Dynamic Fine-grained SLA Management for 6G eMBB-plus Slice using mDNN & Smart Contracts

Sadaf Bukhari, Kashif Sharif, *Senior Member, IEEE,* Liehuang Zhu, *Senior Member, IEEE,* Chang Xu, Fan Li, *Member, IEEE*, and Sujit Biswas, *Member, IEEE*

***Abstract*—The advent of 6G networks promises revolutionary advances in dynamism, intelligence, and decentralization. Realizing the full potential of 6G requires adaptable service level agreements (SLAs) that can optimize performance based on dynamic network conditions. In this paper, we suggested a method based on the Hyperledger Sawtooth blockchain's smart contract with the Reptile meta-learning algorithm to solve the rigidity of static SLA and centralization problems. In order to sustain the quality of service in the radio access network and core network domain of 6G networks, this work focuses on SLA management for efficient resource allocation for the eMBB-plus slice. Our approach entails breaking down static SLAs into finer-grained components, transferring those components onto Hyperledger Sawtooth smart contracts, and using the Reptile meta-learning algorithm to forecast SLA metrics and resource requirements. A dynamic tariff model, also proposed within the smart contract, handles increased user demands. We evaluate the solution by analyzing Reptile performance, resource allocation, and SLA violations under dynamic demands. Results demonstrate the efficiency of this AI-driven, blockchain-based approach for automated, optimized 6G eMBB-plus resource management adhering to dynamic fine-grained SLAs. This work highlights the synergistic potential of AI and blockchain for trusted and intelligent 6G service delivery.**

***Index Terms*—SLA Management, Blockchain, Smart Contracts, eMBB-Plus, Network Slicing, 6G.**

I. Introduction

With the unprecedented demand for faster, more reliable, and efficient communication services, the revolutionary vision of 6G networks is set to deliver a paradigm shift in the way we communicate, meeting the ambitious demands of the 2030s and beyond. The race towards developing 6G networks has captured the attention of academia, standardization bodies, governments, and industrial organizations since 2018 [1]. Although 5G networks deliver substantial capabilities, they have shortcomings in massive machine connectivity, scalability, real-time latency, throughput, security, availability, access control, and bandwidth efficiency [2]. Overcoming these limitations requires harnessing emerging technologies like Blockchain (BC), Edge Computing (EC), Network Slicing (NS), and Artificial Intelligence (AI) to unlock the full potential of 6G for hyper-connected intelligence.

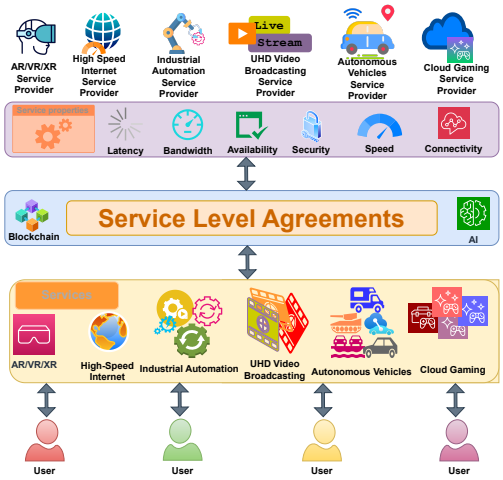


Figure 1: Working, communication, & interaction framework of a 6G service-oriented system, augmented with the proposed solution framework.

The main objective of 5G-A & 6G networks is to offer improved worldwide coverage, highly reliable and secure communication, faster data rate transmission, enhanced privacy, and extremely low latency. The potential applications for 6G networks are eXtended/Mixed Reality (XR, MR), autonomous systems, wireless brain-computer interaction, smart cities, artificial technologies, holographic and human-centered communication, indoor positioning, and vertical industry-supported communications. To fulfill the requirements of these applications, five services have been identified as the upgraded versions of 5G network services: enhanced Mobile Broadband-plus (eMBB-plus), Secure Ultra-Reliable Low Latency Communication (SURLLC), Three-Dimensional Integrated Communication (3D-Intecom), Unconventional Data Communication (UCDC), and Big Communication (BigCom) [3].

Service Level Agreements (SLAs) define the negotiated terms of service and agreements between service providers, consumers, and infrastructure entities. They outline different performance metrics, quality guarantees, and other technical/legal aspects. Traditional SLAs are static, lacking adaptability to changing user needs and services, which is inadequate for dynamic service provisioning. These challenges highlight the need for more flexible and secure SLA frameworks in next-generation services for 6G systems. This paper

The work is in part supported by Beijing Natural Science Foundation (IS23056) and National Natural Science Foundation of China No. 62232002. S. Bukhari is the primary contributing student. Co-corresponding authors: K. Sharif & L. Zhu.

S. Bukhari, K. Sharif, and F. Li are with School of Computer Science and Technology, Beijing Institute of Technology, Beijing, China (email: sadaf@bit.edu.cn; kashif@bit.edu.cn; fli@bit.edu.cn).

L. Zhu and C. Xu are with the School of Cyberspace Science and Technology, Beijing Institute of Technology, Beijing, China (email: liehuangz@bit.edu.cn; xuchang@bit.edu.cn).

S. Biswas is with the School of Science and Technology, CITY, University of London, London, United Kingdom (email: sujit.biswas@city.ac.uk).

focuses on eMBB-plus slices, the successor to 5G eMBB. eMBB-plus optimizes networks for interference, handover, big data transmission, and processing. It enables highly immersive extended reality applications with capabilities like ultra-high peak data rates, reduced latency, increased capacity, and throughput [4]. eMBB-plus guarantees seamless user experiences across diverse mobility scenarios, including remote, commercial airplanes, unmanned aerial vehicles (UAVs), and ship environments [5]. In this work, we advocate that 6G networks should leverage blockchain for security and decentralization, while AI/ML for design and optimization [6]. Figure 1 illustrates the overall working and communication framework, including SLAs with blockchain and Machine Learning (ML) technologies. SLAs form an essential bridge between heterogeneous users requesting diverse eMBB-plus services from multiple Service Providers (SPs).

Integrated Technologies: Blockchain enables immutable distributed record-keeping using cryptographically chained blocks of replicated transactions. These transactions are validated and added to the ledger by decentralized peer nodes based on the network type [7]. An essential BC component is the smart contract (SC), which automates the verification and execution of transactions on the chain securely and efficiently. These self-executing scripts have the agreement terms directly coded into them. Once deployed on the blockchain, SCs can autonomously validate transactions as per the encoded business logic and trigger any resulting actions [8]. The immutable BC combined with tamper-proof SCs enables reliable and transparent automation for processes like SLA enforcement. Blockchain solutions like Hyperledger Sawtooth can provide customized modular architecture supporting various consensus protocols and facilitate SLA automation using SCs [9], [10]. 5G has seen limited integration of ML, while 6G is planned to fully utilize AI, particularly meta-learning [11]. It enables gathering experiential meta-data to solve related problems with minimal training data [12]. Meta-learning improves deep neural network performance with small datasets, overcoming deep learning's requirement for large data, which can otherwise lead to inaccurate outputs. Along with BC and AI, network slicing enables diverse new 6G services and Quality of Service (QoS) levels [13], [14]. It involves creating isolated virtual networks over shared infrastructure, with each slice governed by an SLA spanning radio access, transport, and core domains. Slices have customized resource and Key Performance Indicator (KPI) requirements per service type [15].

SLA Limitations: QoS guarantees are ensured through the SLAs. Traditionally, SLAs are inflexible for real-time environments. As technology has advanced, SLAs have transformed from static to dynamic structures. Dynamic SLAs exhibit adaptability in response to changing demands [16], [17]. However, managing dynamic SLAs and Service Level Objectives (SLOs) for effective resource allocation is difficult. Dynamic resource allocation needs adaptable SLAs with adjustable conditions and prices [18]. In order to manage sliced 6G networks effectively, dynamic fine-grained SLAs are to be prioritized. The term '*fine-grained*' denotes a meticulous approach for monitoring and maintaining service levels [6]. However, centralized SLA management systems often encounter trust-related challenges. Therefore, BCs and SCs emerge as a solution to effectively manage SLAs and make it easier to distribute resources in a cost-effective and decentralized manner. Coupled with deep and meta-learning for optimization, it will further improve the Quality of Experience (QoE) significantly [19]–[22].

Problem & Solution: Technically, in this paper, we address the problems related to the management of eMBB-plus slices SLA in Radio Access Network (RAN) and Core Network (CN) dynamic environments, i.e., dynamic SLA management, consistent rates of services, trust, and resource allocation. In order to address these issues, our paper proposes a dynamic fine-grained SLA for the slices of the eMBB-plus service for the 6G RAN and CN, as well as its administration through the Reptile meta-learning algorithm and SCs of Sawtooth blockchain. To the best of our knowledge, there is no prior work on maintaining a dynamic fine-grained SLA for a particular eMBB-plus slice for the RAN and CN domain and how to map that SLA slice to SCs using the Sawtooth blockchain. Existing works in [15], [18], [23]–[32] try to address these challenges from different aspects, but none of them gives a complete and comprehensive solution. Our methodology involves three main tasks for each domain: SLO target prediction, optimal resource prediction, and predicted resource allocation. The other functions, i.e., monitoring system, dynamic tariff model, resource allocation & execution sub-system, and payment & penalty functions, are also implemented as part of the overall solution. We evaluate our methodology based on the performance of the training model, resource allocation function, SLA violations, and BC network. Our approach shows promising results for effectively managing SLAs in a dynamic and fine-grained manner and adapting to changes in network conditions to meet SLOs.

Contributions: This paper makes several notable contributions that advance the state-of-the-art in dynamic fine-grained SLA management for 6G networks.

- First, this paper presents a novel approach for breaking down conventionally inflexible, static SLAs into finer-grained, more flexible, and adjustable components. This involves clearly defining the conditions, performance objectives, and service metrics that make up a digitized SLA.
- Second, we segment the dynamic fine-grained SLA for the eMBB-plus slice into two domains, i.e., the RAN domain and the CN domain. Further, we categorized each domain with respect to resource demands and performance metrics.
- Third, we select peak data rate and latency for RAN domain SLA, and peak bandwidth and availability for CN domain SLA. We then developed fine-grained SLA components to the needs of each domain & resource.
- Fourth, leveraging the Reptile-based meta-learning algorithm, we enable intelligent prediction of domain metrics and resources, allowing for proactive SLA optimization in both the RAN and CN domains.
- Fifth, we use SCs as digitized, self-executing, and monitoring SLAs on Hyperledger Sawtooth for the predicted metrics and the resource allocation process. The SCs also

contain proposed payment and penalty functions and a dynamic tariff model.
- Finally, we implement a comprehensive prototype testbed to thoroughly evaluate various aspects of the proposed system. The detailed results demonstrate that this approach significantly enhances service management efficiency and effectiveness.

