang et al. [34] address missing acceleration data imputation for Structural Health Monitoring when sensors fail or data are lost. It proposes a time–frequency-aware GAN, with a generator which is a stacked LSTM aiming to capture long-range temporal dynamics. Training combines adversarial learning with both time-domain reconstruction (MSE) and frequency-domain reconstruction by adopting power spectral density (PSD). This method was verified in SHM scenarios, such as a three-span bridge and reference structures (e.g., ASCE/QUGS configurations). Compared to CAE-based GAN variants and a VAE baseline, the LSTM–time–frequency approach usually achieves lower RMSE/MAE and higher R^2. It better preserves the modal/spectral content (peaks and dynamics) in comparison with approaches that only matching the waveform in time. Inference is reported to be fast (millisecond-scale), enabling near real-time use. The main drawbacks are the higher computational cost (PSD–LSTM) and sensitivity to the amount/quality/placement of available healthy sensors and noise.

Shekarian et al. [35] present an IoT-based smart greenhouse monitoring system designed to keep operating reliably even when some sensors fail or produce missing/abnormal readings. Its architecture connects sensor nodes to a sink node, then to an Edge/Cloud pipeline (MQTT/Node.js with database storage) for remote monitoring and analysis. The main technique is an approach based on artificial intelligence to detect faulty sensors. From a methodological point of view, four 1D-CNN models are trained: three regression models to estimate indoor/outdoor temperature and humidity, and a classification model to predict discretized CO and brightness levels. The reported performance is strong for regression (sub-degree temperature RMSE and a few-percent humidity RMSE) and good for classification (around 90% for luminosity and 83% for CO). Compared to simpler baselines, such as linear regression, CNN models generally improve accuracy and remain usable in different weather conditions, even with multiple faulty sensors. The limitations of the approach include weaker CO/lux results due to class imbalance and sensor quality, plus added system and computational complexity from model training and deployment.

Isern et al. [36] present a cyber–physical system (CPS) to protect critical smart grid infrastructures by integrating remote control/protection with smart video surveillance over a single communication network. The enabling networking technology is Time-Sensitive Networking (TSN), used to ensure deterministic latency and bandwidth for

time-critical substation control while coexisting with bandwidth-hungry video streams. Reliability in the control subsystem is strengthened via high-availability seamless redundancy (HSR) among remote terminal units (RTUs), providing seamless failover under single communication failures. On the surveillance side, Edge nodes (NVIDIA Jetson SoCs) perform real-time person detection using a MobileNetV2-based DCNN, while a Cloud server performs multi-person tracking and perimeter monitoring and issues alarms. Limitations include a lot of false negatives (partially visible people are ignored by design) and added system complexity from coordinating TSN scheduling, Edge/Cloud analytics, and RTU interoperability.

Almasoudi [29] target power grid resilience by using AI. The work uses supervised deep learning with hybrid feature extraction and sequence modeling: the author trains three CNN-based hybrids, CNN–RNN, CNN–LSTM, and CNN–GRU, where 1D convolution layers learn local patterns from the fault related time-series and the recurrent block models temporal dependencies before a final sigmoid output for fault classification. The experimentation uses real operational data collected from a working Saudi grid station during 2017–2022, with variables such as the number of faults, the number of users, including VIP users, time and recorded actions, split 70% training and 30% validation and test, and trained in Python with Keras and TensorFlow on a GPU workstation. The results indicate that the CNN–GRU variant achieves the best performance, reaching a prediction accuracy of 93.92% and outperforming CNN–LSTM and CNN–RNN in terms of error metrics, with MAE of 0.14, loss of 0.10, and RMSE of 0.05. Furthermore, it shows a significant improvement over earlier non-CNN baselines on the same dataset, supporting the effectiveness of combining CNN-based feature extraction with GRU-based temporal modeling for reliable fault detection and timely remediation. Limitations are that the study is tied to one station and a small set of aggregated variables and faults are rare, so broader generalization needs further validation, and the paper itself notes that additional research is needed to confirm findings and test other models and settings.

For the convenience of the reader, we summarize, in Table 3, the most used AI-based techniques for monitoring and anomaly detection along with their main characteristics, while in Table 4 we summarizes, on the basis of our recent discussion, the pros and cons of the different AI models analyzed for monitoring and fault detection.

Table 3. Recurring AI macro-techniques for monitoring and fault/anomaly detection.

Macro-Technique	Input	Models/Mechanisms	Main Use
Supervised ML	Telemetry/features	DT, RF [31]; XGBoost, C5.0 [23]; linear/ridge [33]	Fault/anomaly classification; interpretable drivers
Residual/statistical	I/O signals	LS residuals–χ^2 test [32]	Fault decision via test statistic (distributed-friendly)
Distance/thresholds	Sensor streams	Mahalanobis–adaptive EWMA–thresholds [19]	Fast anomaly flagging; adaptive false-alarm control
Deep sequence models	Logs/time-series	TCN+attn (+forecast) [18]; Transformers [24]; dilated BiLSTM+attn [20]; CNN–GRU/CNN–LSTM hybrids [29]	Temporal anomaly detection and early warning; fault classification from sequential telemetry

Table 3. *Cont.*

Macro-Technique	Input	Models/Mechanisms	Main Use
Imputation/ reconstruction	Gappy time-series	LSTM-GAN–PSD [34]; 1D-CNN surrogates [35]	Recover missing data to keep monitoring reliable
Detection–mitigation	Context–alarms	Predictive control [32]; RL–metaheuristic routing [27]; consensus/isolation [19]; remedial action enablement via fast classification [29]	Trigger recovery actions; limit propagation

Table 4. Practical trade-offs of common model families for monitoring and fault detection.

Family	Advantages	Typical Limitations
Trees/ensembles (RF, boosting)	Strong performance on tabular telemetry; handles nonlinearities; some interpretability; efficient inference	Sensitive to labeling/preprocessing and imbalance choices [23]; may degrade under drift if not recalibrated [33]
Statistical/residual methods	Explicit decision statistic (thresholding); good for distributed/engineering settings [32]; controllable false alarms	May rely on structural assumptions; can be restrictive under complex topologies [32]
Deep sequence models (TCN/LSTM/Transformers; CNN–RNN hybrids)	Learns temporal features; strong on logs/time-series; supports early warning via forecasting [18,24]; effective fault classification from sequential telemetry with hybrid CNN–GRU/LSTM designs [29]	Heavier deployment; log parsing/template dependence; portability issues across sources [18,24]; may be station/system-specific and sensitive to rare-fault regimes without careful validation [29]
Generative/imputation models	Maintains monitoring under missing data; can preserve spectral/physical properties [34]	Additional component to train/monitor; sensitive to sensor placement/noise [34]

3.3. Deployment Considerations in Cloud and Edge

Deploying AI-based monitoring and fault detection in Cloud and Edge environments requires addressing constraints that go beyond offline model accuracy, including data collection and preprocessing, latency and throughput, robustness to drift, and safe integration with operational control loops. From a systems perspective, Cloud deployments typically demand high throughput ingestion and low end-to-end latency to reduce mean time to detect: transformer-based log classifiers can be effective but require careful batching, resource provisioning, and monitoring of inference latency under peak load [24], while cascaded forecasting-plus-detection stacks can introduce additional compute and operational complexity despite enabling earlier warning [18].

At the Edge, constraints shift toward limited computing resources, power, and intermittent connectivity, promoting lightweight models, local processing, and selective offloading placed on the Cloud. Practical designs often place fast, simple detection close to sensors (e.g., distance-based scoring with adaptive smoothing and thresholds) and reserve heavier validation or coordination for both distributed or back end components, in order to balance false alarms with communication costs [19]. Similarly, IoT deployments frequently combine protocol heterogeneity (e.g., MQTT/Zigbee gateways) with periodic retraining and conservative thresholds to remain stable under drift and configuration changes [33]. Edge–Cloud splits are common in CPS scenarios where near-real-time inference runs on em-

bedded devices while global aggregation/tracking or alert orchestration runs remotely; this improves responsiveness but increases integration complexity and requires clear contracts for timing, bandwidth, and failure modes [36]. In industrial monitoring where remedial actions depend on fast classification, hybrid CNN–RNN models may be attractive because they can be implemented efficiently at inference time once trained, but they still require disciplined retraining and monitoring for drift when operating conditions, protection policies, or reporting practices change [29]. Missing or faulty sensor data is another recurring deployment issue: models may need to impute or estimate key signals to keep downstream monitoring functional, but these components add training/maintenance overhead and can be sensitive to sensor placement and noise [34,35].

3.4. Discussion

The surveyed literature confirms that AI-enabled monitoring and fault detection is maturing from isolated, offline predictors into system-level pipelines that must operate under realistic constraints such as rare failures, heterogeneous observability, and continuously evolving workloads. A first clear takeaway is that classical supervised learning remains highly competitive and often preferable in operational settings when reliable features and labels can be obtained. Tree-based methods and ensembles provide strong accuracy with comparatively low inference cost and offer operator-friendly explanations (e.g., split rules and feature importance), which supports practical alert tuning and incident triage [23,31]. However, these approaches are also the most sensitive to dataset construction: labeling policies, aggregation windows, and imbalance treatments can materially change results, and overly similar train/test splits can yield optimistic estimates that may not hold after scheduling or workload changes [23]. This highlights a methodological gap between benchmark-style evaluations and production conditions, where temporal validation, drift monitoring, and periodic recalibration are essential.

