electronics

MDPI

Article

Slicing the Core Network and Radio Access Network Domains through Intent-Based Networking for 5G Networks

Khizar Abbas, Muhammad Afaq, Talha Ahmed Khan, Adeel Rafiq and Wang-Cheol Song *

Department of Computer Engineering, Jeju National University, Jejusi 63243, Korea; khizar.abbas@jejunu.ac.kr (K.A.); afaq@jejunu.ac.kr (M.A.); talhajadun@jejunu.ac.kr (T.A.K.); adeel.rafiq@jejunu.ac.kr (A.R.)

* Correspondence: philo@jejunu.ac.kr

Received: 9 September 2020; Accepted: 14 October 2020; Published: 18 October 2020

Abstract: The fifth-generation mobile network presents a wide range of services which have different requirements in terms of performance, bandwidth, reliability, and latency. The legacy networks are not capable to handle these diverse services with the same physical infrastructure. In this way, network virtualization presents a reliable solution named network slicing that supports service heterogeneity and provides differentiated resources to each service. Network slicing enables network operators to create multiple logical networks over a common physical infrastructure. In this research article, we have designed and implemented an intent-based network slicing system that can slice and manage the core network and radio access network (RAN) resources efficiently. It is an automated system, where users just need to provide higher-level network configurations in the form of intents/contracts for a network slice, and in return, our system deploys and configures the requested resources accordingly. Further, our system grants the automation of the network configurations process and reduces the manual effort. It has an intent-based networking (IBN) tool which can control, manage, and monitor the network slice resources properly. Moreover, a deep learning model, the generative adversarial neural network (GAN), has been used for the management of network resources. Several tests have been carried out with our system by creating three slices, which shows better performance in terms of bandwidth and latency.

Keywords: 5G networks; E2E network slicing; service orchestration; intent based networking; software defined networking; network function virtualization

1. Introduction

It is expected that the fifth-generation mobile network (5G) can cover a wide range of industrial use cases and improve network performance significantly. The current mobile networks are no more efficient at handling the multi-service demands and providing different qualities of service (QoS) to their users [1]. The virtualized and programmable nature of the 5G mobile network is the best and most cost-effective solution for ensuring on-demand services. Network slicing is an important feature of the 5G network that allows you to create multiple isolated logical networks, referred to as slices over the same physical infrastructure, for the mission of supporting multiple use cases, with diverse service requirements. The network slicing in a 5G network is achieved using network function virtualization (NFV) and software-defined network (SDN) technologies [2].

Just as 3G and 4G mobile networks revolutionized social behavior by enabling social networking on mobile phones, the 5G mobile network is also expected to connect everything, our environment, our businesses, our cities, and our industries will be revolutionized. The 5G network will not only

give users high-speed throughput like 4G but will also make the industries and cities smarter. It will also encounter the diverse use cases of smart factories as well. The 5G network design can meet the two new services' requirements, unlike enhanced mobile broadband (eMBB): the large variety of connected objects, known as massive machine type communication (mMTC), and ultra-reliable low latency (URLLC)—which requires high reliability and low latency. The main three slice categories in the 5G network are presented in Figure 1.

The 5G network architecture is designed on the service-oriented pattern that can provide better and different QoS to users for different kinds of application scenarios—data transmission and low latency, etc. The 5G network should be able to provide all kinds of services to every kind of user requirement, such as in the highly-dense area where the cellular traffic is very high. In those cases which require seamless service with high bandwidth, the indoor environment should provide seamless high bandwidth, even with fast speed mobility [3–5]. On the other hand, in the cases of large numbers of widespread low power-constrained devices (sensors, smartphones, security cameras, etc.) that need a reliable connection, 5G networks assure the connectivity of millions of these low powered and constrained-in-resources devices at very low cost. The other use-cases are connected vehicle, smart factories and real-time applications (health care monitoring) which require ultra-low latency; 5G networks fulfill the requirements of low latency application and provide secure communications by introducing the URLLC network slice [6].

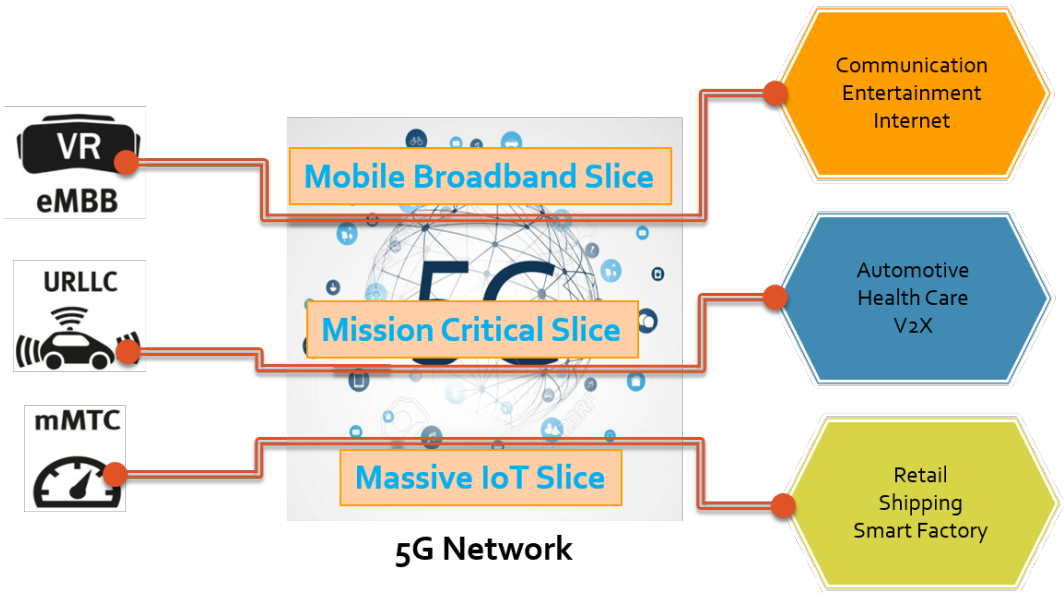

Figure 1. Types of slices in 5G network.

We have proposed and developed an automated framework named intent-based network slicing in which the intent-based networking (IBN) tool is used for providing the upper-level slice configurations to the network orchestrator. This slicing system encompasses four major modules: the IBN tool, the Network Orchestrator Open-Source Mano (OSM), the radio access network (RAN) controller FLEXRAN, and the machine learning module. This system can perform E2E slicing for both the core network and the access network. 5G networking requires one to have an automated management system to create, delete, and update the on-demand network slices by providing just abstract-level configurations. Thus, our system can automate the procedure of network slice creation with the help of the IBN tool. The IBN tool has the ability to take high-level configurations for a slice and generate a network slice template according to the network orchestrator acceptable format, for example, a JSON string for the OSM orchestrator and the JSON file for the RAN controller FlexRAN. It also contains a machine learning module that continuously monitors the network resources' statistics and stores them in the IBN database resource repository. Whenever a new slice request arrives, it is

also given to a machine learning module to decide whether the request is admitted or not by checking the status of the RAN and core network resources. If enough resources are available then the slice is admitted successfully and otherwise rejected. Hence, our system can automate the slice creation process completely.

To summarize, the following are the major contributions of our work:

- An intent-driven approach for the automated E2E slice configuration and the resource orchestration is the main contribution of our system. It induces a generic mechanism that can orchestrate resources at multiple domains. Additionally, it abstracts the underlying platforms and enables the configuration of different vendor-specific platforms. A few of the example platforms are CORD, OSM, and FlexRAN. Additionally, the best thing about this mechanism is that it eliminates the manual effort of work for creating, updating, and deleting network slices. Hence, it completely automates the lifecycle management of network services while keeping the IETF standard-based intent lifecycle mechanism.
- For the lifecycle management of intents, continuous monitoring is essential. Hence, while having been limited to open-source platforms this system introduced the monitoring and management of both the access and core network resources.
- The standards for network automation insist on introducing intelligence into different domains of the networks. Hence, AI-driven slice resource assurance is achieved by using the GAN deep learning model.

The organization of this article is as follows. Section 1 presents the brief introduction of the proposed system. In Section 2, background literature related to core and RAN network slicing is explained. The design and architecture of the E2E network slicing system are presented in Section 3. The components of the slicing system are well explained in Section 4. Section 5 explains the implementation of the proposed system and its working principle. Section 6 presents the slice instantiation process with the implemented test-bed. The performance results generated by the proposed system are discussed in Section 7. Section 8 presents the discussion of the proposed system with existing works. The final section contains the conclusion and future work of the presented work.

2. Related Work

2.1. Intent-Based Networking (IBN)

Intent-based networking (IBN) has emerged as an innovative technology for automation and orchestration of networks that may change the current network architectures and technologies. It is an intelligent networking system that can convert, deploy, and configure the network resources according to operator intentions automatically. This system can overcome and control abnormal events or failures. Moreover, it can also monitor the network resources continuously and collects the statistics for performing life cycle management [7–10]. Many industrial organizations have developed IBN-based systems but Cisco, Huawei, and IETF (Internet Engineering Task Force) are the main contributors to the development of intelligent IBN systems. The road-map of the IBN-based systems in the literature are presented in Figure 2.

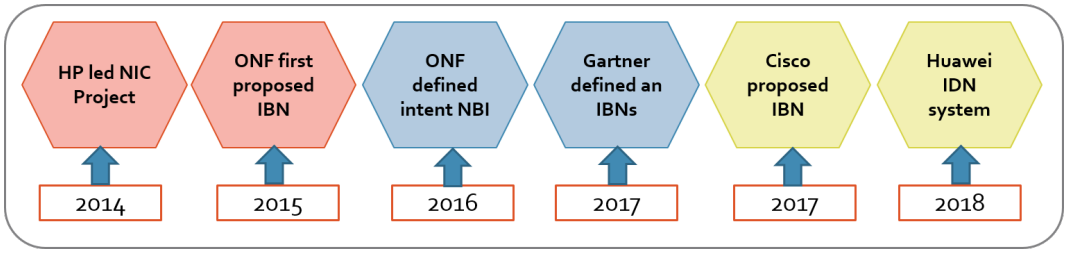

Figure 2. IBN system development road-map.

Cisco introduced the first IBN system in 2017; it is a closed-loop system and consists of three main modules: translation, activation, and assurance [11]. It facilitates the users of the network to input higher-level configurations in the form of intents, and the system can automatically translate those configurations into a system understandable format. Finally, the translated configurations are used for the deployment of resources. It is an intelligent and self-organizing system where the networks are planned, designed, and operated automatically. The translation module of the IBN system provides an easy way for network operators to express their intent in a flexible and declarative way. However, the network operator can consider the business objectives rather than thinking about how to configure the network resources to achieve the desired goal. On the other hand, the activation module of the IBN system can convert the captured intents into network policies that can be further used for deploying the network resources. Furthermore, with the help of domain orchestrators, these policies can be deployed into virtual and physical infrastructure. The assurance module is for monitoring the deployed resources, and in case of failure, it performs automatic recovery.