The rest of the paper is structured as follows: In Section II, we discuss related works to provide a context for our research. Section III describes the problem statement and outlines our proposed dynamic fine-grained SLA management solution. Section IV provides a detailed explanation of the fine-grained SLA components. Section V delves into the decomposition of the SLA for resource allocation. In Section VI, we explain the mapping process for Smart Contracts (SCs). Section VII presents a comprehensive evaluation of our proposed solution. Finally, in Section VIII, we conclude the paper and suggest directions for future research.

II. Related works

AI and blockchain technologies will serve as vital enablers for realizing the envisioned capabilities of 6G networks. However, a literature review reveals a significant gap in research *intersecting* AI, blockchain, and mobile computing for dynamic fine-grained SLA management and resource allocation. Much of the current work focuses *singularly* on either blockchain, deep neural networks, or 5G network slicing. Hence, exploring the integration of AI and blockchain is essential to address the challenges of adaptive SLA management and resource provisioning for emerging 6G services like eMBB-plus slicing. Below, we have grouped the literature into related subsections, while a summary of comparative analysis of the most relevant existing works and the proposed work is listed in Table I.

A. SLA Management & Resource Allocation using Blockchain

Blockchain and smart contracts create a seamless infrastructure for reliable record-keeping and automated transactions. With its decentralized network of nodes in a peer-to-peer structure, blockchain ensures that data is unchangeable and universally distributed [33]. On the other hand, SCs streamline transactions by encoding agreement terms into self-executing code. Once activated, these contracts independently verify and initiate actions based on predefined logic [34].

Blockchain technology provides a decentralized and secure approach to managing SLAs, enhancing trust and reliability in the management process. The work in [23] discusses similar issues as targeted in this paper, i.e., centralized SLA management. However, their problem was presented in the context of a dynamic cloud environment without AI/ML and resource allocation. Contrary to this, our research focuses on the issues by managing dynamic fine-grained SLA through blockchain for a network slice, specifically in 6G networks. The work in [24] has contributed by introducing an SC-based trusted payment system customized for dynamic SLAs. [26] emphasized the necessity of SLAs in 6G networks regarding infrastructure and resource sharing, and proposed a blockchain-rooted framework that enables network sharing while ensuring accountability and transparency using SCs. However, the framework is very abstract with over reliance on orchestrator.

SCs can be used across a spectrum of application frameworks. The work in [25] offered a blockchain-based solution for fog computing environments that enables autonomous SLA management, using a blockchain-driven SLA registry and an SC-based SLA enforcement mechanism. It uses SCs to automates compensation processing against SLO violations [35]. The work in [27] also manages SLA through BCs. It uses Ethereum SCs to handle the billing, penalties, and compensation process. However, in both the above solutions the SLA-SC mapping is hard coded manually, which is then enforced. In contrast, our solution is more comprehensive as described earlier.

In 6G networks, SLAs with static pricing models are becoming insufficient due to their rigid and pre-determined rates that do not cater to changing user demands, network conditions, or resource requirements. Dynamic pricing models are surveyed in [18], such as usage-based, congestion-based, and auction-based models. This article concludes that the dynamic pricing models can help improve SP's profitability by adjusting service priority based on subscribers' changing networking requirements. 6G networks offer a plethora of services, each with varying QoS requirements, KPIs, and resource needs. Network slicing has emerged as a crucial feature in 6G networks to efficiently cater to these diverse service demands. [15] reviews requirements and challenges related to network slicing. It concludes that network slicing enables dynamic resource allocation that adapts to changing user demands, tenant requirements, and network conditions. Both of the above articles have provided the technical argument that the dynamic pricing and network slicing must be addressed in the proposed solution of our work.

SCs can ensure that resources are distributed and utilized according to predefined rules and conditions, which can help prevent fraudulent behavior & over/under-utilization of resources. A blockchain-based mechanism is proposed in [28] for exchanging RAN resources (infrastructure or spectrum) for beyond 5G networks. It utilized SCs to facilitate flexible and dynamic resource-sharing between SP and users. [29] presented a system for allocating spectrum for 6G communication networks based on blockchain. It only proposes a plan that uses BC to create solutions based on SCs that are secure & transparent and enhance resource distribution. Moreover, the work in [30] proposed a blockchain-powered fog radio access network (BF-RAN) design, which utilizes blockchain technology to improve resource allocation and provide secure and transparent SC-based solutions. All these works establish that SCs can be effectively used to manage network resources.

B. Resource allocation using Blockchain & Machine Learning

Integrating blockchain technology and AI in 6G network slicing can provide a comprehensive resource allocation and usage solution for SPs and users. Additionally, AI can be employed to predict traffic patterns, which is immensely beneficial in dynamic real-time environments [36]. Work in [14] proposed an AI-driven approach for intelligent network

Table I: Comparative analysis of related works and the proposed solution.

Ref.	Contributions	AI/ML	BC	SLA	Resource Allocation	Implementation/Evaluation Mechanics
[15] ‡∗	Network slicing requirements and challenges	✓	✓	✓	✓	None
[18] ‡∗	Classifies dynamic pricing strategies and examines price modification in response to demand, resource utilization, & current circumstances	✗	✗	✓	✓	Theoretical evaluation on static and dynamic pricing schemes
[23] †	Framework for dynamic SLA mgt. in cloud SC & BC for dist. SLA adaptation	✗	✓	✓	✗	Only a proposal with no evaluation
[24] †	BC and SC for dynamic payment automation in SLA lifetime	✗	✓	✓	✗	Synthetic usecase & minor evaluation of QoS SLA (client-SP) usecase
[25] ‡	BC based structure for SLA fulfillment in fog computing	✗	✓	✓	✗	Ethereum based evaluation. Three use cases for measuring sub-transactions, SLA registration, execution, & detection gas, and miner fee
[26] †	BC based architecture (BEAT) for transparent & accountable network sharing	✗	✓	✓	✓	Simulated DL node & packet processor for SLA execution time & CPU utilization
[27] †	Trust architecture for SLA mgt. using automated penalties & compensations	✗	✓	✓	✗	Evaluated for uRLLC & eMBB slices w.r.t latency & throughput
[28] †	BC enabled reverse auction mechanism for RAN sharing in beyond 5G networks	✗	✓	✓	✓	Evaluated BC performance, user's performance, and behavior of traditional & virtual MNOs
[29] †	BC Integration in 6G spectrum for secure & transparent band allocation among telecom providers	✗	✓	✗	✓	Evaluated for static/dynamic resource block allocation, trust probability, & overhead on base station
[30] ‡	Secure and trustless BC based architecture for resource allocation in fog RANs	✗	✓	✗	✓	Architecture only with no evaluation
[31] ‡	BC & AI powered DRS architecture for dynamic spectrum sharing with DRL	✓	✓	✗	✓	Evaluated for user profit ratio, facilitation of acquiring training data, BC overhead
[32] †	BC used for resource allocation & double spending prevention ML for QoS prediction and SLA negotiation	✓	✓	✓	✓	Evaluated four use cases for double spending of resources and attacks
This Work	·Comprehensive solution for SLA management of eMBB-plus slice for 6G networks. ·Decomposition of static SLA into fine-grained components. ·Dynamic fine-grained SLA for RAN & CN domains w.r.t performance metrics & resources. ·Dynamic fine-grained SLA component mapping to Hyperledger Sawtooth SC. ·SC modeling for predicted performance metrics & resource allocation functions, monitoring of SLOs, dynamic tariff model, & payment/penalty functions. ·Domain metric/resource prediction allocation in eMBB-plus slice.	✓	✓	✓	✓	·Performance evaluation using reptile based meta learning algorithm. ·Performance analysis of resource allocation w.r.t SLA violation. ·Performance analysis of blockchain network.

BC: Blockchain, **SC**: Smart Contract, **DL**: Distributed Ledger, **SLA**: Service Level Agreement, **QoS**: Quality of Service, **SP**: Service Provider, **DRL**: Deep Reinforcement Learning, †: Conference/Magazine article, ‡: Journal article, ∗: Survey/analysis article.

slicing in 6G systems. It uses an AI slicer architecture with reinforcement learning and knowledge graphs to dynamically optimize slices based on diverse QoS metrics. AI can also enhance the decision-making powers of SCs; hence, in [31], a framework employing blockchain and reinforcement learning is proposed for dynamic resource sharing in 6G networks and beyond. Machine learning coupled with big data can provide robust support for enhancing blockchain technology in areas such as prediction modeling, network stability, and operational maintenance, as explored in [37], [38]. In the context of wireless environments, the synergy of blockchain technology and machine learning is examined in [32], for thwarting double spending of resources. This work makes a strong case for use of ML in resource management with SLA guidance. Finally, deep learning algorithms hold significant promise in tackling challenges 6G networks, including resource optimization, adaptability to dynamic environmental conditions, and the establishment of intelligent network infrastructures [39].

III. Problem Statement and Solution Architecture

A. Problem Identification & Challenges

An SLA represents a contract between service providers (SP) and customers, defining the delivery terms of services. Conventionally, SLAs have been static, with fixed terms established at the outset for the duration of the agreement. These static SLAs outline service metrics, quality guarantees, and legal and financial details upfront, locking the service into these predefined specifications regardless of any changes that may occur during the contract period. This rigidity renders static SLAs inadequate for the dynamic requirements of next-generation service environments, where adaptability and responsiveness are essential [40], [41]. Another major challenge with traditional SLAs is trust, due to reliance on third-party auditing and centralized management. SLAs typically reside on centralized servers controlled by the provider, raising concerns about reliability and security. This centralization exposes agreements to risks like malicious attacks, single points of failure, and potential data tampering by the provider, thereby eroding trust.