Deep learning approaches are most beneficial when the monitored signal is strongly sequential or weakly structured, especially for logs and multivariate time-series where manual feature engineering is brittle. Sequence models (TCNs/LSTMs) and transformer-based encoders can provide high detection quality and earlier warning through forecasting, and attention mechanisms partially address interpretability by highlighting salient segments [18,20,24]. Nevertheless, these gains come with operational costs: heavier inference, additional preprocessing (e.g., log parsing/template vocabularies), and potentially reduced transferability across environments with different logging rhythms or templates [18,24]. Hybrid CNN–RNN models for fault classification, as in grid telemetry, occupy a middle ground: they can be effective on sequential numeric signals and relatively efficient at inference, but their reported performance may depend strongly on the station/site, the aggregation level of variables, and the rarity of faults, reinforcing the need for temporal splits and robustness checks before operational use [29]. This suggests that DL systems should be evaluated not only on F1/accuracy but also on deployment-centric metrics such as end-to-end latency, retraining frequency, robustness to missing data, and performance decay under drift.

Across domains, missing data and sensor faults emerge as a practical bottleneck that directly affects monitoring reliability. Two complementary strategies appear: (i) making detection robust to missingness through model design and masking, and (ii) reconstructing or estimating missing signals via predictive/generative models so that downstream monitoring continues to function [18,34,35]. In contrast, reconstruction-based approaches can preserve important physical characteristics, they introduce an additional component that must itself be monitored and maintained, and their reliability depends on sensor placement and noise conditions [34]. In IoT deployments, simpler predictive surrogates can be a pragmatic com-

promise, but they may struggle under imbalance or poor sensor quality [35]. In station-level industrial telemetry, where variables may be aggregated and faults are rare, similar concerns apply: performance can look strong in samples while still being vulnerable to distribution shift and low event counts, so robustness and recalibration strategies remain central [29].

Finally, the architecture choice is fundamental. Centralized pipelines simplify management and enable heavy models, but they depend on stable connectivity and can become bottlenecks at scale [24]. Edge and distributed designs reduce latency and improve privacy, yet require careful engineering of protocols, gateways, and configuration, and often rely on conservative thresholds or periodic retraining to remain stable [19,33]. Overall, the evidence suggests that future progress will come less from proposing new standalone models and more from end-to-end, reproducible evaluations that account for drift, imbalance, missingness, and the safety of automated mitigation actions, ideally validated on long-running deployments rather than short, curated experiments [23,29,33].

4. AI-Based Fault Prediction

Failure prediction is a strategy designed with the aim of improving the reliability of high-availability (HA) systems by predicting potential failures before they occur. Predictive mechanisms trigger automatic repair, adaptive resource allocation, or failover activation, reducing downtime and operational impact. In modern Cloud, Edge, and IoT infrastructures, traditional threshold-based methods are insufficient, and AI-based techniques are increasingly being adopted.

4.1. Machine Learning Techniques

Machine learning (ML) methods have become a key element in predicting failures in HA systems thanks to their ability to model complex, nonlinear dependencies in operational data. Traditional HA mechanisms often rely on static thresholds or predefined rules, which do not take into account the dynamic, multidimensional, and often stochastic nature of modern Cloud, Edge, and distributed systems. ML algorithms, in contrast, can learn patterns from historical telemetry, logs, and performance metrics, enabling proactive fault detection and prediction that can significantly reduce downtime and improve system reliability [37–41].

Several studies have highlighted the effectiveness of classical supervised learning methods in fault prediction processes. Mohammed et al. [42] applied SVM, random forest, k-Nearest Neighbors, decision trees, and Linear Discriminant Analysis to virtualized HPC systems. Their findings highlight that SVMs often achieve the highest predictive accuracy. This is particularly true with datasets characterized by high dimensionality and mixed feature types. In addition to Cloud computing, ML has been applied to industrial systems for predictive maintenance.

The adoption of ML in Software Defect Prediction (SDP) provides further evidence of their utility. Surveys indicate that SVM, RF, logistic regression (LR), Naïve Bayes (NB), and neural networks remain the dominant algorithms for predicting fault-prone modules in software repositories [43–45]. Hybrid approaches that combine ensemble learning with feature selection or metaheuristic optimization (e.g., PSO, genetic algorithms) have been shown to improve accuracy, especially in high-dimensional datasets [46]. Class imbalance remains a major challenge across HA and SDP applications, as failure events are typically rare. Techniques such as the Synthetic Minority Over-Sampling Technique (SMOTE), cost-sensitive learning, and adaptive class weighting have been employed to mitigate bias towards the majority class [47].

Temporal dependencies represent another limitation of classical ML models. Although engineered features such as sliding windows and aggregated statistics can capture short-term

dependencies, they may fail to account for longer temporal correlations that precede system failures [48]. To address this, several studies integrate ML with unsupervised or semi-supervised techniques. Isolation forests, autoencoders, and clustering methods (e.g., DBSCAN, fuzzy clustering) have been applied to identify latent anomalies in telemetry data, providing early warnings of system degradation [25,26]. These anomaly detection approaches complement supervised learning by highlighting previously unseen failure patterns.

The operational integration of ML models into HA systems is critical. Yang and Kim [43] proposed a predictive framework that interfaces with OpenStack and Kubernetes, using supervised ML output to trigger automated recovery actions. Similarly, Uppal et al. [49] applied random forests and SVMs to real-time sensor data in hospital IoT systems, demonstrating that ML predictions can be integrated into live monitoring dashboards to alert operators before service degradation occurs. These studies show that beyond accuracy, deployment considerations such as inference latency, scalability, and interpretability are essential for practical adoption.

Several works have explored hybrid approaches that combine ML with optimization or heuristics. Carter et al. [26] investigated predictive observability by coupling ML models with statistical process control, highlighting the role of ML in supporting proactive maintenance strategies. Borghesi et al. [50] demonstrated online fault classification in HPC systems using ML trained with fault injection data, showing that real-time learning pipelines can achieve low-latency predictions while continuously updating models with new telemetry. Netti et al. [51] applied ML in fault injection analytics to discover previously unknown failure modes in Cloud infrastructures, reinforcing that ML can facilitate exploratory and predictive tasks.

The interpretability of ML models is another critical concern. Tree-based models such as random forests inherently provide feature importance metrics, offering insights into which operational parameters most strongly influence predictions. In contrast, more complex models such as neural networks or gradient boosting machines, while often achieving higher predictive performance, require post-hoc explainability techniques to ensure that system operators can trust automated recovery decisions [26,44].

Finally, cross-domain studies highlight that ML models trained on historical telemetry from one environment (e.g., Cloud HPC clusters) can sometimes generalize to similar operational contexts, although adaptation via transfer learning or fine-tuning is often necessary [47,52]. This indicates the potential for reusable ML frameworks that can accelerate fault prediction across heterogeneous HA systems while reducing the cost of data collection and labeling.

In conclusion, ML techniques offer a versatile and empirically validated toolkit for fault prediction in HA systems. From Cloud job and task failures to industrial machinery and IoT sensor networks, classical supervised learning remains foundational due to its interpretability, robustness, and efficiency, while hybrid and ensemble methods enhance predictive accuracy under high-dimensional, noisy, or imbalanced datasets. Integration with unsupervised anomaly detection, online learning pipelines, and operational monitoring systems further extends the applicability of ML, establishing it as a critical enabler for proactive and data-driven HA management in modern computing infrastructures.

4.2. Deep Learning Approaches

Deep learning (DL) approaches have increasingly been adopted in fault and failure prediction tasks for high-availability (HA) systems due to their ability to automatically learn hierarchical and temporal features from raw telemetry, without the need for manual feature engineering [44]. Unlike traditional machine learning, DL models can capture complex, nonlinear dependencies and interactions in multivariate time-series collected from Cloud infrastructures, industrial machinery, and IoT devices [53].

Recurrent neural networks (RNNs), including long short-term memory (LSTM) and gated recurrent unit (GRU) architectures, are particularly suitable for sequential data, as they capture long-term dependencies that often precede system faults [25].

Hybrid architectures, such as CNN–LSTM and CNN–GRU models, have demonstrated superior performance in scenarios where spatial and temporal correlations are important. CNN layers extract local patterns across multiple sensor channels, while LSTM or GRU layers capture temporal evolution. In industrial predictive maintenance studies, CNN–LSTM hybrids outperformed standalone CNN or LSTM models, achieving accuracy up to 96% and F1-scores near 95%. Similar results have been reported in the predictive maintenance of wind turbines, where CNN–LSTM models effectively learned degradation patterns from SCADA telemetry data [44].

Optimization techniques, such as particle swarm optimization (PSO), have been used to fine-tune Bi-LSTM networks, producing improved precision and recall for fault prediction in charging stations and other distributed systems [54]. LSTM encoder–decoder architectures have also been used to reconstruct normal operational sequences, enabling anomaly detection even with limited labeled failure data [53].

Despite their advantages, DL models face challenges. Large labeled datasets are required for training, which can be difficult to obtain in HA systems where failures are rare [44]. Moreover, deep architectures are computationally intensive, limiting real-time deployment in Edge and Cloud environments. Interpretability is another concern: CNN–LSTM or LSTM models are often black boxes, making it difficult for operators to understand the basis for predictions. To address this, Explainable AI (XAI) techniques, such as attention mechanisms, are integrated to highlight influential features and time segments that contribute to predictions.

Recent research also explores alternative deep learning models, including autoencoders, deep belief networks, and transformer-based architectures, for fault prediction and remaining useful life estimation [55,56]. These models provide complementary benefits, such as unsupervised anomaly detection, but may impose higher computational costs.