After the CISCO, Huawei [12] introduced a similar kind of system named the Intent-Driven Network (IDN). It is an autonomous driving network system which aims to develop a future generation network with lifecycle management support, intelligent operation and management, closed-loop, multi-service support, E2E slicing support, and automatic failure and recovery. It can efficiently deploy business intents into physical networks and facilitate users with better QoS and QoE. The IDN network has a smart brain layer named Network Cloud Engine (NCE) that performs intelligent network control and management. It performs the operations related to the translation of intents and activation dynamically. It can also monitor the health of the virtual and physical resources of the network in real-time. It is an intelligent system due to AI and big data analytics modules. Furthermore, the SDN-based controllers are also integrated with the NCE platform for better control.

2.2. End-to-End Network Slicing

The end to end (E2E) network slicing is a very important use-case to fulfill the requirements for service-oriented 5G networks. Many industrial organizations and standardization bodies related to network communications, such as the Third-Generation Partnership Project (3GPP), the Fifth Generation Partnership Project (5GPPP), the International Telecommunication Union (ITU), and the Next-Generation Mobile Networks (NGMN), also outline the E2E network slicing for future generation networks (FGN). Many systems have been proposed for core network slicing. For example, 3GPP explains very well slicing the core network, but slicing the access network is a very challenging task for the industry. Moreover, radio access network slicing still fails to provide different isolation levels and sharing to permit the slice owner to customize separated user plane (UP), control plane (CP), and control logic (CL) for specific services. Providing separate resources can also increase the RAN network utilization in terms of spectrum and resource blocks [13–17].

Some open-source network orchestrators enable the network operators to slice the core and RAN network resources to achieve the E2E network slicing. ETSI provides an open-source orchestrator for slicing the network, which has been developed based on an NFV specification named OSM [18]. The other open-source solution is ONAP [19] which was developed by the Linux Foundation to provide E2E network orchestration of virtual appliances. Moreover, Cloudify [20] and OpenBaton [21] are also open-source solutions for network orchestration, although Cloudify does not work on the basis of NFV MANO. JOX [22] is another open-source orchestrator developed by EURECOM and MOSAIC5G [23]. It is an event-driven orchestrator for network slicing implemented with JujuCharms NFs. The MOSAIC community [23] has also provided a solution for slicing the RAN resources with the FlexRAN controller. Many research studies related to the slicing of core, RAN, and transport networks are briefly explained below. In this paper, the authors [6] developed an E2E network slicing orchestrator which is capable of slicing the core, RAN, and transport network resources. Further, they have tested the proposed framework with three different case studies such as the eMBB slice in the stadium full of people for a sports event, a URLLC slice for performing remote surgery, and an eMBB slice for power meter reading

scenario. The framework performs well and provides isolated resources for each slice successfully. The author [2] presented a complete survey on network slicing and current challenges while slicing the 5G networks. Thus, the outcome of this survey is that efficient resource allocation and management framework is required for E2E network slicing.

Another slicing system presented by Meneses et al. [24] is known as SliMANO, which is an ETSI-based E2E network orchestrator that can perform commissioning and decommissioning of the network slices. It performs slicing with the help of MANO, an SDN controller, and an RAN controller as well. It can also monitor the slice resources and reconfigure the slices in case of failure. Thus, this system is completely developed based on the NFV and ETSI standards. SliceMano is quite similar to our work but our work is a complete package to automate the configuration procedure of network slices and have deep learning modules for efficient resource management [25]. This article [26] explains the network slicing framework for the 5G networks and describes the basic design challenges for performing RAN slicing. Therefore, a solution has been proposed for the configurations of RAN slices inside the radio protocol layers by defining a well-organized set of policies, descriptors, and resources. This work is well organized and contains a complete analysis that attempts to explain how to slice the RAN resources and how to customize multiple RAN slices. In the end, the applicability of the proposed framework for RAN slicing has been explained. This paper [27] presents a management and orchestration framework for the slicing of next-generation RAN. It defines the important interfaces and information models for the provisioning of RAN slices dynamically. Additionally, the authors have discussed the feasibility and requirements for the automation and orchestration of RAN slices. Moreover, a self-planning mechanism is illustrated which can handle all issues related to the management of the RAN network slicing domain [1].

This paper [28] investigated the issues and problems in the RAN slicing domain. The main challenge is how to allocate and share the spectrum for different kinds of slices to meet diverse user requirements. The other key issue is how to provide openness for third parties to use the same spectrum such as support for multi-tenant. The multi-tenant support enables third parties to use and share the infrastructure of different network operators, instead of deploying new infrastructure. In this system [29], the author has proposed an application-specific RAN slicing framework in which the radio spectrum can be assigned concerning application-specific radio resource blocks (RRBs). Additionally, an application-specific scheduling policy for the distribution of the RAN spectrum was introduced. Hence, the comparison between the conventional spectrum allocation mechanism and application-specific RRB distribution and scheduling was carried out, and the results show the efficiency and feasibility of the proposed system. In the end, an intelligent network architecture based on application-specific RAN slicing and machine learning was introduced.

In this paper [28], a RAN slicing system is presented with two services supports, such as eMBB and V2X. Additionally, this paper explains the issue and challenges that occur during the implementation of RAN slicing. A less complex heuristic-based algorithm has been developed with the RL (reinforcement learning) approach, which can allocate the optimal resources for each slice. It ensures the availability of RAN resources for each slice to meet the traffic requirements and maximizes the utilization of RAN resources. A simulation setup has been deployed to test the performance of the system. These results show that the proposed system can improve the performance of the network concerning latency, data rate, and resource utilization. In this paper [30], the author proposed a novel RAN slicing framework with haptic communications for 5G networks named Hap-SliceR. Firstly, an RL-based RAN resource slicing strategy has been implemented which can optimally allocate the radio resources to different slices according to the specified requirements. Secondly, this sliceR is capable of customizing the RAN resources for haptic transmission and communications. For the allocation of RAN resources, a heuristic-based algorithm has been designed and developed. The performance of the Hap-SliceR can be evaluated by comparing it with a well-known 5G air-interface design.

STORNS is an analytical framework for RAN slicing which works based on stochastic geometric theory [31]. They have developed a mathematical model to prove the benefits of slices RAN compared

to non-sliced RAN. This system can decide according to the provided service level agreement (SLA) and ensure average throughput per slice. Whenever a slice request arrives, it acts as an admission control block and allocates the resources for the admitted slice. Moreover, it will optimally allocate the bandwidth and transmit power for each tenant by introducing a new radio resource allocation technique. The performance of the STORNS can be validated with the help of numerical results by comparing the slicing and non-slicing systems. In this work [32], the authors have presented an optimization approach for resource allocation and mode selection in fog RAN. It also provides a RAN slicing mechanism for F-RAN where fog UEs and legacy UEs are served by the implemented slices. The paper formulates it as an integer programming optimization problem; it also provides a sub-channel allocation scheme to ensure slice isolation [33]. For solving this complex optimization problem, two RL algorithms have been proposed and implemented, covered by the fog and traditional UEs requirements. The major contribution of this work is to generate an optimal policy for mode selection according to the reward provided by the RL models and apply that reward to the environment. The simulation results generated with this system can validate the performance in terms of optimal queue delay and power consumption [34].

This work [35] proposes a RAN slicing system for 5G networks named iRSS (intelligent resource scheduling strategy) that works based on deep learning (DL) algorithms with the combination of the RL. In addition, large time scale resource allocation has performed using DL and small time scale dynamics, such as online resource scheduling, inaccurate prediction, and unexpected errors related to network states have performed by the RL. The DL algorithm is trained on historical traffic data and it predicts according to the amount of training data, while iRSS system compares the prediction results with RL online decision engine and adjusts the values flexibly. Hence the combined iRSS system can assist the RAN in deciding the resource scheduling. Compared with other algorithms, the proposed iRSS system iRSS shows better performance for online resource scheduling and increases resource utilization. It can also ensure the complete isolation among RAN slices. In this study [36], the authors have designed a novel framework for the fog RAN slicing domain which can orchestrate the network slices. For the case study, two slices were created and realized for hotspot and vehicle to infrastructure use cases. Additionally, this system considered RAN slicing as an optimization problem to deal with mode selection and content caching. The diverse user requirements and constrained resources cause this optimization problem much complexity so traditional approaches fail to solve it [37]. To solve this difficulty, a reinforcement learning (RL)-based algorithm has been proposed in which the cloud server is responsible to decide on mode selection and content caching. It can boost the reward by checking the status of the channel and cache dynamically. Hence, results show that this system can significantly improve the performance in terms of hit ratio and transmission rate. The comparative analysis of the proposed system with existing studies is presented in Table 1. This comparison is done on the basis of literature review with important parameters for network slicing systems.

Table 1. Comparative analysis of proposed work with existing methods.

Work	Core Slicing	RAN Slicing	Transport Slicing	Orchestration & Automation	Resource Monitoring	Intelligent Resource Management	Intent-Based Networking	Life Cycle Management
[1]	JUJU NFV management framework	FlexRAN Controller	No	JOX, JUJU Framework	No	No	No	Yes
[6]	Yes	Yes	Yes	Huawei Orchestrator	No	No	No	No
[23]	Dedicated Core Network	FlexRAN Controller	No	No	Monitioring using ELK	No	No	Yes
[24]	Yes	Yes	No	OSM MANO	No	No	No	No
[26]	No	Yes	No	No	No	No	No	No
[27]	No	Yes	No	Yes	Yes	No	No	No
[28]	No	Yes	No	No	Yes	RL & heuristic based algorithm	No	No
[29]	No	Yes	No	No	Yes	Intelligent Networking	No	No
[30]	No	Yes			Yes	RL algotithm	No	No
[35]	No	Intelligent Resources Scheduling	No	No	No	DL & ML algorithm	No	No
Our Work	OSM	FlexRAN Controller	No	OSM, FlexRAN & IBN	Openstack & FlexRAN	GAN model	Yes	Yes

3. Design and Architecture of the Network Slicing System

The network slicing system consists of four major modules named IBN tool, OSM network orchestrator, FlexRAN controller, and deep learning model. This system can perform the E2E slicing mechanism for both the core network and the access network. Moreover, our first module IBN is an automated tool to deploy higher-level network configurations in deployment and execution mode. It enables the network service providers to configure the network resources by just providing higher-level network configurations for both the core network and access network. It also contains a hybrid deep learning model GAN that continuously monitors the network resources statistics and stores them to the IBN database resource repository. Another module is the OSM framework which provides core network slicing support. The OSM policy configurator of the IBN tool can convert the higher-level configurations into JSON string format because the OSM orchestrator accepts the slice configurations in the form of JSON strings. The core NFVs are deployed by providing the information to the OSM orchestrator. After that, the OSM NFVO dispatches those slice configurations to OpenStack for the deployment of network functions (NFs). The core network slicing is managed by the OSM framework, where we deployed three separate core network EPC (Evolved Packet Core) functions such as MME, HSS, and SPGW for each slice. On the other hand, the FlexRAN controller which is an SDN-based controller that is used for managing and creating the access network slicing. We develop a specific RAN slicing policy configurator for access network to convert the high-level slice configuration provided at the IBN tool to JSON slice template format and send it to the underlying FlexRAN controller. Further, the FlexRAN controller deploys these configurations at eNodeB for slice creation. The complete E2E design and architecture of network slicing system is presented in Figure 3.