These challenges are significantly amplified in mobile communication, where SLAs between clients (mobile devices) and service providers (network carriers) are highly dynamic. The sheer volume of users and the diversity of their requirements introduce two critical issues: the need for dynamic SLAs and optimizing resources to meet them. This requires breaking down traditional, static SLAs into more granular and adjustable components capable of responding to varying per-

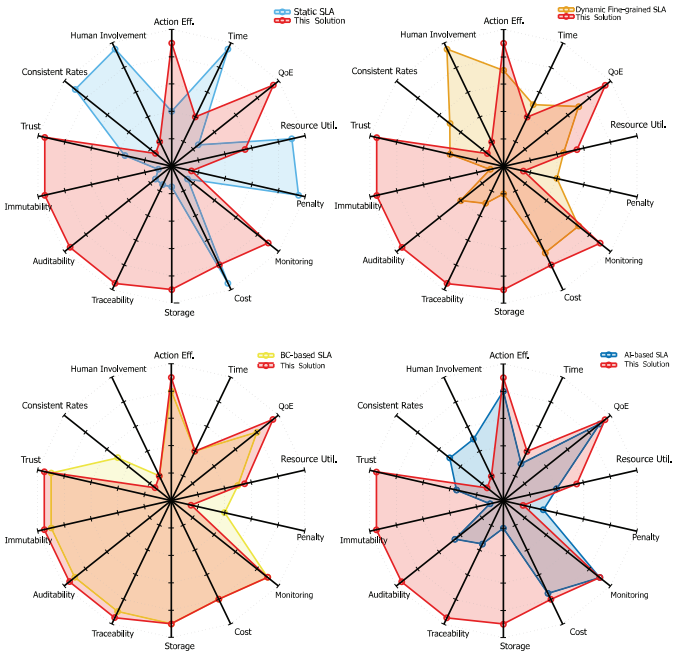


Figure 2: Relative comparison of different SLAs for KPI impact. Except *storage* (which is centralized/decentralized), all KPIs indicate relative low/high impact.

formance metrics such as latency, bandwidth, and data rates. Furthermore, this involves real-time prediction and allocation of resources across the RAN and CN domains, ensuring that service levels are consistently met end-to-end.

Figure 2 presents an analytical comparison of various SLA implementation technologies versus our proposed solution, using KPIs such as human involvement, efficiency, time, QoE, resource usage, penalties, monitoring, cost, storage, traceability, auditability, immutability, and trust. The comparison is relative, highlighting the impact on KPIs without exact scales, as quantitative values vary based on factors like organizational scale, service parameters, technical solutions, and system specifications. Most KPIs show a low-high impact scale, except for *storage*, which is centralized or decentralized. Dynamic fine-grained SLAs reduce costs, time, and penalties while improving consistency compared to static SLAs. Blockchain-based SLAs enhance trust, immutability, auditability, traceability, automation, and decentralized storage. AI-powered SLAs boost QoE, resource utilization, monitoring, and efficiency while reducing penalties. Static SLAs require significantly more human intervention and are more expensive and less efficient than advanced SLAs. This comparison underscores the advantages of moving from rigid, manual static SLAs to automated, optimized SLAs using blockchain and AI, as demonstrated by our proposed solution.

B. Multi-Tier Solution

This paper proposes a dynamic fine-grained SLA approach implemented via SCs and powered by AI techniques to address the challenges of static SLAs and centralized SLA management. Specifically, we employ fine-grained SLAs optimized for the eMBB-plus service across RAN and CN domains. The overall process can be broadly divided into two stages. Firstly, we meticulously decompose the SLA into granular components, including service type, network domain, dynamic performance metrics, and resource allocation. Within this stage, we achieve two critical tasks: *a)* Real-time prediction of performance metric targets based on network conditions, and *b)* Optimal resource allocation aligned with these predicted targets. Leveraging the meta-deep neural network (mDNN) Reptile algorithm enables dynamic forecasting, enhancing adaptability and responsiveness. Secondly, we seamlessly integrate these fine-grained SLAs with SCs to facilitate autonomous blockchain-based management. This approach ensures efficient and secure execution of SLA terms transparently.

The system architecture and the end-to-end workflow are shown in Figure 3. The initial parts of the solution (as described above) are encapsulated in the SLA-SC Orchestrator, whose primary roles include user/system interaction and integration. The orchestrator also implements various SLA components, policies, and business logic. Fully automating an SLA-based system is complex; therefore, this work focuses on automating key aspects: fine-grained decomposition, dynamic tariff model implementation, Reptile training, resource allocation and execution, and Blockchain interaction APIs. However, the architecture of the complete system is designed and shown in Figure 3. The Automated Information Extractor

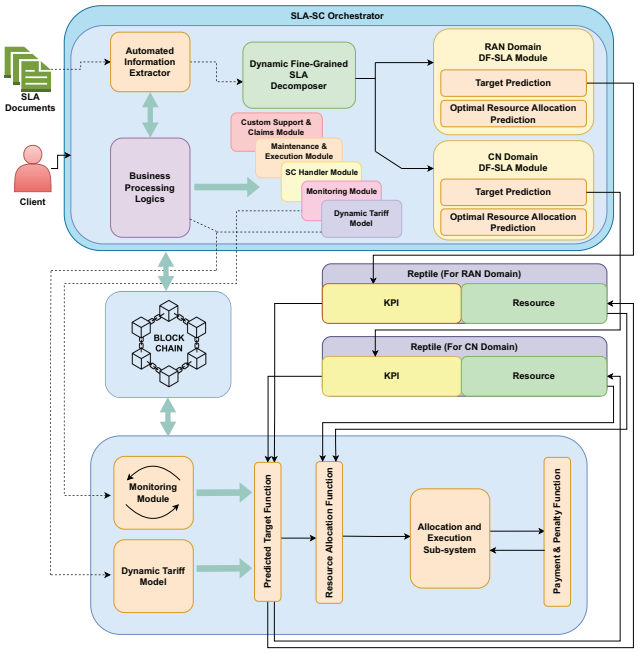


Figure 3: End-to-end system design and working architecture for fine-grained SLA management.

is a sub-system that uses NLP and SC templates for various commercial BC solutions to convert traditional paper-based SLAs into ready-to-use SCs[2]. Finally, in the implementation of our solution (for prototyping and evaluation), the orchestrator directly controls the Blockchain system through special APIs. The orchestrator/BC manager is usually available in commercial BC solutions, which can be augmented with the proposed solution.

The dynamic fine-grained SLA is divided into domain-specific SLAs, i.e., RAN and CN, at the beginning of the workflow. The dynamic fine-grained SLA generates predictions using the Reptile-based meta-learning algorithm in the CN and RAN domains. The section on mapping to SC is shown at the bottom of the figure. It contains the predicted target function that starts the resource allocation function. The other three functions in this block are monitoring of SLOs, dynamic tariff model, and payment & penalty functions. The monitoring subsystem provides regular SLO updates every n minutes, where n is a critical factor in prototype implementation. Reptile ingests these real-time network states to predict optimized SLA targets and required resources. The SCs leverage the Reptile forecasts to automate resource allocation, adhering to the fine-grained SLA specifications encoded in the logic. This combination of fine-grained SLA decomposition, machine learning prediction models, and blockchain-based SCs overcomes the limitations of static SLAs and centralized management.

The following sections detail the complete SLA-to-SC framework implementation.

Solution through Dynamic Fine-grained SLA: To enable responsive, stable performance under diverse real-time network conditions, we implement dynamic fine-grained SLAs. We employ a highly granular approach to dynamic SLA for real-time adjustment of specific service metrics based on current demands. This will allow precision tuning of resources

to maintain high service quality aligned to needs. However, our proposed solution uses static SLAs as an essential baseline or starting point for dynamic fine-grained SLAs. Building upon this base, our proposed dynamic fine-grained SLA introduces enhanced flexibility and adaptability and prepares for real-time conditions like workload, traffic patterns, or other performance factors. Overall, combining a standardized static SLA and an adaptive, granular dynamic SLA in our solution provides stability and responsiveness critical for next-generation services. Sections IV and V cover the details on the granularity, process of decomposition, and resource allocation, respectively.

Solution through Smart Contracts: To address challenges in SLA management related to trust, privacy, immutability, monitoring, self-execution, and complete automation, this work utilizes SCs based on blockchain technology. SCs reduce costs and conflicts between SP and customers. Each network slice has an individual SLA tailored to its service type, where resource metrics such as ultra-high data rate, ultra-low latency, ultra-high reliability, and others must be optimized. We implement a permissioned blockchain network to execute the SCs that encode the SLAs. User information like name, location, service type, and timestamp is stored off-chain, along with SP data. The SC facilitates resource allocation from providers to users. Section VI provides further details on how SCs are mapped, configured, and executed.

IV. Identification of SLA components

This section provides an in-depth analysis of SLAs and their individual components. The aim is to deliver a comprehensive breakdown mechanism of SLA to operate in dynamic environments and their influence on resource allocation and management. Specifically, we emphasize the utilization of fine-grained SLA analysis to enable optimized performance and resource management. By dissecting the key constituents of an SLA, this section establishes a framework for engineering fine-grained SLAs tailored to emerging SC-based management systems. The granular perspective of each component facilitates adaptable SLAs that can respond in real-time to changing network and user conditions for next-generation services. Below, we first give details of the components that we have identified after analyzing more than a hundred SLAs for network services and establishing their correlations and interdependencies. Following this, we give details of the SLO and the specific parameters needed for the 6G eMBB-plus slice.

A. SLA Component Classification & Correlation

Service agreements can be complicated documents usually drafted for legal purposes. Hence, SLAs for 6G services, especially slices, must first be dissected to a fine granular level for enforcement through SCs. Here, we identify 19 components that are usually (and should be) present in eMBB-plus service SLAs. These components have been identified by studying more than 100 SLAs[2] for telecom, Internet, and computing services and then cross-referencing them from literature in computing and engineering management domains. It is important to note that these individual components are highly interconnected and, hence, require a complete analysis

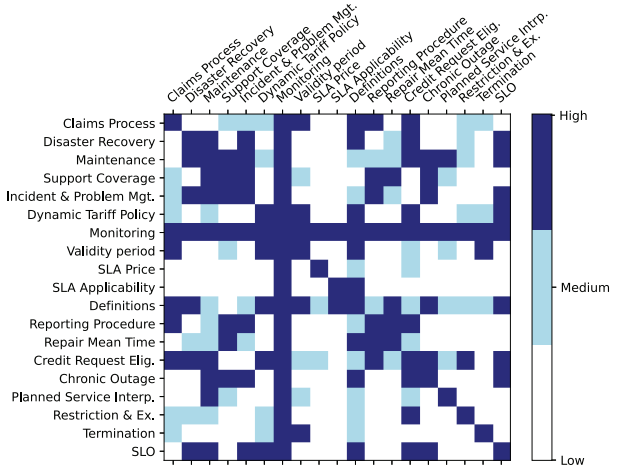


Figure 4: Fine-grained components of SLA and their correlation and interdependencies.

before they can be mapped to SCs. Figure 4, in this regard, presents a correlation matrix between these components. For simplicity and in the interest of space, we only show low (or no) correlation, medium, and high relationship/impact among the components, and give a detailed description of each one of these in Appendix A[2].