In summary, DL approaches—from RNNs and GRUs to hybrid CNN–LSTM networks—have substantially advanced fault prediction in HA and predictive maintenance applications [44]. They automate feature extraction, model complex temporal and spatial dependencies, and enhance early warning capabilities. However, challenges remain in data availability, computational cost, interpretability, and deployment in real-time HA environments, motivating ongoing research in optimization, model compression, and explainable AI.

4.3. Incorporating Software Defect Prediction

Software Defect Prediction (SDP) has become a mature research area in software engineering, focusing on identifying defect-prone modules prior to deployment. The methodologies developed in SDP are highly relevant to high-availability (HA) systems, as they provide transferable strategies for anticipating rare but critical failure events. Traditional SDP approaches used classical machine learning models such as decision trees, random forests, support vector machines (SVMs), and logistic regression [25,47]. Over time, deep learning approaches, including CNN, LSTM, Bi-LSTM, and hybrid architectures, have been increasingly adopted to capture complex patterns in source code representations [57]. These models can automatically learn feature hierarchies, minimizing the need for manual feature engineering and enabling the detection of subtle signals associated with defects or anomalies.

A core insight from SDP is the challenge of class imbalance, where defective modules are often a small fraction of the code base. Techniques such as SMOTE (Synthetic Minority Oversampling Technique), cost-sensitive learning, and ensemble weighting have been systematically studied to address this problem [47]. Similarly, in high-availability (HA) systems, failure events are inherently rare, and the resulting imbalance in telemetry data

can significantly degrade the performance of predictive models if not properly addressed. Several studies highlight that, in the absence of explicit imbalance handling, models may achieve high overall accuracy while failing to effectively identify the minority class. This limitation underscores the importance of adopting evaluation metrics such as recall, F1-score, and the Matthews Correlation Coefficient, which provide a more reliable assessment in imbalanced scenarios.

Feature representation learning is another key lesson from the SDP. Early SDP approaches relied on manually engineered metrics such as cyclomatic complexity, code churn, and module size. However, more recent research demonstrates that embedding-based representations of code tokens, abstract syntax trees (ASTs), and graph structures can significantly improve prediction performance [57]. Applying this insight to HA systems, telemetry data streams can be encoded using embeddings or graph representations to automatically extract informative features for fault prediction, reducing the reliance on domain-specific feature engineering.

SDP also highlights the challenge of cross-project generalization. Models trained on a single software project may not generalize to other projects due to differences in programming practices, code bases, or frameworks [57]. This has direct relevance for HA systems, which operate on heterogeneous workloads, Cloud nodes, and Edge devices. Transfer learning, domain adaptation, and multi-task learning strategies from SDP can be leveraged to improve generalizability, allowing models to transfer knowledge from previously observed systems to new operational environments [58]. Empirical studies have shown that fine-tuning pretrained models in target datasets can substantially increase predictive accuracy and reduce false negatives [57].

The integration of ensemble and hybrid modeling is another lesson from SDP. Ensembles of ML and DL models can improve robustness, reduce variance, and mitigate the impact of noisy data [47]. In HA systems, ensembles can combine predictions from different models trained on diverse telemetry modalities (e.g., CPU load, network traffic, sensor readings), enhancing early warning capabilities. Additionally, hybrid approaches that combine traditional ML with DL models enable the exploitation of both interpretability and hierarchical feature extraction [57].

Evaluation rigor in SDP has been a strong focus, with systematic studies advocating for cross-validation, cross-project testing, and statistically robust metrics [58]. For HA systems, such evaluation protocols can ensure models are not overfitting to specific nodes or workloads and can reliably predict rare failures across different operational contexts. Recent work has also explored semi-supervised learning for SDP, leveraging unlabeled code to improve predictions, which may inspire similar approaches in HA systems where error labels are scarce [57].

Finally, SDP studies provide insights into interpretability and actionable insights. While deep learning models excel in predictive performance, they often lack transparency. Explainable AI techniques, including attention mechanisms, feature attribution, and saliency maps, have been successfully applied to SDP to help developers understand why a module is flagged as defect-prone [57]. Translating this to HA systems, interpretable models can help operators identify which telemetry signals most influence fault predictions, increasing trust and enabling more effective preemptive actions.

In conclusion, incorporating methodologies from Software Defect Prediction provides HA systems with a wealth of strategies: handling class imbalance, learning hierarchical feature representations, applying transfer learning for heterogeneous systems, leveraging ensembles for robustness, and maintaining rigorous evaluation protocols. By adapting these lessons, predictive models in HA environments can achieve higher reliability, lower

downtime, and more actionable insights, supporting operational continuity in Cloud, Edge, and IoT infrastructures [47,57].

4.4. Application Scenarios

AI-based fault prediction techniques have been widely applied across diverse domains characterized by high availability requirements and complex operational dynamics. In Cloud computing environments, predictive models are typically used to forecast job and task failures to support proactive remediation such as dynamic migration, resource scaling, and load balancing. Ensemble methods (e.g., random forest, gradient boosting) and deep learning architectures have shown robust performance on large public traces such as Google Cluster, Trinity, and Mustang, enabling anticipatory actions that prevent severe service disruptions [44,59].

In industrial predictive maintenance, AI models analyze multivariate time-series from sensors embedded in machinery, production lines, and critical infrastructure. Hybrid deep learning frameworks (e.g., CNN–LSTM) are particularly effective in these settings due to their ability to learn spatial–temporal patterns inherent in sensor data, outperforming traditional ML models when forecasting equipment degradation and imminent failures. For instance, predictive maintenance of wind turbines and heavy industrial machinery has benefited from deep architectures that integrate local feature extraction with temporal modeling, reducing unplanned downtime and maintenance costs.

IoT and Edge computing contexts pose additional challenges due to heterogeneous data sources, intermittent connectivity, and resource constraints. Here, lightweight AI models and federated learning paradigms are increasingly adopted to enable real-time fault prediction without centralizing sensitive data. Research has shown that CNN and RNN variants can accurately forecast sensor faults and network anomalies in smart environments—such as healthcare IoT systems—enabling Edge agents to trigger local remediation actions before system degradation becomes critical [49,60].

Applications extend to software reliability and defect forecasting, where AI models originally developed in Software Defect Prediction (SDP) are repurposed to anticipate failure-prone modules in HA software systems. By identifying components with high fault likelihood, HA orchestration platforms can adapt deployment strategies or allocate additional redundancy to prevent outages [47].

Finally, high-performance computing (HPC) infrastructure benefits from AI-driven online fault classification and anomaly detection, where predictive models operate on streaming logs and telemetry to detect fault precursors with minimal latency. Fault injection datasets and real-time analytics frameworks demonstrate that ML and DL models can be integrated into monitoring pipelines to provide continuous, context-aware fault predictions [50,51].

Overall, these application scenarios illustrate the breadth of environments where AI-based fault prediction enhances availability, from Cloud and Edge to industrial systems and software stacks, by enabling proactive interventions, reducing downtime, and optimizing operational workflows.

4.5. Performance Metrics and Evaluation

Evaluating AI-based fault prediction models in high-availability (HA) systems requires a comprehensive approach that considers both classification accuracy and operational relevance. While traditional metrics like accuracy provide a general overview of model performance, HA environments are inherently imbalanced: faults occur rarely, yet have critical consequences. As a result, metrics such as precision, recall, F1-score, AUC–ROC, and Matthews Correlation Coefficient (MCC) are essential to capture the model's ability to correctly identify rare failure events without generating excessive false alarms [25,47].

Table 5 summarizes the key classification metrics commonly reported in the literature. Accuracy, while widely reported, can be misleading in imbalanced datasets where the majority class dominates. Precision and recall provide complementary views: precision measures the fraction of correctly predicted faults among all predicted faults, while recall reflects the fraction of actual faults that are successfully predicted. F1-score balances these two, offering a more robust measure in rare-event scenarios. MCC is particularly recommended in HA contexts because it considers all four outcomes (true positives, true negatives, false positives, false negatives) and remains informative under severe class imbalance [47].

Table 5. Classification metrics commonly used in HA fault prediction.

Metric	Description and Critical Commentary
Accuracy	Ratio of correctly classified instances to total instances. Misleading in imbalanced datasets where the majority class dominates.
Precision	Proportion of true positives among all positive predictions. High precision reduces unnecessary interventions.
Recall (Sensitivity)	Proportion of actual faults correctly detected. Critical in HA as missed faults may lead to outages.
F1-score	Harmonic mean of precision and recall. Provides a balanced measure, especially when both false positives and false negatives are important.
AUC–ROC	Evaluates classification quality across thresholds. Useful for selecting operating points in probabilistic models.
MCC	Balanced metric suitable for imbalanced datasets. Provides more reliable comparison across models than accuracy.

In addition to classification performance, temporal and operational metrics are critical for evaluating the real-world utility of predictive models. Metrics such as lead time, mean time to prediction (MTTP), and false alarm rate directly influence how predictive results translate into concrete actions. For example, a model with a high F1 score but minimal lead time may detect failures too late to enable automatic resolution, thus limiting its practical benefits [53]. Table 6 summarizes these operational metrics and their relevance.

Several studies illustrate the interplay between classification and operational metrics. For instance, ensemble ML models applied to Google Cluster traces achieve high F1-scores, but their operational utility depends on lead time: predictions must occur early enough to enable failover or migration [47]. In IoT predictive maintenance, CNN–LSTM models can forecast component degradation accurately, yet the computational cost and latency must be balanced to ensure real-time applicability.