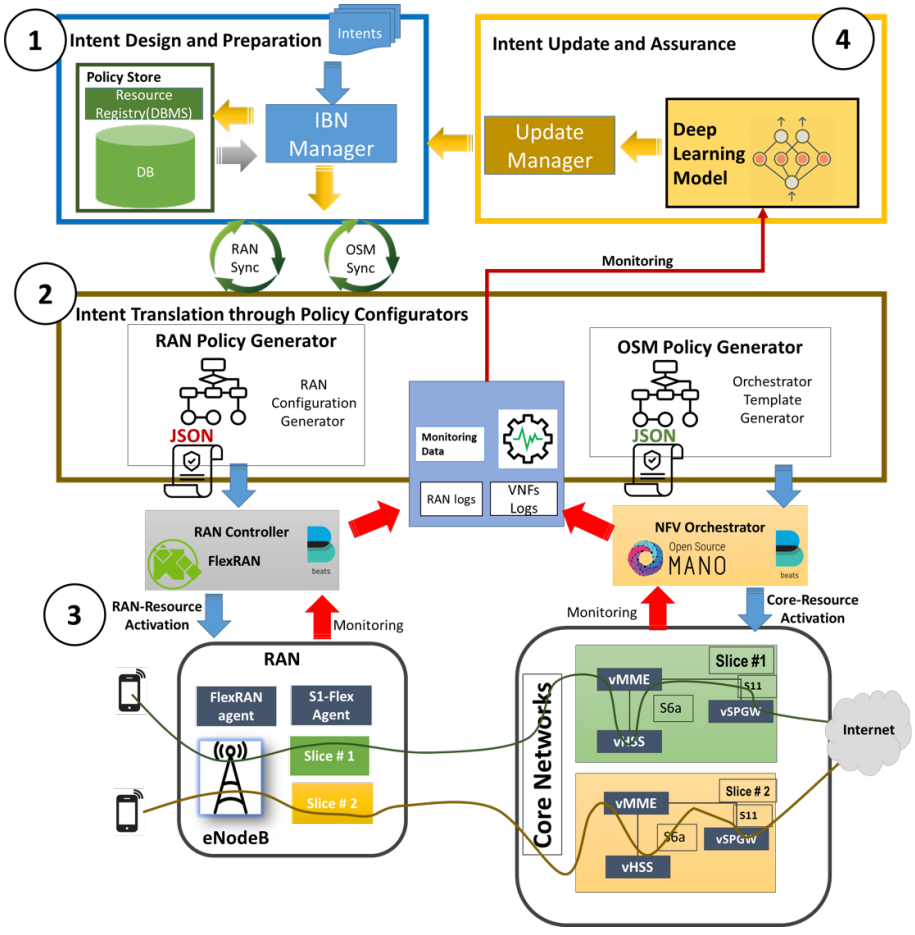

Figure 3. End to End network slicing architecture with intent-based networking.

4. Major Components of Network Slicing System

Our proposed system has four major modules: OSM (Open Source MANO), an intent-based networking tool, SDN controller FLEXRAN, and a deep learning (DL) module. This section contains the details of each module and how these modules work together and perform radio and core network slicing efficiently.

4.1. Open Source Mano (OSM)

The open-source management and orchestration (OSM) platform is a project of ETSI which provides telco operators an E2E network service orchestrator for the deployment of network services automatically. OSM has been developed based on NFV standards and facilitates the orchestrator to interact with other components, such as infrastructure managers VIM, NFVI, and network functions (PNFs and VNFs). Moreover, it provides the feature of on-demand creation of networks, e.g., network as a service (NaaS). It also provides an interactive GUI where the client can easily deploy, manage, and monitor the network resources. It can also support and facilitate the clients to deploy E2E network slices according to their needs at run-time, although the complete life cycles of network slices can also be managed by the OSM [18]. In our network slicing system, the IBN component OSM policy configurator can interact with the orchestrator with the help of a REST API by generating the slice configuration in JSON string and sending them to the orchestrator for NFs deployments. After that, the orchestrator deploys the required slice resources with the help of VIM (OpenStack).

4.2. Intent-Based Networking (IBN) Tool

The IBN tool encompasses six major components named the intent manager, the policy store, the contract design, the resource manager, the OSM policy configurator, the RAN policy configurator, and the DL Module. The IBN system can automate the network slicing procedure by just providing higher-level configurations from the front-end of the IBN tool. The functionality of each component is as follows; the contract design provides an easy way for users (operators, subscribers) of the system to define their intentions by using the front-end of the IBN tool. Users of the system define their resource requirements for creating a slice by just dragging and dropping at the front-end portal of the system. The intent manager acts as a coordinator for all the components and communicates directly with them. When a contract comes from the operator, it fetches the architecture information from the policy store and generates a graph by mapping both contract information and architecture information; after that, it sends that graph information to the policy configurator; the RAN related requirements can be sent to the RAN policy configurator and core and to the OSM core policy configurator as well. Moreover, the network functions information, version information, IP address scheme, and network instances images are stored in the policy store database repository. The policy store is a well-organized database repository that has all the information related to core network functions and registered eNodeB information. It stores the information according to the 5G architecture design. Further, the policy configurator extracts the information related to resources from the graph which is provided by the intent manager and converts that information in the form of a slice template. That slice template is according to the network orchestrator acceptable format; for example, for theOSM core, the policy configurator can give the slice template in the form of a JSON file which contains the information related to the deployment of the network functions and their mapping. The design of the OSM policy configurator is well explained in our paper [38–40]. On the other hand, the RAN policy configurator also generates the policies' template in the form of JSON which could be further sent to the FLEXRAN controller for slice creation at eNodeB. Afterward, FlexRAN deploys those configurations to the eNB through the master controller and agents via Southbound API. Finally, the requested network slice can be deployed at the access network. In this way, dedicated EPC and RAN resources can be assigned to a requested slice using specified S-NSSAI in the slice template. The IBN tool not only automates the policies for the deployment of the network slices but is a generic system for multiple orchestrators to

manage the whole network slicing procedure; it also monitors the network slice resources with the DL model during execution and updates the resources in case of failure or resource over-utilization. Hence, the DL model continuously monitors the resources and updates accordingly. The complete working functionality of each component is explained below.

4.2.1. User Intents/Contract Design

The contract design is a very important part of the IBN system because it will define the underlying platform resources requirement in an easy way which is understandable by the network operators. Basically, contract design is the user intent for creating a slice (resource requirements). This user contract design has four major fields: S-NSSAI (single network slice assistance information), architecture ID, up-rate, and down-rate. First, field S-NSSAI is used for identification of the service domain for each slice (URLLC, eMBB, mMTC, etc.) which is a standardized approach provided by 3GPP. The second field defines the network architecture type; it will be LTE, 5G, LTE advanced, etc.; architecture-ID is different for each network architecture. The last field is to provide QoS parameters up-rate and down-rate data requirements for the requested slice by specifying the bandwidth and latency requirements. The network operators can control the bandwidth, spectrum, and other resources by just specifying the up-rates and down-rates. Thus, the IBN framework has an interactive web-based front-end for network operators where they can define the user contracts by input the information of these four fields. These four important requirements are enough for the IBN framework to generate the service graph which will allocate the network resources (RAN, core) and suitable network functions for a requested slice. The procedure to process a contract with other modules of the IBN framework is shown in Figure 4.

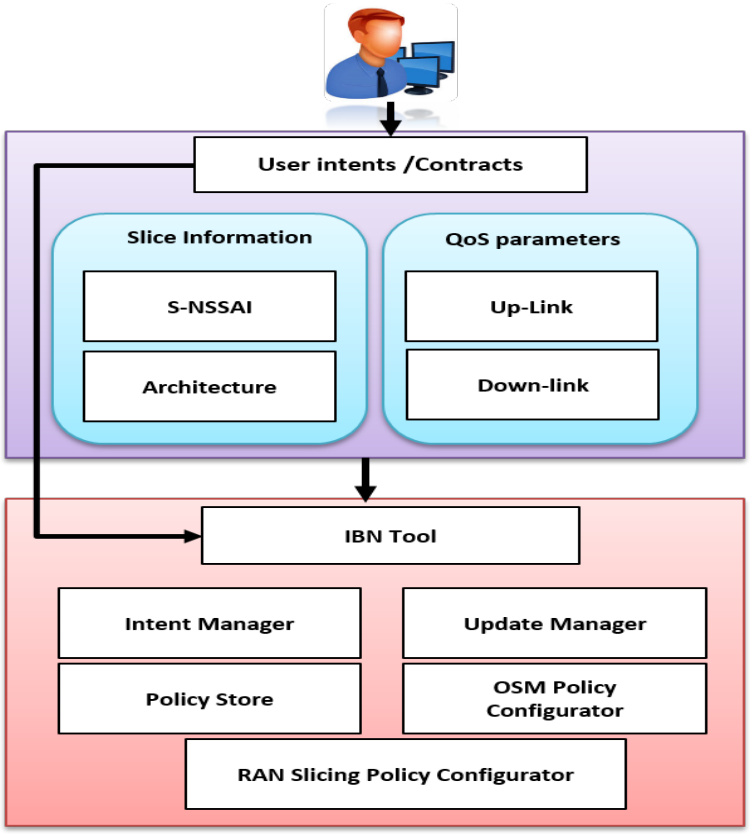

Figure 4. User intents or contracts design with required attributes.

4.2.2. Intent Manager

The second component of the IBN framework is the Intent manager which acts as a manager of the system. This intent manager can interact with all components of the IBN system and Network orchestrators as well with the help of REST API. The step by step working of this manager is as follows; first of all, it receives the user contracts from the top and then extracts the contract information and validates the required resources by querying the available resources from the policy store. The policy store is a database repository that contains information about all underlying resources. Thus, the intent manager will validate the required resources with available resources from the database. After validating the resources, extracted contract information could be used to make a service graph for the deployment of resources. it will make two service graphs for RAN and core network resources. Finally, the intent manager will send the service graphs to the policy configurators. Both the service graphs (RAN, core) are sent to the RAN policy configurator and core policy configurators for further processing. Furthermore, these policy configurators convert the service graph information according to the underlying platform compatible format.

