AI-driven Self-Healing in Cloud-native 6G Networks through Dynamic Server Scaling

Anastasios Giannopoulos
R&D Department
Four Dot Infinity
Athens, Greece
angianno@fourdotinfinity.com

Sotirios Spantideas
R&D Department
Four Dot Infinity
Athens, Greece
sospanti@fourdotinfinity.com

Panagiotis Trakadas
R&D Department
Four Dot Infinity
Athens, Greece
ptrak@fourdotinfinity.com

Jesus Perez-Valero
Dept. of Information and Comm. Eng.
University of Murcia
Murcia, Spain
jesus.perezvalero@um.es

Ginés Garcia-Aviles
Dept. of Information and Comm. Eng.
University of Murcia
Murcia, Spain
gigarcia@um.es

Antonio Skarmeta Gomez
Dept. of Information and Comm. Eng.
University of Murcia
Murcia, Spain
skarmeta@um.es

2025 IEEE 11th International Conference on Network Softwarization (NetSoft) | 979-8-3315-4345-7/25/$31.00 ©2025 IEEE | DOI: 10.1109/NETSOFT64993.2025.11080631

***Abstract*—The increasing complexity of cloud-native 6G networks necessitates intelligent resource management to optimize scalability, energy efficiency, and service reliability. This paper presents an AI-driven self-healing mechanism for dynamic server activation within the a cloud-native system. The proposed framework integrates three key frameworks: the Management and Orchestration Framework (MOF) for policy-based network service orchestration, the Cloud Continuum Framework (CCF) for dynamic resource scaling, and the Artificial Intelligence and Machine Learning Framework (AIMLF) for predictive analytics and anomaly detection. By leveraging AI models, the system continuously monitors workload variations, forecasts resource demand, and dynamically scales computing resources, ensuring optimal energy efficiency and SLA compliance. The proposed self-healing workflow enables proactive server activation and deactivation, addressing load bursts and underutilization scenarios. Numerical evaluations, including real-world traffic data analysis, demonstrate that our approach significantly improves power consumption, load balancing, and resource utilization compared to traditional static resource allocation methods.**

***Index Terms*—6G network, dynamic server activation, energy efficiency, load forecasting, proactive scaling, resource allocation, resource scaling, SLA enforcement**

I. Introduction

A. Cloud-native 6G Networks

Cloud-native 6G networks represent the next evolution in mobile communication, built on scalable, software-defined architectures that seamlessly integrate cloud, edge, and network services [1]. Unlike previous generations, 6G networks are designed to be highly flexible, self-healing, driven by Artificial Intelligence (AI), and energy efficient, enabling adaptive service provisioning in real time.

In principle, a 6G system comprises four primary frameworks that collectively manage network resources: (i) Management and Orchestration Framework (MOF), which is responsible for high-level policy enforcement and network service orchestration, interacting with AI-driven modules to optimize service availability and energy efficiency; (ii) Cloud Continuum Framework (CCF), which ensures seamless integration between cloud and edge computing resources, dynamically managing computational workloads, handling real-time resource allocation and scaling based on AI-driven predictions; (iii) Artificial Intelligence and Machine Learning Framework (AIMLF) [2], which provides AI-based analytics, anomaly detection, and predictive modeling to optimize performance, while also continuously refining Machine Learning (ML) models for adaptive network optimization [3]; (iv) Radio Access Network/Core Network (RAN/CN) Services (NSs), which comprises RAN and CN functions, providing the foundational virtualized connectivity infrastructure, integrating with the other frameworks to ensure a complete service provision with optimized traffic handling and robust network performance. These four frameworks are deployed on virtualized cloud or edge resources and work in tandem to create an intelligent, self-healing 6G network that autonomously adapts to varying traffic patterns, minimizes energy consumption, and enhances overall service reliability.

This work was supported in part by the 6G-Cloud Project funded from the European Union's HORIZON-JU-SNS-2023 programme under grant agreement No 101139073 (www.6g-cloud.eu).