B. Service Level Objectives & 6G eMBB-plus Parameters

SLOs are the targeted service levels provided to end users, along with upper and lower bounds. The SLA specifies the acceptable range of variation for the SLO metric, which helps ensure that the service meets the user's requirements. SLOs (dependent on the service type) are categorized as operational or performance objectives. Moreover, they include metrics and target parameters. Metrics are defined according to identifiers (i.e., latency, throughput, and reliability), and target values are maximum, minimum, and average values used to evaluate the metrics.

Operational & Performance Objectives: For the case of eMBB-plus application scenario, the performance metrics are peak & experienced data rate, area traffic capacity, user & control plane latency, peak spectral efficiency, 5th percentile user spectral efficiency, and energy efficiency. Moreover, availability is an added operational objective, while service degradation is taken as a performance objective [3]. These objectives are based on the parameters (KPIs) listed above for the eMBB-plus slice in 6G networks and their values mentioned in Table II. In the interest of space, we have given details of each of these in Appendix B[2].

V. Decomposition of Dynamic Fine grained SLA w.r.t Resource Allocation in RAN & CN

Decomposing the SLA by specific service and network domains offers enhanced performance and resource management benefits. It enables optimal resource allocation by providing greater visibility into domain-specific resources and KPIs for individual services. This helps prevent under or over-provisioning of resources. Decomposition also exposes the objectives most critical to users, facilitating granular SLA

Table II: Recommended KPI values of eMBB-plus service in future 6G system.

No.	6G KPIs	Values
1	Maximum Bandwidth †‡	400MHz - sub-6 GHz band 3.25GHz - mmWave band 100GHz - THz band
2	Peak data rate	≥1 Tb/s
3	Experienced data rate	≈1 Gb/s
4	Area traffic capacity	1 Gb/s/m^2
5	Latency §	10-100 µs
6	Jitter	≈1 µs
7	Peak spectral efficiency	60 b/s/Hz
8	Experienced spectral efficiency *	11-16 b/s/Hz
9	Energy Efficiency	1 Tb/joule

† Band dependent approximations; ‡ Carrier bandwidth is not defined; § Control plane latency may be higher; * 2x of 5G.

optimization for efficient resource utilization, including network bandwidth, processing, and SLA life cycle management. Decomposed SLAs allow for the adaptation of performance metrics and resources per domain as network conditions change. To decompose the SLA for eMBB-plus service resource allocation in RAN and CN domains of a 6G network, we break it down based on domain-specific performance metrics and resource allocation policies. Below, we describe the Reptile mDNN Training model for the proposed system and individually dissect the SLA for RAN and CN.

A. Reptile based mDNN Training

We first explain the mDNN training process and the algorithm for training, so that understanding SLA decomposition and performance metrics becomes easier. Figure 5 provides an overview of the proposed deep neural network architecture based on the Reptile meta-learning algorithm to train specialized models for predicting RAN and CN performance metrics and resource requirements. Unlike traditional machine learning methods that require large datasets for effective training, Reptile algorithm [42] enables rapid adaptation to new tasks even with limited samples. This is achieved through a unique mechanism that allows models to quickly grasp new concepts by iteratively updating their gradients across a distribution of related tasks.

Objective: The mDNN model is employed for predicting SLA metrics and allocating resources for slices with dynamic needs and intermittent demands, i.e. when the user requests a change in SLOs or the network conditions violate the SLOs for any SLA. Specifically, when an eMBB-plus slice customer initially on-boards, historical data to train tailored models is exceptionally scarce. Reptile empowers our system's SLA forecaster and resource manager to quickly adapt during the first few hours of the new slice's lifetime, learning optimized policies even with little operational data. Compared to alternatives (such as model-agnostic meta-learning and meta-gradient descent), Reptile provides simplicity of implementation and few-shot solid learning performance. As shown in Figure 5, the left side enumerates the specific tasks and associated training data corresponding to each domain's metrics and resource types. This encompasses key performance metrics like peak data rate $\mathcal{P}_d$, latency $\mathcal{L}$, bandwidth $\mathcal{B}$, and availability $\mathcal{A}$, as well as their corresponding resources. The input layer size depends on the number of features in the domain-specific datasets. The right side depicts the Reptile-based

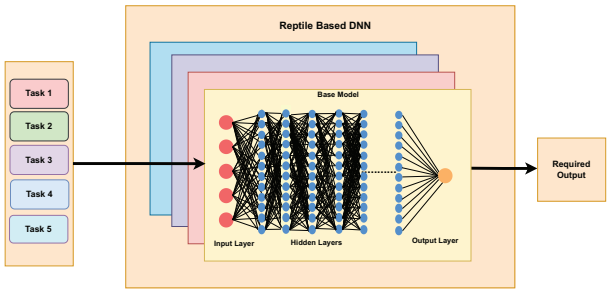


Figure 5: Reptile based mDNN for Radio Access Network and Core Network.

deep neural network architecture with a shared base model. The base model distills transferable knowledge across tasks during meta-training to learn a robust initialization. The output layer size is set to the number of target classifiers. Relu and Tanh activation functions are utilized in hidden and output layers, respectively. The networks are trained on meta-training data with random initialization of weights and biases via forward/backward propagation.

Algorithm 1 mDNN Training Algorithm

Require: Tasks: $\mathrm{T}_n = \{\tau_1, \tau_2, \ldots, \tau_n\}$
Require: α, β: Step size hyperparameters
Require: m: Number of inner gradient steps
Require: θ: Initial meta-parameters
Ensure: Updated meta-parameters θ
while not converged **do**
  Sample tasks $\tau_1, \tau_2, \ldots, \tau_n$
  for all T_n **do**
    $\theta_n \leftarrow \theta$
    for $k = 1, \ldots, m$ **do**
      Sample support set S_n from T_n
      Update $\theta_n = \theta_n - \alpha \nabla_\theta L_{\mathrm{T}_n}(f_{\theta_n}; S_n)$
    end for
    Sample query set Q_n from T_n
    Evaluate $L_{M_n} = L_{\mathrm{T}_n}(f_{\theta_n}; Q_n)$
    $\theta \leftarrow \theta + \beta(\theta_n - \theta)$
  end for
end while
return θ

The complete training algorithm is shown in Algorithm 1. The initial lines outline the requisite inputs, encompassing relevant elements such as related tasks T_n, step size hyperparameters α and β, and the number of inner gradient steps m. The initial parameter θ serves as the meta-parameter for the model, representing the adaptable parameters refined through meta-learning. The algorithm initiates a loop that persists until a predefined stopping criterion is satisfied. Within this loop, the support set $\mathrm{S_n}$ and query set $\mathrm{Q_n}$ are systematically sampled from the task. The $\mathrm{S_n}$ plays a crucial role in the inner gradient update, contributing to the adaptation of the task-specific parameters. Simultaneously, the $\mathrm{Q_n}$ calculates the query loss based on the updated task parameters θ_n. The $\mathrm{L_{T_n}}$ and $\mathrm{L_{M_n}}$ represent the task-specific loss and meta task loss (i.e., loss on $\mathrm{Q_n}$), respectively. In the final stages of the algorithm, the meta-parameter θ undergoes an update via a meta-update rule. The algorithm's output is the updated meta-parameters, which encapsulate the model's enhanced adaptability acquired through the iterative process of meta-learning.

B. SLA Decomposition for RAN

In the following sections, we investigate the performance metrics and resource requirements related to the RAN domain for delivering the eMBB-plus service.

1) Performance metrics: After extensive analysis, we determine that the key performance metrics relevant to the eMBB-plus service in the RAN domain for 6G networks will include $\mathbb{P}^r = \{\mathcal{D}, \mathcal{L}, \mathcal{S}, \mathcal{T}, \mathcal{P}_d, \mathcal{E}_d, C\}$. These are data throughput, latency, signal strength, user throughput, peak data rate, user-experienced data rate, and connection density, respectively. Each of these is a metric. However, in order to enforce one metric, fine-tuning of many target variables is required. By focusing the training on these essential RAN service metrics, the resulting models can effectively forecast performance and, in turn, enable SC-based SLA governance and responsive resource allocation for the eMBB-plus service. In order to simplify the explanation and for prototype implementation, we select two vital performance metrics, i.e., $\mathcal{P}_d$ and $\mathcal{L}$, and explain the targets and resource parameters for them. Note that the following process will be done for all the key performance metrics identified above.

2) Targets: The $\mathcal{P}_d$ and $\mathcal{L}$ target values are set dynamically based on real-time network conditions. We employ the Reptile meta-deep neural network model described earlier to predict these target values. The input dataset features for forecasting $\mathcal{P}_d$ include $\mathcal{P}_d = \{\mathcal{P}'_d, N_{\text{rb}}, N_{\text{s}}, N_{\text{b}}, N_{\text{a}}, N_{\text{l}}, N_{\text{m}}\}$. From the above set, the $\mathcal{P}'_d$ is the historic values of $\mathcal{P}_d$, N_{rb} is the number of physical resource blocks, N_{s} is the number of symbols per N_{rb}, N_{b} is number of bits per N_{s}, N_{a} is number of antennas, N_{l} is number of layers, and N_{m} is modulation order. Similarly, the dataset for predicting $\mathcal{L}$ includes $\mathcal{L} = \{\mathcal{L}'_h, d_{\text{p}}, d_{\text{t}}, d_{\text{r}}, d_{\text{q}}\}$. Here, $\mathcal{L}'_h$ are the historic latency values, while the rest are propagation, transmission, processing, and queuing delay, respectively. By training on these relevant RAN features, the meta-learning model can effectively predict optimized $\mathcal{P}_d$ and $\mathcal{L}$ targets tailored to the current network state. This enables smart contract-encoded SLAs to dynamically adapt resources on performance metrics to maintain QoS.

