ML-based Inter-Slice Load Balancing Control for Proactive Offloading of Virtual Services

Felipe S. Dantas Silva[a,b], Sérgio N. Silva[a,c], Lucileide M. D. da Silva[a,b], Ayuri Bessa[a,c], Samuel Ferino[a,c], Pablo Paiva[a,c], Marcos Medeiros[a,c], Lucas Silva[a,c], José Neto[a,c], Kevin Costa[a,c], Charles Santos[a,c], Eduardo Aranha[a,c], Allan Martins[a,c], Uirá Kulesza[a,c], Roger Immich[a,c], Augusto V. Neto[a,c], Ramon Fontes[a,c], Vicente Sousa[a,c], Marcelo A. C. Fernandes[a,c,*]

[a] *Leading Advanced Technologies Center of Excellence (LANCE), Federal University of Rio Grande do Norte (UFRN), Natal, 59078-900, RN, Brazil*
[b] *Federal Institute of Rio Grande do Norte (IFRN), Natal, 59015-300, RN, Brazil*
[c] *Federal University of Rio Grande do Norte (UFRN), Natal, 59078-900, RN, Brazil*

Abstract

In the new 5G networking digital era, Network Slicing is pivotal in delivering new visionary applications (e.g., holographic calls, tactical Internet, immersive video, and expanded reality) as well as driving new business models and market opportunities. To pave the way for delivering new applications, the Quality of Network-Slice Service (QoNSS) must be maintained over time, which requires intelligent decision-making logic at the network edge for agility. In this paper, we propose a new Network Slice control-plane function that lies on predictive analysis using Machine Learning (ML) over throughput and packet loss rate Key Quality Indicators (KQIs) at slice-delivered service application data traffic granularity to guarantee QoNSS over time. The study brings the following significant contributions to the field of dynamic network traffic management within a network slicing framework: (*i*) assessing the performance of three popular machine learning models to predict QoNSS-aware data traffic patterns at slice-delivered applications granularity; (*ii*) designing an intelligent decision-making function capable of proactively fostering inter-slice offloading of slice-delivered applications flows in response to QoNSS degradation predictions; and, (*iii*) providing testing outcomes to offer a better understanding of the different mapping features displayed by each machine learning model. The experiments were conducted atop different testbed settings featuring real-world slice-defined network

*mfernandes@dca.ufrn.br

and computing infrastructure. Detailed mappings and statistical outcome analyses suggest that the proactive load-balancing function outperformed a regular rule-based approach by significantly reducing the total packet loss of the network system while allowing high accuracy and anticipation reaction time.

Keywords: Network Slicing, Load Balancing, Service Quality Control, Machine Learning, QoS
PACS: 0000, 1111
2000 MSC: 0000, 1111

---

1. Introduction

Slicing has become a pivotal focus and an essential part of 5G network ecosystem design due to its potential to drive new business models and market opportunities. Slicing networks [1] lay on an ecosystem of technologies that interwork to overlay multiple virtual network partitions (i.e., slices) atop the same physical network infrastructure. Network Function Virtualization (NFV) and Software-Defined Networking (SDN) [2] are among these enabling technologies. Each network slice is assumed to deliver particular services and applications, thus yielding different requirements holding proper characteristics, resources, policies, and others defined at the level of mobile users, applications verticals, customers, and other use-cases [3].

The network-slicing heterogeneity nature demands computing, storage, virtual services, and network resource to be configured in the supporting infrastructure in a monolithic manner to meet each particular instance's demands. A network slice instance comprises manageable constituent parts called slice parts, a service-delivering point within an end-to-end network slice structure. Network Slice parts can be provisioned to run on Commercial Off-The-Shelf (COTS) equipment distributed along on-path facilities spanning across backhaul, fronthaul, and Radio Access Network (RAN) infrastructure domains [4].

A slice part structure comprises service components, such as applications and network functions, operating in virtual mode for service provisioning, along with representations of physical objects like digital twins [5], which are mapped to physical resources. During the orchestration lifecycle of a network slice, all constituent slice parts are digitally associated to form Service Functions Chaining (SFC), which handle the data-plane traffic that the network slice yields systematically and under personalized and isolated use perspectives [6]. Hence, the ability to notice whether each network slice meets the performance and reliability requirements of slice part-running services and applications is denoted in this paper as QoNSS.

In order to yield predictable QoNSS levels, which must comply with data rate and packet loss settings established in Service-Level Agreements (SLAs), the network slicing control plane must include methods tailored for QoNSS-aware slice parts service delivery management and control operations. The transformation of networks into programmable, software-driven, service-based, and holistically managed architectures paved the way for the need for network slice's monolithic management and operations to be conducted autonomously, namely in terms of self-configuration, self-composition, self-monitoring, self-optimization, and self-elasticity. The Zero touch network and Service Management (ZSM)[1], states that "Full end-to-end automation of network and service management has become an urgent necessity for delivering services with agility and speed and ensuring the economic sustainability of the very diverse set of services offered by Digital Service Providers." Last but not least, the dynamic patterns that slice part-running services and applications yield a hot research area.

The state-of-the-art in the field of monolithic and autonomous network slicing service delivering control plane reveals that reactive adjusting operations, namely load balancing and elasticity, can keep network slices activated at a given performance level. However, reactive decision-making methods are unsuitable for stringent resource-demanding use cases by assuming that the temporary occurrence of service outages while employing relief functions is tolerable. This might be unacceptable for the 5G scenario since the 3GPP (3rd Generation Partnership Project) design foresaw fostering visionary applications, such as in mission-critical services (e.g., autonomous vehicles and smart factories), intense-bandwidth experience-enhanced multimedia services (e.g., holographic calls and immersive video), and ultra-reliable connected massive Internet of Things (IoT) deployments, to name a few currently incapable of delivering in market [7]. Considering the need to cater to the stringent requirements (highly intense bandwidth and ultra-low latency) that such visionary applications yield, we hypothesize that intelligent decision-making is best fitted to prevent actions over the anomalies that struggle to maintain QoNSS levels.

The ability to predict QoNSS degradation before it happens is essential to speed up network slice data traffic adjustment to guarantee QoNSS and optimize the resource utilization of the underlying network infrastructure as a whole. In this context, incorporating ML supported decision-making provides the prospect of deploying intelligent mechanisms that are not just prepared to solve complex problems, but also can enhance the understanding of complex situations, improve the accuracy of decision-making, and ultimately lead to more accurate outcomes,

[1]http://www.etsi.org/technologies/zero-touch-network-service-management

unsuitable, for instance, in regular rule-based solutions [8, 9, 10]. The complexity behind the problem of keeping QoNSS over time requires optimizing the network slice resources efficiently. To achieve this, the prediction of QoNSS levels will potentially promote Network Slice data plane proactive control without incurring undesirable service outages.

This paper fills the above-described gap by devising an intelligent decision-making scheme, which fosters a proactive inter-slice load-balancing function that offloads, at running time and in a QoNSS-driven manner, a given slice-delivered application service data traffic to another network slice instance best fitting the predicted QoNSS demand. The QoNSS predictions are driven by analyzing throughput and packet loss rate KQIs, at slice-delivered application granularity. The inter-slice load-balancing solution is designed on an agent-based architecture that can be provisioned at any part of the cloud continuum (core or edge facilities), and operates autonomously and in a monolithic Slice approach to promote seamless reactions [11]. In the scientific methodology applied, we first evaluated the capabilities that three fundamental ML models (namely, Artificial Neural Networks (ANN), k-Nearest Neighbors (kNN), and Support Vector Machines (SVM) take in predicting QoNSS degradation for employing intelligent inter-slice load balancing decisions [12, 13, 14, 15]. We present the outcomes of the training phase of the ML models, using Key Performance Indicators (KPIs), such as the coefficient of determination (R^2), to assess the accuracy of the models. The results reveal remarkable consistency and effectiveness of the models in predicting QoNSS degradation associated with active slice-part-running applications [16, 17, 18].

Subsequently, the trained ML models were rigorously tested, whilst the obtained results were compared with regular rule-based load-balancing decision-making algorithms, unequivocally suggesting the superiority of the ML-based models. This study offers an in-depth understanding of how ML-based models enhance slice-delivered applications' responsiveness and data traffic management capacities in slice-defined environments through detailed mappings and statistical analyses. The presented results provide a solid foundation for practically integrating these innovative techniques into next-generation communication infrastructures. The ML-based approaches, including ANN, kNN, and SVM, showcased significantly reduced cumulative packet loss rates compared to a regular rule-based strategy. Additionally, the ML models exhibited consistent and timely action anticipation in offloading the data traffic of affected slice-delivered applications to another suitable Network Slice instance. This indicates the effectiveness of ML-based mechanisms in proactively responding to predictions of slice-delivered application throughput and packet loss patterns while highlighting their potential in optimizing QoNSS over

time.

The present study makes several significant contributions to the field of dynamic network traffic management within a network slicing framework:

- Introducing and evaluating three ML models (ANN, kNN, and SVM) for predicting QoNSS levels of active slice-delivered applications, taking the predictive analysis over throughput and packet loss rates at slice-delivered applications granularity. The evaluation results strongly suggest that these models succeed with high accuracy, with R-squared values close to 1.0, indicating a substantial correlation with the data.

- Proposing an ML-based decision-making strategy tailored to the proactive QoNSS-aware inter-slice load-balancing problem. This solution keeps quality-guaranteed slice-delivered application data traffic over time while raising high agility at edge network premises.

- The research offers a better understanding of each ML model's different mapping features. It also shows the advantages and disadvantages behind a Network Slicing system when managing Slice-part network traffic autonomously.

The approach stands out from the current state of the art by offering several significant advantages. Firstly, it enhances the agility and accuracy of QoNSS predictions, which is crucial for maintaining service quality amidst dynamic network conditions. Secondly, through comprehensive experimental validation, the framework demonstrates its effectiveness in reducing packet loss and optimizing resource utilization—areas where many existing studies fall short. Lastly, the work addresses the lack of real-world applicability in prior models, providing a practical, scalable, and adaptive solution that significantly improves network management and QoNSS. By directly addressing these gaps, the research advances the theoretical understanding of network slicing and provides a robust, empirically validated framework for its application in real-world scenarios.

The paper's organization falls under the following structure. Section 2 outlines the study of the most relevant related works in application service control in slice-defined networks. Details about our solution are delineated in Section 3. Section 4 presents the methodology for developing the proposed solution and conducting evaluation experiments. Section 5 presents the suggested insights from the analysis with the experimental results collected during our solution evaluations. Finally, but not least, Section 6 wraps up our work and provides suggestions for future work.

2. Related Work Analysis

In recent years, significant research efforts have been devoted to enhancing the Quality of Service (QoS) in the context of 5G technology. This section provides an overview of the relevant literature in the domain of efficiently delivering network slice services, highlighting key contributions and approaches. The related work has been confronted on the basis of their supporting capabilities attempting to provision QoS-aided slice-delivered.