4.2.3. Policy Store

The policy store is a well-managed database repository that can store all the information related to network functions, RAN resources, orchestrator information, and network architecture information. To store the network resources information in a database repository is a unique idea where network operators can easily get the available network resources information and can create a service graph for a specific network slice. The policy store was designed by keeping the NFV dependencies with each other. It encompasses many database tables: Predefined xontracts, modules_architecture, modules_relations, architectures_supported, modules_versions, and snssais_supported tables etc. The information stored in each table is as follows; the xontracts table stores the information related to the contracts which are inserted from the GUI interface of the IBN system. The architectures_supported table stores the information of all registered architecture with name and unique ID; each architecture has its unique-ID. The module_architecture table can store the information of all modules (NFVs, eNBs) and their dependent nodes i.e., the network function SPGWC is a dependent node on MME and SPGWU network functions. This table stores the relation of each node to the other which will help to compute a service graph. The fourth important table named a modules_relations table that can store the information related to link specification between different modules. Lastly, the table snssais_supported stores the information related to each slice doamin with specific SNSSAI id. The design of the Policy store is shown in Figure 5 with a table definition and their attributes etc.

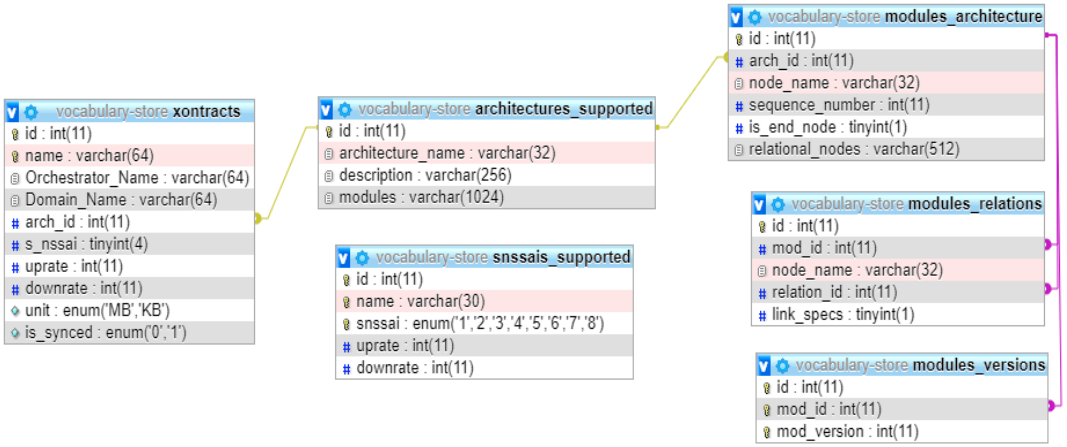

Figure 5. Design of the policy store database.

4.2.4. OSM Policy Configurator

The IBN tool is a general system that supports multi-platforms, which requires a variety of policy configurators for each platform. The responsibility of these policy configurators is to converts the user contracts into an appropriate format required for a specific platform. In our work, we have developed three policy configurators for different platforms OSM, M-CORD, and RAN but we used two policy configurators OSM and RAN for developing the E2E Network slicing system. In this section, we will discuss the OSM policy configurator and how it works. The working details of this configurator is as follows; the main purpose of OSM based policy configurator is to converts the service graph information into the JSON string format because OSM accepts policy descriptors in this form.

OSM policy configurator consists of three different engines, namely, VNF descriptor creation engine, NS descriptor creation engine, and NST Creation engine. Moreover, the first VNF descriptor engine contains three modules named basic information, VDU information, and connection point. The working of each module is as follows; the basic information module stores basic information, for example, name, description, and ID, etc.; similarly the second module VDU information can be used to record the information of resources, for example, required storage and RAM, etc.; thirdly, the connection point module stores the information related to ports and interfaces which are required for making connections. After that, all module performs their tasks and produce the JSON string that contains all configuration required for a VNF. The NS descriptor creation engine encompasses four main components: such as basic information, VNF descriptor (VNFD) information, connection ports information, and VLD information respectively. The working of the first two components are similar to the VNF descriptor engine but the third component VNFD information stores the information about specific VNFD for a specific NS descriptor. The last component stores the information of virtual links between different VNFDs.

The last module named a network slice template (NST) generation which consists of three components basic information, NS descriptor information, and VL descriptor. This module uses the information which is provided in the NS descriptor previously and generates the network slice information in the form of the template. The next VL descriptor information component stores the information for establishing a connection between different network services. After that, it can transfer the NS template information into the next module OSM for further processing. All the modules work together and generate the descriptors in JSON format, these modules are also dependent on each other because the next module has used the descriptor information of the previous module. Afterward, the generation of all descriptors in the JSON string format will forward to the onboarding module for storing them in the OSM repository. These onboarding modules communicate with OSM with the help of REST API and sends the request to OSM five-time in case of any error or failure. After that, OSM REST API notifies the internet manager about the failure and the next process will be stopped immediately. Moreover, onboarding of all the descriptors are required for the instantiation of NST. The OSM uses these descriptors for the deployment of network slice template NST or network service. After onboarding all the descriptors, the policy configurator sends the specific service deployment request to the OSM. At the end, the required network service is deployed according to NST configurations. The architectural design of policy configurator for OSM is presented in our paper [38,41].

4.2.5. RAN Policy Configurator

The RAN policy configurator is an important component of the IBN system because it is responsible for slicing the resources of eNB. This policy configurator can converts the requested slice requirements in terms of bandwidth, spectrum allocation, uprate, and downrate that are provided through IBN GUI in the form of contract by the clients. It converts the contracts into the JSON file because FlexRAN accepts the slice information in the JSON template as a network slice template. Afterward, FlexRAN deploys those configurations to the eNB through the master controller and agents via Northbound API. Finally, the requested network slice can be deployed at the access network. In this

way, dedicated EPC and RAN resources could be assigned to a requested slice using specified S-NSSAI in the slice template.

4.3. SDN Controller FLEXRAN for RAN Slicing

FLEXRAN is an open-source and highly programmable Software-defined RAN (SD-RAN) platform that separates the RAN control plane to the data plane. It provides programmability support at two levels: the first one is to develop control and management applications over the FlexRAN controller and the second is to develop applications within the controller for performing the automatic deployment of control functions. It can also control the multiple distributed base station and their coordination among each other.

The FlexRAN consists of two main components, namely, FlexRAN control plane and FlexRAN agent API. The first component FlexRAN control plane composed of a FlexRAN master controller which is further connected to FlexRAN agents. Although, one FlexRAN agent is for each eNodeB and also acts as a local controller while communicating with the master controller and other agents. The second component FlexRAN agent API is a southbound API which separates the data plane to control plane such as the separation of FlexRAN control plane to the eNB data plane. On the other hand, the FlexRAN- protocol is responsible for the two communication between agents and master-controller of FlexRAN. The master controller can control and manages the operation of each eNB with the help of FlexRAN-protocol. The abstract view of FlexRAN controller and its communication with eNodeB data plane is shown in Figure 6. On the top, some monitoring and control applications communicate with the master controller by using the northbound-API interface. The responsibility of these applications are to modify the RAN resources by checking the statistics and monitoring logs of the eNB in the FlexRAN control plane [42].

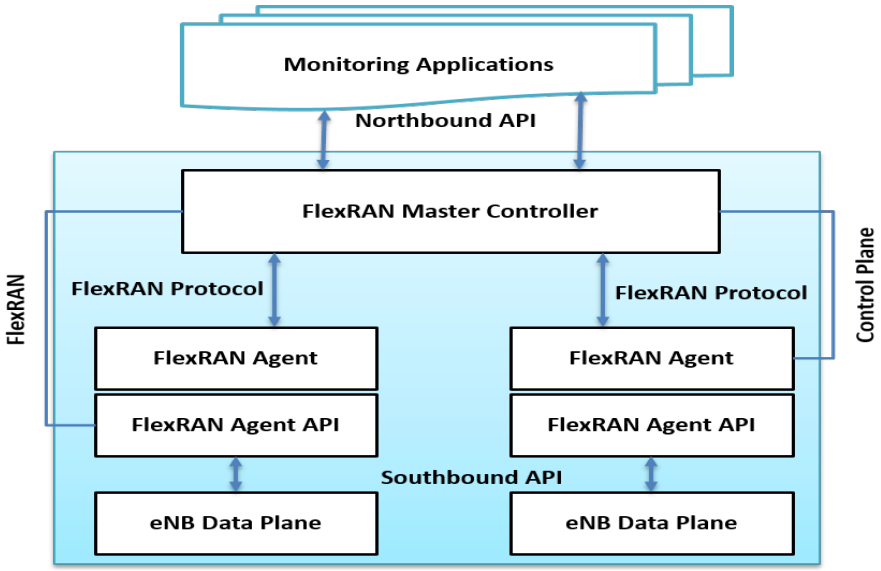

Figure 6. FlexRAN controller abstract view.

The role of the FlexRAN controller in our network slicing system is to slice the RAN by getting the network slice template from the IBN component RAN slicing configurator. The FLEXRAN accepts the policies in the form of a JSON file. Thus, our RAN slicing configurator converts the slice request which is generated through the IBN system. Finally, after getting the slice configurations from the RAN policy configurator the FlexRAN controller deploys those slices configurations to eNB. Furthermore, FlexRAN controller can be able to share the RAN resources with the help of the master controller and the agents. Although the controller can also monitor and control the RAN resources, these monitoring logs can be further used for deep learning models for the slice admission decision.

4.4. Advanced Deep Learning Module

In this section, we are going to explain the deep learning Model GAN and dataset used for training and testing purposes. GAN is a very famous deep learning model that is mostly used for image processing, video processing, and computer vision tasks but we have used the GAN model for predicting and forecasting resource utilization for the next period. It has two modules, namely, generator (G) and discriminator (D): one neural network will be used for generator and one for the discriminator part. The first part generator generates new data instances and the discriminator model acts like a decider whether the data is authentic (real) or generated [43,44]. The pseudocode of GAN model is presented in Algorithm 1. In our system, we have used the LSTM (long-short term memory) model in the Generator part and CNN (convolutional neural network) model in the Discriminator part for the prediction of network resource utilization (CPU, RAM, etc.) is illustrated in Figure 7.

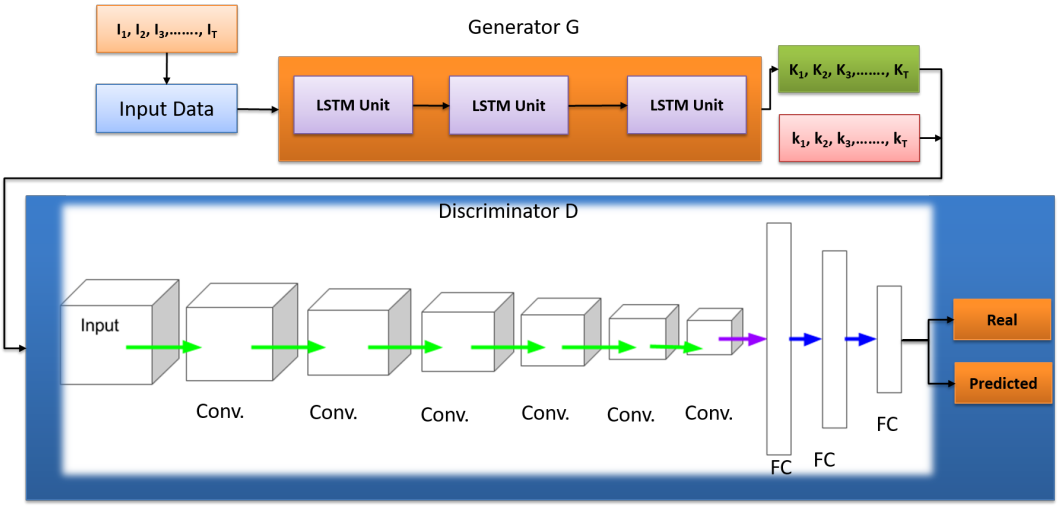

Figure 7. Design of GAN deep learning model.