B. AI-aided Self-healing of Computing Resources

In Cloud-native 6G networks, computing resources must dynamically adapt to fluctuating traffic demands while ensuring low latency and energy efficiency. Traditional static resource management approaches often lead to over-provisioning, underutilization, and performance bottlenecks due to their inability to anticipate workload variations. Self-healing mechanisms [4] address this challenge by autonomously detecting anomalies, predicting future demand, and applying real-time corrective actions to restore optimal network conditions. Given the complexity and scale of 6G infrastructures, AI-driven decision-making is essential to enable real-time data analytics, pattern recognition, and adaptive optimization [5]. ML models

can continuously analyze traffic loads, predict performance degradations, and proactively adjust server activations, workload balancing, and anomaly mitigation strategies.

C. Related Work

Self-healing and proactive resource management techniques have been extensively explored in dynamic computing environments, including fault tolerance, load balancing, and energy efficiency via AI-driven and rule-based approaches. First, Lee et al. [6] proposed a collaborative resource allocation mechanism for self-healing in self-organizing networks, where network nodes dynamically adjust resource allocation to compensate for failures and restore service availability. Mashaly and Kühn [7] explored a load balancing strategy for cloud-based content delivery networks via adaptive server (de-)activation, dynamically adjusting active servers based on real-time traffic conditions. Later, Kuehn and Mashaly [8] extended this work, proposing an automatic energy efficiency management framework that enables data centers to activate or deactivate servers based on load-dependent sleep modes.

Beyond static threshold-based management, AI-driven approaches have significantly enhanced predictive self-healing. Ghahremani et al. [9] introduced a utility-driven self-healing framework for large-scale adaptive architectures through reinforcement learning (RL). In the cloud-edge continuum, Giannopoulos et al. [10], [11] proposed a distributed deep RL for delay-aware computation offloading, optimizing workload distribution between cloud and edge servers based on real-time traffic and latency constraints. Also, Schuler et al. [12] further explored RL-based auto-scaling in serverless computing, showcasing how AI-driven adaptive resource allocation outperforms traditional reactive methods in handling workload fluctuations. Finally, self-healing mechanisms have been considered for Mission-Critical Service (MCS) provisioning, where AI-assisted decisions have been used to preemptively scale up the computational resources of MCS in the presence of traffic bursts [13], [14].

D. Paper Outline and Contributions

This paper presents an end-to-end self-healing control loop combining real-time monitoring, AI inference, and Service Level Agreement (SLA)-driven resource orchestration. A comprehensive AI-driven framework for self-healing resource management is also proposed in the frame of Cloud-native 6G networks.

The key contributions of this work include:

- A closed-loop AI-driven control system for real-time server activation and deactivation is outlined, following a load-aware decision-making process. Specifically, we propose a general-purpose architectural workflow of AI-driven CCF optimization under SLA demands (policies) and dynamic compute traffic.
- Collaborative interactions among different 6G frameworks, including MOF, CCF, and AIMLF, are highlighted to enable self-configurable and automatic resource scaling.
- A comparative evaluation of static (or reactive), predictive (or proactive), and ML-aided adaptive threshold-based resource allocation strategies is presented. To this end, we conducted a quantitative and comparative evaluation of ML impact on the number of active servers and the system's energy efficiency.
- A proof-of-concept demonstration of significant power savings and optimized resource utilization through AI-enhanced decision-making is provided, considering real compute traffic patterns.

II. Architecture Model Overview

A. High-level 6G Architecture

The high-level architecture of the Cloud-native 6G (6G-Cloud) system integrates multiple frameworks and functional components to enable self-healing and energy-efficient NS manageability. The system is structured around three primary frameworks, namely the MOF, CCF and AIMLF, all of which interact seamlessly with the 6G Core and RAN NSs to ensure optimal performance. A single NS at the CCF level represents a logical grouping of resources, called Resource Partition (ResP), and functionalities designed to provide a specific network capability. These resources can span across multiple cloud/edge servers or data centers and are orchestrated dynamically based on demand, SLAs, and system requirements. An NS can consist of a set of virtualized network functions (VNFs) that run on one or many servers. AIMLF functionalities are considered as overlay functions extending the capabilities of the CCF or MOF modules. Their instantiation can be done after interaction with AIMLF Orchestrator.