In [22], the Quality of Experience/Quality of Service (QoE/QoS) of 5G-enabled optical networks is studied, which focuses on the E2E (end-to-end) service delivery. An architecture of network slice provisioning with QoS guarantee was presented, supporting 5G service chaining in cross-domain optical networks. This setup facilitated linking 5G services through a policy-driven monitoring and control system to uphold the specified QoS standards for end-to-end network slices. Nevertheless, this system lacked a mechanism for SDN controllers and NFV entities to communicate and enforce QoS decisions on the occurrence of infrastructure topology events that affect the network slice. In [20], authors address the optimal allocation of a slice in 5G core networks by tackling two challenges, namely function isolation and guaranteeing end-to-end delay for a slice.

The authors of [21] elaborate on the necessity of automating network functions related to the design, construction, deployment, operation, control, and management of network slices. It revisits machine-learning techniques applicable to the automation of network functions and then presents an ML-based framework for the operation and control of network slices by continuously monitoring workload, performance, and resource utilization and dynamically adjusting the resources allocated to network slices. The authors of [19] thoroughly discuss the challenges that network slicing brings in the different network parts and design a cooperative game to study the potential cooperation aspects among the participants. Although these works have notorious relevance, they do not seek to validate the proposals through realistic use case studies. They limit themselves only to evaluating the performance of their proposals. In some cases, they are just architectural models.

In [23], a stateful backward recursive path procedure was used to maintain the E2E connection services. Experimental results indicated that this solution can support the automatic establishment of QoS-based E2E connections across multi-operator network domains. However, the orchestration scheme was not flexible enough to support the scalability of the advertisement for resources and dynamic connection services. A novel SliceNet framework is introduced in [24], based on advanced and customized network slicing, to address some challenges in the QoS for emergency service operators in migrating eHealth telemedicine services to 5G

networks. Experimental results empirically validate the prototyped enablers and demonstrate the applicability of the proposed framework. Although these solutions seek to do some experimental validation analysis, there is no integration between SDN controllers and NFV entities. Furthermore, they also do not consider the use of AI in decision-making.

In [25], the authors propose a novel QoS framework of network slice in 5G and beyond networks based on SDN and NFV to guarantee key QoS indicators. The performance evaluation was executed in the Mininet emulation tool, which shows that the proposed QoS framework can steer different flows into different queues of OpenVSwitch (OVS), schedule network resources for various network slice types, and provide reliable E2E QoS for users according to pre-configured QoS requirements. This work comes closest to ours but lacks a mechanism to automate traffic flow between services in a virtual network through SFC.

The paper presented in [27] introduces an innovative approach to enhance the efficiency of congestion control algorithms by employing Recurrent Neural Networks (RNNs) for bandwidth prediction. Diverging from prior studies that primarily relied on simulated data, the authors validate the accuracy of their predictions in a real-world environment, capturing and converting packet data for training their model. The results demonstrate a maximum correct response rate of 79.71%, showcasing the method's effectiveness in a real-world context and comparable performance to simulated data results. The paper also examines congestion control in TCP, exploring various algorithms that dynamically adjust data transmission in response to congestion levels. Additionally, it references a previous study that utilized RNNs to address transmission reduction in congestion control algorithms.

In [28], the authors present a significant contribution to 5G experimentation platforms by designing a scalable, flexible, and reliable analytics service supporting end-user experiments and verticals. It provides valuable insights into KPIs for network setup, ensuring correct operation and operational efficiency. Additionally, it demonstrates the application of the analytics tool in various environments and releases it as open-source in the Open5Genesis suite. The analytics component is developed as a set of microservices, promoting modularity and scalability. The analytics module incorporates descriptive, diagnostic, and predictive functionalities based on microservices and containerized modules, including standard machine learning, visualization, and reporting libraries. The paper also outlines a methodology for applying machine learning and artificial intelligence techniques in networks, covering data collection to model validation. While the paper showcases promising results, highlighting the effectiveness of the analytics tool in practical scenarios, there are limitations, such as the absence of a comprehensive

evaluation of the performance and scalability of the analytics service and the need to address privacy and security concerns associated with data collection and analysis. Furthermore, it would be beneficial to explore the limitations of the machine learning and artificial intelligence techniques employed and compare them with existing analytics solutions for 5G experimentation platforms to contextualize the proposed approach's advantages better.

The work presented in [29] investigates the application of diverse machine and deep learning techniques, including Decision Trees, Random Forest, Support Vector Machines, 1D Convolutional Neural Networks, Multi-Layer Perceptron, and k-nearest Neighbours, to enhance network prediction performance in 5G cellular networks. Introducing a bagging-based ensemble learning approach, which outperforms existing methods for network load prediction, emerges as a notable contribution. The study addresses load prediction and anomaly detection, emphasizing the importance of accurate anomaly detection in mitigating network load. Additionally, incorporating standardized 3GPP protocols enhances the realism of the dataset used. However, the paper lacks in-depth discussions on model limitations, dataset analysis, scalability considerations, resource constraints, and comparisons with state-of-the-art techniques, highlighting areas for potential improvement.

The paper [30] makes notable contributions to the SDN security field. It introduces two datasets generated using Mininet and Ryu controllers, comprising regular traffic and various attack types, for training supervised binary classification machine learning algorithms. The study evaluates multiple algorithms, with the decision tree algorithm standing out for achieving impressive scores, including an F1 score of 0.9995 for the attack class. However, it acknowledges limitations such as the need for a more comprehensive algorithm comparison, reliance on generated datasets, and potential scalability challenges in large-scale SDN environments. Nevertheless, the paper represents a significant advancement in applying machine learning techniques to bolster security in SDN infrastructures, paving the way for future research and refinement in this domain.

The articles address resource management and orchestration within 5G networks, focusing on dynamic network slicing for performance optimization. [32] proposes combining a tailored HARQ scheme with hierarchical resource scheduling for 5G network orchestration, emphasizing efficient resource utilization and responsive service delivery. [33] introduces dynamic single-tenant radio resource orchestration for eMBB traffic in multi-slice scenarios, aiming to ensure optimal service levels by monitoring performance indicators. Both emphasize the importance of inter-slice coordination for flexible and efficient resource allocation across diverse service demands in 5G networks.

In Table 1, we perform a comprehensive comparative analysis of the existing literature to evaluate the strengths and limitations of current solutions in comparison to our proposed approach.

Table 1: Comparison between existing solutions and our proposed solution.

References	Dynamic multi-path routing	Cooperation of multiple SDN controllers	Integration between SDN controllers and NFV entities	Service Function Chaining	NFV in Cloud	Experimental validation	Algorithms based on AI
Vincenzi et al. 2017 [19]					■		
Sattar and Matrawy 2019 [20]					■		
Kafle et al. 2019 [21]				■			■
Montero et al. 2019 [22]					■		
Sgambelluri et al. 2019 [23]		■				■	
Wang et al. 2019 [24]				■	■	■	
Shu and Taleb 2020 [25]	■	■	■		■	■	
Khan et al. 2021 [26]					■		■
Kazama et al. 2022 [27]						■	■
Aumayr et al. 2022 [28]					■	■	■
Haider et al. 2023 [29]					■	■	■
Alzahrani and Alenazi 2023 [30]					■	■	■
Maule et al. 2022 [32]			■	■			
Maule et al. 2021 [33]			■	■		■	
Our Solution	■	■	■	■	■	■	■

The findings from our comprehensive comparative analysis, as presented in Table 1, reveal that while numerous innovative contributions have been made, many research works have struggled to address the ever-expanding complexities and emerging challenges of effectively delivering network slice services. Overall, related works leverage standard functionalities for slice provisioning, such as dynamic multi-path routing, collaboration among multiple SDN controllers, integration between SDN controllers and NFV Management and Orchestration (MANO), SFC, experimental validation, and AI-based algorithms.

Therefore, we assert that our approach represents a significant advancement in dynamic network traffic management within network slicing structures, fostering the provisioning of QoS-guaranteed slice-delivered services over time. By providing detailed QoNSS prediction, customized decision-making strategies, optimized resource utilization, and insights into ML model mapping features, we foresee to pave the way for more efficient, adaptive, and resilient network operations in the era of 5G and beyond.

3. Description of the Proposed Solution

To ensure slice-delivered application data traffic with guaranteed quality over time, we designed a novel function that operates virtually on COTS data center equipment, which are distributed across a programmable and cloud-native network infrastructure. Our solution is centered around an agent-based virtual service function that incorporates predictive QoNSS-aware inter-slice load-balancing decision-making logic. This study emphasizes deploying the function on edge-premised equipment for high-agility perspectives. The functional requirements of our solution are outlined as follows:

- can run at either edge or core cloud computing premises;
- carries out ML-based predictive analysis on active slice-delivered application data flows in the packet loss and data rate patterns;
- predicts QoNSS levels of active slice-delivered applications at data traffic, QoS, and QoE levels;
- operates autonomously to offload slice-delivered data flows to another best-fitted network slice;
- operates autonomously to promote seamless and monolithic reactions.

Figure 1 depicts the functional architecture of the QoNSS-aware inter-slice load-balance solution, considering inner building blocks and internal/external communication interfaces. The main functionalities of the proposed solution are described as follows:

- Data Manager: The Data Manager building block is pivotal for orchestrating slice-delivered application data management functions, namely gathering, organizing, and storing all data in the State Table. Thus, it aims to ensure that all data is readily available for the Model Training building block. By centralizing this information, the Data Manager enables seamless access to processed data for further predictive analysis, foreseen to enhance the system's ability to leverage ML effectively, enabling accurate predictions and proactive load balancing based on real-time and historical data insights.
- Model Training: This component plays a crucial role in the functional architecture by acting as a foundation for creating a robust ML system to achieve optimal performance and predictive accuracy. Model training entails