LTSM (long short term memory) is a popular deep learning model mostly used for time series data analysis, prediction, traffic forecasting, natural language processing, and handwriting recognition. LSTM belongs to the RNNs family (Recurrent Neural Networks). RNN models can store and keep track of all the previous data points thats why they are the best choice for time series dataset operations. In our case, network traffic or resource utilization data is time series, LTSM is the best choice to use as a generative model to generate the resource utilization output data based on input dataset parameters. Moreover, for controlling, updating, forgetting, and storing the data every hidden node in the LSTM model has its gates or memory cell. There are three kinds of memory cells or gates for storing data dependencies in LSTM such as Forget F_t, output O_t and input gate I_t. The activation function for these gates is the sigmoid function and t is the current time. The working of each gate in the LSTM unit is illustrated below.

$$I_t = \sigma(U_i x_t + W_i h_{t-1} + b_i) \quad (1)$$

$$F_t = \sigma(U_f x_t + W_f h_{t-1} + b_f) \quad (2)$$

$$A_t = tanh(U_c x_t + W_c h_{t-1} + b_c) \quad (3)$$

$$C_t = I_t * A_t + F_t * C_{t-1} \quad (4)$$

$$O_t = \sigma(U_o x_t + W_o h_{t-1} + V_o C_{t-1} + b_0) \quad (5)$$

$$H_t = O_t * tanh(C_t) \quad (6)$$

The function of the input gate in the LSTM unit is shown in Equation (1). Where U_i and W_i are the input weights and b_i is the bias value. Equation (2) represents the forget gate function where U_f and W_f are the weights parameters and b_f is the bias value. Equation (3) presents the function of intermediate cell state A_t where U_c and W_c are the weight parameters and b_c is the bias value. Equation (4) presents the function of cell state C_t. Equation (5) presents the function of output gate O_t where U_o, W_o, and V_o are the weight parameters and b_o is the bias value. Equation (6) shows the function of the new hidden state H_t. further, (σ) and (tanh) are the activation functions in the LSTM unit. In all equations h_{t-1} shows the current time in hidden state. In Equations (4) and (6) $(*)$ operator is used as element wise multiplication operator.

Algorithm 1 Pseudo-code for the generative adversarial network (GAN) model.

1: **Probability distribution for noise generation** $P_g(z)$ **and probability distribution of training data** $p_{data}(x)$**.**

2: **Output: Trained Generator (Gen) and discriminator (Dis) modules.**

3: **Begin**

4: **for** number of training iterations {

5: **for** j=1 to m {

6: Divide the data into minibatch $\{z^1, z^2, z^3, ..., z^n\}$ of n noises sampled from noise prior $P_g(z)$

7: Divide the data into minibatch $\{x^1, x^2, x^3, ..., x^n\}$ of n noises sampled from noise prior $p_{data}(x)$

8: Adjust the parameters θ_{Dis} of the Dis module with the following averaged stochastic gradients $\theta \longleftarrow \theta d + \eta \bigtriangledown \theta_d \frac{1}{\mathrm{m}} \sum_{i=1}^{m} \left[\log D(x^i) + \log(1 - D(G(z^1)))\right]$

9: } **end of for loop**

10: Divide the data into minibatch $\{z^1, z^2, z^3, ..., z^n\}$ of n noises sampled from $P_g(z)$

11: Adjust the parameters θ_{Gen} of the Gen module with the following averaged stochastic gradients $\theta \longleftarrow \theta g - \eta \bigtriangledown \theta_g \frac{1}{\mathrm{m}} \sum_{i=1}^{m} \left[\log(1 - D(G(z^1))\right]$

12: } **end of for loop**

13: Terminate (conditions satisfied)

14: **End**

In the second discriminator part, we have used the CNN model for the prediction of resource utilization that takes the input data or generated data from the generator module. CNN takes three-dimensional input for training purposes and also has fully connected (FC) layers for better prediction. This part estimates the probability of whether the input data are real or predicted from the previous module. Both the models (G, D) are trained on real and generated datasets by using stochastic gradient descent (SGD) model. The final output is the resource utilization statistics of the underlying VMs for each slice next time t, after that the predicted values will be used by the resource update manager of the IBN tool. Those predicted statistics are very useful for network slice acceptance, resources scale-up, failure recovery, and resource management. The training of the generator and discriminator is summarized in Algorithm 2, which is an iterative process by partitioning the data into mini-batches of size n. We have used a stochastic gradient descent method for generalizing the algorithm into mini-batches of size n. The generator module should minimize the adversarial loss as much as possible so that the discriminator module will not perform prediction correctly. The generator module is trained by giving input data in the LSTM unit and it generates similar kinds of data samples for the discriminator module. After that in the discriminator module, this generated data samples and original input data have been used to confuse the discriminator for predicting either the date is fake or

original. The CNN model in the discriminator module classify and discriminate between two datasets and predict the resource utilization accurately.

Algorithm 2 Pseudo-code for the training of Generator and Discriminator.

1: **Step 1: Initialize the learning rates for Generator Gen and Discriminator Dis.**
2: adjust the parameters
3: set the weights W_{Gen} and W_{Dis}
4: While not satisfied do
5: **Step 3: Update the Generator Gen Module:**
6: Get n new data samples
7: Get updated weight W_{Gen} for Generator
8: **Step 4: Update the Discriminator Dis Module:**
9: Get n new data samples
10: Get updated weight W_{Dis} for Discriminator
11: **Step 5: End While**

We have used an open-source dataset provided by the MATERNA datacenter which contains performance metrics of 547, 520, and 527 data center virtual machines (VMs). MATERNA is a well-known service provider and IT product manufacturing organization. The dataset was collected in three traces within a period of three months, so one trace for one month timestamp. More detail about dataset collection is in this paper [25]. Moreover, the dataset has 12 attributes related to resource utilization such as the timestamps (ms), CPU cores (number of virtual cores), CPU usage (MHz), CPU capacity (MHz), CPU capacity (percentage), memory requested (KBs), memory usage (KBs), memory usage (%), received throughput(KBs), transmitted throughput(KBs), disk writeup throughput (KBs) and disk size (GBs). Thus, we trained our model at this dataset and our deep learning model can predict the future state of the resources and network operator used these predicted statistics for better resource management, i.e., scaling up or scaling down the resources. Moreover, update manager of the IBN system updates the resources' statistics in our IBN policy store. Additionally, the IBN system considers these statistics from the database while preparing and designing user intents. It designs the slice intent by looking at the statistics of the resources, if there are enough resources then process the slice request otherwise rejected. Thus, in this way IBN system can easily scale up and down the resources; in case of failure, it will notify the IBN system and request to update the resources. In the future, we have plan to used reinforcement learning with the GAN model in our system for better QoS, slice admission, slice control, and slice management. On the other hand, the training and testing accuracy of the deep learning model is presented in Figure 8, which shows almost 87% training accuracy and 82% testing accuracy. However, our GAN shows satisfactory performance while predicting VNFs resources—CPU, memory, etc.

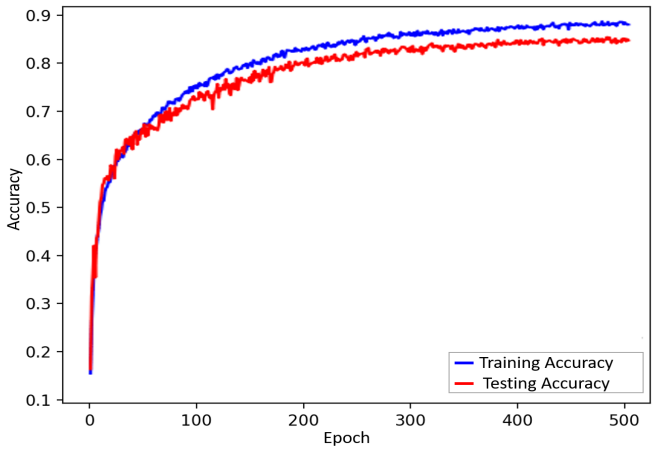

Figure 8. Test and training accuracy of the GAN model.

5. Radio Access Network Slice Instantiation Process

The user first defines the slice requirements on the GUI of the IBN tool and includes downlink, uplink, starting time, ending time, and slice SNSSAI information. After that, the intent manager of the IBN system interacts with the resource repository and policy store for the creation of the VNFFG type policy graph. The created policy graph contains the information of requested core and RAN resources and links between them. This policy graph is sent to policy configurators for further processing. These IBN policy configurators are a very important part of the IBN system. As discussed above, there are two types of policy configurators in our IBN system; one is the OSM policy configurator and the other is the RAN slicing configurator.

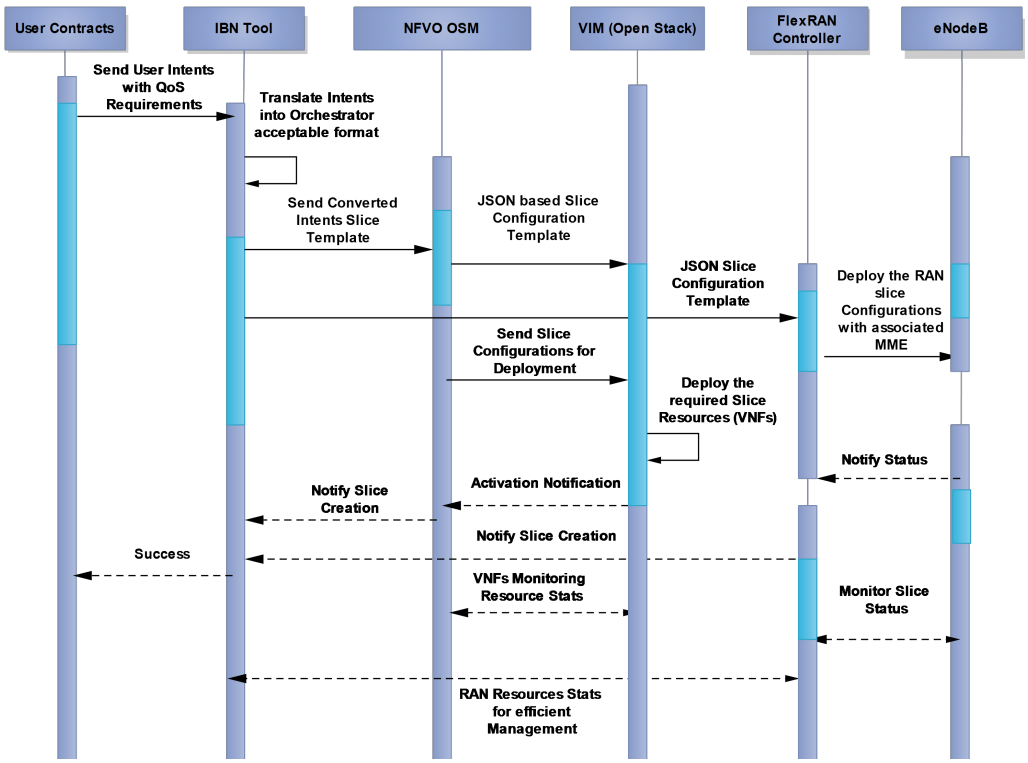

Figure 9. Network slice instantiation process.