The general architecture is depicted in Fig. 1, while the key functionality of the three frameworks is outlined as follows.

1) **Management and Orchestration Framework (MOF):** This framework handles policy-based orchestration and SLA enforcement. It consists of several subcomponents, such as the NS Directors, Service Orchestrator (SO), which manages a single NS, and the Master Service Orchestrator (MSO), which oversee the dynamic management of NSs and resource provisioning. It interacts with both AIMLF and CCF to coordinate resource scaling actions. User or system administrator requirements are collected through the Global Operational Support System (G-OSS) or Global Business Support System (G-BSS).
2) **Cloud Continuum (CCF):** This framework aggregates and manages all types of resources (far/near edge, central cloud) across a multi-provider environment, while it is responsible for real-time resource monitoring and allocation across cloud and edge servers. It includes Autonomic CCF Operations, a Resource Database (DB), and multiple Resource Orchestrators (ROs) that dynamically assign resources, potentially based on AI predictions.
3) **Artificial Intelligence and Machine Learning Framework (AIMLF):** This framework is responsible for

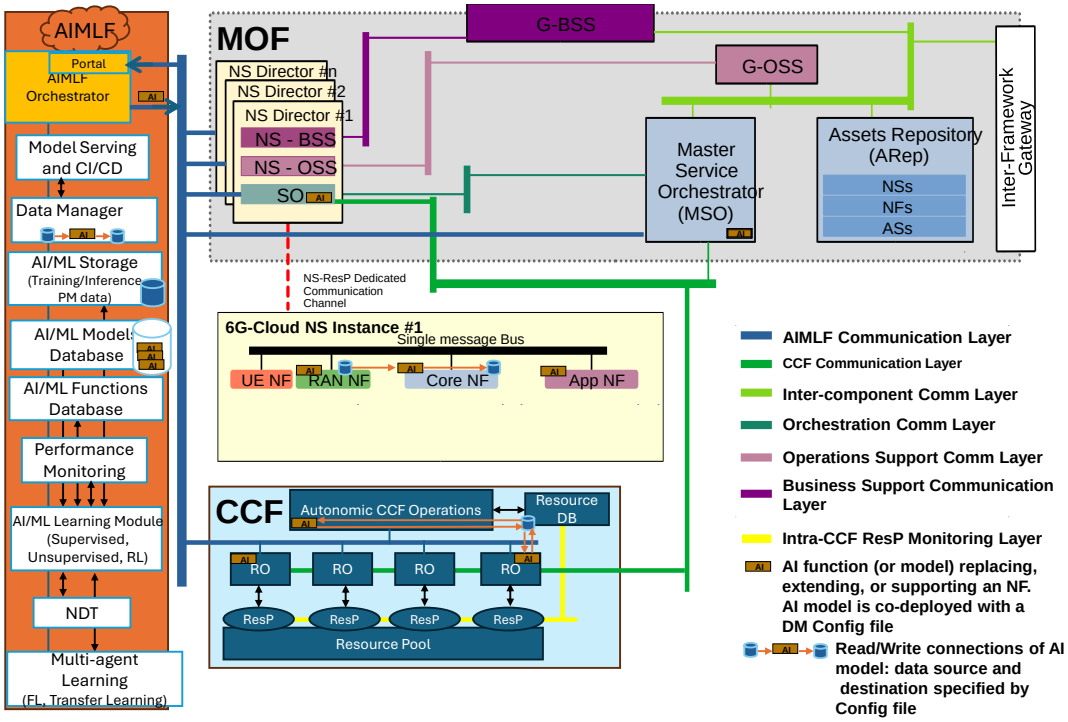

Fig. 1. High-level architecture of the multi-framework 6G-Cloud system.

AI-driven decision-making processes by providing AI functions (AIFs) and pipelines, including model training/inference, performance monitoring, and predictive analytics. It includes components such as the AI/ML Learning Module, Model Storage, Performance Monitoring, Network Digital Twin (NDT), and Model Serving to allow for deployment of AI workloads.