In conclusion, the evaluation of AI-based failure prediction models requires a dual focus: traditional classification metrics able to quantify the statistical performance of models, and operational metrics—measuring their practical value in high-risk HA environments. By combining these perspectives it is possible to ensure that these models are not only accurate in detecting failures, but they result also effective in reducing downtime, preventing malfunctions, and enabling timely resolution, which nowadays are essential features for modern Cloud, Edge, and IoT infrastructures.

Table 6. Operational and temporal metrics for HA fault prediction.

Metric	Use in HA Systems and Critical Insights
Lead Time	Time difference between prediction and actual fault. Longer lead times enable proactive interventions such as migration or resource reallocation.
Mean Time to Prediction (MTTP)	Average duration between fault detection and occurrence. Short MTTP may reduce operational benefit despite high classification accuracy.
Mean Time to Recovery (MTTR) Impact	Quantifies reduction in recovery time due to early fault prediction. Links predictive performance to measurable operational gains.
False Alarm Rate	Frequency of incorrect fault predictions. High rates can cause unnecessary remediation, wasting resources and potentially destabilizing HA systems.
Prediction Horizon Coverage	Range over which model predictions remain reliable. Longer horizons are desirable for strategic planning but may decrease precision.

4.6. Comparative Analysis and Critical Commentary

When reviewing the literature on AI-based fault prediction, several trends and distinctions emerge that are essential for both researchers and practitioners. First, classical ML models (e.g., RF, SVM) remain foundational due to their interpretability and relatively low computational requirements, especially when engineered features effectively capture failure precursors [42]. Ensemble variants often outperform single classifiers because in such a way it is possible to reduce variance and improve robustness across heterogeneous datasets. Unfortunately, their reliance on manual feature engineering and inability to learn temporal dynamics intrinsically limit their performance in scenarios dominated from sequential patterns.

Deep learning architectures—particularly LSTM, GRU, and CNN–LSTM hybrids—are excellent in capturing temporal and spatial correlations in complex telemetry. Empirical studies report that these models achieve higher recall and F1-scores in predictive maintenance and IoT scenarios with high-dimensional time-series. The trade-off, however, is increased training complexity, data requirements, and computational cost. For real-time HA systems, deploying deep models demands careful design to balance latency and prediction quality.

Integration of unsupervised and semi-supervised learning (e.g., autoencoders, isolation forests) has shown promise for anomaly detection and forecasting without abundant labeled failure data [25,26]. These methods can identify latent failure patterns that supervised approaches might miss, especially in environments where labeling is costly or infeasible.

Cross-domain adaptability is another dimension of comparison. Techniques from Software Defect Prediction (SDP) inform strategies for transfer learning and domain adaptation, helping models trained on one system generalize to another [58]. However, domain shift remains a challenge: models often require fine-tuning or additional calibration to maintain performance across different infrastructures.

Interpretability and operator trust are recurring themes. ML models like random forests provide feature importance scores, supporting actionable insights. Deep models, despite higher accuracy, require Explainable AI (XAI) techniques (e.g., attention mecha-

nisms) to yield interpretable predictions, which is crucial in HA settings where false alarms or incorrect actions can have significant operational costs [44].

In synthesizing these comparisons, it is clear that no single method universally dominates. Instead, a hybrid and integrated approach—combining classical ML for baseline interpretation, DL for complex pattern extraction, and unsupervised methods for anomaly detection—often yields the best practical performance. Evaluation using both classification and operational metrics (as in Tables 5 and 6) ensures that models are judged not only by statistical accuracy but also by *real-world impact* on system availability and maintenance efficiency.

5. AI-Driven Recovery Mechanisms

AI-driven recovery mechanism leverages machine learning to choose and execute recovery actions that restore service health with minimal downtime and user impact. Automated remediation strategies focus on corrective interventions such as triggering failover, rolling back a faulty release, restarting or isolating unhealthy components, and applying pre-approved mitigation based on the most likely incident cause. Adaptive resource management instead targets resilience through dynamic capacity and control tuning by adjusting autoscaling, resource quotas, concurrency limits, and back-pressure so the system stays within reliability and performance objectives even under degraded conditions [61].

5.1. Automated Remediation Strategies

Song et al. [62] propose ACPR, an adaptive predictive repair mechanism for erasure coded distributed storage that reduces recovery cost by using failure prediction outputs to decide what to repair immediately and what to delay: it assumes an ML based predictor can identify soon to fail nodes and their fault type: immediate reboot, slow reboot, or forcible decommission.The authors use an upstream node failure predictor which outputs both the next failure time and the failure type of each node, where the type is a multi-class classification among immediate reboot, slow reboot, and forcible decommission, while ACPR is a rule-based classification and scheduling layer that couples two repair actions, migration from the at risk node and pipeline reconstruction across surviving nodes, and dynamically monitors low risk stripes to avoid unnecessary traffic. They implement ACPR on HDFS with RS(3,2) and evaluate on 9 Alibaba Cloud ECS instances with a 90 GB file and repeated repairs of 60 blocks, comparing against FastPR and ECPipe, showing up to 15.2% and 33.5% lower per block repair time, up to 83% and 86.2% lower total repair time, and up to 74.1% and 84.4% lower repair traffic when only high risk blocks are repaired. Limitations are that effectiveness hinges on the accuracy and timeliness of the external ML failure prediction and on correctly estimating access heat, experiments are on a small RS(3,2) cluster with larger scale claims mainly supported by modeling assumptions, and if prediction is wrong or the node fails early ACPR can degenerate toward pure pipeline reconstruction behavior.

Zota et al. [63] define a practical framework for building an Agentic AIOps system whose recovery mechanism is closed-loop IT operations. The paper explicitly proposes a discriminative ML component for incident classification and service-element identification, a retrieval augmented generation module for diagnosis and resolution selection, and an LLM-based component for ticket communication, all coordinated by an orchestrator and connected to automation tooling such as playbooks or pipelines to execute the actual remediation. Key limitations are that the work is a forward-looking framework without empirical performance results, so effectiveness depends on future implementations and data quality, and the paper explicitly notes that security aspects are out of scope and would require separate governance and controls in real deployments.

Toprani et al. [64] propose an agentic AI recovery mechanism for Cloud infrastructure security, where resilience is achieved by automatically detecting and correcting misconfigurations in Infrastructure as Code before deployment, reducing the need for manual incident response after release. The paper exploits a report agent for generating developer ready remediation actions, then implemented by means of Anthropic Claude Sonnet 3.5 v2 via Amazon Bedrock and Titan Text Embeddings v2 for retrieval. Limitations have mainly practical nature: performance and accuracy depend on keeping the knowledge base current, the system can struggle with conditional or highly parameterized templates, remediation can be overly conservative in some cases, and the evaluation set is small thus generalization and scalability need a broader test phase.

Al-Na'amneh et al. [65] presents ML-Heal, a system for automatic fault management in Cloud service compositions. It uses a continuous loop to monitor the system, detect issues, and apply recovery actions to avoid failure propagation. The recovery strategy is based on a reinforcement learning approach (in a simplified form such as Q-learning). The evaluation is performed in a simulated environment with different types of faults, including crashes, resource exhaustion, network issues, bugs, and database failures, and compares ML-Heal with simpler baselines. Results show improvements in recovery time and overall workflow success rate. However, the evaluation is mainly simulation-based, and aspects like real-world deployment, scalability, and learning reliability are not fully validated.

In the study by Boubiche et al. [19], anomalies are detected using the Mahalanobis distance over multivariate sensor features also combined with an adaptive EWMA whose smoothing factor changes with local variance, while Bayesian optimization continuously tunes key thresholds and coefficients in order to balance sensitivity and false alarms under changing conditions. For robust isolation, it adds a distributed reputation-weighted consensus with Byzantine fault tolerance, so that only sufficiently trusted neighbors contribute to confirm anomalies before a suspect node is locally isolated and later re-evaluated for reintegration.

In the study by Isern et al. [36], moving-object ROIs are extracted with a Mixture-of-Gaussians background model and then classified as person or not using a deep CNN based on MobileNetV2, followed by a second deep CNN that produces a 128D appearance embedding for re-identification; multi-person tracking is completed in the Cloud by matching detections to tracks with a distance matrix solved by the Hungarian algorithm, generating MODBUS/TCP alarms toward the control subsystem.

For recovery, Kaur et al. [27] select replacement cluster heads and alternative forwarding paths using a hybrid reinforcement learning–whale optimization algorithm (RLWOA) executed centrally at the sink. Routing decisions optimize a multi-objective fitness that considers residual energy (lifetime), delay, and congestion, enabling fast rerouting when links break.

5.2. Adaptive Resource Management

Ipek et al. [66] propose AIRSDN, an AI-based routing approach for SDN that adapts path selection to mitigate performance degradation using a logistic regression model trained on selected traffic features. While the results show clear improvements in RTT, throughput, and video quality over baseline methods in a Mininet-based setup, the evaluation remains limited in scope. The use of a small, synthetic dataset and a controlled simulation environment raises concerns about generalizability to real networks, where traffic dynamics and variability are more complex. Moreover, the choice of a relatively simple model, although effective in this setting, may not capture more intricate network behaviors. Scalability and deployment aspects are also not fully addressed, particularly considering the reported CPU

fluctuations at the controller level. Overall, the approach is promising but still lacks strong evidence of robustness and applicability in realistic, large-scale scenarios.

Zhao et al. [28] propose an AI-based framework for adaptive resource management in CPS, combining time-series prediction with reinforcement learning for allocation decisions. Although the results show higher rewards and faster convergence compared to simple baselines, the evaluation is entirely simulation-based and relies on synthetic demand patterns. The use of reward as the main metric, without direct measurement of system-level performance (e.g., latency or utilization), limits the practical interpretation of the results. Moreover, prediction quality is not rigorously assessed, and the generalization to real-world CPS scenarios remains unclear.