The OSM policy configurator's job is to take this policy graph and translate it into a JSON string format. In contrast, the RAN policy configurator converts the policy graph information into TOSCA and then into the JSON file. Moreover, the OSM policy configurator transfers these configurations to OSM NFVO for the deployment of core EPC resources. On the other hand, the RAN configurator sends the JSON file to the FlexRAN controller and the controller deploys the slice QoS to eNodeB. It also contains dedicated MME information and slice SNSSAI Which connects the RAN slice with the core EPC. The core EPC VNFs are deployed with the help of OSM NFVO and VIM (OpenStack) using the JSON string format slice template. After the slice creation, the user is notified with the help of IBN. This is how a network slice is created, which does not require any manual work or expertise. Hence, users can define slice QoS requirements in a higher-level language through drag and drop from GUI of IBN, and the system automatically translates these QoS requirements in the form of a slice template and deploys resources accordingly. The slice instantiation procedure is presented in Figure 9.

6. Experimental Test-Bed Implementation

Our implemented test-bed consisted of the IBN tool, OSM platform, FlexRAN, OAI EPC core network functions, and OAI eNB for achieving E2E network slicing. The other module is the integration of machine learning model GAN which predicts resource utilization—CPU, RAM, etc. This system aims to automate the network slice configuration and management process and provide the clients with an easy way to create network slice according to their requirements. For testing purposes, we created three types of network slices eMBB, IoT, and URLLC by inserting higher-level configurations in the form of user intents/contracts from the IBN tool. The IBN tool dispatcheed those configurations to the OSM network orchestrator and SDN-RAN controller FlexRAN for further processing. Moreover, IBN provided higher-level contract configurations in the JSON file format to OSM and FlexRAN.

Table 2. Test-bed components configurations and specifications.

Component	System Specifications
FlexRAN	OS: UBUNTU 16.04 RAM: 16 GB CPU: CORE-i5 3.0 GHZ SSD: 500 GB
OSM	OS: UBUNTU 18 LTS RAM: 252 GB CPU: 32 Cores 2.10 GHZ H/D: 2 TB OSM Version: 7 Openstack Version: stein
IBN tool	OS: Window 10 RAM: 16 GB CPU: Core I5 3.0 GHZ H/D: 1 TB Programming languages: PHP, JAVA Database: MYSQL
SDR USRP B210	Frequency Range: 70 MHz-6 GHz Channels: 2TX * 2RX

Afterward, the OSM orchestrator deploys the generated JSON string slice template configurations to the physical layer with the help of the VIM manager OpenStack. On the other hand, FlexRAN enforces the slice template to underlying eNBs to allocate specific spectrum and resource blocks with the help of the master controller and local agents. The slice template generated through the RAN slicing configurator also provides the configurations (IP addresses of EPC functions) for the connectivity of dedicated EPC with selected eNB. In this way, a specific slice is created and

stitches together with core and access network automatically. The whole slice creation process is automated where the user just used IBN GUI and defined his contract for the slice and IBN automatically deployed those configurations to the physical layer with the help of OSM and FlexRAN. Furthermore, we have used the EPC eNodeB provided by the well-known open source community OpenAirInterface (OAI) [45]. Further, two eNodeBs were deployed with the help of SDR USRP B210. Three core networks have been deployed on different VMs by using VIM OpenStack for the creation of three slices. Smartphones and OAI simulated UEs were also used for testing and validation purposes. The test-bed components with their configurations and specifications are illustrated in Table 2.

7. Experimental Results

The interactive GUI of the IBN application is shown in Figure 10 where users can submit their intents in the form of contracts. After that these contracts are translated with the help of both policy configurators into a JSON format string for OSM and FlexRAN respectively. Furthermore, these converted configurations are applied to NFVO of OSM and FlexRAN for the creation of network slices dynamically. IBN policy configurators communicate with NFVO and FlexRAN with the help of southbound REST API.

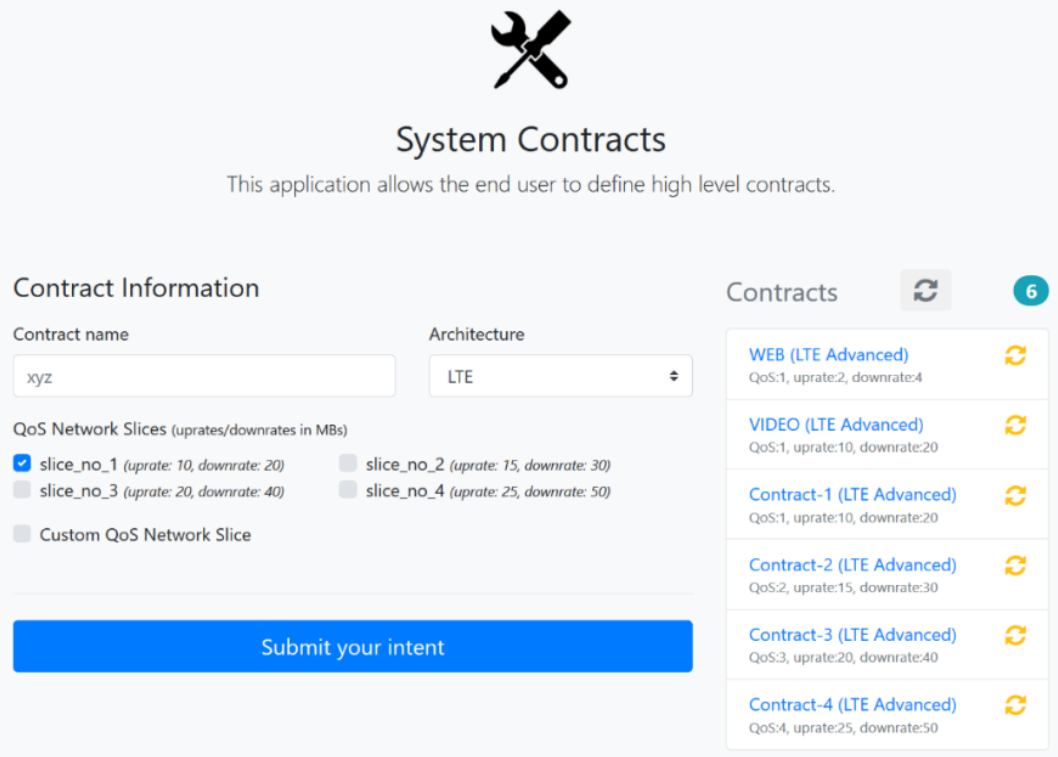

Figure 10. GUI of intent-based networking tool.

For our experimental setup, we have created three slices with different QoS requirements, namely, eMBB, IoT, and URLLC slices. The QoS requirements for slice 1 in user contract is specified as 40 MB/s downlink speed and 20 MB/s uplink, slice 2 is 40 MB/s downlink and 20 MB/s uplink, and slice 3 is 20 MB/s downlink and 10 MB/s uplink respectively. we have performed the Iperf test to verify the stability of each slice separately. Further, the simulated OAI UEs were also used to test the performance of each slice. IBN tool has the GUI, where the users can define slice contracts by just providing higher-level configurations. Furthermore, the specific contract is dispatched to OSM orchestrator and FlexRAN for the deployment of network slice dynamically. Finally, the specified uplink and downlink are reserved for that slice with required core EPC functions (vHSS, vMME, vSPGW).

Figure 11 shows the downlink throughput of all three slices, we have achieved a maximum of 32 MB/s throughput for eMBB, 26 MB/s for IoT, and 16 MB/s for URLLC slice downlink throughput while performing the Iperf test for each slice. For the downlink case, we have already specified 40, 40, and 20 MB/s of throughput in slice contract for each slice. However, the configurations provided from the IBN tool were properly deployed and configured the resources for each slice.

The uplink throughput speed tested with Iperf for each slice is presented in Figure 12, as mentioned above the up-rate throughput specified in the network slice template was 20, 20, and 10 MB/s respectively. Although we have got a maximum of 19 MB/s for slice#1 and 17 MB/s for slice#2 and 7 MB/s for slice#3. In our experimental setup, we have 100 MB/s connection and SDR USRP B210 also provides up to 100 MB/s spectrum support. The results show the performance of our system is stable and the required slices were created dynamically.

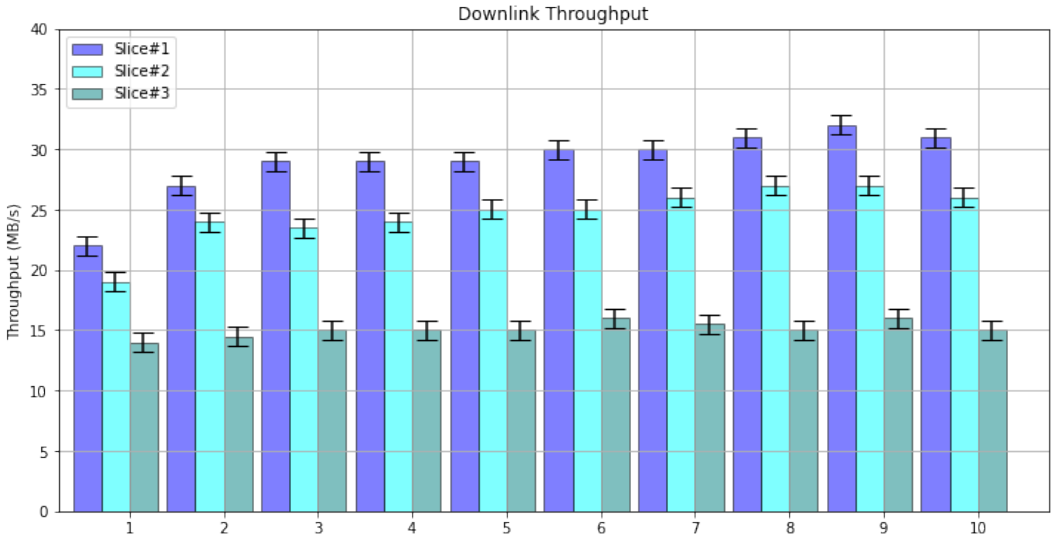

Figure 11. Iperf downlink throughput for three slices.