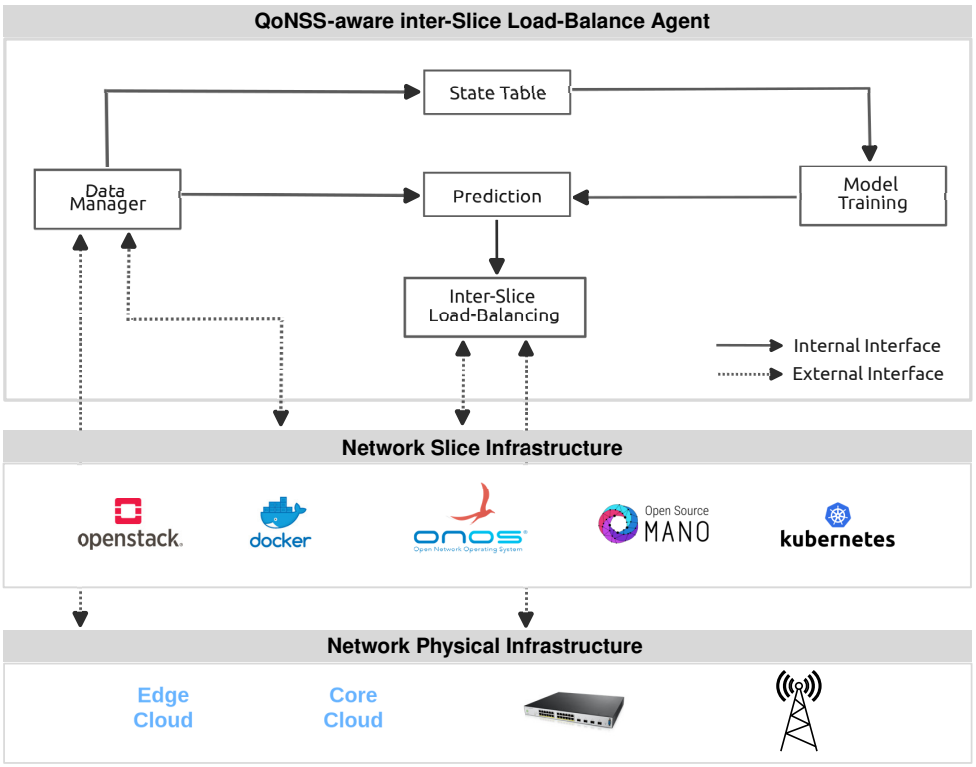


Figure 1: Architecture of the QoNSS-aware inter-Slice load-balance agent solution.

the logic tailored to undertake model training capabilities efficiently to handle the complexities of data pre-processing effectively and yield high-accurate predictions. The data pre-processing steps are integrated seamlessly to enhance data quality, thereby enabling the extraction of meaningful insights.

- Prediction: Distinct from the Model Training, the Prediction building block analyzes real-time data to make highly accurate predictions based on the ML model's learned patterns. Hence, it aims to ensure that the ML model can adapt effectively to dynamic load conditions, optimize resource allocation, and enhance overall load balancing performance by harnessing sophisticated ML algorithms and techniques. This architectural component is fundamental in validating the model's predictive capabilities and its ability to handle varying workloads efficiently in real-world scenarios.

- Inter-slice Load-Balancing: As the architecture's central decision-making point, the Inter-slice Load-Balancing building block utilizes predictions generated by

the system to determine the most suitable network slice for performing optimal load balancing. This component is responsible for dynamically configuring the infrastructure to effectively implement the predicted load-balancing adjustments. This building block completes the load-balancing lifecycle by laying the foundation for predictive traffic steering of flows delivered by affected network slices, aiming to afford optimal network performance and resource utilization while responding proactively to changing traffic conditions.

- State Table: The State Table component in the QoNSS-aware inter-slice load balancing functional architecture, plays a pivotal role in facilitating the decision-making processes by allowing real-time data on the status and performance metrics of each network slice to be maintained, gathered, and used by both the Data Manager and the Model Training building blocks. By leveraging the State Table, the system can dynamically allocate resources, adjust load balancing strategies, and ensure QoS requirements are met across different slices. Additionally, the State Table component interfaces with other modules within the architecture to enable seamless communication and coordination, contributing to the overall effectiveness and responsiveness of the inter-slice load balancing solution.

- Interfaces: Within our QoNSS-aware inter-slice load balancing functional architecture, internal interfaces serve as the communication channels that interconnect the system's building blocks. External interfaces, on the other hand, allow external systems to interact with the QoNSS solution. These interfaces adhere to RESTful principles to ensure standardized and efficient communication. The seamless operation of these interfaces is essential for facilitating QoNSS-aware inter-slice load balancing operations.

As Figure 1 illustrates, the interaction among the AI-based Virtual Network Functions (VNFs) is facilitated by well-defined interfaces structured according to REST API principles [RFC 9205]. These interfaces are systematically crafted to provide a streamlined mechanism for QoNSS-embedded services to communicate internally with architectural building blocks, as well as be accessed seamlessly by external entities. Furthermore, the interaction with the network slice and the physical network infrastructure elements plays a crucial role throughout the entire QoNSS system orchestration lifecycle. For example, the system orchestration functionalities leverage ONOS's advanced management, monitoring, and programmability capabilities designed to enhance network performance and efficiency. As a result, essential tasks such as keeping abreast of network resource

availability and enforcing slicing-defined flow rule control become more manageable. It is worth mentioning that ONOS has been widely adopted as the main control layer solution for 5G networks (i.e. SD-FABRIC[2]).

Figure 2 depicts a baseline scenario for applying the QoNSS-aware inter-slice load balancing solution, which incorporates network intelligence service provisioning operating virtually within edge computing nodes of the cloud infrastructure. This approach aims to improve traffic management in slice-enabled network environments by integrating NFV and ML technologies, particularly focusing on AI-supported VNFs. By strategically placing these AI-based VNFs at the edge premises, intelligent decisions can be made quickly and in real-time, analyzing numerous KQIs to optimize slice-delivered services for QoNSS guarantees. Through softwarization facilities, ML models can be generated and updated instantaneously, addressing challenges associated with data drift.

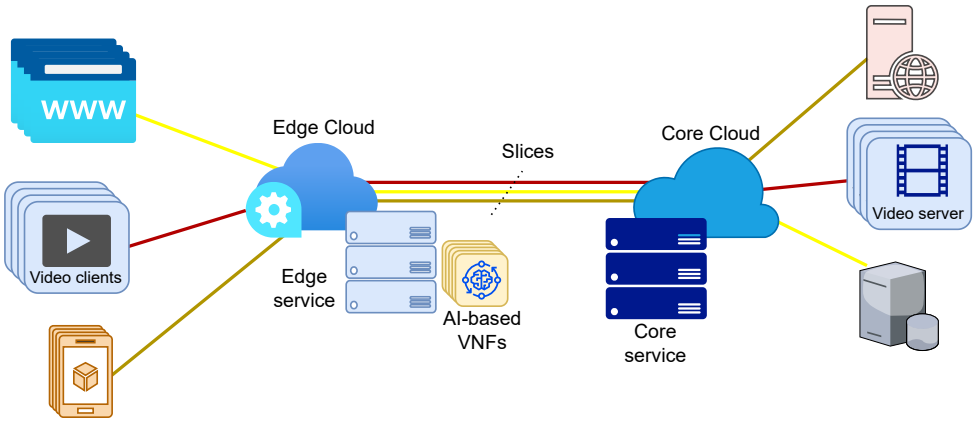


Figure 2: Operational framework of the QoNSS-aware inter-slice load-balance solution and edge network traffic management, integrating AI-based VNFs within the edge service in the edge cloud.

Although the applicability of our intelligent approach is not logically limited to particular parameters and can be extended to situations with multiple input parameters, we designed our predictive analysis based on throughput and packet loss rates, which are performed at the slice-delivered service application granularity. It is essential to stress that using intelligent decision-making processes provides a more comprehensive understanding of the challenges in guaranteeing QoNSS over time. This, in turn, leads to improvements in the patterns of slice-delivered services, ultimately enhancing the overall network performance.

[2]https://opennetworking.org/sd-fabric/

4. Methodology

This section provides a comprehensive overview of the scientific methodology applied to validate the efficacy of the proposed intelligent QoNSS-aware inter-slice load-balance function. The validation process adopts a two-step approach, starting with creating a comprehensive dataset that reflects the behavior of data flows that each slice-delivered application yields.

All the experiments were executed within an emulated testbed scenario, enclosing a core cloud and interconnected edge network parts, as illustrated in Figure 2. The experimental setup was built using an Ubuntu 22.04 virtual machine configured with eight vCPUS and 32GB of RAM, which was managed and hosted at an OpenStack premisse. The Containernet[3] tool emulated the network topology, thus providing connectivity to two pairs of Docker[4] containers attached as hosts. The first Docker pair serves as a video streaming server and client. In contrast, the second pair serves as a traffic generator (by employing the iPerf3[5] tool), used to consume slicing resources. The control plane management was performed by an ONOS SDN Controller operating as a WAN Infrastructure Manager (WIM), which was responsible for creating and managing slicing capabilities provisioned by the Virtual Private LAN Service (VPLS). SDN rules and applications were built following the OpenFlow v1.3 standard.

To create the dataset, we conducted experiments using the iPerf3 traffic generator tool, resulting in a dataset of 9,030 rows. This dataset serves as the foundation for the subsequent phase of the validation process, where we analyze the performance of the intelligent QoNSS-aware inter-slice load-balance function.

Employing a meticulously crafted dataset enables a thorough assessment of the proposed solution, guaranteeing the reliability and practical applicability of the outcomes in real-world contexts. By focusing on the throughput and packet loss rates KQIs at the slice-delivered service application granularity, we can assess the effectiveness of the intelligent decision-making process in improving the patterns of slice-delivered services and guaranteeing QoNSS over time.

Overall, the scientific methodology employed in this study provides a robust framework for evaluating the proposed intelligent QoNSS-aware inter-slice load-balance function and its potential impact on network performance and user experience. The second phase shifted the focus to ML models. Three distinct

[3]https://containernet.github.io/
[4]https://docker.com/
[5]http://iperf.fr

ML techniques (kNN, ANN and SVM) were carefully trained using the enhanced dataset. These models were designed to predict the cumulative load associated with the slice-delivered application data traffic, a critical factor in the decision-making process to keep QoNSS over time.

Subsequently, the model's predictive accuracy was performed in evaluation scenarios that were not encountered during its training phase. This process involved introducing novel and previously unseen load conditions to assess the model's ability to generalize and make accurate predictions beyond its training data. The models' performance, measured using suitable KQIs, demonstrated their ability to accurately predict QoNSS degradation (packet loss rates) allowed by new flow demands, thus allowing the proposed intelligent load-balancing solution to proceed with the inter-slice offloading. This comprehensive approach provides the basis for the paper's results. It illustrates the integration of advanced technologies in tackling network resource optimization and video streaming quality enhancement challenges.