Lv et al. [67] introduce an AI-driven resource management approach for healthcare SDN–IoT systems using aerial nodes to handle demand fluctuations and maintain service quality. While the reported improvements in energy consumption and latency suggest potential benefits, the study is constrained by simulation-based validation and the use of a repurposed dataset. In addition, the lack of clear details on model design and training reduces reproducibility, and the impact of computational overhead and scalability in real deployments is not thoroughly addressed.

5.3. Integration with AI Monitoring and Detection

AI-driven recovery is most effective if it is designed as a closed loop together with AI monitoring and detection, where observability signals are first transformed into actionable incident context and then into safe remediation or control actions. From a practical viewpoint, monitoring outputs rarely give a single definitive fault label; it produces anomaly scores, predicted failure probabilities, forecasts, and correlated symptoms across components. Integration therefore requires mapping these outputs into a compact context that includes scope (affected components), likely fault type/cause, urgency (time-to-SLO violation), and uncertainty (expected false positives), so that the recovery layer can decide whether to auto-act, escalate, or wait for confirmation [18,24].

A recurring pattern is to align detection outputs with the control surface of recovery. Residual/statistical detection with an explicit test statistic can trigger fault-tolerant control reconfiguration in a principled way [32]. Failure-time/type prediction can feed cost-aware repair scheduling in storage (proactive vs. lazy repair) [62]. In distributed sensing, fast local anomaly flags are often coupled with reputation/consensus validation before isolation or reintegration to reduce acting on spurious alerts [19]. In networked systems, learned path-quality scoring embedded in controllers can treat routing itself as a recovery mechanism under degradation [66]. Agentic AIOps frameworks make the loop explicit by combining incident classification, diagnosis (often retrieval-augmented), and an orchestrator that executes pre-approved playbooks with auditing and policy checks [63].

Because recovery actions can be disruptive, integration must include guardrails and tiered automation. Low-risk actions (restart stateless services, reroute traffic, apply back-pressure) can be automated when confidence is high, while high-impact actions (rollback, isolation of critical data services, security posture changes) should require stronger evidence (cross-signal correlation, statistical thresholds, consensus) or human approval [19,32]. Drift couples both layers: changes in workloads, logging templates, or telemetry distributions can degrade detection and consequently mis-trigger remediation, motivating continuous validation and periodic recalibration before allowing fully automated recovery in production [29,33].

5.4. Discussion

The papers discussed show that the main risk is not failing to propose an action, but triggering the wrong action at the wrong time: false positives can be expensive because remediation changes the system state and can amplify instability. Approaches that keep an explicit notion of evidence and confidence, through statistical decision rules, corroboration across signals, or distributed validation, tend to offer a clearer path to trustworthy automation because operators can tune risk and understand why an action was taken [19,32]. Framework-oriented AIOps work pushes in the same direction by placing orchestration and policy checks between diagnosis and execution, but it also makes clear that governance, secure tool interfaces, and high-quality operational knowledge bases are prerequisites for safe autonomy [63].

The analyzed literature showed that the integration of detection with mitigation must be done with caution. Automated responses (rerouting, isolation, or control reconfiguration) can reduce downtime, but risk cascading effects if alerts are noisy or poorly calibrated. Approaches that include explicit decision logic, consensus validation, or statistically grounded tests can improve trustworthiness, while data-driven control and routing policies must be evaluated for stability and safety under partial observability and changing operating conditions [19,27,32]. In grid settings, the intended operational value often comes from the faster activation of remedial actions after classification, confirming the need to link false positives and quantify uncertainty before integrating model output into protection or operator workflows [29].

Predictions become operationally valuable when they map directly onto decisions such as what to repair now versus later, or which path to route traffic through under congestion [62,66]. Conversely, when the mapping is indirect, recovery relies on brittle heuristics and the benefits of smarter detection can be lost in translation. Finally, nearly all approaches are vulnerable to drift and rare-event regimes: workloads change, telemetry pipelines evolve, and true faults remain sparse, so models that perform well offline may degrade silently and start mis-triggering interventions. This reinforces the practical need for conservative automation tiers, continuous validation, and periodic recalibration before allowing aggressive self-healing in production, particularly in settings where actions are high-impact or data are strongly site-dependent [29,33].

6. Research Questions Discussion

This section synthesizes the findings of the systematic analysis by addressing the four research questions defined in Section 2.1. Rather than providing a descriptive recap of individual studies, the discussion abstracts recurring architectural patterns, methodological convergences, and operational implications emerging across Cloud, HPC, CPS, and IoT high-availability systems. The objective is to identify how AI is concretely embedded within HA architectures (RQ1), which methodological families dominate monitoring and prediction tasks (RQ2), how intelligent outputs are coupled with recovery mechanisms (RQ3), and which structural limitations and open challenges still constrain large-scale adoption (RQ4).

By organizing the evidence along these dimensions, the section highlights not only what has been implemented but also the systemic design principles and research gaps that define the current state of AI-driven operational management in high-availability infrastructures.

- **RQ1:** AI-based solutions for operational management are primarily adopted in high-availability architectures characterized by distribution, scalability requirements, and partial observability of internal states. In Cloud-native environments and large-scale data centers, AI is embedded within microservices and container-orchestrated infrastructures to enhance autoscaling, anomaly detection, and traffic engineering. These architectures typically rely on layered observability pipelines (metrics, logs, traces)

feeding ML models that operate either centrally (control-plane intelligence) or in a distributed fashion close to execution nodes. In HPC systems, AI is integrated into job schedulers and log analysis frameworks to predict node failures, classify runtime anomalies, and optimize workload placement under reliability constraints [24]. In cyber–physical systems (CPSs) and Industrial IoT deployments, AI-enhanced HA architectures combine Edge-level inference for low-latency anomaly detection with Cloud-level aggregation for model retraining and long-term optimization [19]. Distributed storage systems adopting erasure coding incorporate predictive models to anticipate disk or node failures and proactively migrate fragments to preserve data availability [62]. Similarly, SDN-enabled Cloud networking architectures embed learned path-quality estimators within controllers to maintain performance and resilience under congestion or partial link failures [66]. Across these domains, AI adoption is most prevalent in architectures featuring (i) software-defined control planes, (ii) high telemetry availability, and (iii) orchestration layers capable of executing automated remediation actions, thus enabling closed-loop intelligent operational management.

- **RQ2:** AI for monitoring and fault detection in Cloud, CPS, and IoT environments tends to converge on a small set of recurring macro-techniques that differ mainly in (i) data processing, (ii) type of models, and (iii) where computation is placed. On the classical ML side, supervised pipelines based on feature engineering remain common because they are efficient and relatively interpretable: tree-based models such as decision trees and random forests are frequently used for anomaly detection and anomaly-type classification when deviations can be summarized into discriminative features [31], while boosted ensembles (e.g., XGBoost and C5.0) are often top performers for failure/performance prediction from infrastructure telemetry, especially when combined with explicit imbalance handling through over/under-sampling [23]. Linear models also appear as lightweight predictors or baselines in operational analytics, often coupled with periodic retraining to cope with drift in day-to-day deployments [33]. Complementary to purely discriminative learning, several works adopt residual- and statistics-driven detection, where faults are identified through model mismatch rather than directly learned labels: least-squares identified residual generators paired with a χ^2 decision statistic support distributed fault detection in interconnected systems [32], and distance-based scoring (Mahalanobis distance) combined with adaptive EWMA smoothing and dynamic thresholds enables low-latency anomaly flagging in sensor networks [19]. Fault tolerance is also treated as a decision/optimization problem beyond detection: once faults are detected, data-driven predictive control can compute corrective inputs to recover performance [32], while in WSN-assisted IIoT routing, reinforcement learning combined with metaheuristic optimization is used to rapidly reconfigure routes under node/link faults while balancing delay and energy objectives [27].

  Deep learning approaches are mainly adopted when the monitored signals are strongly sequential or weakly structured, as per logs and multivariate time-series, making manual feature extraction brittle: temporal convolutional and attention-based detectors are used to operate online, sometimes preceded by multi-step forecasting to anticipate anomalies ahead of time [18], and transformer encoders adapted via domain/task pretraining support scalable log-based failure classification in HPC settings [24]. For time-series fault classification, sequence models such as multi-scale dilated BiLSTM with attention are used to enlarge receptive fields and focus on salient temporal segments [20]. In addition, hybrid convolutional–recurrent architectures (e.g., CNN–GRU/CNN–LSTM) are often selected in industrial monitoring settings where local temporal patterns and longer dependencies must both be captured, as shown in power

grid fault classification from station telemetry [29]. Finally, robustness to missing data and sensor failures is often handled through prediction or generation, enabling monitoring continuity even when sensors drop out: time–frequency-informed GANs with LSTM generators reconstruct missing structural responses while preserving spectral characteristics [34], and compact 1D-CNN models can estimate key environmental variables or discretized states in IoT deployments to tolerate faulty readings [35]. Overall, classical ML dominates when features are well-defined and interpretability is important, statistical/residual methods provide controllable decision rules in distributed settings, and deep sequence or hybrid convolutional–recurrent models are preferred for complex temporal dependencies, unstructured logs, or scarce/rare fault signatures, albeit with higher deployment and maintenance complexity [18,19,29].