3) Resource requirements: The key resources for the RAN domain include: bandwidth, processing power (CPU cycles, memory, storage), radio resources (frequency bands, transmission power, antennas, modulation/coding), and edge computing resources. To optimally allocate these resources per the dynamic targets, we apply the mDNN model. The dataset features for predicting $\mathcal{P}_d$ related resources such as N_{B} and N_{s} based on these input features $\mathcal{P}'_d$, N_{b}, N_{a}, N_{l}, $N_{\text{s}'_h}$, and $N_{\text{B}'_h}$. Here, N_{s}, N_{B}, $N_{\text{s}'_h}$, and $N_{\text{B}'_h}$ are the storage, bandwidth and their respective historic values respectively. Similarly, for $\mathcal{L}$-based resource allocation, the dataset incorporates $\mathcal{L}'_h$, d_{p}, d_{t}, d_{r}, $N_{\text{B}'_h}$, and d_{q}, for training, and then the mDNN can effectively forecast resource N_{B}. This enables smart contract-driven autonomous resource provisioning to deliver the eMBB-plus service per SLA.

C. SLA Decomposition for CN

The following section examines the performance metrics and resource requirements of the CN domain for the eMBB-plus service.

1) Performance metrics: The primary performance metrics for evaluating the CN are $\mathbb{P}^c = \{\mathcal{L}, \mathcal{B}, \mathcal{J}, \mathcal{P}_l, \mathcal{A}\}$, where $\mathcal{B}$ is bandwidth, $\mathcal{J}$ is jitter, $\mathcal{P}_l$ is packet loss, and $\mathcal{A}$ is availability. Again, each of these elements is a metric in itself; however, in order to enforce one metric, fine-tuning many target variables is required. For this work, we select two vital metrics, i.e., $\mathcal{A}$ and $\mathcal{B}$ to focus the training of the prediction models. The rationale behind selecting two performance metrics is to streamline the execution of our suggested solution, though alternative metrics may also be utilized. By concentrating on these essential CN service metrics, the resulting models can effectively forecast performance and facilitate smart contract-based SLA and resource governance for CN components delivering the eMBB-plus service.

2) Target: To dynamically adapt $\mathcal{A}$ and $\mathcal{B}$ targets in the CN per changing conditions, we utilize the same mDNN model; however, it is not the same instance. As the CN domain is separate and its orchestrator is also independently working, this is a separate instance specific to the CN domain. The input dataset features for predicting $\mathcal{A}$ are $\mathcal{A} = \{\mathcal{A}'_h, \mathcal{B}, N_{\text{a}}\}$ where $\mathcal{A}'_h$ are historic availability values. For forecasting $\mathcal{B}$, the dataset contains, $\mathcal{B} = \{\mathcal{B}'_h, B_{\text{ch}}, N_{\text{mc}}, N_{\text{rb}}\}$ where $\mathcal{B}_h$ are historic peak bandwidth values, B_{ch} is channel bandwidth and N_{mc} is modulation and coding schemes. By training on these relevant domain features, the mDNN can effectively predict optimized $\mathcal{A}$ and $\mathcal{B}$ targets tailored to the current CN state.

3) Resource requirements: The input dataset features for predicting $\mathcal{A}$-based resource such as $\mathcal{B}$ include: $\mathcal{A}'_h$, $\mathcal{B}'_h$, and N_{a}. For $\mathcal{B}$ resource allocation B_{ch}, the dataset contains $\mathcal{B}'_h$, $B_{\text{ch}'_h}$, and N_{mc}. Here, the $B_{\text{ch}'_h}$ are the historic channel bandwidth values. By training on these relevant domain features, the meta-learning model can effectively predict resource requirements tailored to the dynamic $\mathcal{A}$ and $\mathcal{B}$ targets. This allows smart contract-based autonomous resource provisioning to deliver the eMBB-plus service meeting CN SLAs.

VI. Smart Contracts & System Modules Design

In this section, we discuss the complete design of the SLA to SC mapping process for the proposed slice and explain the functionality of different modules as shown in Figure 3. We describe all the modules from the working and SC perspectives.

We employed Hyperledger Sawtooth to map fine-grained SLA components into executable smart contract logic on the blockchain. The smart contract encoded all signed SLAs, validity periods, SLOs, pricing, compensation values, and customer/provider/monitoring system addresses. In our Sawtooth system model, each participating node comprises of four key components - client, REST API, validator, and transaction processor. The client groups transactions into batches, digitally signs these batches, and then submits them to validator through a REST API. The validator verifies the batch integrity and routes it to the appropriate transaction processor. After validating the batch, the transaction processor commits it on-chain. The implementation of the same is visible in Figure 8, which shows the containerized nodes and the control elements

(part of client, REST API, and data capturing hooks). Further implementation details are given in Section VII. The reasoning for the selection of Sawtooth is already discussed in Section I.

A. Mapping of SLA to Smart Contract

In our presented model, the input received from the SLA-SC orchestrator is decomposed into dynamic fine-grained SLAs for eMBB-plus service for RAN and CN networks. These decomposed SLAs are mapped separately to two different smart contracts for RAN and CN domains, respectively. However, the framework for mapping the SLA to a smart contract is the same for both decomposed SLAs. The Reptile-based learning algorithm is employed outside the smart contract, and its output (prediction) is taken by the specific smart contract for the execution of tasks, i.e., execution of target for optimal resource allocation.

Figure 6 illustrates the framework for mapping dynamic fine-grained SLA components (identified in Section IV.A) onto Sawtooth-based smart contracts[1]. The left side represents the SLA elements to encode, while the right depicts a Sawtooth node. The SLA validity period is programmed as a timer with start and end times. The service type name extracts the transaction family for the validator to route transaction batches. The public and private keys, which are generated from customer and provider information, enable transaction signing. These fields populate the transaction header. SLA organizations, definitions, and descriptions are mapped to comments. SLOs map to performance monitoring functions, while the pricing models map to dynamic tariff logic. Chronic outage and eligibility of credit map to payment and penalty functions. Planned interruptions and restrictions are unmapped exceptions.

Once the SC is created and executed, it is responsible for ensuring that SLOs are met, and if any violation occurs, corrective measures are taken. Transactions created in the client connect to the validator and transaction processor via the REST API. The transaction processor implements the business logic smart contract, defining the actions on transactions per the SLA. This includes monitoring metrics, predicting targets, and allocating resources, payments, and penalties. A dynamic tariff policy is also encoded.

B. Monitoring System

The monitoring system is the primary subsystem that keeps a check on the SLO targets and allocation of resources according to the SLA terms. It is important to note that the parameter n is crucial and needs to be fine-tuned. As discussed earlier, n is the time difference after which any SC checks for SLO violations across all KPIs and resources. Smaller values will lead to quicker prediction and corrective measures, lesser downtime and penalties, and a better quality of experience for the user. However, this will require the monitoring system

[1]Framework does not limit the solution to Sawtooth. In Figure 3, an Automated Information Extractor module is shown, which is a separate solution (separate publication). It can generate three different types of SC for different BCs in multiple languages. However, mapping is the critical design challenge.

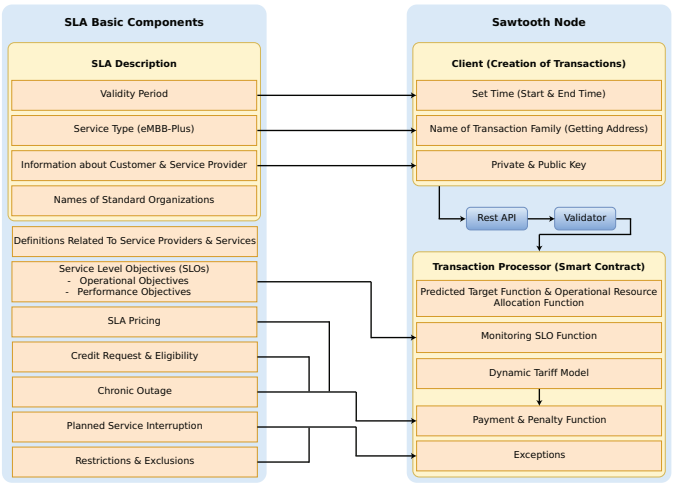


Figure 6: Mapping of SLA's components to smart contract.

to utilize more resources on the orchestrator end. A larger n value will utilize fewer such resources, but the downtime and service degradation will be much more severe, resulting in more penalties to the SP. In the evaluation section, we show both these aspects and discuss the consequences.

The payments and penalties are dependent on the monitoring function evaluations. Once the violation is detected, the prediction resource allocation modules are triggered, which are explained subsequently.

C. Dynamic Tariff Model

Our proposed solution-based smart contract architecture also supports dynamic pricing when users request services beyond the currently defined SLA. When such an extra service request is received, the smart contract calculates the additional cost for the resources based on the pricing model encoded into the contract logic. It queries the user's account balance on the blockchain to verify sufficient funds for this ad-hoc purchase. If an adequate balance exists, the smart contract facilitates the allocation of the extra requested resources to the user. However, if the user's account lacks enough funds, the smart contract immediately notifies them of insufficient balance to fulfill the request. The request is suspended until the user can deposit adequate funds to cover the additional cost. This integrated dynamic pricing capability enables users to purchase extra on-demand services while the smart contract mediates transparently. Furthermore, once the request is approved, the prediction and allocation module will start to ensure this smooth transition.

D. Predicted Target & Optimal Resource Allocation

The core focus of this mapping is the resource allocation policy, which is depicted in the flowchart shown in Figure 7. This mapping ensures that when a service starts, it considers the available resources and conditions specified in the SLA. When the predicted parameter value x_p reaches the smart contract, it is checked to determine if it falls within the SLO range, demonstrating good QoE for the user. If the x_p falls outside the SLO range, it is sent to training modules for resource prediction R_p. Once the smart contract receives the

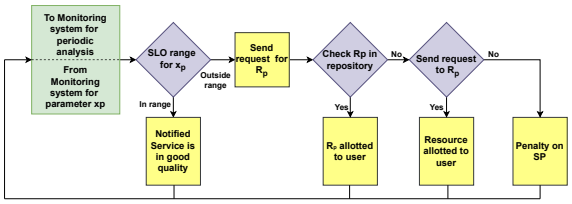


Figure 7: Control flow and policy for resource allocation in the orchestrator.

R_p requirements, it checks its repository to see if those resources are available. The smart contract allows the resources to the user if the resources are present. If the resources are unavailable, the smart contract requests them from the resource provider. If the resource provider supplies the resources, they are allotted to the user, and the SP pays the resource provider. Otherwise, a penalty is levied on the SP by the user for failing to deliver upon resources to maintain the service quality.

E. Payments & Penalty Function

The SLA components, i.e., pricing, credit request, eligibility criteria, and chronic outage, are programmed in payments and penalty functions. It gets the input from the monitoring system and dynamic tariff model through the resource allocation subsystem. This function calculates conditions, fees, and penalties against the SLA violations for performance metrics and resource allocation. The other components' planned service interruption, restrictions, and exclusions are programmed into exceptions. These parts are not in the scope of this work, as they are highly dependent on service providers, local laws, and market conditions. However, this does not mean that they cannot be implemented. The proposed framework already considers them as part of mapping.