4.1. Architecture

This research sets out two network slice instances on top of the same underlying communication infrastructure, s_{be} and s_2. The slices s_{be} and s_2 are set up to a bandwidth of the B_{be} and B_2 Mbps, respectively. The best-effort network slice, s_{be}, contains the primary data flow ϕ_p and a set of secondary data flows ϕ_s. The total data flow in s_{be} can be expressed by

$$\phi_{be} = \phi_p + \phi_s \tag{1}$$

where

$$\phi_s = \sum_{i=1}^{N} \phi_{s,i} \tag{2}$$

where the $\phi_{s,i}$ is the i-th flow, whilst N stands for the number of the flows associated with the secondary flow, ϕ_s. The ϕ_p is essential for providing the main video streaming service in HTTP Live Streaming (HLS) video flow and is carefully balanced with the ϕ_s within the same slice. An important part of the proposed approach is the ML-driven decision-making process, which can move the ϕ_p from s_{be} to the alternate slice s_2 based on predictive analysis. The flows ϕ_p and ϕ_s are distinguished by their bandwidths, B_p Mbps and B_s Mbps, respectively. The alternate slice s_2 flow is expressed as ϕ_2. The decision to move flows from one slice to another rather than expanding the slice's capabilities and experiencing performance issues was made for simplicity, but slice scheduling can be performed through the REST APIs provided by the control plane if necessary.

This slice guarantees a bandwidth of approximately B_p Mbps for the primary flow, which ensures that video quality is maintained within the specified range. In contrast, the composite secondary flow, ϕ_s, comprises various data streams that create a variable network demand. The bandwidth requirements of the secondary flow, estimated to be around B_s Mbps, are not limited by the allocated best-effort slice bandwidth (B_{be}). This necessitates using the intelligent agent, embedded within the ONOS-based[6] WIM-controlled environment, to predict the secondary flow's bandwidth consumption.

It is worth mentioning that the architectural model considered in this work is not limited to just two slices. The number of slices is managed by the ONOS controller, and this abstracts any complexity in managing slices from the control plane. The ONOS controller has been proposed by the Open Networking Foundation (ONF) with the support of many of the most important telecommunications vendors and operators. Specifically, ONOS is part of a broader array of initiatives promoted by the ONF, including SD-Fabric[7], which facilitates the operation of a 4G/5G mobile core User Plane Function (UPF) within the packet processing pipeline of switches. It is seamlessly linked with ONOS through the ONF's SD-Core project[8].

The intelligent agent, in turn, can then migrate the primary video flow, ϕ_p, to an alternate slice, s_2, to respond to changing traffic dynamics. This strategy ensures optimal network resource utilization and sustained video quality within the orchestrated network-slicing ecosystem. The process involves activating network slicing orchestration capabilities enforced by the Inter-slice Load-Balancing mechanism, which handles the necessary flow rules within network elements, ensuring continuous data flow transmission for the video streaming services. As a result of our self-managing network slicing orchestration approach, the video streaming service is delivered with the appropriated QoNSS without introducing any noticeable service interruption.

The flow ϕ_p is characterized by the video stream in HLS, which is the main element of the streaming service. On the other hand, ϕ_s is generated using iPerf3, a popular tool for measuring network performance. iPerf3 is set up to generate traffic with a bandwidth of B_s within the network slice s_{be}, simulating a range of secondary flows and their effect on the resource allocation of the network slice.

This work assumes a certain limit, B_l, and if the bandwidth of ϕ_s, B_s, surpasses this limit, the primary flow ϕ_p should be moved to the slice s_2, that is, the quality

[6]https://opennetworking.org/onos/

[7]https://docs.sd-fabric.org/master/index.html

[8]https://docs.sd-core.opennetworking.org/master/index.html

of the video, in flow ϕ_p, decreases if $B_s > B_l$. The intelligent agent predicts the B_s value by ML ($\tilde{B}_s$) and decides the slice change using B_l. At each time instant t, the ML takes the packet loss rate, $s_{\text{pl}}(t)$, and throughput, $s_{\text{th}}(t)$, KPIs from the secondary flow, ϕ_s, as input. These KPIs are essential metrics that provide the ML models with information about the quality and effectiveness of the network resources allocated to the flows within the slice.

Figure 3 illustrates the architecture scheme used to test the paper proposal. The decision rules use the B_l value and decide about primary flow, ϕ_p, changing. The ML output, $\tilde{B}_s$, is the predicted value of B_s. It is important to understand that the secondary flow, ϕ_s, is also affected by the primary flow, ϕ_p, since both initially share the same slice. Therefore, when the ϕ_s demands more bandwidth beyond the best-effort slice limit, the packet loss rate of the ϕ_s will also increase, similar to the loss of video quality. The flow ϕ_2 in the alternative slice, s_2, can be expressed as

$$\phi_2 = \begin{cases} 0 & \text{if } D(\tilde{B}) < 0 \\ \phi_p & \text{if } D(\tilde{B}) > 0 \end{cases} \tag{3}$$

where $D(\tilde{B})$ is the decision function used by the intelligent agent. The $D(\tilde{B})$ can be characterized by

$$D(\tilde{B}_s) = \begin{cases} +1 & \text{if } \tilde{B}_s >= B_l \\ -1 & \text{otherwise} \end{cases}. \tag{4}$$

4.2. Dataset generation

Developing a reliable and representative dataset is key to verifying the proposed approach. Several experiments were conducted in the network slice environment to create this dataset, which encompasses a wide range of load conditions, reflecting the diversity of real-world network traffic. This variety of data is essential for providing ML models with a broad range of scenarios, allowing them to recognize the intricate connections between KPIs and flow loads. To this end, we used real media with H.264 encoding, 1920×1080 pixels resolution, bitrate of 3 Mbps, frame rate of 60 frames per second, and duration of 30 seconds.

Experiments that lasted one minute each were performed to promote the diversity of network dynamics. IPerf was employed to generate traffic load to degrade network QoS and, consequently, the video streaming service QoE. To this end, the traffic load parameters were set up with gradual modifications (0.2 Mbps increments) so that the dataset could include knowledge about conditions ranging from situations (from 0.6 Mbps to 3.4 Mbps) that were least harmful to QoE to highly compromising ones

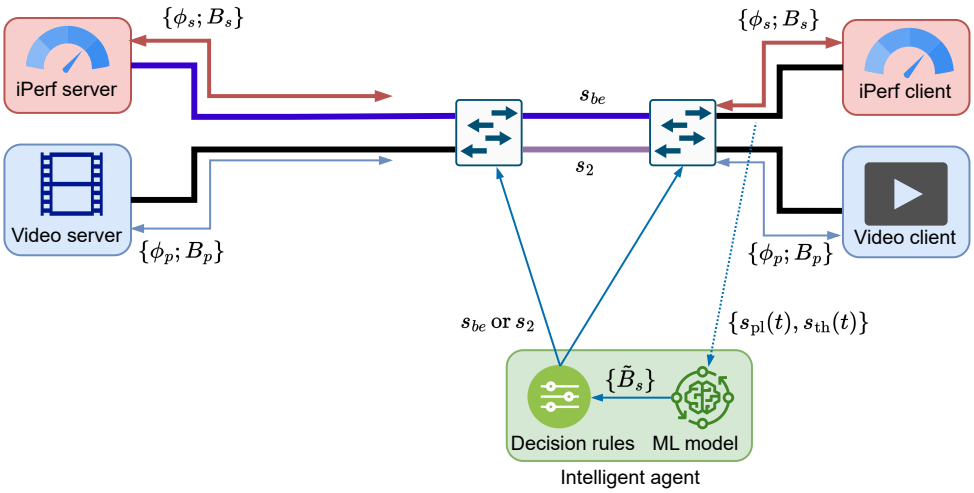


Figure 3: Architectural scheme of the implemented testbed for Proposal evaluation: Integration of primary and secondary flows via ML for network resource optimization.

(given the slice capabilities and the media bitrate requirements). In other words, it means an amount of traffic that, when jointly added to the best-effort slice during the video streaming transmission, could cause impairment to the streaming service.

For each k-th load configuration, $s_{\text{pl},k}(t)$ and $s_{\text{th},k}(t)$ KPIs were collected, at each time instant t, from slice s_{be}, including ϕ_p and ϕ_s flows. This thorough data collection process resulted in a dataset of 9,030 distinct instances, each with its associated KPI values and load levels. In each k-th load experiment, a secondary flow $\phi_{s,k}$ is generated with a bandwidth $B_{s,k}$ Mbps, where $B_{s,k} \in \{0.6, 0.8, 1.0, \ldots, 3.2, 3.4\}$ for $k = 1, \ldots, K$, where K represents the number of experiments. In total, $K = 14$ load experiments were conducted. Figure 4 illustrates the scheme employed for dataset generation.

The resulting dataset serves as the foundation for the next step, where the ML models are trained and tested, thus equipping the intelligent agent with the ability to make informed decisions about flow management to improve video streaming quality.

4.3. ML Training Methodology

Three ML models performed regression tasks during the training phase, uncovering patterns in the collected data. These models were ANN, kNN, and SVM. The training dataset included KPI values and their corresponding load levels, which the models were adjusted to identify the connections between the input KPIs and the flow loads. Iterative optimization strategies were used to refine the models and

reduce the prediction errors using the data in the training dataset. This process of refinement and cross-validation enabled the models to accurately predict the load of secondary flows based on real-time KPI measurements. These trained ML models form the basis of the intelligent agent's decision-making, allowing it to assess changing network conditions and make decisions that optimize the quality of the primary video streaming service. Table 2 presents the hyperparameters used to train the ML models.

Table 2: Hyperparameters used for each ML model.

Model	Hyperparameters
ANN	Hidden Layers: 2 (20, 10) Activation Function: ReLU Learning Rate: 0.001 Epsilon: 1e-8 Optimizer: Adam
kNN	Distance Function: Euclidean Number of Neighbors: 3 Neighbor Weights: Uniform Leaf Size: 10
SVM	Kernel: RBF Regularization Parameter (C): 1.0 Kernel Degree: 3 Epsilon: 0.1 Gamma: Auto

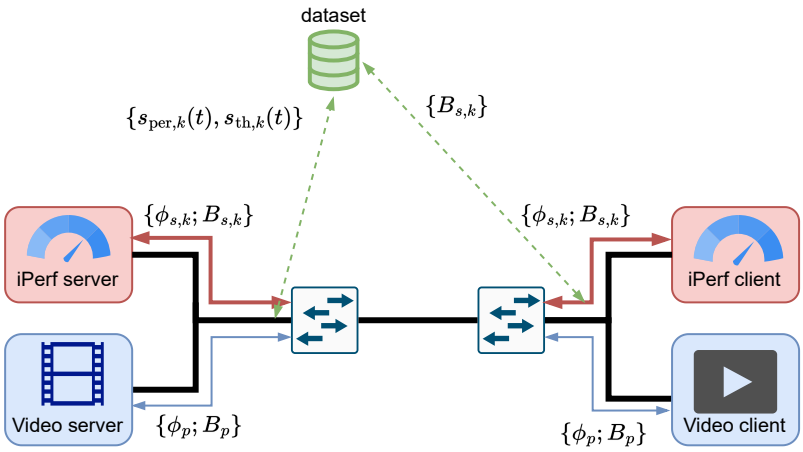


Figure 4: Experimental scheme for sataset generation: Collecting $s_{\text{pl},k}(t)$ and $s_{\text{th},k}(t)$ KPIs from s_{be} slice across $K = 14$ load configurations.