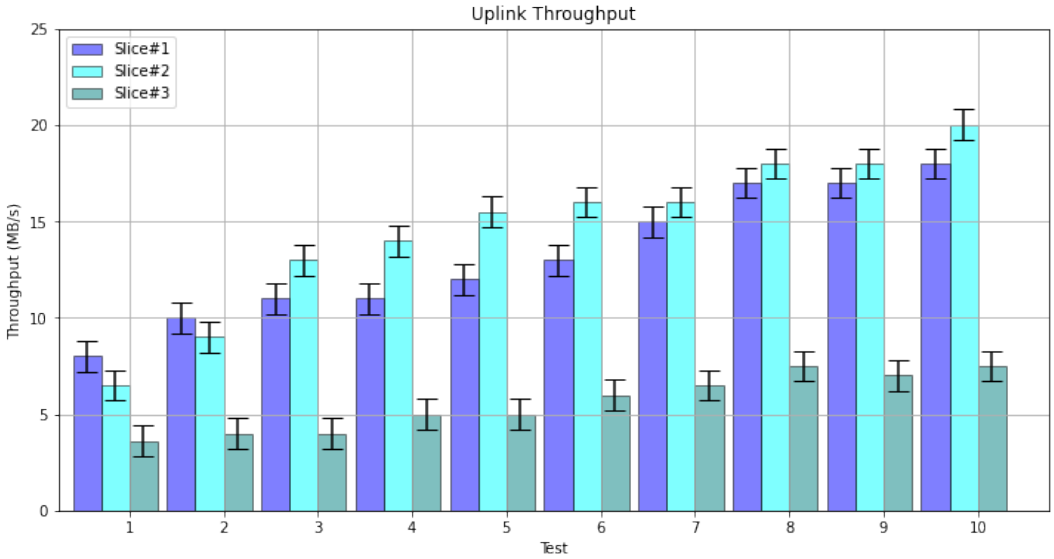

Figure 12. Iperf uplink throughput for three slices.

Figure 13 presents the downlink and uplink throughput test performed with Iperf only at access network eNB. We just provide the configurations to the RAN slicing configurator with different QoS, such as radio resource block (RBs) and 20, 40, 60, 80, and 100% spectrum resources of the RAN eNB. You can see that by increasing the resources of RAN, you can increase the uplink and downlink

throughput of the network slice. When we have created a slice with 20% resources of eNB, it provides low 4.5 MB/s and 9 MB/s throughput speed for UL and DL respectively. On the other hand, by using the 100% resources of RAN, it shows 69 MB/s of DL and 41 MB/s of UL speed. Thus, after getting the slice template configurations from the RAN slicing configurator, FlexRAN can deploy those configurations at physical eNB and share the resources accordingly.

The downlink throughput with simulated UEs is presented in Figure 14 where four simulated OAI UEs are connected with the deployed RAN slice successfully. The Iperf results show that the maximum of 12 MB/s downlink speed throughput was achieved for one UE in our test-bed. This test has been carried out with multiple time slots and UEs are connected properly all the time. Although, sometimes the connection problem occurs due to poor signal strength of SDR USRP B210. We test our slicing system with eight simulated OAI UEs and it shows promising performance in terms of bandwidth, latency, and delay. Thus, our system is totally dependent on open-source systems like OSM, FlexRAN, OAI EPC, OAI eNodeB, and OAI UEs. Hence, our IBN system can automate the network configurations and generate a slice template with the help of policy configurators and deploy these resources requirements into underlying infrastructure by using OSM orchestrator and FlexRAN controller.

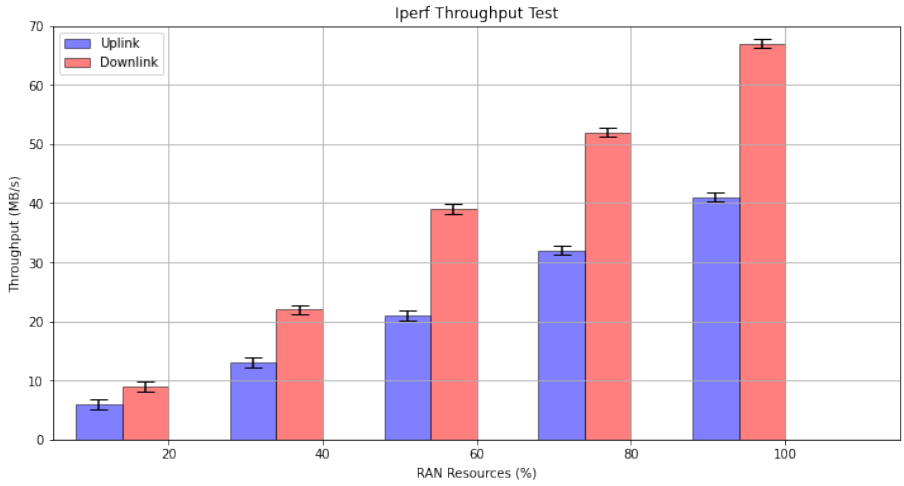

Figure 13. Iperf downlink and uplink throughput test for eNB resource sharing.

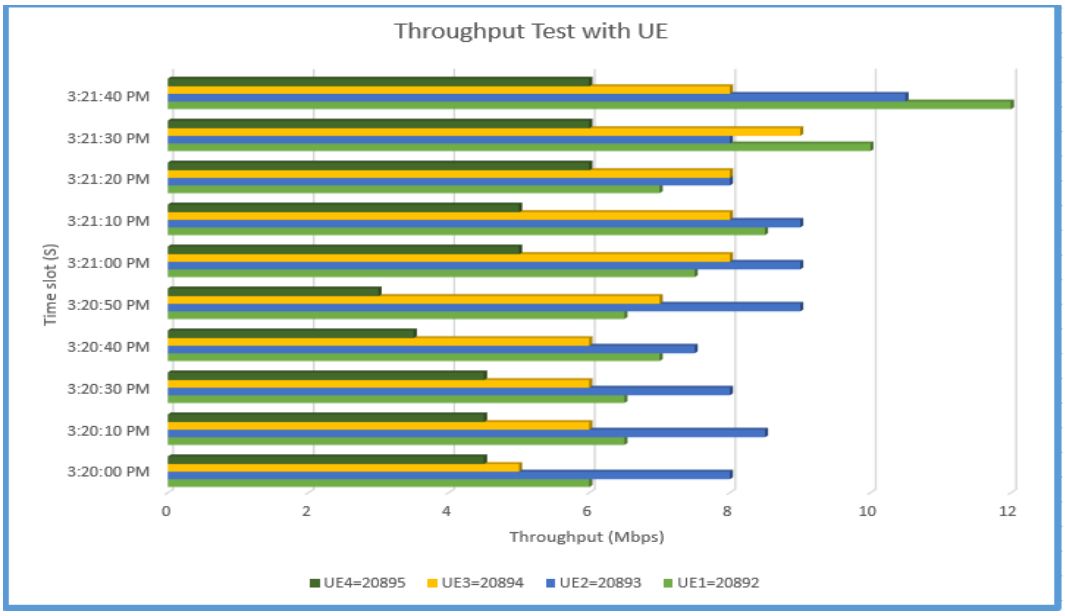

Figure 14. Iperf throughput test results with simulated UEs.

8. Discussion

From the literature, there is a need for an automated and zero-touch system, where the user needs to provide its intention for resources, and in return, the system will deploy the resources dynamically. The automatic policy generation, resource activation, resource assurance, and resource monitoring are important features for a zero-touch system. However, there is a lack of automation in the previous studies and none of them uses IBN-based approach for performing E2E network slicing. There are various methods with RL for slicing the RAN and core domains separately, but only few perform slicing for both domains. Such methods lack the existence of any domain orchestrator for controlling the slicing mechanism for both domains. However, in our case, we have domain orchestrators, such as OSM MANO and FlexRAN controller for handling core and RAN domains slicing. Moreover, our IBN system automates the process of policy generation for RAN and core domains. As aforementioned, the IBN system has three major components, i.e., translation, activation, and assurance. Similar to the standard IBN system, our proposed system also has an intent translation module, which can translate user requirements into policies with the help of policy configurators. The activation module deploys these policies over the infrastructure for the creation of slice. In addition, our system facilitates monitoring of virtual and physical resource with the help of OpenStack Neutron and FlexRAN controller monitoring application. The DL module forecasts resources for efficient resource management. Moreover, unlike previously developed systems, our proposed system automates policy generation and slicing creation process through IBN-based approach.

9. Conclusions

This paper describes the design and implementation of the intent-based E2E network slicing platform that facilitates the network operators to deploy the network services in a flexible and customizable manner. The network operators instantiate the network slice by using the GUI of IBN application, wherein users just input the higher-level configurations in the form of QoS requirements. IBN system can designs the slice template with the help of policy configurators (OSM core policy configurator, RAN slicing policy configurators) and dispatches those configurations to the OSM network orchestrator and FlexRAN controller for the deployment of resources. More precisely, our system allows network operators to automate the network configuration and slice creation process. Moreover, deep learning model forecasts and predicts the network resources state on runtime, which helps the intent manger in deciding slice admission. We have performed multiple tests to check the working of our system, which shows promising results for the creation and management of the network slices. In the future, we intend to extend our system by adding more features related to the complete life cycle management of network slicing.

Author Contributions: Conceptualization, K.A.; Data curation, T.A.K.; Formal analysis, M.A.; Funding acquisition, M.A. and W.-C.S.; Investigation, K.A., T.A.K. and A.R.; Methodology, K.A. and T.A.K.; Project administration, M.A. and W.-C.S.; Supervision, W.-C.S.; Validation, M.A.; Writing—original draft, K.A.; Writing—review & editing, A.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2020-2017-0-01633) supervised by the IITP (Institute for Information and communications Technology Planning and Evaluation). This research was supported by the 2020 scientific promotion program funded by Jeju National University.