- **RQ3:** AI-based methods reduce service downtime in HA systems by coupling AI monitoring outputs to automated or semi-automated recovery actions. On the monitoring side, log/time-series models (e.g., transformer classifiers or TCN-based detectors, sometimes preceded by forecasting) detect and localize incidents earlier than static thresholds, while statistical/residual techniques provide explicit decision statistics for reliable triggering [18,24,32]. These signals feed recovery mechanisms that either execute pre-approved remediation (restart, isolate, rollback, failover) via an orchestrator, or adapt system behavior through controllers. For example, failure-time/type prediction can drive proactive repair and migration decisions in erasure-coded storage to prevent data unavailability [62]; residual-based detection can directly trigger fault-tolerant control inputs to contain fault propagation [32]; and in networked/IoT settings, anomaly confirmation combined with consensus can justify isolating faulty nodes, while RL-based routing reconfigures paths to maintain delivery under node/link failures [19,27]. In Cloud networking, learned path-quality scoring embedded in an SDN controller can reroute traffic away from congestion, acting as a rapid recovery from performance degradation [66]. Across these approaches, downtime is reduced by (i) anticipating failures, (ii) narrowing action scope to the affected components, and (iii) gating automation with confidence/validation to avoid disruptive false positives [63].
- **RQ4:** The main challenges and open research issues in applying AI to operational management of HA systems concern reliability guarantees, life cycle management, scalability, and trustworthiness. First, AI models introduce probabilistic decision making into infrastructures that traditionally rely on deterministic guarantees; ensuring bounded risk in anomaly detection and prediction remains an open problem, particularly under concept drift and non-stationary workloads [33]. Second, maintaining model validity over time requires continuous retraining, validation, and monitoring pipelines (MLOps integration), which increases architectural complexity and operational overhead. Third, scalability constraints arise when deep sequence or transformer-based models process high-volume telemetry streams, potentially conflicting with latency requirements of HA control loops [24]. Data sparsity for rare fault events further limits supervised approaches, motivating research into self-supervised, transfer, and few-shot learning paradigms. Explainability and accountability also remain critical challenges, especially in CPS and IIoT contexts where automated remediation may affect physical processes [32]. Security concerns—including adversarial manipulation of telemetry data or poisoning of training pipelines—represent an emerging research frontier, as compromising the AI layer may indirectly degrade availability policies. Finally, hybrid architectures that combine rule-based safeguards with learning-based adaptation are increasingly recognized as necessary to prevent the intelligence layer itself from becoming a single point of failure. Addressing these

issues requires advances in robust learning, drift-aware adaptation, certifiable AI, and dependable AI–orchestrator co-design.

Building on the evidence synthesized across RQ1–RQ3, Figure 2 proposes a reference architecture for AI-driven HA systems, abstracting the recurring design patterns identified in the reviewed literature.

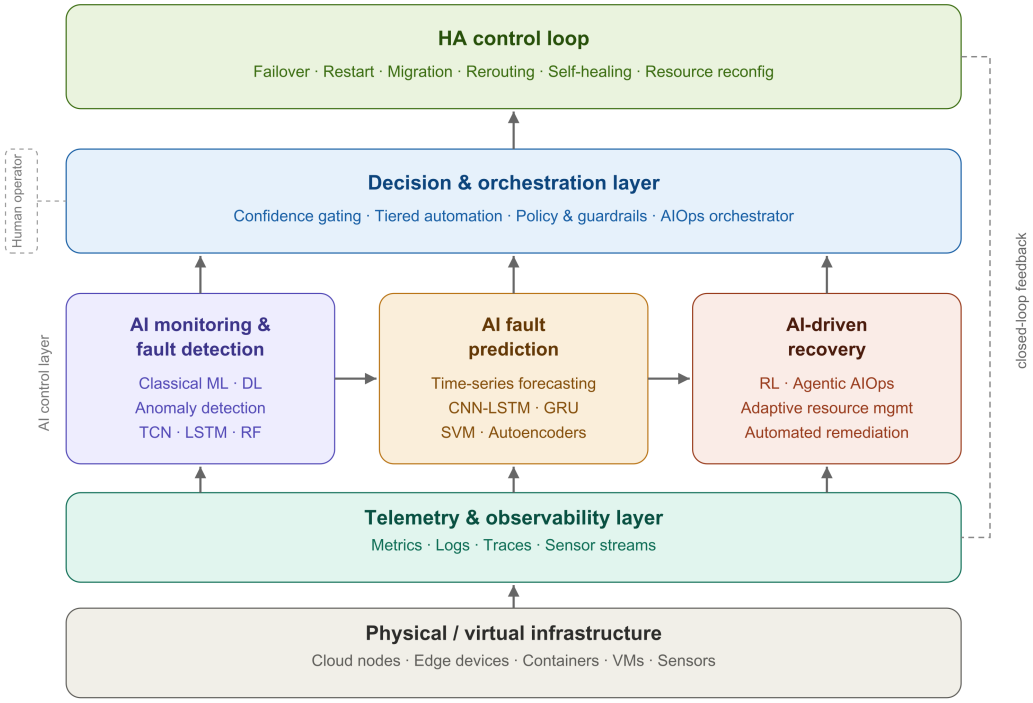

Figure 2. Conceptual architecture of AI-driven operational management in HA systems.

Figure 2 presents a reference architecture for AI-driven high-availability systems, synthesized from the recurring design patterns identified across the 42 studies reviewed. Rather than prescribing a specific implementation, the framework abstracts the structural and functional relationships that consistently emerge in the literature. The architecture is organized as a layered control structure. At its foundation, a heterogeneous physical and virtual infrastructure, spanning Cloud nodes, Edge devices, containers, and sensors, generates the operational signals that drive all higher-level functions. These signals are collected and normalized by a telemetry and observability layer, which supplies the continuous stream of metrics, logs, and traces required by AI-based components. The core of the framework is the AI control layer, composed of three functionally distinct but tightly coupled modules corresponding to the thematic categories identified in Section 2.5: AI-based monitoring and fault detection, AI-based fault prediction, and AI-driven recovery. As highlighted in RQ2 and RQ3, these modules are not independent pipelines but operate within a closed control loop, where detection outputs inform prediction models and predicted failure contexts directly condition recovery decisions. Above the AI layer, a decision and orchestration layer mediates between probabilistic AI outputs and deterministic HA actions, implementing the confidence gating, tiered automation, and governance guardrails discussed in Section 5.4. This layer also maintains a human operator interface for high-impact decisions, reflecting the practical necessity of supervised autonomy in production-grade deployments. Finally, a feedback path from the HA control loop back to the telemetry layer, represented by the dashed arc in Figure 2, captures the continuous retraining and recalibration cycle that several reviewed works identify as essential for maintaining model validity under concept

drift and evolving workloads. Taken together, this reference architecture provides a structured vocabulary for comparing existing approaches and positioning future work within a coherent operational framework.

7. Conclusions

In this review paper, we have discussed the state of the art of AI-based solutions for HA systems, by focusing in the particular and important areas of the application of AI for the main operational core mechanisms named monitoring, failure detection/anomaly prediction, and recovery. We have based our study on four main research questions which mainly focused on (i) the main architectures adopted in AI-based solutions for operational management, (ii) AI techniques employed for monitoring, fault detection, and fault prediction, (iii) how AI-based methods are used to support recovery processes and, finally, we focused on (iv) the main challenges and open research issues in applying AI to operational management of HA systems. For our analysis we selected a total of 42 studies after filtering an initial set of 5800 papers, and we classified those studies into three main categories reflecting our main interests: AI-based monitoring and fault detection, AI-based fault prediction and, finally, AI-driven recovery mechanisms in HA systems. For Q1, after our analysis, we observed that AI adoption is most prevalent in architectures featuring (i) software-defined control planes, (ii) high telemetry availability, and (iii) orchestration layers capable of executing automated remediation actions, thus enabling closed-loop intelligent operational management. For Q2 our analysis allowed us to classify a number of recurring macro-techniques for monitoring and fault detection Cloud and IoT environments that mainly differ in the data processing techniques (machine learning vs. deep learning models). For Q3, our analysis revealed that AI-based recovery methods mostly rely on coupling on AI-based automated or semi-automated recovery actions. Finally, in order to give an answer to the last question (Q4), we observed that AI models introduce probabilistic decision making into infrastructures that traditionally rely on deterministic guarantees; moreover we observed that the adoption of AI techniques requires continuous retraining, validation, and monitoring pipelines and there may arise scalability constraints when high-volume streams are analyzed by the AI. There are also open issues of data sparsity, explainability and accountability, as well as security concerns.

Finally, this work proposes a reference architecture for AI-driven HA systems, synthesizing the recurring design patterns identified across the reviewed studies into a unified layered framework covering telemetry, monitoring, fault prediction, and automated recovery.

The analysis presented in this manuscript highlights that, in fact, a significant amount of research has been conducted on AI-based approaches to support high availability in critical systems. Anyway, as finally stated in the response in Q4, some aspects should be further explored along the dimension of AI-based HA systems as, for example, the probabilistic nature of the decision-making process and the security concerns. In a future work we aim to explore the integration of the agentic AI system paradigm [68] to achieve complex, multi-step goals aiming to support the high availability of specific systems and with minimal human supervision and also to assist humans in recovering decisions critical real-time scenarios.

Author Contributions: Conceptualization, L.F., R.G., F.M., D.R. and G.M.L.S.; methodology, L.F., R.G., F.M., D.R. and G.M.L.S.; investigation, L.F., R.G., F.M., D.R. and G.M.L.S.; writing—original draft preparation, L.F., R.G., F.M., D.R. and G.M.L.S.; writing—review and editing, L.F., R.G., F.M., D.R. and G.M.L.S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the project PIACERI 2024-26, funded by the University of Catania.