During the training phase, a k-fold cross-validation approach was employed for each machine learning model, with k set to 10. This methodology is a commonly used practice for assessing the performance of ML models and ensuring robust evaluation. This approach divides the dataset into ten (10) equally sized subsets, or folds. The model is trained and evaluated 10 times, each using a different fold as the validation set and the remaining nine as the training set.

k-fold cross-validation is a beneficial technique for assessing and selecting models in machine learning, as it helps to reduce the risk of overfitting. It also provides a more comprehensive evaluation of the model's performance, as it is tested on multiple subsets of data. This ensures that the model's performance metrics are more accurate and reflect its ability to generalize. Ultimately, k-fold cross-validation is a valuable tool for model evaluation and selection in machine learning.

4.4. ML Test Methodology

Following the training phase, tests were conducted to evaluate the models and compare their performance to the traditional approach, in which a decision system decides when to switch the primary flow, ϕ_p, to another slice based on the throughput of the secondary flow, ϕ_s. Since the secondary flow, ϕ_s, is the variable component of the system, it is essential. On the other hand, the ML proposed in this work considers the throughput of the secondary flow, ϕ_s, and the associated packet loss rate. This is noteworthy as ML can anticipate the need for a flow switch by considering an extra variable.

Five separate experiments were conducted to evaluate the performance of the tests, with the traffic load on iPerf ranging from 1.6 Mbps to 3.6 Mbps in 0.1 Mbps increments. Each increment lasted for 60 seconds, resulting in 21 minutes (1260 seconds) for each experiment. In this evaluation, the architectural parameters were set to $B_p = 3$ Mbps and $B_{be} = 5$ Mbps. The outputs of the three ML models were recorded for each experiment, allowing for a comparison between the conventional agent and ML-based approaches and providing insight into their effectiveness.

The conventional agent was characterized as a decision system that evaluates both the packet loss rate, $s_{\mathrm{pl}}(t)$, and the throughput, $s_{\mathrm{th}}(t)$, of the secondary flow, ϕ_s, at each time instant t. Based on predefined threshold values, this system determines whether to move the primary flow, ϕ_p, to the alternative slice, s_2. Tests were conducted for two decision rules, called R_1 and R_2. The rule R_1 is based solely on the throughput of the secondary flow, and the R_2 is based on both metrics, i.e., the packet loss rate and the throughput of the secondary flow. The decision function

associated with R_1 rule can be expressed by

$$D_1(s_{\text{th}}(t)) = \begin{cases} +1 & \text{if } s_{\text{th}}(t) >= B_l \\ -1 & \text{otherwise} \end{cases} \tag{5}$$

and the decision function associated with R_2 rule is expressed as

$$D_2(s_{\text{pl}}(t), s_{\text{th}}(t)) = \begin{cases} +1 & \text{if } s_{\text{th}}(t) >= B_l \text{ AND } s_{\text{pl}}(t) >= P_l \\ -1 & \text{otherwise} \end{cases}. \tag{6}$$

The flow ϕ_2 in the alternative slice, s_2, can be expressed as

$$\phi_2 = \begin{cases} 0 & \text{if } D_1(s_{\text{th}}(t)) < 0 \\ \phi_p & \text{if } D_1(s_{\text{th}}(t)) > 0 \end{cases} \tag{7}$$

and

$$\phi_2 = \begin{cases} 0 & \text{if } D_2(s_{\text{pl}}(t), s_{\text{th}}(t)) < 0 \\ \phi_p & \text{if } D_2(s_{\text{pl}}(t), s_{\text{th}}(t)) > 0 \end{cases} \tag{8}$$

for R_1 and R_2, respectively.

Figure 5 illustrates the conventional agent in the test scheme. These experiments aim to compare the effectiveness of the conventional approach against the proposed ML-based strategy, considering different decision criteria.

5. Assessment Results

The results were conducted in two stages. Initially, the outcomes related to the training of the ML techniques described in the methodology section will be presented. Subsequently, the test results of the trained models will be discussed.

5.1. ML Training Results

The effectiveness of the ML models in forecasting bandwidth utilization of composite secondary flows in a network slicing environment was evaluated using R-squared (R^2) as a metric. R^2 gauges the accuracy of the models to the data, with values closer to 1.0 indicating a strong correlation.

As shown in Table 3, the results for each ML model are as follows:

- **ANN** achieved a mean R^2 of approximately 0.9945 and a low standard deviation of 0.0015, highlighting a strong and consistent fit to the dataset.

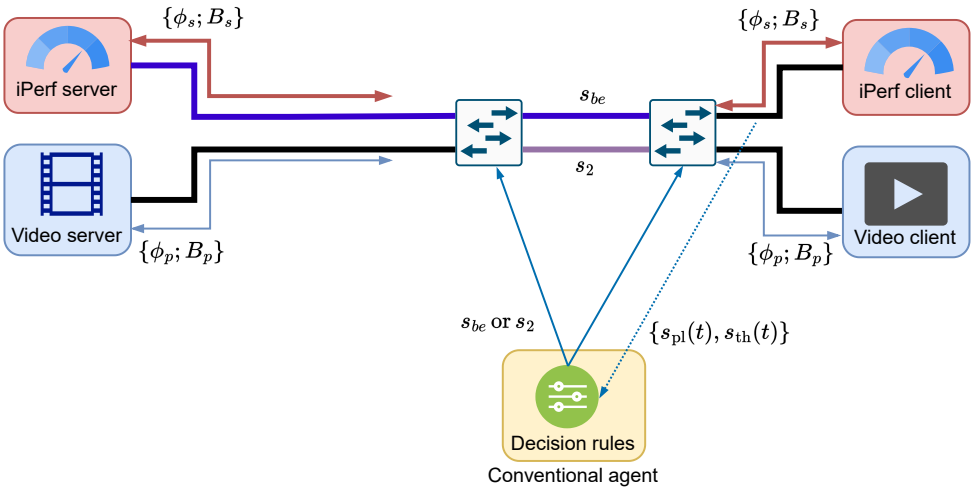


Figure 5: Conventional agent test scheme: Comparing the conventional approach against the ML-based strategy using diverse decision criteria.

- **kNN** also exhibited exceptional performance with a mean R^2 value of around 0.9951 and a standard deviation of 0.0018, showcasing an exceptional and consistent fit to the data.
- **SVM** demonstrated a mean R^2 value of approximately 0.9940 with a standard deviation of 0.0015, indicating a robust and consistent alignment with the observed data. This performance reflects minimal variability, positioning the SVM model closely alongside the other models presented.

After analyzing the mean and standard deviation of the R^2 for three techniques namely ANN, kNN, and SVM, it was observed that the differences in the R^2 means were insignificant. Additionally, the standard deviations for ANN and SVM were the same, with only a slightly higher value for kNN. From a mathematical perspective, these minor differences do not provide sufficient evidence to declare any technique as superior to the others. However, these findings underscore the effectiveness of these models in supporting dynamic network traffic management within a network slicing framework.

Table 4 displays extra performance metrics for the most successful folds of each machine learning model. These metrics are essential for evaluating the models' capacity to forecast bandwidth utilization in a network-slicing setting. The metrics taken into account include Mean Squared Error (MSE), Root Mean Squared Error (RMSE), and the Coefficient of Determination (R^2 Score).

Table 3: Mean R-squared (R^2) and Standard Deviation (SD) for ML each model used in this work.

Model	Mean of R^2	SD
ANN	0.9945	0.0015
kNN	0.9951	0.0018
SVM	0.9940	0.0015

Table 4: Performance metrics for best folds of ML models.

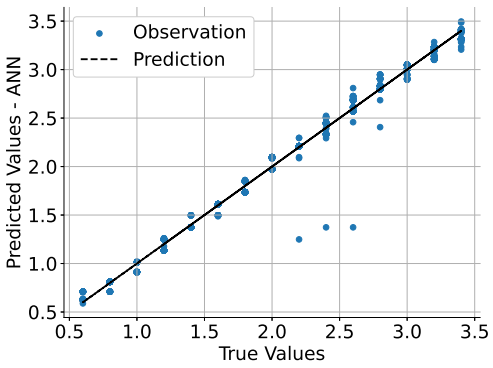
Model	MSE	RMSE	R^2 Score
ANN	0.0047	0.0683	0.9937
kNN	0.0028	0.0530	0.9962
SVM	0.0046	0.0678	0.9938

For the ANN model, it showcased an MSE of 0.0047, an RMSE of 0.0683, and an R^2 Score of 0.9937 in its best fold. These metrics reflect the model's proficiency, where the lower MSE and RMSE values indicate minimal prediction errors. At the same time, the high R^2 Score signifies the model's capacity to effectively explain variations in the data. In the other way, the kNN model demonstrated remarkable results, with an MSE of 0.0028, an RMSE of 0.0530, and an R^2 Score of 0.9962 in its best fold. These metrics demonstrate the model's impressive predictive capabilities, with low MSE and RMSE values indicating minimal prediction errors and a high R^2 Score indicating the model's remarkable capacity to explain data variability. Lastly, the SVM model revealed an MSE of 0.0046, an RMSE of 0.0678, and an R^2 Score of 0.9938 in its best fold. These results align closely with the ANN model and demonstrate SVM's remarkable performance in terms of prediction accuracy and fitting to the data.

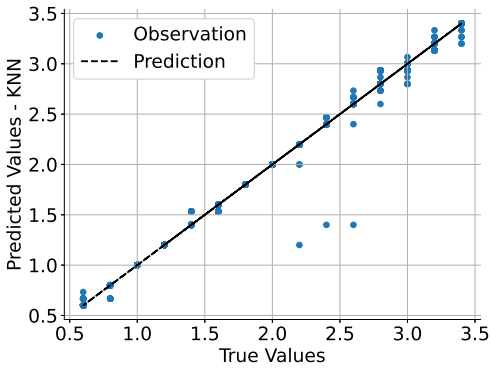
The ML models demonstrated remarkable accuracy in forecasting bandwidth utilization in a network-slicing environment. The kNN model was particularly impressive, with the lowest MSE and RMSE values. All models achieved R^2 Scores close to 1.0, indicating their remarkable capacity to explain data variations. These findings confirm the efficacy of these models in enabling dynamic network traffic management within a network-slicing framework.