VII. Prototype Implementation & Evaluation

This section describes the prototype implementation and evaluates the proposed Reptile-based meta-learning solution for predicting the network parameters and resources. The evaluation focuses on assessing the performance of the trained forecasting models on the defined SLA metrics of $\mathcal{P}_d$, $\mathcal{L}$, $\mathcal{A}$, and $\mathcal{B}$ for the RAN and CN domains, respectively. Additionally, the SLA violations and system performance are measured for the BC network.

A. Setup & Implementation

Testbed for Training & Blockchain: The testbed is built using multiple systems in a LAN environment connected via 1 Gbps Ethernet links, and TL-SG1024D managed switch, as shown in Figure 8. The training modules are powered by a 2.4GHz 10-core Xeon server with a RTX3080 12GB GPU and 40GB DDR4 RAM labeled S1. In addition, the Blockchain system is built using a 3-node Hyperledger Sawtooth network (N1-N3), where each node has an i5-13500 2.5Ghz processor with an RTX3060 12GB GPU and 16GB DDR4 RAM. Moreover, all systems utilize NVMe SSDs; hence, IO operations have a minimum possible delay. In this setup, the S1 Xeon server hosts the orchestrator module for the proposed system,

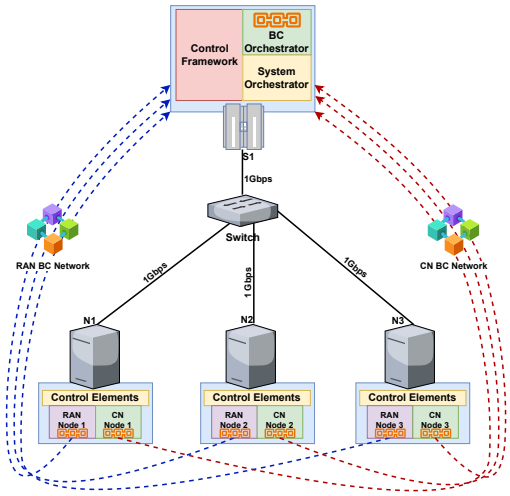


Figure 8: Testbed configuration and topology for prototype implementation. The dashed lines show the logical connectivity of separate BC networks.

the three systems act as pure Hyperledger Sawtooth 1.2 nodes, and all of them run Ubuntu 22.04.0 with kernel version 5.15. Furthermore, they model the RAN and CN separately; hence, each physical system hosts containers for Sawtooth nodes. To capture the raw data parameters and evaluate the system, we have separate application hooks built into the system for reporting as a control framework.

Dataset & Training: To facilitate training of Reptile models, synthetic datasets[2] are used for each parameter and associated resource for the RAN and CN domains. The training data forms the basis for evaluating Reptile's prediction capabilities on SLA metrics and resources. Separate multilayer neural network models are created for each domain's parameters and resource combinations. Each domain-specific performance metric and its resources-related tasks, each of which contains 70 K data points, are used to train the models. The models are then tested using new tasks (30K) not included in the training dataset. The optimal convergence of the models is ensured by careful hyperparameter tuning during training. The performance analysis examines whether the accuracy attained by the tuned Reptile models is sufficient to deliver reliable and trustworthy predictions that can drive automated smart-contract-based SLA management.

Control Framework: We have developed a custom application to act as the client (multiple instances) and feed the orchestrator with SLAs and dynamic changes in requirements. The orchestrator also acts as an eMBB-plus slice manager; however, its implementation is limited for the purpose of the system. In a real 6G environment, this task will be performed by the Network Operator slice manager, with significantly larger functionality. Sawtooth nodes use PoET CFT for consensus of the SC generated by the orchestrator and a validator that checks SC for resource allocation. The validator is modified to mimic an actual client experiencing the impact of resource allocation and then forms a feedback loop with the control application at the orchestrator/slice manager. An

[2]Appendix, related papers, and dataset details are available at: https://github.com/KashifSharif/SLA_DataSet.git

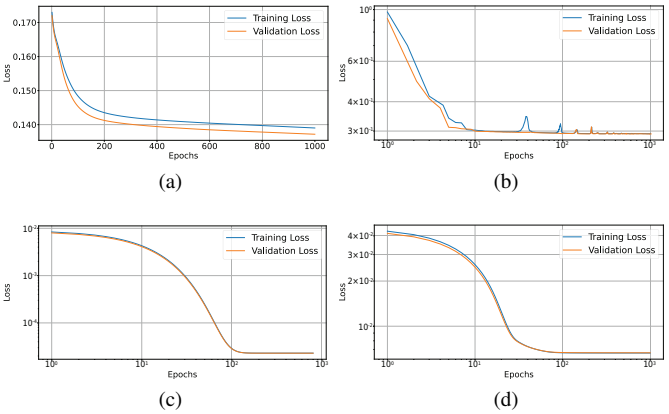


Figure 9: Loss curves in RAN domain: (a) Peak data rate parameter, (b) Peak data rate resources, (c) Latency parameter, (d) Latency resources.

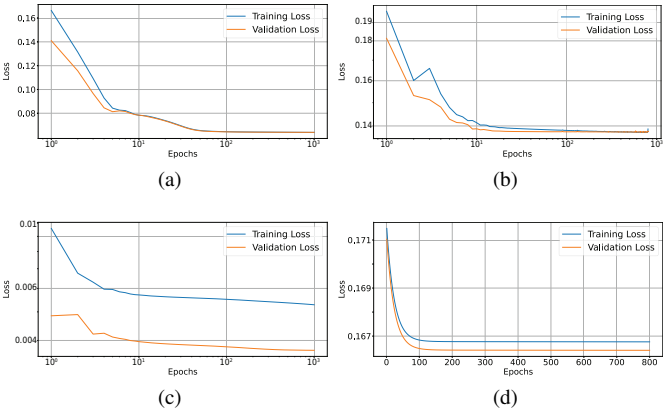


Figure 10: Loss curves in CN domain: (a) Peak bandwidth parameter, (b) Peak bandwidth resources, (c) Availability parameter, (d) Availability resources.

SC is initiated upon confirmation of the contract fee from the transaction payload submitted by the user. This triggers the automated system for mapping SLA components and conditions into executable smart contract logic on Sawtooth. Key SLA aspects, such as resource allocation policies, pricing models, and penalties, are encoded into smart contract functions. Integrated prediction models enable dynamic resource assignment aligned with the SLA. This implementation employs x_p to optimize resource allocation and ensure service quality, particularly when user demand increases.

B. Training Model Test Results

Evaluation of the proposed Reptile-based meta-learning algorithm requires analysis of the training and validation loss curves during the learning process. It provides critical insights into the model's effectiveness for predicting defined SLA parameters and resources. Note that the resources are the different resource parameters described in Section V. The convergence towards minimal loss values proves that the trained model can achieve high accuracy in predicting desired SLA targets such as $\mathcal{P}_d$, $\mathcal{L}$, $\mathcal{B}$ and $\mathcal{A}$ as well as predict associated resource requirements.

Results & Analysis: Figure 9 depicts the performance metrics and their respective resources in the RAN domain for $\mathcal{P}_d$ & $\mathcal{L}$. Figure 9a presents the training and validation loss trajectories for predicting $\mathcal{P}_d$ metric for 1000 epochs. The steady downward progression observed in the training and validation losses provides empirical evidence of our system, especially the trained model's capability to continuously enhance its prediction accuracy on this key RAN metric as more data is processed during meta-training. Figure 9b shows the training and validation losses when tuning the model for predicting the RAN resources required to support the target $\mathcal{P}_d$ per slice. Despite some oscillations (log scale), the declining trend in resource prediction losses validates that the training model can effectively learn specialized models for associated resource allocation. The illustration in Figure 9c shows the training and validation loss trends for forecasting the $\mathcal{L}$ metric in the RAN. In contrast, Figure 9d is for the resource allocation based on target $\mathcal{L}$ constraints. As the epochs increase, a consistent downward progression is evident in both the training and validation loss curves. This signifies the model's capability to continuously enhance its $\mathcal{L}$ prediction accuracy (and corresponding resource requirements) with more experience.

Figure 10 shows the performance of CN-related metrics and resources. Figure 10a illustrates the training and validation loss curves for predicting $\mathcal{B}$. While there is an initial distance between the curves, the gap closes as the epochs increase, showing the improved generalization of the learned representations of the proposed model. Similarly, Figure 10b depicts the training and validation loss trends when tuning the models to predict $\mathcal{B}$ resource parameters for desired SLOs. The consistent descent verifies that it can learn across domain-specific $\mathcal{B}$ resources required to deliver committed service levels. Figure 10c shows the training and validation loss trends to predict $\mathcal{A}$. A downward slope is again observed; however, there is a difference in the actual values of training and validation. Compared to the previous results, this gap is significantly higher. Close observations have shown that the availability parameter is more complex than other direct values, such as bandwidth, data rate, or latency. Hence, minor artifacts in the dataset may have contributed to this gap. However, this requires further investigation and will be part of future work in this project. Figure 10d illustrates the training and validation losses when specializing our model to forecast resource allocation based on the predicted $\mathcal{A}$, and a similar trend is observed as earlier, but without any artifacts.

In summary, in Figures 9 & 10, the losses steadily decline as the training progresses, validating the proposed model's capability to continuously learn. Critically analyzing these graphs proves that the proposed system can be reliably deployed to enable automated blockchain-based SLA governance through accurate service metrics and resource forecasts.