Data Availability Statement: No new data were created or analyzed in this study.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Bartlett, J.; Gray, J.; Horst, B. Fault tolerance in tandem computer systems. In *The Evolution of Fault-Tolerant Computing: In the Honor of William C. Carter*; Springer: Berlin/Heidelberg, Germany, 1987; pp. 55–76.
2. Somasekaram, P.; Calinescu, R.; Buyya, R. High-availability clusters: A taxonomy, survey, and future directions. *J. Syst. Softw.* **2022**, *187*, 111208. [CrossRef]
3. Vega-Luna, J.; Salgado-Guzmán, G.; Cosme-Aceves, J.; Tapia-Vargas, V.; Andrade-González, E. Linux cluster programming for high availability with an Oracle instance. *J. Appl. Res. Technol.* **2025**, *23*, 108–119. [CrossRef]
4. Hark, R.; Koziolek, H.; Yussupov, V.; Eskandani, N. Kubernetes High-Availability Software Architecture Options for Two-Node Clusters in IoT Applications. In Proceedings of the 2025 IEEE 22nd International Conference on Software Architecture Companion (ICSA-C), Stuttgart, Germany, 31 March–4 April 2025; IEEE: Piscataway, NJ, USA; pp. 69–76.
5. Malleni, S.S.; Sevilla, R.; Lema, J.C.; Bauer, A. Bridging Clusters: A Comparative Look at Multi-Cluster Networking Performance in Kubernetes. In Proceedings of the Proceedings of the 16th ACM/SPEC International Conference on Performance Engineering, Toronto, ON, Canada, 5–9 May 2025; pp. 113–123.
6. Nguyen, T.N.; Lee, J.; Vitumbiko, M.; Kim, Y. A design and development of operator for logical kubernetes cluster over distributed clouds. In Proceedings of the NOMS 2024–2024 IEEE Network Operations and Management Symposium, Seoul, Republic of Korea, 6–10 May 2024; IEEE: Piscataway, NJ, USA; pp. 1–6.
7. Pereira, P.; Araujo, J.; Melo, C.; Santos, V.; Maciel, P. Analytical models for availability evaluation of edge and fog computing nodes. *J. Supercomput.* **2021**, *77*, 9905–9933. [CrossRef]
8. Sangolli, D.R.; Ravindrarao, N.M.; Patil, P.C.; Palissery, T.; Liu, K. Enabling high availability edge computing platform. In *Proceedings of the 2019 7th IEEE International Conference on Mobile Cloud Computing, Services, and Engineering (MobileCloud), Paris, France, 24–27 June 2019*; IEEE Computer Society: Los Alamitos, CA, USA; pp. 85–92.
9. Swetha, R.; Thriveni, J.; Venugopal, K. Resource Utilization-Based Container Orchestration: Closing the Gap for Enhanced Cloud Application Performance. *SN Comput. Sci.* **2025**, *6*, 191. [CrossRef]
10. Liu, X.; Cheng, B.; Wang, S. Availability-aware and energy-efficient virtual cluster allocation based on multi-objective optimization in cloud datacenters. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 972–985. [CrossRef]
11. Sivakumar, J.; Salman, N.R.; Salman, F.R.; Salimova, H.R.; Ghimire, E. AI-driven cyber threat detection: Enhancing security through intelligent engineering systems. *J. Inf. Syst. Eng. Manag.* **2025**, *10*, 790–798. [CrossRef]
12. Gbenle, P.; Abieba, O.A.; Owobu, W.O.; Onoja, J.P.; Daraojimba, A.I.; Adepoju, A.H.; Chibunna, U.B. A Conceptual Model for Scalable and Fault-Tolerant Cloud-Native Architectures Supporting Critical Real-Time Analytics in Emergency Response Systems. 2021. Available online: https://worldscientificnews.com/a-conceptual-model-for-scalable-and-fault-tolerant-cloud-native-architectures-supporting-critical-real-time-analytics-in-emergency-response-systems/ (accessed on 10 January 2026).
13. Toumi, N.; Bagaa, M.; Ksentini, A. Machine learning for service migration: A survey. *IEEE Commun. Surv. Tutor.* **2023**, *25*, 1991–2020. [CrossRef]
14. Endo, P.T.; Rodrigues, M.; Gonçalves, G.E.; Kelner, J.; Sadok, D.H.; Curescu, C. High availability in clouds: Systematic review and research challenges. *J. Cloud Comput.* **2016**, *5*, 16. [CrossRef]
15. Maciel, P.; Dantas, J.; Melo, C.; Pereira, P.; Oliveira, F.; Araujo, J.; Matos, R. A survey on reliability and availability modeling of edge, fog, and cloud computing. *J. Reliab. Intell. Environ.* **2022**, *8*, 227–245. [CrossRef]
16. Hasan, M.; Goraya, M.S. Fault tolerance in cloud computing environment: A systematic survey. *Comput. Ind.* **2018**, *99*, 156–172. [CrossRef]
17. Kitchenham, B. *Procedures for Performing Systematic Reviews;* Technical Report; Keele University: Keele, UK, 2004.
18. Guo, Y.; Sun, Y.; Xiong, P. Research on Online Log Anomaly Detection Model Based on Informer. *Concurr. Comput. Pract. Exp.* **2025**, *37*, e70300. [CrossRef]
19. Boubiche, S.; Boubiche, D.E.; Toral-Cruz, H. ASOCIDA: Adaptive self-optimizing approach for anomaly detection and collaborative isolation in wireless sensor networks. *Ad Hoc Netw.* **2025**, *178*, 103959. [CrossRef]
20. Selvaraj, C.; Justin, J. Automatic fault detection in self healing network using multiscale dilated bidirectional long short-term memory with attention mechanism. *Signal Image Video Process.* **2025**, *19*, 907. [CrossRef]
21. Zhuang, N.; Ren, Z.; Yang, D.; Tian, X.; Wang, Y. A Knowledge-Guide Data-Driven Model with Selective Wavelet Kernel Fusion Neural Network for Gearbox Intelligent Fault Diagnosis. *Sensors* **2025**, *25*, 7656. [CrossRef]
22. Tran, D.H.; Nguyen, V.L.; Nguyen, H.; Jang, Y.M. Self-Supervised Learning for Time-Series Anomaly Detection in Industrial Internet of Things. *Electronics* **2022**, *11*, 2146. [CrossRef]