The plots in Figure 6 demonstrate the accuracy of the three ML models in predicting bandwidth utilization. The x-axis of each graph shows the model's predictions, while the y-axis displays the actual values from the dataset. The diagonal line in the center of each plot is the estimated regression line, and the data points close to this line indicate the model's accuracy. All three models have data points tightly clustered around the regression line, indicating their solid predictive capabilities. This visual evidence confirms the effectiveness of the ANN, kNN, and SVM models

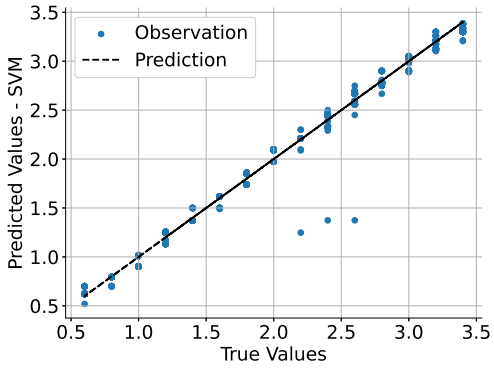
in predicting network traffic in network-slicing environments.



(a) ANN regression model.



(b) kNN regression model.

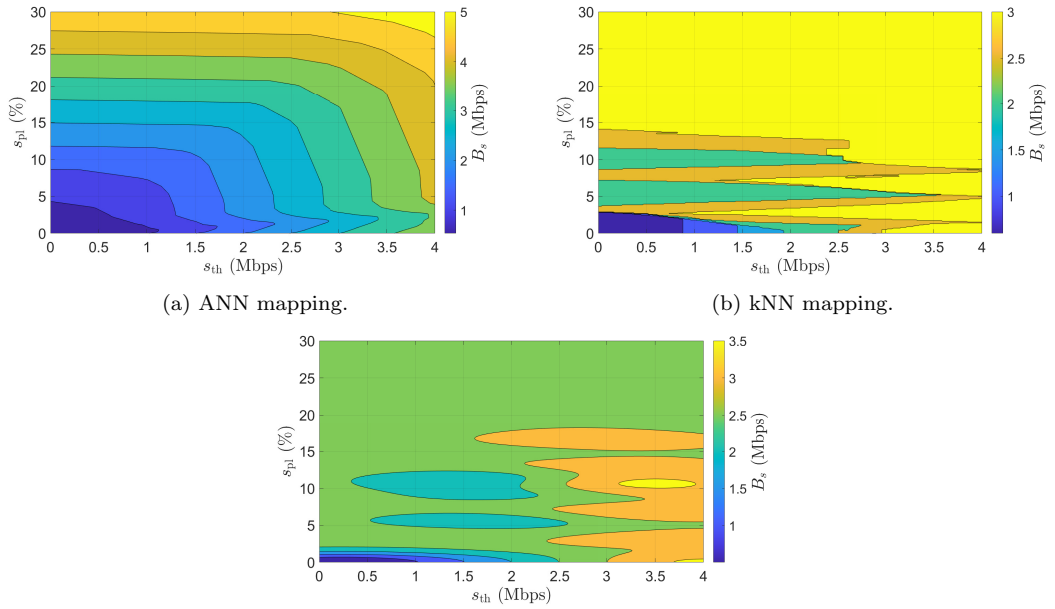


(c) SVM regression model.

Figure 6: Predicted values.

Figures 7a, 7b, and 7c show the mapping done by the ML techniques at different throughput and packet loss rate points. These maps visually demonstrate the connection between the two key metrics. The maps created by each technique (ANN, kNN, and SVM) have distinct patterns and decision boundaries. These visualizations provide useful information on how each ML model categorizes different parts of the feature space, helping to comprehend their respective advantages and drawbacks in this context. The mappings generated by the ML models reveal distinct approaches to interpreting the relationship between throughput and packet loss rate. The ANN model utilizes its layered structure to capture intricate, non-linear associations, leading to consistent, continuous mappings. On the other hand, kNN, being an instance-based approach, produces mappings characterized by localized decision regions, making it vulnerable to local changes in the feature space. Focusing

on finding optimal hyperplanes, SVM generates mappings with distinct decision boundaries, providing a more organized and separable view of the data. These distinctions in mapping characteristics come from the unique techniques underlying each ML technique, emphasizing the significance of selecting the most appropriate model for a given problem domain.



(a) ANN mapping.

(b) kNN mapping.

(c) SVM mapping.

Figure 7: Comparison of throughput and packet loss rate mappings by different machine learning models.

5.2. ML Test Results

Figures 8 and 9 depict the measured throughput values and cumulative packet loss rate over the entire simulation without using any decision scheme for primary flow switching. These results validate the experiment, as they demonstrate that as the bandwidth of the secondary flow increases, the cumulative packet loss rate and measured throughput also rise. Figure 10 illustrates the optimal decision points for switching the primary flow to slice s_2. It is observed that before 2 Mbps, the packet loss rate begins to escalate, emphasizing the importance of the decision system anticipating this change. However, conventional decision systems rely solely on a priori knowledge regarding the maximum bandwidth limit, (B_l), that the secondary flow, ϕ_s, can achieve, which is determined by measuring the throughput of the

secondary flow as described in Equations (5) and (6) for decision rules R_1 and R_2, respectively.

Thus, it is already evident that conventional systems will exhibit high packet losses before switching the primary flow to slice s_2. On the other hand, as will be shown in the forthcoming results, the ML-based intelligent agent will proactively respond to packet loss, initiating the switch just before the bandwidth limit, B_l, resulting in superior performance in terms of packet loss compared to conventional techniques. For the tested experiment, the value is approximately $B_l = 2$ Mbps, considering that the best-effort slice, s_{be} has a bandwidth of $B_{\text{be}} = 5$ Mbps, and the video associated with the primary flow, ϕ_p, has a bandwidth of $B_p = 3$ Mbps.

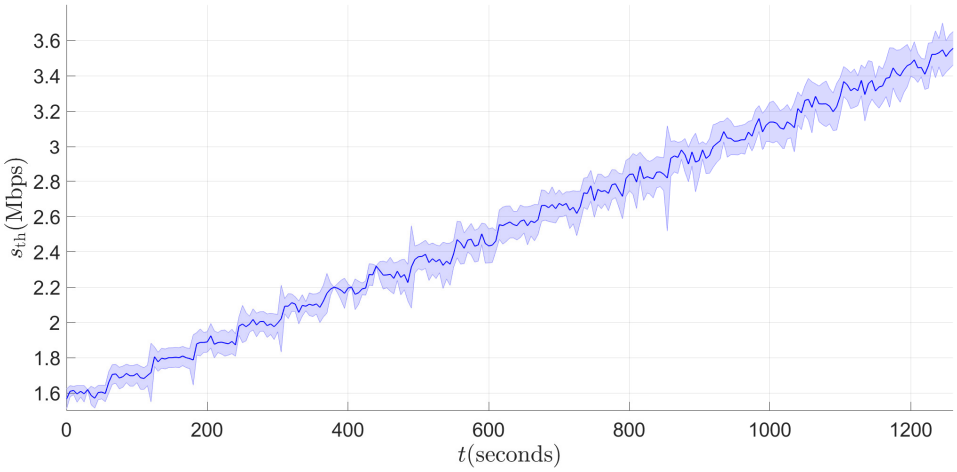


Figure 8: The measured throughput associated with the 5 test experiments conducted. The graph depicts the average throughput value every 5 seconds.

Figure 11 presents each decision strategy's cumulative packet loss rate distribution until the switch time to slice s_2. It can be observed that the ML-based strategies exhibited lower cumulative loss compared to conventional decision strategies based on rules R_1 and R_2. This is attributed to the anticipatory switching performed by the ML model. Figures 12 and 13 display the distribution of anticipation time for ML-based decision techniques about the conventional decision based on R_1 and R_2, respectively. It is crucial to emphasize that the ML prediction criterion was based on training from the previously generated data, as previously outlined. Tables 5, 6 and, 7 summarize the statistical results of the distributions presented in Figures 11, 12, and 13.

The R_1 strategy had an average cumulative loss rate of approximately 0.6011%,

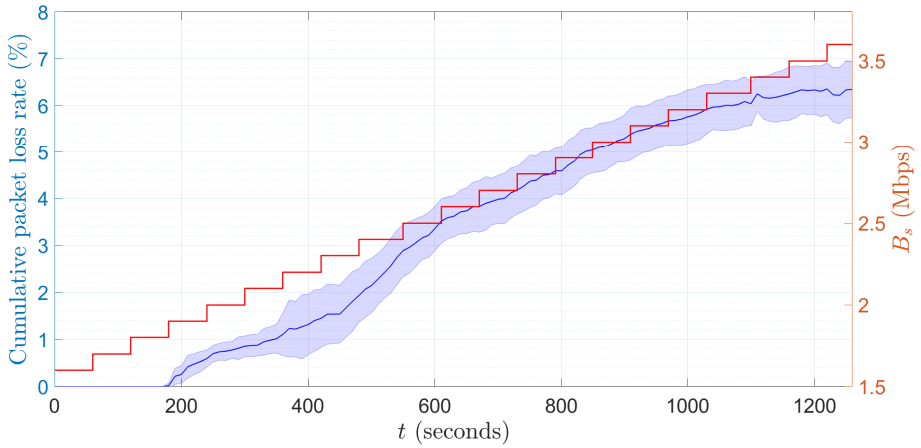


Figure 9: The cumulative packet loss rate and the bandwidth of the secondary flow associated with the 5 experiments without using any decision system for switching the ϕ_p.

Table 5: Statistical results of each decision strategy's cumulative packet loss rate until switching to slice s_2, associated with all 5 experiments using $B_l = 2$ Mbps and $P_l = 2\%$.

Decision	Mean (%)	Standard deviation (%)	Median (%)
R_1	0.6011	0.2092	0.6527
R_2	0.4678	0.5777	1.0190
ANN	0.0917	0.0897	0.0787
kNN	0.1269	0.0866	0.0907
SVM	0.0287	0.0092	0.0287

with a standard deviation of 0.2092% and a median of 0.6527%. In contrast, the R_2 strategy had an average cumulative loss rate of about 0.4678%, with a much higher standard deviation of 0.5777%, and a median of 1.0190%, indicating a considerable variation in performance. On the other hand, the machine learning approaches demonstrated significantly better performance. The ANN strategy achieved an average cumulative loss rate of only 0.0917%, with a standard deviation of 0.0897% and a median of 0.0787%. The kNN strategy recorded an average rate of 0.1269%, with a standard deviation of 0.0866% and a median of 0.0907%, while the SVM strategy had the lowest average cumulative loss rate, at just 0.0287%, with a standard deviation of 0.0092% and a median of 0.0287%.