B. Role of Key Components

Although all the architectural components are required to form a complete 6G-Cloud system, this section outlines the role of key components engaged in the proposed control loop. These interconnected components form a closed-loop system where AI enhances adaptability, self-healing, and energy-efficient operations across the 6G-Cloud infrastructure.

First, in the CCF level, the critical components are: **(i) Resource Database** which collects real-time metrics, including energy consumption, resource utilization (e.g. CPU percentage utilization), and workload distribution from cloud/edge servers; **(ii) Resource Orchestrator** which allocates resources dynamically to NSs (e.g. server de-activation) and enforces reconfigurations based on AI model outputs; **(iii) Autonomic CCF Operations** which is considered to host the training or inference modules (e.g. CPU usage prediction models) using historical or real-time data from resource partitions. Regarding the AIMLF, the following key building blocks are identified: **(i) AI/ML Training Module** which trains or continuously refines prediction models using historic or new data from CCF Resource Database; **(ii) AI/ML Models Database** which stores trained models (e.g. CPU forecasting model) ready for deployment; **(iii) AI Performance Monitoring** which monitors the accuracy of models under training or the performance of running models. In the MOF level, key components include: **(i) Global OSS** which provides policy-based configurations such as service availability or energy efficiency, and interacts with external stakeholders; **(ii) Service Orchestrator (SO)** which orchestrates NS deployment and SLA policy enforcement by identifying threshold exceedance, and interfaces with the CCF to ensure resource availability; **(iv) Anomaly Detector** which resides inside SO and is responsible for detecting workload anomalies based on thresholds or predictive outputs from AI models.

III. AI-driven Dynamic Server Activation

In a nutshell, the proposed end-to-end resource allocation loop is illustrated in Fig. 2, where different coloring is used to notify the functionality per framework. Evidently, the process starts by monitoring several metrics from resource partitions. These VNF compute metrics (e.g., CPU utilization) exhibit temporal variations due to the dynamic nature of the underlying cellular traffic data. Then, a self-healing loop uses these data to initially train ML models based on historical datasets, whereas during the inference phase, real-time samples are fed in the trained ML model to predict future values of the servers' load. Then, according to SLA or policy-based thresholds set via the G-OSS, we perform prediction-based checks to detect potential violations of the existing SLA policies and identify impending anomalies in the NSs (e.g., the overall load of an NS exceeds a predefined limit). To ensure preemptive resource scaling (e.g., adding more computational power to a certain

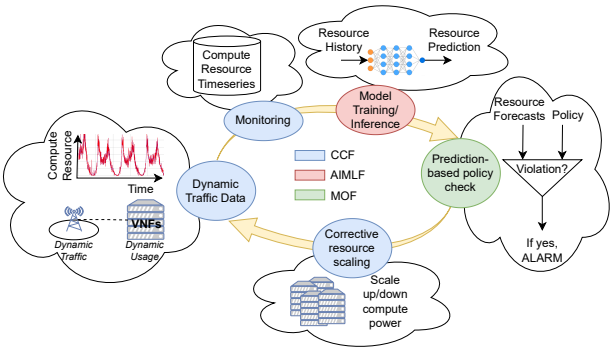

Fig. 2. End-to-end resource allocation loop for self-healing actions encompassing 6G-Cloud frameworks.

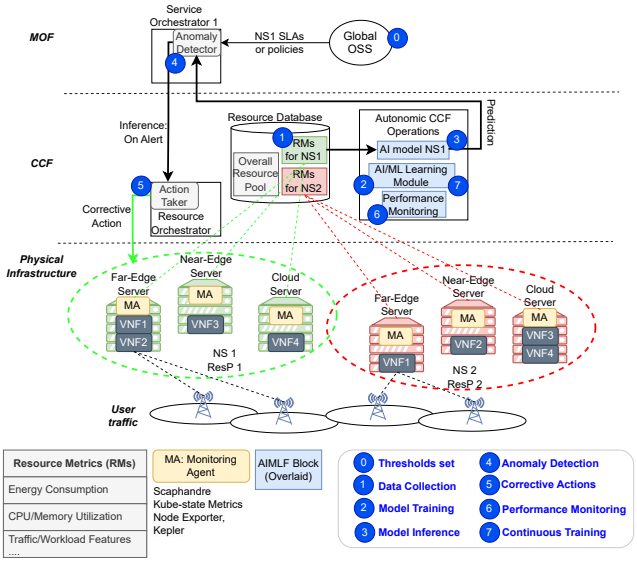

Fig. 3. Step-by-step procedure for dynamic server activation within 6G-Cloud system.