23. Kalaskar, C.; Thangam, S. Fault Tolerance of Cloud Infrastructure with Machine Learning. *Cybern. Inf. Technol.* **2023**, *23*, 26–50. [CrossRef]
24. Bang, B.; Kim, J.; Song, U.; Lee, J.; Ma, J. HAMS: An AI-driven framework for real-time failure detection in HPC system logs. *J. Supercomput.* **2025**, *81*, 1364. [CrossRef]
25. Gollapalli, M.; Al Metrik, M.; Alnajrani, B.; Alomari, A.; AlDawoud, S.; Almunsour, Y.; Abdulqader, M.; Aloup, K. Task Failure Prediction Using Machine Learning Techniques in the Google Cluster Trace Cloud Computing Environment. *Math. Model. Eng. Probl.* **2022**, *9*, 545–553. [CrossRef]
26. Carter, A.; Imtiaz, S.; Naterer, G.F. Review of interpretable machine learning for process industries. *Process Saf. Environ. Prot.* **2023**, *170*, 647–659. [CrossRef]
27. Kaur, G.; Chanak, P. An Intelligent Fault Tolerant Data Routing Scheme for Wireless Sensor Network-Assisted Industrial Internet of Things. *IEEE Trans. Ind. Inform.* **2023**, *19*, 5543–5553. [CrossRef]
28. Zhao, X.; Guo, F.; Huang, A.; Ding, J.; Yan, C.; Yuan, W.; Su, Y.; Li, Q.; Zhang, Q. Adaptive resource management in dynamic Cyber–Physical Systems using Artificial Intelligence. *Eng. Appl. Artif. Intell.* **2025**, *162*, 112409. [CrossRef]
29. Almasoudi, F.M. Enhancing Power Grid Resilience through Real-Time Fault Detection and Remediation Using Advanced Hybrid Machine Learning Models. *Sustainability* **2023**, *15*, 8348. [CrossRef]
30. De la Cruz Cabello, M.; Prince Sales, T.; Machado, M.R. AIOps for log anomaly detection in the era of LLMs: A systematic literature review. *Intell. Syst. Appl.* **2025**, *28*, 200608. [CrossRef]
31. Odyurt, U.; Pimentel, A.D.; Gonzalez Alonso, I. Improving the robustness of industrial Cyber–Physical Systems through machine learning-based performance anomaly identification. *J. Syst. Archit.* **2022**, *131*, 102716. [CrossRef]
32. Li, B.; Li, W.; Yang, Y. Data-Driven Distributed Fault Detection and Fault-Tolerant Control for Large-Scale Systems: A Subspace Predictor-Assisted Integrated Design Scheme. *IEEE Trans. Cybern.* **2025**, *55*, 5346–5357. [CrossRef]
33. Khomenko, Y.; Babichev, S. Modular IoT Architecture for Monitoring and Control of Office Environments Based on Home Assistant. *IoT* **2025**, *6*, 69. [CrossRef]
34. Wang, Z.; Kachireddy, M.; Mondal, T.G.; Tang, W.; Jahanshahi, M.R. Time-frequency informed stacked long short-term memory-based generative adversarial network for missing data imputation in sensor networks. *Eng. Appl. Artif. Intell.* **2025**, *155*, 110973. [CrossRef]
35. Mohammadhossein Shekarian, S.; Aminian, M.; Mohammad Fallah, A.; Akbary Moghaddam, V. AI-powered sensor fault detection for cost-effective smart greenhouses. *Comput. Electron. Agric.* **2024**, *224*, 109198. [CrossRef]
36. Isern, J.; Jimenez-Perera, G.; Medina-Valdes, L.; Chaves, P.; Pampliega, D.; Ramos, F.; Barranco, F. A Cyber-Physical System for Integrated Remote Control and Protection of Smart grid Critical Infrastructures. *J. Signal Process. Syst.* **2023**, *95*, 1127–1140. [CrossRef]
37. Comi, A.; Fotia, L.; Messina, F.; Rosaci, D.; Sarné, G.M. Grouptrust: Finding trust-based group structures in social communities. In *Proceedings of the International Symposium on Intelligent and Distributed Computing, Paris, France, 21–23 September 2016*; Springer: Cham, Switzerland; pp. 143–152.
38. Comi, A.; Fotia, L.; Messina, F.; Pappalardo, G.; Rosaci, D.; Sarné, G.M. Forming homogeneous classes for e-learning in a social network scenario. In *Proceedings of the Intelligent Distributed Computing IX: Proceedings of the 9th International Symposium on Intelligent Distributed Computing–IDC'2015, Guimarães, Portugal, 21–23 September 2015*; Springer: Cham, Switzerland; pp. 131–141.
39. Comi, A.; Fotia, L.; Messina, F.; Pappalardo, G.; Rosaci, D.; Sarné, G.M. An evolutionary approach for cloud learning agents in multi-cloud distributed contexts. In *Proceedings of the 2015 IEEE 24th International Conference on Enabling Technologies: Infrastructure for Collaborative Enterprises, Larnaca, Cyprus, 15–17 June 2015*; IEEE: Piscataway, NJ, USA; pp. 99–104.
40. Comi, A.; Fotia, L.; Messina, F.; Pappalardo, G.; Rosaci, D.; Sarné, G.M. Using semantic negotiation for ontology enrichment in e-learning multi-agent systems. In *Proceedings of the 2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems, Blumenau, Brazil, 8–10 July 2015*; IEEE: Piscataway, NJ, USA; pp. 474–479.
41. Comi, A.; Fotia, L.; Messina, F.; Rosaci, D.; Sarnè, G.M. A qos-aware, trust-based aggregation model for grid federations. In *Proceedings of the OTM Confederated International Conferences "On the Move to Meaningful Internet Systems", Amantea, Italy, 27–31 October 2014*; Springer: Berlin/Heidelberg, Germany; pp. 277–294.
42. Mohammed, B.; Awan, I.; Ugail, H.; Younas, M. Failure prediction using machine learning in a virtualised HPC system and application. *Clust. Comput.* **2019**, *22*, 471–485. [CrossRef]
43. Yang, H.; Kim, Y. Design and Implementation of Machine Learning-Based Fault Prediction System in Cloud Infrastructure. *Electronics* **2022**, *11*, 3765. [CrossRef]
44. Asmawi, T.N.T.; Ismail, A.; Shen, J. Cloud failure prediction based on traditional machine learning and deep learning. *J. Cloud Comput.* **2022**, *11*, 47. [CrossRef]
45. Albattah, W.; Alzahrani, M. Software defect prediction based on machine learning and deep learning techniques: An empirical approach. *AI* **2024**, *5*, 1743–1758. [CrossRef]

46. Yu, H.; Zhang, Y.; Li, Q.; Wang, X.; Liu, Z.; Chen, Y. Research on software defect prediction based on machine learning. *J. Chongqing Univ.* **2025**, *48*, 10–21. [CrossRef]
47. Pachouly, J.; Ahirrao, S.; Kotecha, K.; Selvachandran, G.; Abraham, A. A systematic literature review on software defect prediction using artificial intelligence: Datasets, Data Validation Methods, Approaches, and Tools. *Eng. Appl. Artif. Intell.* **2022**, *111*, 104773. [CrossRef]
48. Ubal, C.; Di-Giorgi, G.; Contreras-Reyes, J.E.; Salas, R. Predicting the Long-Term Dependencies in Time Series Using Recurrent Artificial Neural Networks. *Mach. Learn. Knowl. Extr.* **2023**, *5*, 1340–1358. [CrossRef]
49. Uppal, M.; Gupta, D.; Juneja, S.; Sulaiman, A.; Rajab, K.; Rajab, A.; Elmagzoub, M.A.; Shaikh, A. Cloud-Based Fault Prediction for Real-Time Monitoring of Sensor Data in Hospital Environment Using Machine Learning. *Sustainability* **2022**, *14*, 11667. [CrossRef]
50. Borghesi, A.; Libri, A.; Benini, L.; Bartolini, A. Online Anomaly Detection in HPC Systems. In Proceedings of the IEEE International Conference on Artificial Intelligence Circuits and Systems (AICAS), Hsinchu, Taiwan, 18–20 March 2019; pp. 229–233. [CrossRef]
51. Netti, A.; Kiziltan, Z.; Babaoglu, O.; Sîrbu, A.; Bartolini, A.; Borghesi, A. FINJ: A Fault Injection Tool for HPC Systems. In *Proceedings of the Euro-Par 2018: Parallel Processing Workshops, Turin, Italy, 27–31 August 2018;* Springer: Cham, Switzerland; Volume 11339, pp. 800–812. [CrossRef]
52. Jorayeva, M.; Akbulut, A.; Catal, C.; Mishra, A. Machine Learning-Based Software Defect Prediction for Mobile Applications: A Systematic Literature Review. *Sensors* **2022**, *22*, 2551. [CrossRef] [PubMed]
53. Yang, Y.; Wang, H. Random Forest-Based Machine Failure Prediction: A Performance Comparison. *Appl. Sci.* **2025**, *15*, 8841. [CrossRef]
54. Ji, Y.; Gu, L.; Huang, H.; Wang, W.; Zhang, W. Research on fault prediction and management of charging station combined with deep learning model. *Int. J. Low Carbon Technol.* **2025**, *20*, 848–854. [CrossRef]
55. Wang, L.; Zhu, Z.; Zhao, X. Dynamic predictive maintenance strategy for system remaining useful life prediction via deep learning ensemble method. *Reliab. Eng. Syst. Saf.* **2024**, *245*, 110012. [CrossRef]
56. Akheel, M.; Anjanikar, A.; Navale, S.; Sairam, V.; Pareek, P.; Wagle, S.A.; Ansari, M.A. Proactive maintenance for water pumps: Multitask neural network for fault detection and RUL estimation. *Life Cycle Reliab. Saf. Eng.* **2025**, *15*, 249–260. [CrossRef]
57. Abdu, A.; Zhai, Z.; Abdo, H.A.; Algabri, R.; Lee, S. Graph-Based Feature Learning for Cross-Project Software Defect Prediction. *Comput. Mater. Contin.* **2023**, *77*, 161–180. [CrossRef]
58. Zhang, W.; Zhao, J.; Qin, G.; Wang, S. Cross-project defect prediction based on autoencoder with dynamic adversarial adaptation. *Appl. Intell.* **2025**, *55*, 324. [CrossRef]
59. Bommala, H.; Udaya, M.V. Machine learning job failure analysis and prediction model for the cloud environment. *High Confid. Comput.* **2023**, *3*, 100165. [CrossRef]
60. Rafique, S.H.; Abdallah, A.; Musa, N.S.; Murugan, T. Machine learning and deep learning techniques for internet of things network anomaly detection—Current research trends. *Sensors* **2024**, *24*, 1968. [CrossRef]
61. Bellini, E.; D'Aniello, G.; Flammini, F.; Gaeta, R. Situation Awareness for Cyber Resilience: A review. *Int. J. Crit. Infrastruct. Prot.* **2025**, *49*, 100755. [CrossRef]
62. Song, Y.; Zheng, P.; Tian, Y.; Wang, B. ACPR: Adaptive Classification Predictive Repair Method for Different Fault Scenarios. *IEEE Access* **2024**, *12*, 4631–4641. [CrossRef]
63. Zota, R.D.; Bărbulescu, C.; Constantinescu, R. A Practical Approach to Defining a Framework for Developing an Agentic AIOps System. *Electronics* **2025**, *14*, 1775. [CrossRef]
64. Toprani, D.; Madisetti, V.K. LLM Agentic Workflow for Automated Vulnerability Detection and Remediation in Infrastructure-as-Code. *IEEE Access* **2025**, *13*, 69175–69181. [CrossRef]
65. Al-Na'amneh, Q.; Aljawarneh, M.; Hazaymih, R.; Alsarhan, A.; Alnafisah, K.; Alshammari, N.; Alshammari, S. Autonomous Self-Adaptation in the Cloud: ML-Heal's Framework for Proactive Fault Detection and Recovery. *Int. J. Adv. Comput. Sci. Appl.* **2025**, *16*. [CrossRef]
66. İpek, A.D.; Cicioğlu, M.; Çalhan, A. AIRSDN: AI based routing in software-defined networks for multimedia traffic transmission. *Comput. Commun.* **2025**, *240*, 108222. [CrossRef]
67. Lv, J.; Babbar, H.; Rani, S. AI-Driven Resource Management for Energy-Efficient Aerial Computing in Large-Scale Healthcare SDN–IoT Systems. *IEEE Internet Things J.* **2025**, *12*, 23536–23549. [CrossRef]
68. Acharya, D.B.; Kuppan, K.; Divya, B. Agentic AI: Autonomous intelligence for complex goals—A comprehensive survey. *IEEE Access* **2025**, *13*, 18912–18936. [CrossRef]