The ML techniques displayed a consistent performance in anticipation time for the R_1-based strategy. The ANN approach had an average anticipation time of around 52.6 seconds, with a standard deviation of 14.9 seconds and a median of 58.6 seconds. Similarly, the kNN strategy had an average anticipation time of approximately 52.6

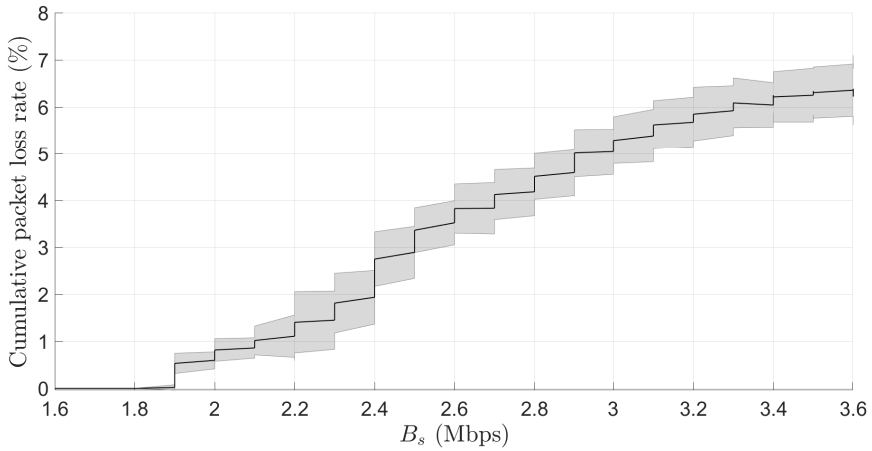


Figure 10: The cumulative packet loss rate plotted against the bandwidth of the secondary flow associated with the 5 experiments without using any decision system for switching the ϕ_p.

Table 6: Statistical results of the anticipation time for the ML-based decision strategies compared to the conventional R_1-based strategy. The distribution was based on the five experiments using $B_l = 2$ Mbps.

ML-based decision	Mean (seconds)	Standard deviation (seconds)	Median (seconds)
ANN	52.5932	14.8756	58.6454
kNN	52.6471	16.3828	60.0358
SVM	52.6437	14.2399	57.9705

seconds, with a standard deviation of 16.4 seconds and a median of 60.0 seconds. The SVM strategy showed an average anticipation time of roughly 52.6 seconds, with a standard deviation of 14.2 seconds and a median of 58.0 seconds. These results suggest that the ML-based strategies provide consistent anticipation times when compared to the traditional R_1-based strategy.

The R_2-based strategy was found to have a consistent anticipation time, with an average of 199.8 seconds, a standard deviation of 48.5 seconds, and a median of 213.1 seconds. Similarly, the ML techniques also demonstrated similar consistency in anticipation time, with the ANN, kNN, and SVM strategies having an average anticipation time of 199.8 seconds, a standard deviation of 48.5 - 49.5 seconds, and a median of 213.1 - 213.8 seconds. These results suggest that the ML-based strategies are comparable to the R_2-based strategy regarding anticipation time.

The use of machine learning approaches is effective in making more precise decisions than traditional rule-based decision strategies. This has resulted in a significant decrease in the total packet loss rate, which is essential for improving the

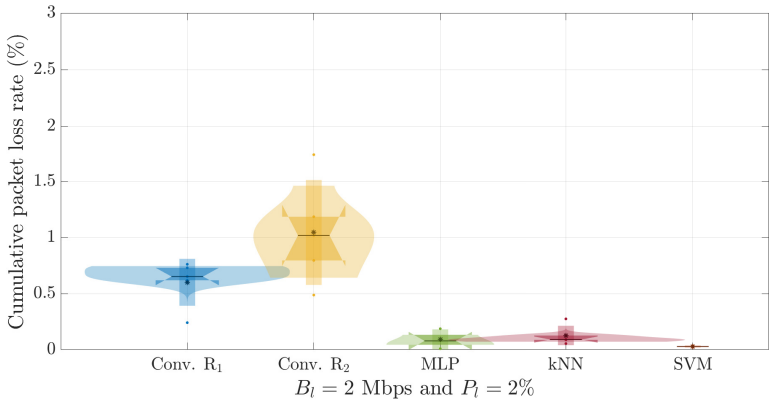


Figure 11: Distribution of the cumulative packet loss rate for each decision strategy until switching to slice s_2, associated with all 5 experiments using $B_l = 2$ Mbps and $P_l = 2\%$.

Table 7: Statistical results of the anticipation time for the ML-based decision strategies compared to the conventional R_2-based strategy. The distribution was based on the five experiments using $B_l = 2$ Mbps and $P_l = 2\%$.

ML-based decision	Mean (seconds)	Standard deviation (seconds)	Median (seconds)
ANN	199.756	48.4610	213.078
kNN	199.810	49.4990	213.789
SVM	199.807	48.6130	213.096

dependability and efficiency of networks in changing and ever-evolving conditions.

The SVM model has the lowest cumulative packet loss rate at 0.0287%, followed by the ANN model at 0.0917% and the kNN model at 0.1269%. The conventional rule-based strategies R_1 and R_2 have notably higher loss rates, indicating the superiority of ML-driven decision-making. All three ML models (ANN, kNN, SVM) have similar anticipation times, ranging from 52.6 to 199.8 seconds, demonstrating consistent and timely decision-making.

In conclusion, the results of this study demonstrate that ML models are more effective than rule-based strategies in terms of both cumulative packet loss rate and anticipation time. The SVM model had the lowest loss rate, and all ML models showed reliable and consistent anticipation times. These results highlight the usefulness of ML-driven approaches in dynamic network environments.

Figure 14 demonstrates the mapping done by traditional decision makers based on the rules (Equations 5 and 6) and those based on Machine Learning (intelligent agent). The mapping shows when each decision maker changes the primary flow, ϕ_p, from the best-effort slice, s_{be}, to the alternate slice, s_2. ML techniques create a

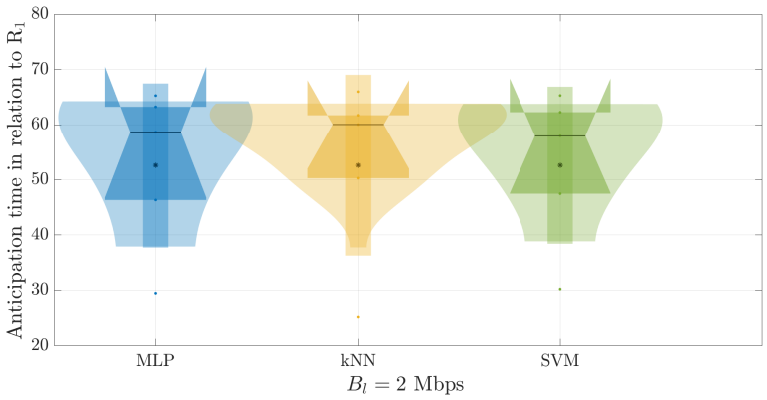


Figure 12: Distribution of the anticipation time for the ML-based decision strategies compared to the conventional R_1-based strategy. The distribution was based on the 5 experiments using $B_l = 2$ Mbps.

non-linear mapping based on the training data (Figures 14c, 14d, and 14e), while the conventional decision maker creates a linear mapping as seen in 14a and 14b. From the mappings, it can be concluded that ML has learned the behavior of the tested flows and can generate a more personalized and appropriate result for the flow switch between the slices, in this case, the shift of the primary flow, ϕ_p, from the best-effort slice, s_{be}, to the alternate slice, s_2.

6. Conclusions and Future Work

The findings of this study demonstrate the potential of machine learning-based techniques in dynamic network settings, highlighting their capacity to improve network traffic management and resource allocation. Using these models, network administrators can create more adaptive and responsive network slicing configurations, improving user experiences and operational effectiveness.

Exploring the potential of more sophisticated machine learning techniques and ensemble methods could further enhance the accuracy and speed of the intelligent decision-making agent. Additionally, adapting the machine learning-based system for real-time use is a key next step, given the time constraints of network operations. Finally, broadening the analysis to include more network performance metrics and security considerations will help to create a more comprehensive understanding of the effectiveness of network slicing.

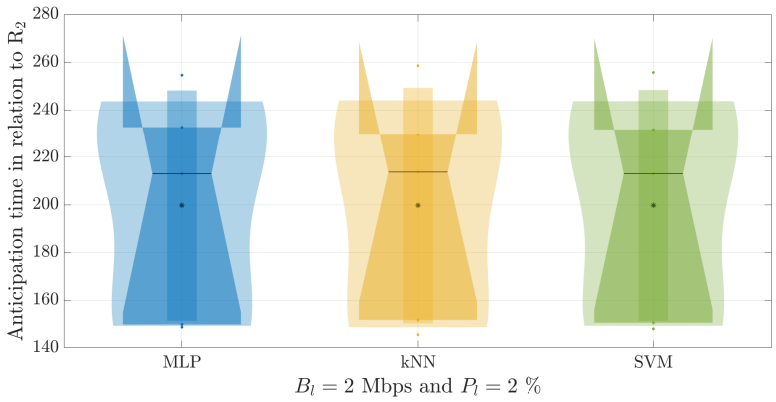


Figure 13: Distribution of the anticipation time for the ML-based decision strategies compared to the conventional R_2-based strategy. The distribution was based on the five experiments using $B_l = 2$ Mbps and $P_l = 2\%$.

Acknowledgment

This research was partially funded by Lenovo, as part of its R&D investment under Brazilian Informatics Law, and by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - Finance Code 001.

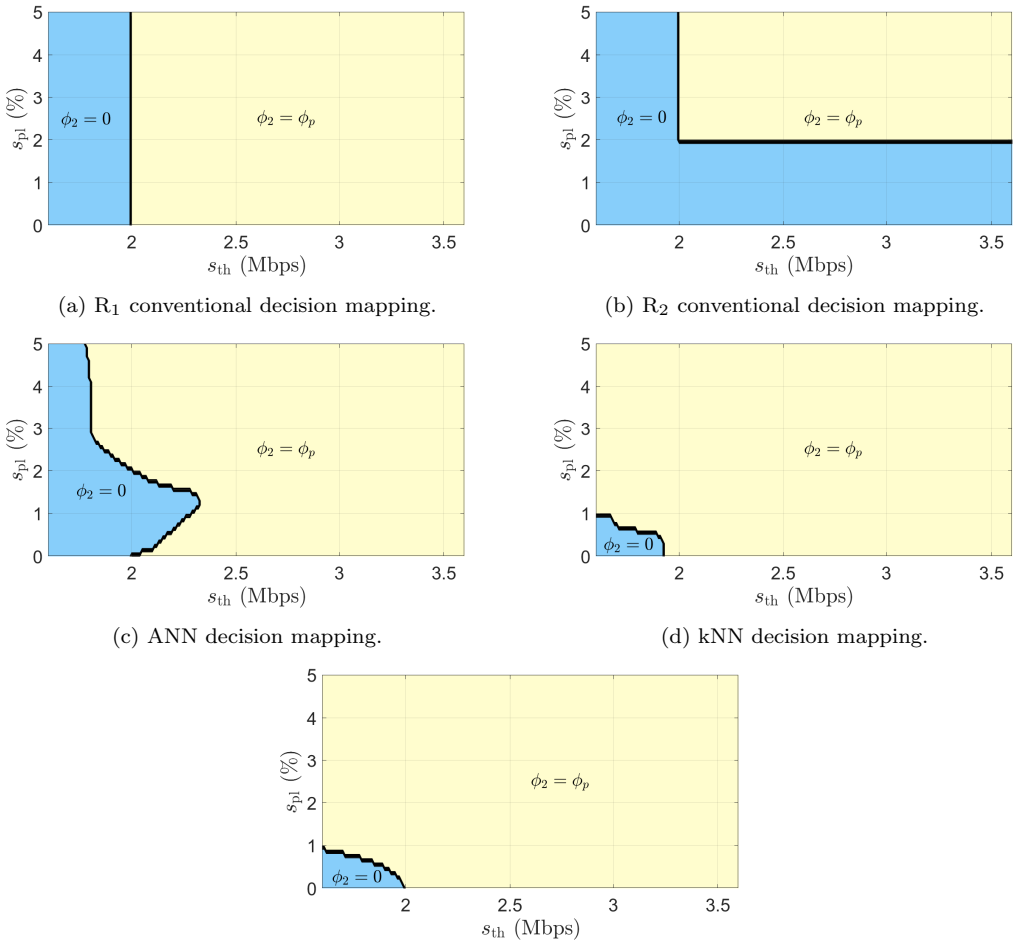
References

[1] Ibrahim Afolabi et al. Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. IEEE Communications Surveys Tutorials, 20(3):2429–2453, 2018. doi: 10.1109/COMST.2018.2815638.