NS), corrective actions are then performed by the ROs to guarantee adaptation to the upcoming compute traffic demands dynamically.

A. Resource Allocation Workflow

To concretely describe the proposed scenario workflow, Fig. 3 shows the architecture system model, isolating the 6G-Cloud components of interest. The physical infrastructure layer includes the user devices served by radio units, and the edge/cloud servers that host the VNFs of an NS. The resources marked in green constitute a resource partition for NS 1, whereas other resource partitions for other NSs may also be considered and are shown in red. A monitoring agent (MA) software component is deployed in each edge/cloud server, collecting timeseries metrics in the form of Resource Metrics (RMs) report. The CCF layer manages the physical infrastructure through the ROs. The AIMLF blocks shown in blue are overlaid in the Autonomic CCF operations, which means that they are directly deployed in the CCF. Finally, the MOF layer includes the modules responsible for the NS orchestration, as well as the SLA policy enforcement. The latter includes the identification of server overload to notify the RO for further scaling corrections.

Considering a single NS and its associated ResP, the proposed resource allocation workflow is unfolded as follows:

Step 1: The first step is to collect the timeseries of the CPU utilization metric associated with the ResP of NS 1. The collected data are stored in the Resource Database of the CCF. To ensure consistency, the latter tags the time-series dataset with contextual information such as server type, and location.

Step 2: The timeseries historical dataset is utilized to train an ML model that, based on a history lookback window, provides forecasts of the total CPU utilization of NS 1. The training process is performed offline in the AI/ML learning module of the AIMLF that either is instantiated in the Autonomic CCF operations or is located in a cental server. During the training phase, the AI/ML learning module subscribes through the RO to the Resource Database of the CCF in order to gather the training data. This is achieved by obtaining the configuration parameters from the Data Manager that specifies the input/output of the AI/ML nodes. Then, the training pipeline commences in the AI/ML Learning module, including the exact labeling of the data, the feature extraction of the dataset, as well as the fine-tuning of the ML model parameters. The output of this process is the trained and fine-tuned ML model for CPU utilization forecasting, which is sent and stored to the AI/ML Models Database.

Step 3: The trained ML model is used for inference using real-time data from the Resource Database, acknowledging the predicted CPU utilization values in the MOF's SO.

Step 4: Predictions are analyzed by the MOF's Anomaly Detector component to identify resource partitions with potential CPU overload. Anomaly thresholds are defined by SLAs or policies provided by the Global OSS.

Step 5: If CPU overload or anomaly (e.g. SLA violation) is identified by SOs, then Action Taker performs corrective actions, including dynamic scaling. This means that resources are scaled up/down in NSs by adding a new server (i.e. Server Activation) or putting into low-power states (or deactivating) temporarily the underutilized servers (i.e. Server Deactivation).

Step 6: Input and output (i.e. model predictions) data are gathered by the Performance Monitoring (PMon) Client module inside Autonomic CCF Operations and are sent to the Performance Monitoring Server (inside AIMLF) together with the actually observed output values. PMon Server provides performance data (i.e. error between actual and predicted values) and initiates model retraining upon model performance degradation.

Step 7: Training data is continuously gathered from the Resource Database in the form of RMs reports, and, periodically, are fed back into the AI/ML Learning Module. At

predefined regular periods, models are retrained to adapt to changing workload patterns and environmental conditions. In this sense, AIMLF ensures that the AI models remain accurate and effective over time by saving their most recently trained versions in the AIMLF Models Database.

This seven-step workflow is continuously repeated during the real-time network operation, ensuring flexible, proactive, and up-to-date scaling corrections based on AI-assisted decisions.