Unseen SLA Validation: In addition to training and validation, the adaptability of the learned models is critically evaluated by testing performance on new examples not encountered during training. For each model trained on a specific SLA parameter and resource, the test loss is measured for the

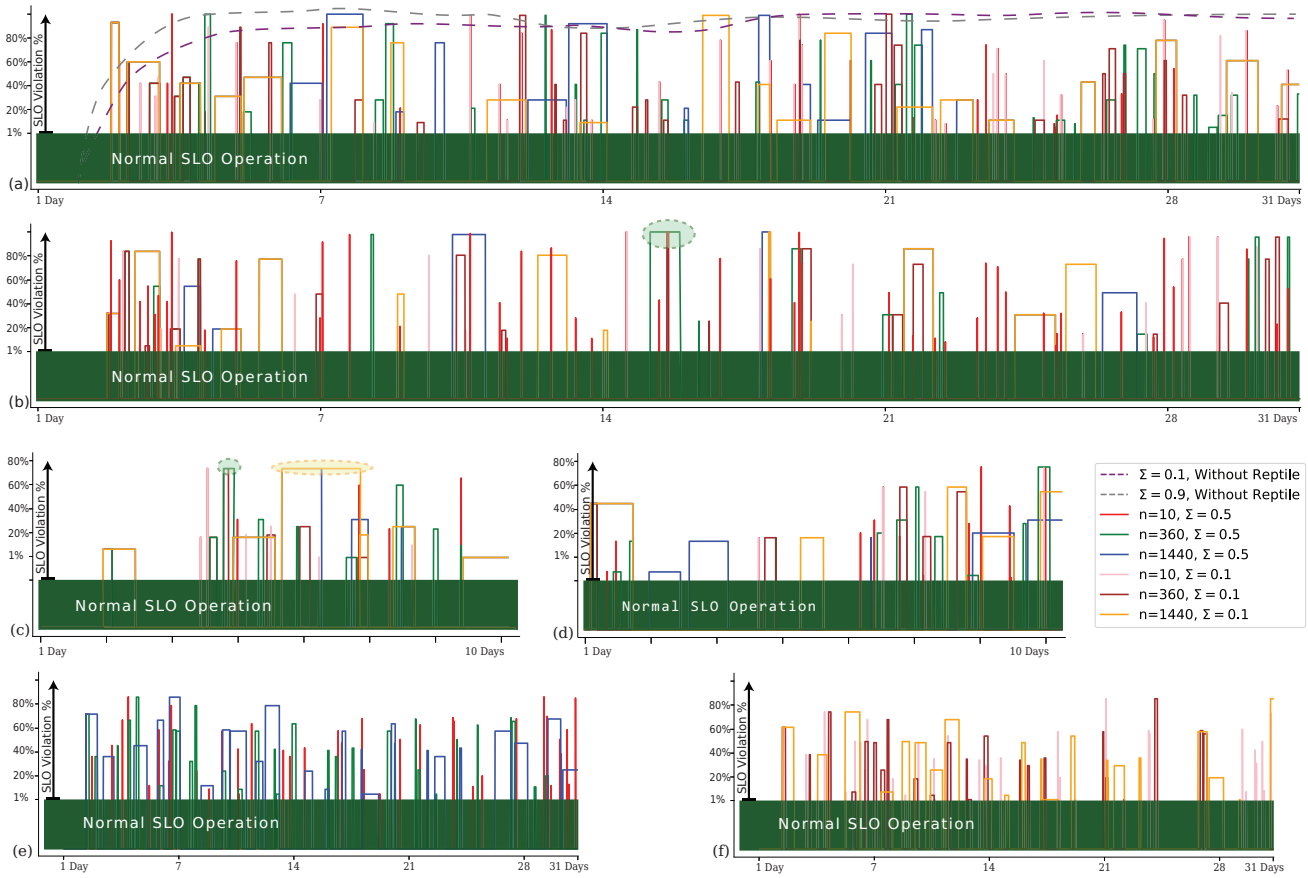


Figure 11: SLO Violations of: (a) i^{th} RAN SLA, (b) i^{th} CN SLA, (c) j^{th} RAN SLA, (d) j^{th} CN SLA, (e) k^{th} RAN SLA with $\Sigma = 0.5$, and (f) k^{th} RAN SLA with $\Sigma = 0.1$.

20 unseen examples. As highlighted in Table III, the models trained with the proposed algorithm exhibit significantly reduced test losses compared to the baseline models without meta-learning. This demonstrates the ability of the proposed system to produce generalized models that can maintain robust prediction accuracy. When deployed for real-time SC-based SLA management, this adaptability is key where inputs may differ from training.

C. SLO Violation and Reptile Execution Results

To evaluate the overall system functionality and test the trained models on real SLAs, we used 100 real-world SLAs[2] in the proposed system. Figure 11 shows the results for three randomly selected SLAs (i^{th}, j^{th}, & k^{th}). The system models the execution of the SLA for 31 days (mimicking one month of service), and all the SLAs are executing in parallel, i.e. the SCs for all are active and their uniqueness guaranteed.

Table III: Loss with unseen data for a random test SLA.

No.	Parameter Names	Without Learning	With Proposed Learning Model
1	Peak data rate $\mathcal{P}_d$	0.0877	0.0767
2	Peak data rate $\mathcal{P}_d$ resources	0.3269	0.2966
3	Latency $\mathcal{L}$	$3.9920\text{x}10^{-06}$	$1.5837\text{x}10^{-07}$
4	Latency $\mathcal{L}$ resources	0.0118	0.0109
5	Peak bandwidth $\mathcal{B}$	0.0192	0.00608
6	Peak bandwidth $\mathcal{B}$ resources	0.1411	0.08465
7	Availability $\mathcal{A}$	2.4286	0.0404
8	Availability $\mathcal{A}$ resources	0.1773	0.16787

The evaluations focused on the *monitoring interval* n, *SLO violation severity* (y-axis), and the network dynamicity Σ. n is varied between 10 min, 360 min (6 hours), & 1440 min (24 hours). Note that this is where a corrective measure is taken for any violation, while the violation may have occurred any time after the last monitoring point. Violation severity is measured as a percentage of a deviation from the acceptable value until the service becomes unavailable (100% violation). The network dynamics Σ, which forces the violations & severity, is varied between 0.1 and 0.9, where 0.1 means that 10% of the resource parameters of the network are continuously changing. This mimics client joining and leaving, network connections, and data usage. In the implementation, we ensure that the parameters selected for change are the ones that impact $\mathcal{P}_d$ & $\mathcal{L}$ for RAN, and $\mathcal{B}$ & $\mathcal{A}$ for CN experiments. The granularity of the discrete-time scheduler is 1 sec in the control framework, and the RAN and CN domains are modeled separately for the same SLAs.

Results & Analysis: Figure 11 depicts the SLO violation when it occurs (real-time) for the three SLAs. The shaded area represents the threshold of SLOs for the RAN and CN domains. Figures 11a & 11b show the i^{th} SLA for the RAN and CN domains, respectively. The dashed lines in Figure 11a show the SLO violations without any training model for different values of Σ. As expected, once a violation occurs, the violation continues unless corrective measures are taken. In traditional systems, human intervention is required

(after reporting by the client) to address this issue. Although Figure 11b is for the same SLA, we see fewer violations than Figure 11a. This difference in the CN and RAN domains is due to the significantly dynamic nature of RAN and the more stable nature of the CN. It is important to note that the implementation test bed is part of a larger project capable of modeling 5G-Advanced slices; hence, the parameter adjustment for RAN and CN is quite realistic. We also note that the value of n is critical in removing the violations. Lesser n values resolve this more quickly than larger values; however, this is at the cost of running a continuous monitoring system.

Figures 11c & 11d show zoomed-in 10-day SLO violations for j^{th} SLA in RAN and CN domains, respectively. The objective is to highlight an anomaly in the violation resolution process. Also visible in Figure 11b and indicated by shaded ovals, we found that at some points, the training system did not predict the resources correctly, or resources were unavailable to satisfy the SLA conditions. Hence, the violation is prolonged until the subsequent monitoring interval and resolved. Such occurrences are rare, but this does show that higher values of n will experience longer QoS degradation (and the SP will be penalized more). Furthermore, we also observed from different SLAs that they occur when the violation is usually above 70%. This indicates a scarcity of resources for allocation and highlights the increased complexity of QoS degradation. When violations become more extreme, simply adjusting the resource allocation may not be sufficient. The key takeaway is that while the proposed system is adept at responding to transient violations, even with intelligent automation, major SLO breaches will still necessitate more arduous mitigation processes spanning hours or days, depending on scale and complexity.

Figures 11e & 11f show the k^{th} SLA for the RAN only by separating the graphs for Σ 0.1 and 0.5. This SLA had one of the highest areas under the curves (boxes) among all the 100 SLAs. Careful examination proves the effectiveness of our training model, response to dynamicity, monitoring system, and prediction and resource allocation modules. 99.9 of the violations are resolved across the 31-day evaluation in the RAN for both Σ values. Figure 11 shows the violation and its severity for each of the four observed parameters. It does not show the actual violated conditions and the values after remedial action. In Table IV, we list the exact values of the regular SLO operation (shaded region in Figure 11), violation value, values after prediction, predicted resource location, and penalty amount. As it is impossible to show all the data, we randomly selected the violation instance from the k^{th} SLA shown in Figure 11e & 11f for the RAN. It can be observed from the table that for each parameter, the predicted values for resource allocation from the trained model are within the acceptable range. This demonstrates the critical capability of the system, such as leveraging tight monitoring feedback and swift automated mitigation procedures to maintain SLOs, thus averting prolonged periods of degraded user experience.

D. Blockchain Network Results

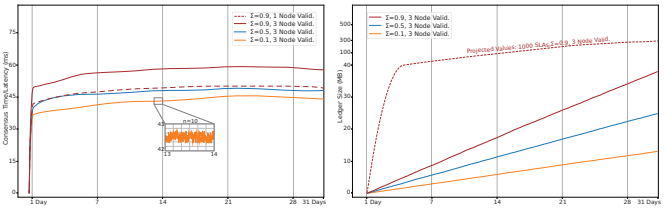
The prototype is implemented using Hyperledger Sawtooth; observing its impact on nodal performance is essential. Figure 12 shows the statistics of the BC nodes for 31 days of SLA execution. The time is measured from the point when the predicted values are returned and the batch is confirmed, i.e., BC orchestrator receives values; a transaction is created to record the changes, then committed to the batch for consensus (PoET CFT), and finally recorded in the blockchain (batch confirmed). The resource allocation module of the proposed system triggers after the batch is confirmed. We plot this BC latency/consensus time against time for both the RAN and CN domains and compare them against different values of Σ. In blockchain terms, higher values of Σ indicate higher TPS (transactions per second). It can be observed from Figure 12a that the latency is quite acceptable for $n = 10$. Although higher TPS will require more time, reducing the number of validation nodes may be possible with a secure and controlled SP. Hence, the dashed line shows a significant time reduction for $\Sigma = 0.9$. The figure scale shows smooth lines; however, the magnification shows the fluctuations per batch on a single chain. Furthermore, from Figures 12a and 11a we can deduce that without the SC, the SLOs will remain violated until human intervention is done. Figure 12b shows the ledger size for the RAN domain for varying values of Σ. The transaction details are the changes in the parameter values, which occupy a few bytes of data; hence, the overall ledger size with $n = 10$ increases steadily. Even with 1000 SLAs, the projected storage requirements are insignificant. In summary, the BC-based system provides the benefits of smart contracts and secure storage of records with acceptable delays and storage needs.