[2] Evangelos Haleplidis, Kostas Pentikousis, Spyros Denazis, Jamal Hadi Salim, David Meyer, and Odysseas Koufopavlou. Software-Defined Networking (SDN): Layers and Architecture Terminology. RFC 7426, January 2015. URL `https://www.rfc-editor.org/info/rfc7426`.

[3] Felipe S. Dantas Silva et al. Network slicing mobility aware control to assist handover decisions on e-health 5g use cases. In 2022 International Wireless Communications and Mobile Computing (IWCMC), pages 1034–1039, 2022. doi: 10.1109/IWCMC55113.2022.9825010.

[4] Felipe S. Dantas Silva et al. Necos project: Towards lightweight slicing of cloud federated infrastructures. In 2018 4th IEEE Conference on Network



(a) R_1 conventional decision mapping.

(b) R_2 conventional decision mapping.

(c) ANN decision mapping.

(d) kNN decision mapping.

(e) SVM decision mapping.

Figure 14: Decision Mapping for All Decision Agents: Conventional and ML-Based. Blue Region: The primary flow has not yet been moved to slice s_2 ($\phi_2 = 0$). Yellow Region: The primary flow has been moved to slice s_2 ($\phi_2 = \phi_p$).

Softwarization and Workshops (NetSoft), pages 406–414, 2018. doi: 10.1109/NETSOFT.2018.8460008.

[5] Faisal Naeem, Georges Kaddoum, and Muhammad Tariq. Digital twin-empowered network slicing in b5g networks: Experience-driven approach. In 2021 IEEE Globecom Workshops (GC Wkshps), pages 1–5, 2021. doi: 10.1109/GCWkshps52748.2021.9682073.

[6] Stuart Clayman et al. The necos approach to end-to-end cloud-network slicing

as a service. IEEE Communications Magazine, 59(3):91–97, 2021. doi: 10.1109/MCOM.001.2000702.

[7] ICT Infra. Sk telecom 6g white paper: 5g lessons learned, 6g key requirements, 6g network evolution, and 6g spectrum. Technical report, SK Telecom, 2023. URL https://newsroom-prd-data.s3.ap-northeast-2.amazonaws.com/wp-content/uploads/2023/08/SKT6G-White-PaperEng_v1.0_web.pdf.

[8] Dennis Overbeck, Niklas A. Wagner, Fabian Kurtz, and Christian Wietfeld. Proactive resource management for predictive 5g uplink slicing. In GLOBECOM 2022 - 2022 IEEE Global Communications Conference, pages 1000–1005, 2022. doi: 10.1109/GLOBECOM48099.2022.10001244.

[9] Guosheng Zhu, Jun Zan, Yang Yang, and Xiaoyun Qi. A supervised learning based qos assurance architecture for 5g networks. IEEE Access, 7:43598–43606, 2019. doi: 10.1109/ACCESS.2019.2907142.

[10] Dimitar Minovski, Niclas Ögren, Karan Mitra, and Christer Åhlund. Throughput prediction using machine learning in lte and 5g networks. IEEE Transactions on Mobile Computing, 22(3):1825–1840, 2023. doi: 10.1109/TMC.2021.3099397.

[11] Rajiv Pandey, Sunil Kumar Khatri, Neeraj Kumar Singh, and Parul Verma. Artificial intelligence and machine learning for EDGE computing. Academic Press, 2022.

[12] Rohit Kumar Gupta and Rajiv Misra. Machine learning-based slice allocation algorithms in 5g networks. In 2019 International Conference on Advances in Computing, Communication and Control (ICAC3), pages 1–4, 2019. doi: 10.1109/ICAC347590.2019.9036741.

[13] Engin Eyceyurt, Yunus Egi, and Josko Zec. Machine-learning-based uplink throughput prediction from physical layer measurements. Electronics, 11(8):1227, 2022.

[14] Samiksha Mathur, Yogesh Chaba, and Amandeep Noliya. Performance analysis of support vector machine learning based carrier aggregation resource scheduling in 5g mobile communication. Procedia Computer Science, 218:2776–2785, 2023.

[15] Hnin Pann Phyu, Diala Naboulsi, and Razvan Stanica. Machine learning in network slicing—a survey. IEEE Access, 11:39123–39153, 2023. doi: 10.1109/ACCESS.2023.3267985.

[16] Dhanashree Kulkarni, Mithra Venkatesan, and Anju.V. Kulkarni. Traffic prediction with network slicing in 5g: A survey. In 2023 Third International Conference on Artificial Intelligence and Smart Energy (ICAIS), pages 1360–1365, 2023. doi: 10.1109/ICAIS56108.2023.10073876.

[17] Saurav Agarwal. An approach of SLA violation prediction and QoS optimization using regression machine learning techniques. PhD thesis, University of Windsor (Canada), 2020.

[18] Abhilasha Sharma, Shweta Pandit, and Salman Raju Talluri. A comparative study to classify and predict the throughput of fifth generation wireless technology using supervised machine learning algorithms. In 2021 Sixth International Conference on Image Information Processing (ICIIP), volume 6, pages 288–292, 2021. doi: 10.1109/ICIIP53038.2021.9702678.

[19] Matteo Vincenzi, Angelos Antonopoulos, Elli Kartsakli, John Vardakas, Luis Alonso, and Christos Verikoukis. Multi-tenant slicing for spectrum management on the road to 5g. IEEE Wireless Communications, 24(5):118–125, 2017.

[20] Danish Sattar and Ashraf Matrawy. Optimal slice allocation in 5g core networks. IEEE Networking Letters, 1(2):48–51, 2019.

[21] Ved P Kafle, Pedro Martinez-Julia, and Takaya Miyazawa. Automation of 5g network slice control functions with machine learning. IEEE Communications Standards Magazine, 3(3):54–62, 2019.

[22] Rafael Montero, Albert Pagès, Fernando Agraz, and Salvatore Spadaro. Supporting qoe/qos-aware end-to-end network slicing in future 5g-enabled optical networks. In Metro and Data Center Optical Networks and Short-Reach Links II, volume 10946, pages 89–95. SPIE, 2019.

[23] A Sgambelluri, O Dugeon, A Muhammad, Jorge Martín-Pérez, F Ubaldi, K Sevilla, OG De Dios, T Pepe, CJ Bernardos, Paolo Monti, et al. Orchestrating qos-based connectivity services in a multi-operator sandbox. Journal of Optical Communications and Networking, 11(2):A196–A208, 2019.

[24] Qi Wang, Jose Alcaraz-Calero, Ruben Ricart-Sanchez, Maria Barros Weiss, Anastasius Gavras, Navid Nikaein, Xenofon Vasilakos, Bernini Giacomo, Giardina Pietro, Mark Roddy, et al. Enable advanced qos-aware network slicing in 5g networks for slice-based media use cases. IEEE transactions on broadcasting, 65(2):444–453, 2019.

[25] Zhaogang Shu and Tarik Taleb. A novel qos framework for network slicing in 5g and beyond networks based on sdn and nfv. IEEE Network, 34(3):256–263, 2020.

[26] Ammara Anjum Khan, Mehran Abolhasan, Wei Ni, Justin Lipman, and Abbas Jamalipour. An end-to-end (e2e) network slicing framework for 5g vehicular ad-hoc networks. IEEE Transactions on Vehicular Technology, 70(7):7103–7112, 2021. doi: 10.1109/TVT.2021.3084735.

[27] Ryu Kazama, Hirotake Abe, and Chunghan Lee. Evaluating tcp throughput predictability from packet traces using recurrent neural network. In 2022 IEEE Symposium on Computers and Communications (ISCC), pages 1–6. IEEE, 2022.

[28] Erik Aumayr, Giuseppe Caso, Anne-Marie Bosneag, Almudena Diaz Zayas, Özgü Alay, Bruno Garcia, Konstantinos Kousias, Anna Brünstrom, Pedro Merino Gomez, and Harilaos Koumaras. Service-based analytics for 5g open experimentation platforms. Computer Networks, 205:108740, 2022.

[29] Usman Haider, Muhammad Waqas, Muhammad Hanif, Hisham Alasmary, and Saeed Mian Qaisar. Network load prediction and anomaly detection using ensemble learning in 5g cellular networks. Computer Communications, 197: 141–150, 2023.

[30] Abdulsalam O Alzahrani and Mohammed JF Alenazi. Ml-idsdn: Machine learning based intrusion detection system for software-defined network. Concurrency and Computation: Practice and Experience, 35(1):e7438, 2023.

[31] Manuel O Alzahrani and Mohammed JF Alenazi. Ml-idsdn: Machine learning based intrusion detection system for software-defined network. Concurrency and Computation: Practice and Experience, 35(1):e7438, 2023.

[32] Massimiliano Maule, John S. Vardakas, and Christos Verikoukis. Multi-service network slicing 5G NR orchestration via tailored HARQ scheme design and hierarchical resource scheduling. IEEE Transactions on Vehicular Technology, 72(4):5021–5034, 2022.

[33] Massimiliano Maule, John Vardakas, and Christos Verikoukis. 5G RAN slicing: Dynamic single tenant radio resource orchestration for eMBB traffic within a multi-slice scenario. IEEE Communications Magazine, 59(3):110–116, 2021.