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

Abbreviations

The following abbreviations are used in this manuscript:

3GPP	Third Generation Partnership Project
5GPP	Fifth Generation partnership Project
IBN	Intent Based Networking
LTE	Long-Term Evolution
EPC	Evolved Packet Core
RAN	Radio Access Network
SDN	Third Generation Partnership Project
NFV	Network Function Virtualization
VNFs	Virtual Network Functions
GAN	Generative Adversarial Network
OSM	Open Source MANO
VIM	Virtual Infrastructure Manager
VNFM	VNF Manager
OAI	OpenAirInterface
DL	Deep Learning
IETF	Internet Engineering Task Force
SDR	Software Defined Radio

References

1. Katsalis, K.; Nikaein, N.; Schiller, E.; Ksentini, A.; Braun, T. Network slices toward 5G communications: Slicing the LTE network. *IEEE Commun. Mag.* **2017**, *55*, 146–154. [CrossRef]
2. Afolabi, I.; Taleb, T.; Frangoudis, P.A.; Bagaa, M.; Ksentini, A. Network Slicing-Based Customization of 5G Mobile Services. *IEEE Netw.* **2019**, *33*, 134–141. [CrossRef]
3. Abbas, K.; Afaq, M.; Ahmed Khan, T.; Rafiq, A.; Iqbal, J.; Ul Islam, I.; Song, W.C. An efficient SDN-based LTE-WiFi spectrum aggregation system for heterogeneous 5G networks. *Trans. Emerg. Telecommun. Technol.* **2020**, e3943. [CrossRef]
4. Abbas, K.; Ahmed, K.T.; Rafiq, A.; Song, W.C.; Seok, S.J. An LTE-WiFi Spectrum Aggregation System for 5G Network: A Testbed. In Proceedings of the IEEE 2020 International Conference on Information Networking (ICOIN), Barcelona, Spain, 7–10 January 2020; pp. 753–755.
5. Saqib, M.; Khan, F.Z.; Ahmed, M.; Mehmood, R.M. A critical review on security approaches to software-defined wireless sensor networking. *Int. J. Distrib. Sens. Netw.* **2019**, *15*, 1550147719889906. [CrossRef]
6. Li, X.; Ni, R.; Chen, J.; Lyu, Y.; Rong, Z.; Du, R. End-to-End Network Slicing in Radio Access Network, Transport Network and Core Network Domains. *IEEE Access* **2020**, *8*, 29525–29537. [CrossRef]
7. Pang, L.; Yang, C.; Chen, D.; Song, Y.; Guizani, M. A survey on intent-driven networks. *IEEE Access* **2020**, *8*, 22862–22873. [CrossRef]
8. Wei, Y.; Peng, M.; Liu, Y. Intent-based networks for 6G: Insights and challenges. *Digit. Commun. Netw.* **2020**, *6*, 270–280. [CrossRef]
9. Zeydan, E.; Turk, Y. Recent Advances in Intent-Based Networking: A Survey. In Proceedings of the 2020 IEEE 91st Vehicular Technology Conference (VTC2020-Spring), Antwerp, Belgium, 25–28 May 2020; pp. 1–5.
10. IETF. Intent-Based Networking. Available online: https://tools.ietf.org/html/draft-irtf-nmrg-ibn-concepts-definitions-01 (accessed on 12 August 2020).
11. Cisco. Intent-Based Networking. Available online: https://www.cisco.com/c/dam/en/us/solutions/collateral/enterprise-networks/digital-network-architecture/nb-09-intent-networking-wp-cte-en.pdf (accessed on 10 August 2020).
12. Huawei. Intent-Driven Network. Available online: https://carrier.huawei.com/~/media/CNBG/Downloads/Spotlight/all-cloud-network-towards-5g/idn-en.pdf (accessed on 3 July 2020).
13. Fossati, F.; Moretti, S.; Perny, P.; Secci, S. Multi-resource allocation for network slicing. *IEEE/ACM Trans. Netw.* **2020**, *28*, 1311–1324. [CrossRef]
14. Shen, X.; Gao, J.; Wu, W.; Lyu, K.; Li, M.; Zhuang, W.; Li, X.; Rao, J. AI-assisted network-slicing based next-generation wireless networks. *IEEE Open J. Veh. Technol.* **2020**, *1*, 45–66. [CrossRef]
15. Khan, P.W.; Abbas, K.; Shaiba, H.; Muthanna, A.; Abuarqoub, A.; Khayyat, M. Energy Efficient Computation Offloading Mechanism in Multi-Server Mobile Edge Computing—An Integer Linear Optimization Approach. *Electronics* **2020**, *9*, 1010. [CrossRef]

16. Devlic, A.; Hamidian, A.; Liang, D.; Eriksson, M.; Consoli, A.; Lundstedt, J. NESMO: Network slicing management and orchestration framework. In Proceedings of the 2017 IEEE International Conference on Communications Workshops (ICC Workshops), Paris, France, 21–25 May 2017; pp. 1202–1208.
17. Afaq, M.; Iqbal, J.; Ahmed, T.; Islam, I.U.; Khan, M.; Khan, M.S. Towards 5G network slicing for vehicular ad-hoc networks: An end-to-end approach. *Comput. Commun.* **2020**, *149*, 252–258. [CrossRef]
18. OSM. Open Source Mano. Available online: https://osm-download.etsi.org/ftp/Documentation/201902-osm-scope-white-paper/#!02-osm-scope-and-functionality.md (accessed on 5 June 2020).
19. ONAP. ONAP: Open Networking Automation Platform. Available online: https://www.onap.org/ (accessed on 5 June 2020).
20. Cloudify. Cloudify: A Open Source Network Orchestrator. Available online: https://cloudify.co/ (accessed on 6 June 2020).
21. OpenBaton. OpenBaton: NFV MANO-Based Framework. Available online: https://openbaton.github.io/ (accessed on 6 June 2020).
22. Katsalis, K.; Nikaein, N.; Huang, A. JOX: An event-driven orchestrator for 5G network slicing. In Proceedings of the NOMS 2018-2018 IEEE/IFIP Network Operations and Management Symposium, Taipei, Taiwan, 23–27 April 2018; pp. 1–9.
23. Mosaic5G. Network Sharing Using FlexRAN. Available online: http://mosaic5g.io/flexran/ (accessed on 10 October 2020).
24. Meneses, F.; Fernandes, M.; Corujo, D.; Aguiar, R.L. SliMANO: An expandable framework for the management and orchestration of end-to-end network slices. In Proceedings of the 2019 IEEE 8th International Conference on Cloud Networking (CloudNet), Coimbra, Portugal, 4–6 November 2019; pp. 1–6.
25. Materna. Dataset. Available online: http://gwa.ewi.tudelft.nl/datasets/gwa-t-13-materna/ (accessed on 2 October 2020).
26. Ferrus, R.; Sallent, O.; Pérez-Romero, J.; Agusti, R. On 5G radio access network slicing: Radio interface protocol features and configuration. *IEEE Commun. Mag.* **2018**, *56*, 184–192. [CrossRef]
27. Ferrús, R.; Sallent, O.; Pérez-Romero, J.; Agusti, R. On the automation of RAN slicing provisioning and cell planning in NG-RAN. In Proceedings of the 2018 European Conference on Networks and Communications (EuCNC), Ljubljana, Slovenia, 18–21 June 2018; pp. 37–42.
28. Elayoubi, S.E.; Jemaa, S.B.; Altman, Z.; Galindo-Serrano, A. 5G RAN slicing for verticals: Enablers and challenges. *IEEE Commun. Mag.* **2019**, *57*, 28–34. [CrossRef]
29. Du, P.; Nakao, A. Understanding Intelligent RAN Slicing for Future Mobile Networks Through Field Test. In Proceedings of the 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS), Matsue, Japan, 18–20 September 2019; pp. 1–6.
30. Albonda, H.D.R.; Pérez-Romero, J. An efficient RAN slicing strategy for a heterogeneous network with eMBB and V2X services. *IEEE Access* **2019**, *7*, 44771–44782. [CrossRef]
31. Sciancalepore, V.; Di Renzo, M.; Costa-Perez, X. STORNS: Stochastic radio access network slicing. In Proceedings of the ICC 2019-2019 IEEE International Conference on Communications (ICC), Shanghai, China, 20–24 May 2019; pp. 1–7.
32. Xiang, H.; Peng, M.; Sun, Y.; Yan, S. Mode Selection and Resource Allocation in Sliced Fog Radio Access Networks: A Reinforcement Learning Approach. *IEEE Trans. Veh. Technol.* **2020**, *69*, 4271–4284. [CrossRef]
33. Marabissi, D.; Fantacci, R. Heterogeneous public safety network architecture based on RAN slicing. *IEEE Access* **2017**, *5*, 24668–24677. [CrossRef]
34. Popovski, P.; Trillingsgaard, K.F.; Simeone, O.; Durisi, G. 5G wireless network slicing for eMBB, URLLC, and mMTC: A communication-theoretic view. *IEEE Access* **2018**, *6*, 55765–55779. [CrossRef]
35. Yan, M.; Feng, G.; Zhou, J.; Sun, Y.; Liang, Y.C. Intelligent resource scheduling for 5G radio access network slicing. *IEEE Trans. Veh. Technol.* **2019**, *68*, 7691–7703. [CrossRef]
36. Xiang, H.; Yan, S.; Peng, M. A realization of fog-RAN slicing via deep reinforcement learning. *IEEE Trans. Wirel. Commun.* **2020**, *19*, 2515–2527. [CrossRef]
37. Raza, M.R.; Natalino, C.; Öhlen, P.; Wosinska, L.; Monti, P. Reinforcement learning for slicing in a 5G flexible RAN. *J. Light. Technol.* **2019**, *37*, 5161–5169. [CrossRef]
38. Rafiq, A.; Mehmood, A.; Ahmed Khan, T.; Abbas, K.; Afaq, M.; Wang Cheol, S. Intent-Based End-to-End Network Service Orchestration System for Multi-Platforms. *Sustainability* **2020**, *12*, 2782. [CrossRef]

39. Khan, T.A.; Mehmood, A.; Rivera, J.J.D.; Song, W.C. Machine Learning Approach for Automatic Configuration and Management of 5G Platforms. In Proceedings of the 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS), Matsue, Japan, 18–20 September 2019; pp. 1–6.
40. Khan, T.A.; Mehmood, A.; Ravera, J.J.D.; Muhammad, A.; Abbas, K.; Song, W.C. Intent-Based Orchestration of Network Slices and Resource Assurance using Machine Learning. In Proceedings of the NOMS 2020-2020 IEEE/IFIP Network Operations and Management Symposium, Budapest, Hungary, 20–24 April 2020; pp. 1–2.
41. Khan, T.A.; Afaq, M.; Abbas, K.; Rafiq, A.; Mehmood, A.; Song, W.C. Generic Intent-Based Networking approach for end-to-end Slice Orchestration and Lifecycle Management. *한국통신학회 학술대회논문집* **2020**, 468–469.
42. Foukas, X.; Nikaein, N.; Kassem, M.M.; Marina, M.K.; Kontovasilis, K. FlexRAN: A flexible and programmable platform for software-defined radio access networks. In Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies, Irvine, CA, USA, 6 December 2016; pp. 427–441.
43. Hua, Y.; Li, R.; Zhao, Z.; Chen, X.; Zhang, H. GAN-powered Deep Distributional Reinforcement Learning for Resource Management in Network Slicing. *IEEE J. Sel. Areas Commun.* **2019**, *38*, 334–349. [CrossRef]
44. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. *Adv. Neural Inf. Process. Syst.* **2014**, 2672–2680.
45. OpenAirInterface. OAI: An Open-Source Community. Available online: https://www.openairinterface.org (accessed on 10 June 2020).