Table IV: SLO violation and prediction values for k^{th} SLA.

Parameter	Acceptable SLO Range	SLO Value Detected	SLO Value After Prediction	Resource	Penalty
Peak data rate ($\mathcal{P}_d$)	0.0138-0.0596 Gbps	0.0045 Gbps	0.0376 Gbps	Edge caching	Nil
Latency ($\mathcal{L}$)	0.35-0.46 ms	0.5708 ms	0.394 ms	Centralized caching	Nil
Peak bandwidth ($\mathcal{B}$)	0.056-0.089 GHz	0.044 GHz	0.06 GHz	Channel Bandwidth	Nil
Availability ($\mathcal{A}$)	97-99 %	96 %	494 GHz	Peak Bandwidth	Nil

VIII. Conclusion & Future Directions

This research aims to advance the foundational understanding of mapping fine-grained SLAs to smart contracts to realize intelligent, trustworthy 6G network slicing. This paper proposes an AI and blockchain-based approach to SLA management for efficient resource allocation in 6G RAN and CN domain for eMBB-plus slices. The main focus has been breaking down static SLAs and translating broken-down components into Hyperledger Sawtooth smart contracts. The



(a) (b)

Figure 12: Blockchain: (a) Consensus time, (b) Ledger size.

evaluation demonstrates that the proposed approach is practical, dynamic, intelligent, and decentralized. Reduced training and validation losses show that the meta-learning algorithm successfully adapts to new tasks. Furthermore, dynamic resource allocation improves the QoE and reduces the penalties to service providers.

Based on this work, we have identified several future directions. Reducing the monitoring parameter will increase the monitoring costs. As we do not isolate the monitoring function, CPU and memory usage must be analyzed in future research. Predictions and corrective measures for all SLAs are done at the same time. If the number of SLAs increases significantly, there may be a delay in completing the prediction and correction processes. Analyzing this and improving the process can be an exciting direction. Furthermore, grouping similar SLOs and predicting resources as a group may further reduce the execution times. The current analysis did not show any recurring patterns in violations. Patterns in network dynamics may also create patterns of violations. Identifying such recurring patterns using a specialized training model may also be an interesting research direction.

REFERENCES

[1] C.-X. Wang *et al.*, "On the road to 6g: Visions, requirements, key technologies, and testbeds," *IEEE Communications Surveys & Tutorials*, vol. 25, no. 2, pp. 905–974, 2023.

[2] H. N. Qureshi *et al.*, "Service level agreements for 5g and beyond: overview, challenges and enablers of 5g-healthcare systems," *IEEE Access*, vol. 9, pp. 1044–1061, 2020.

[3] G. Liu *et al.*, "Vision, requirements and network architecture of 6g mobile network beyond 2030," *China Communications*, vol. 17, no. 9, pp. 92–104, 2020.

[4] H. Technologies, "6g: The next horizon white paper," in *6G — a more advanced next-generation mobile communication system — will go far beyond just communications.* Huawei Technologies, 2022, pp. 1–35.

[5] A. Banchs *et al.*, "Network intelligence in 6g: Challenges and opportunities," in *ACM Workshop on Mobility in the Evolving Internet Architecture*, 2021, pp. 7–12.

[6] T. Hewa *et al.*, "The role of blockchain in 6g: Challenges, opportunities and research directions," *2nd 6G Wireless Summit*, pp. 1–5, 2020.

[7] A. Haddad *et al.*, "Systematic review on ai-blockchain based e-healthcare records management systems," *IEEE Access*, vol. 10, pp. 94 583–94 615, 2022.

[8] S. Wang *et al.*, "Blockchain-enabled smart contracts: architecture, applications, and future trends," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 49, no. 11, pp. 2266–2277, 2019.

[9] B. Ampel *et al.*, "Performance modeling of hyperledger sawtooth blockchain," in *2019 IEEE International Conference on Intelligence and Security Informatics*. IEEE, 2019, pp. 59–61.

[10] G. Perboli *et al.*, "Blockchain-based transaction management in smart logistics: A sawtooth framework," in *44th Annual Computers, Software, and Applications Conference*. IEEE, 2020, pp. 1713–1718.

[11] H. Tataria *et al.*, "6g wireless systems: Vision, requirements, challenges, insights, and opportunities," *Proceedings of the IEEE*, vol. 109, no. 7, pp. 1166–1199, Jul. 2021.

[12] T. Hospedales *et al.*, "Meta-learning in neural networks: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 9, pp. 5149–5169, 2021.

[13] L. U. Khan *et al.*, "6g wireless systems: A vision, architectural elements, and future directions," *IEEE Access*, vol. 8, pp. 147 029–147 044, 2020.

[14] W. Wu *et al.*, "Ai-native network slicing for 6g networks," *IEEE Wireless Communications*, vol. 29, no. 1, pp. 96–103, 2022.

[15] L. U. Khan *et al.*, "Network slicing: Recent advances, taxonomy, requirements, and open research challenges," *IEEE Access*, vol. 8, pp. 36 009–36 028, 2020.

[16] A. F. M. Hani *et al.*, "Renegotiation in service level agreement management for a cloud-based system," *ACM Computing Surveys*, vol. 47, no. 3, pp. 1–21, 2015.

[17] T. Faisal *et al.*, "How to design autonomous service level agreements for 6g," *IEEE Communications Magazine*, vol. 61, no. 3, pp. 80–85, Mar. 2023.

[18] C. A. Gizelis and D. D. Vergados, "A survey of pricing schemes in wireless networks," *IEEE Communications Surveys & Tutorials*, vol. 13, no. 1, pp. 126–145, 2010.

[19] H. Yang *et al.*, "Artificial-intelligence-enabled intelligent 6g networks," *IEEE Network*, vol. 34, no. 6, pp. 272–280, 2020.

[20] K. B. Letaief *et al.*, "Edge artificial intelligence for 6g: Vision, enabling technologies, and applications," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 5–36, 2021.

[21] Q. Guo *et al.*, "Federated reinforcement learning-based resource allocation in d2d-enabled 6g," *IEEE Network*, vol. 37, no. 5, pp. 89–95, Sep. 2023.

[22] A. Alwarafy *et al.*, "Ai-based radio resource allocation in support of the massive heterogeneity of 6g networks," in *2021 IEEE 4th 5G World Forum (5GWF)*. IEEE, 2021, pp. 464–469.

[23] R. B. Uriarte *et al.*, "Towards distributed sla management with smart contracts and blockchain," in *IEEE International Conference on Cloud Computing Technology and Science*. IEEE, 2018, pp. 266–271.

[24] E. J. Scheid *et al.*, "Enabling dynamic sla compensation using blockchain-based smart contracts," in *IFIP/IEEE Symposium on Integrated Network and Service Management*. IEEE, 2019, pp. 53–61.

[25] S. K. Battula *et al.*, "A blockchain-based framework for automatic sla management in fog computing environments," *The Journal of Supercomputing*, vol. 78, no. 15, pp. 16 647–16 677, 2022.

[26] T. Faisal *et al.*, "Beat: Blockchain-enabled accountable and transparent network sharing in 6g," *IEEE Communications Magazine*, vol. 60, no. 4, pp. 52–56, 2022.

[27] S. B. Saad *et al.*, "A trust architecture for the sla management in 5g networks," in *IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[28] F. Wilhelmi and L. Giupponi, "On the performance of blockchain-enabled ran-as-a-service in beyond 5g networks," in *2021 IEEE Global Communications Conference*. IEEE, 2021, pp. 01–06.

[29] F. Patel *et al.*, "Block6tel: Blockchain-based spectrum allocation scheme in 6g-envisioned communications," in *International Wireless Communications and Mobile Computing*. IEEE, 2021, pp. 1823–1828.

[30] Z. Wang *et al.*, "Blockchain-based fog radio access networks: Architecture, key technologies, and challenges," *Digital Communications and Networks*, vol. 8, no. 5, pp. 720–726, 2022.

[31] S. Hu *et al.*, "Blockchain and artificial intelligence for dynamic resource sharing in 6g and beyond," *IEEE Wireless Communications*, vol. 28, no. 4, pp. 145–151, 2021.

[32] A. Adhikari *et al.*, "Wireless network virtualization by leveraging blockchain technology and machine learning," in *ACM Workshop on Wireless Security and Machine Learning*, 2019, pp. 61–66.

[33] Z. Wang *et al.*, "Business innovation based on artificial intelligence and blockchain technology," *Information Processing & Management*, vol. 59, no. 1, p. 102759, 2022.

[34] H. Taherdoost, "Smart contracts in blockchain technology: A critical review," *Information*, vol. 14, no. 2, p. 117, 2023.

[35] E. J. Scheid and B. Stiller, "Automatic sla compensation based on smart contracts," IFI-TecReport No. 2018.02, TEST, Zürich, Switzerland, Tech. Rep., apr 2018. [Online]. Available: https://files.ifi.uzh.ch/CSG/staff/scheid/extern/publications/IFI-2018.02.pdf

[36] H. Xu *et al.*, "Blockchain-enabled resource management and sharing for 6g communications," *Digital Communications and Networks*, vol. 6, no. 3, pp. 261–269, 2020.

[37] M. M. Surya *et al.*, "Converging artificial intelligence and blockchain technology using oracle contract in ethereum blockchain platform," in *International Conference on Informatics and Computing*. IEEE, 2020, pp. 1–5.

[38] D. Kim *et al.*, "Improved raft algorithm exploiting federated learning for private blockchain performance enhancement," in *International Conference on Information Networking*. IEEE, 2021, pp. 828–832.

[39] A. Salh *et al.*, "A survey on deep learning for ultra-reliable and low-latency communications challenges on 6g wireless systems," *IEEE Access*, vol. 9, pp. 55 098–55 131, 2021.

[40] T. Faisal *et al.*, "Ajit: Accountable just-in-time network resource allocation with smart contracts," in *ACM MobiArch Workshop on Mobility in the Evolving Internet Architecture*, 2020, pp. 48–53.

[41] F. Fossati *et al.*, "Multi-resource allocation for network slicing under service level agreements," in *10th International Conference on Networks of the Future*. IEEE, 2019, pp. 48–53.

[42] M. Huisman *et al.*, "A survey of deep meta-learning," *Artificial Intelligence Review*, vol. 54, no. 6, pp. 4483–4541, 2021.