B. Conditional Anomaly Detection Methods

In this section, we present different methods considered for anomaly detection, as part of Step 4 in Fig. 3. To trigger a corrective action, we employ three different approaches, each with increasing sophistication: **(i) Reactive (Without ML):** This is a straightforward method where a new server is activated once CPU utilization reaches a fixed threshold. However, it lacks adaptability to dynamic traffic patterns. Assuming a fixed load threshold C, a new server is added at time $t+1$ when $L(t) > C$, where $L(t)$ is the CPU utilization at time slot t; **(ii) Proactive (Fixed Threshold):** This method still uses a predefined threshold but integrates AI predictions through an LSTM model, offering better anticipation of CPU usage spikes. This is done by comparing the load threshold against the model predictions. Formally, a new server is added at time $t+1$ when $L(t) \leq C$ and $\hat{L}(t) > C$, where $\hat{L}(t)$ is the CPU utilization prediction provided at time slot t concerning the upcoming CPU at time slot $t+1$; **(iii) Proactive (Adaptive Threshold):** This method dynamically adjusts the anomaly threshold based on a moving average of predicted CPU usage. The adaptive threshold is updated at each time step as follows. The threshold $T(t)$ is computed using an exponential moving average $T(t) = 0.9 \times T(t-1) + 0.1 \times \hat{L}(t)$, where $\hat{L}(t)$ represents the current predicted load. A new server is activated when $\hat{L}(t) > T(t)$, and deactivated when $\hat{L}(t) < T(t)$.

IV. Numerical Results

In this section, we numerically evaluate the proposed resource allocation loop against three different methods for anomaly detection, as presented in Section III-B. We employ dynamic server (de-)activation under fluctuating compute traffic at the NS servers. The AI models are trained using real-world traffic flow datasets from Torino city. These datasets capture variations in network traffic, enabling our system to learn effective scaling policies. The dataset consists of traffic flow measurements aggregated over 5-minute intervals from various streets in Torino, spanning several months. These traffic flows generate tasks that require CPU resources (measured as a percentage of usage). In our experiments, we use six-months of data from Corso Agnelli, and one-week of data for testing. Note that this dataset has been extensively used in previous works [15].

Regarding the load thresholds, we use a subset of values defined in [16]. Specifically, we adopt a conservative approach by activating a new server when the current traffic load exceeds 20% of the capacity of the server.

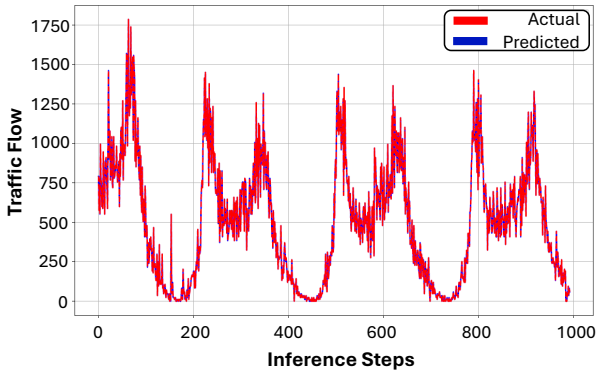

Fig. 4. Actual versus predicted curves of CPU usage considering 1000 inference steps. MSE between red and dashed blue line is below 10^{-3}.

Upon extensive simulations with different hyperparameters, we select a Long Short Term Memory Network (LSTM) as the best CPU forecasting model. The optimal hyperparameter configuration of the LSTM was the following: batch size = 16 (number of samples that are fed at each training step), lookback window = 16 (number of history samples used as input), training epochs = 100 (number of times that the whole dataset was fed for training), learning rate = 0.0001 (affects the backpropagation), hidden neurons per layer = 100, deepness = 5 (number of hidden layers). The mean squared error (MSE) was also used as the loss function, whereas at time instance t, the prediction refers to the CPU utilization at $t+5$ minutes. To illustrate the effectiveness of the trained LSTM in predicting the traffic patterns, Fig. 4 shows the actual and predicted curves considering 1000 inference steps from the testing set. Evidently, the LSTM is capable of accurately predict the actual CPU usage values (MSE below 10^{-3}).

Fig. 5 illustrates the number of active servers over 1000 inference steps for the three anomaly detection methods. In the reactive and the proactive approaches with fixed threshold, we observe similar behavior. This is because the predictions closely align with the actual data, and both approaches employ the same fixed threshold. In contrast, the proactive approach with adaptive threshold results in a significant reduction in the number of active servers. This occurs because the threshold dynamically adjusts based on the current traffic pattern, allowing for a more flexible decision-making process regarding server activation and deactivation, while preventing abrupt changes.

Regarding the power consumption model, we assume that an active server consumes a fixed amount of power, meaning that each server activation incurs a specific power cost. In our experiments, we base our power consumption model on real-world measurements from a Power Consumption Database (see https://www.tpcdb.com/), considering Intel NUC6i7KYK servers. Fig. 6 illustrates the mean power consumption for the three approaches. As expected, the proactive approach with adaptive threshold achieves significant power savings, as it

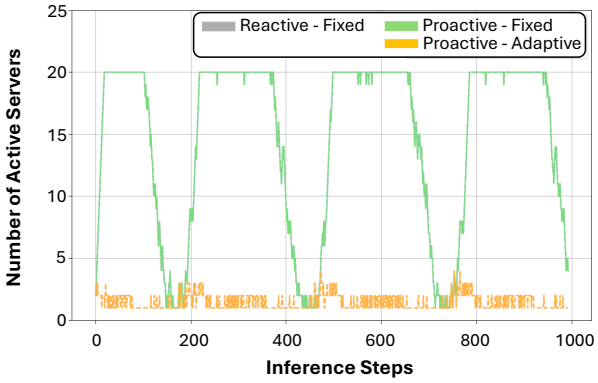

Fig. 5. Number of active servers resulted from three anomaly detection methods in 1000 inference steps. Grey and green lines overlap.

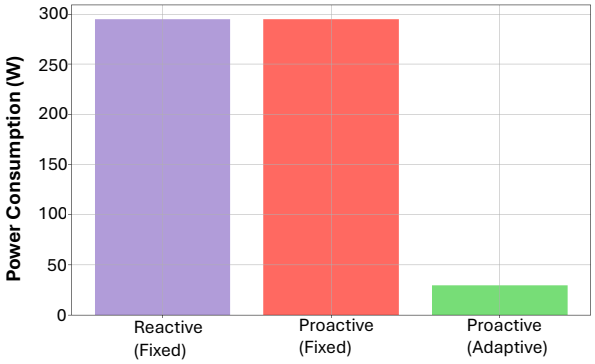

Fig. 6. Power consumption resulted from three anomaly detection methods in 1000 inference steps.

maintains a lower number of active servers. This demonstrates that dynamically adjusting the threshold can lead to significant reductions in power consumption.

V. Conclusion and Future Work

This paper presents an AI-driven self-healing framework for energy-efficient dynamic server activation in 6G-Cloud networks. The approach integrates real-time monitoring, AI-driven workload forecasting, and adaptive anomaly detection to optimize cloud/edge resource allocation.

Future directions of the present work include:

1) **Large-Scale Implementation and Real-World Validation:** Deploying the proposed AI-driven self-healing CCF system in large-scale real-world network environments. This will include testing in multi-operator settings to evaluate its robustness under varying network conditions.
2) **Advanced Learning and Optimization Techniques:** Implementing other AI techniques to enhance the decision-making abilities. Federated Learning (FL) and Transfer Learning (TL) can be studied to enable decentralized training across multiple network environments.
3) **Considering Energy Consumption:** Instead of collecting CPU usage metrics, other indices of compute load such as energy consumption can be also evaluated.

Acknowledgment

The authors warmly thank the partners of 6G-Cloud project for their contribution in the architectural aspects of this article.

References

[1] Q. Li, Z. Ding, X. Tong, G. Wu, S. Stojanovski, T. Luetzenkirchen, A. Kolekar, S. Bangolae, and S. Palat, "6g cloud-native system: Vision, challenges, architecture framework and enabling technologies," *IEEE Access*, vol. 10, pp. 96 602–96 625, 2022.

[2] P. Soto, M. Camelo, G. García-Avilés, E. Municio, M. Gramaglia, E. Kosmatos, N. Slamnik-Kriještorac, D. De Vleeschauwer, A. Bazco-Nogueras, L. Fuentes *et al.*, "Designing the network intelligence stratum for 6g networks," *Computer Networks*, vol. 254, p. 110780, 2024.

[3] N. Nomikos, G. Xylouris, G. Patsourakis, V. Nikolakakis, A. Giannopoulos, C. Mandilaris, P. Gkonis, C. Skianis, and P. Trakadas, "A distributed trustable framework for ai-aided anomaly detection," *Electronics*, vol. 14, no. 3, p. 410, 2025.

[4] Y. Dai, Y. Xiang, and G. Zhang, "Self-healing and hybrid diagnosis in cloud computing," in *IEEE International Conference on Cloud Computing*. Springer, 2009, pp. 45–56.

[5] S. R. Rouholamini, M. Mirabi, R. Farazkish, and A. Sahafi, "Proactive self-healing techniques for cloud computing: A systematic review," *Concurrency and Computation: Practice and Experience*, vol. 36, no. 24, p. e8246, 2024.

[6] K. Lee, H. Lee, and D.-H. Cho, "Collaborative resource allocation for self-healing in self-organizing networks," in *2011 IEEE International Conference on Communications (ICC)*. IEEE, 2011, pp. 1–5.

[7] M. Mashaly and P. J. Kühn, "Load balancing in cloud-based content delivery networks using adaptive server activation/deactivation," in *2012 International Conference on Engineering and Technology (ICET)*. IEEE, 2012, pp. 1–6.

[8] P. J. Kuehn and M. E. Mashaly, "Automatic energy efficiency management of data center resources by load-dependent server activation and sleep modes," *Ad Hoc Networks*, vol. 25, pp. 497–504, 2015.

[9] S. Ghahremani, H. Giese, and T. Vogel, "Improving scalability and reward of utility-driven self-healing for large dynamic architectures," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 14, no. 3, pp. 1–41, 2020.

[10] A. Giannopoulos, I. Paralikas, S. Spantideas, and P. Trakadas, "Cooler: Cooperative computation offloading in edge-cloud continuum under latency constraints via multi-agent deep reinforcement learning," in *2024 International Conference on Intelligent Computing, Communication, Networking and Services (ICCNS)*. IEEE, 2024, pp. 9–16.

[11] A. Giannopoulos, I. Paralikas, S. Spantideas, and P. Trakadas, "Hoodie: Hybrid computation offloading via distributed deep reinforcement learning in delay-aware cloud-edge continuum," *IEEE Open Journal of the Communications Society*, 2024.

[12] L. Schuler, S. Jamil, and N. Kühl, "Ai-based resource allocation: Reinforcement learning for adaptive auto-scaling in serverless environments," in *2021 IEEE/ACM 21st international symposium on cluster, cloud and internet computing (CCGrid)*. IEEE, 2021, pp. 804–811.

[13] S. Spantideas, A. Giannopoulos, M. A. Cambeiro, O. Trullols-Cruces, E. Atxutegi, and P. Trakadas, "Intelligent mission critical services over beyond 5g networks: Control loop and proactive overload detection," in *2023 International Conference on Smart Applications, Communications and Networking (SmartNets)*. IEEE, 2023, pp. 1–6.

[14] S. T. Spantideas, A. E. Giannopoulos, and P. Trakadas, "Smart mission critical service management: Architecture, deployment options, and experimental results," *IEEE Transactions on Network and Service Management*, 2024.

[15] J. Martín-Pérez, K. Kondepu, D. De Vleeschauwer, V. Reddy, C. Guimaraes, A. Sgambelluri, L. Valcarenghi, C. Papagianni, and C. J. Bernardos, "Dimensioning v2n services in 5g networks through forecast-based scaling," *IEEE access*, vol. 10, pp. 9587–9602, 2022.

[16] J. Ortin, P. Serrano, J. Garcia-Reinoso, and A. Banchs, "Analysis of scaling policies for nfv providing 5g/6g reliability levels with fallible servers," *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 1287–1305, 2022.