scientific reports

Check for updates

OPEN

Slice-aware 5G network orchestration framework based on dual-slice isolation and management strategy (D-SIMS)

Sujitha Venkatapathy[1,5], Thiruvenkadam Srinivasan[2,5], Oh-Sung Lee[3], Raju Jayaraman[2], Han-Gue Jo[4]✉ & In-Ho Ra[4]✉

Network slicing is crucial to the 5G architecture because it enables the virtualization of network resources into a logical network. Network slices are created, isolated, and managed using software-defined networking (SDN) and network function virtualization (NFV). The virtual network function (VNF) manager must devise strategies for all stages of network slicing to ensure optimal allocation of physical infrastructure (PI) resources to high-acceptance virtual service requests (VSRs). This paper investigates two independent network slicing frameworks named as dual-slice isolation and management strategy (D-SIMS) and recommends the best of the two based on performance measurements. D-SIMS places VNFs for network slicing using self-sustained resource reservation (SSRR) and master-sliced resource reservation (MSRR), with some flexibility for the VNF manager to choose between them based on the degree to which the underlying physical infrastructure has been sliced. The present research work consists of two phases: the first deals with the creation of slices, and the second with determining the most efficient way to distribute resources among them. A deep neural network (DNN) technique is used in the first stage to generate slices for both PI and VSR. Then, in the second stage, we propose D-SIMS for resource allocation, which uses both the fuzzy-PROMETHEE method for node mapping and Dijkstra's algorithm for link mapping. During the slice creation phase, the proposed DNN training method's classification performance is evaluated using accuracy, precision, recall, and F1 score measures. To assess the success of resource allocation, metrics such as acceptance rate and resource effectiveness are used. The performance benefit is investigated under various network conditions and VSRs. Finally, to demonstrate the importance of the proposed work, we compare the simulation results to those in the academic literature.

Keywords 5G network, Network slicing, Deep neural network, Resource management, VNF manager

In recent years, fifth-generation (5G) communication networks have changed the ICT landscape. The 5G mobile network aims to support multiple vertical markets efficiently and flexibly, such as ultra-reliable low latency communications (uRLLC), enhanced mobile broadband (eMBB), and massive machine type communications (mMTC)[1]. As shown in Fig. 1, network slicing is an effective method that can be used to learn more about how resources could be sliced in order to meet a wide range of customers' needs[2]. Most research on network slicing in the 5G era has been on either radio access network (RAN) slicing or core network (CN) slicing[3]. Research has been carried out on virtualizing and softwarizing the radio resources for RAN slicing[4], while studies known as end-to-end (E2E) network slicing investigates where Virtual Network Functions (VNFs) should be located with

[1]Information Technology, Kumaraguru College of Technology, Coimbatore, Tamilnadu 641049, India. [2]School of Electrical Engineering, Vellore Institute of Technology, Vellore, Tamilnadu 632014, India. [3]School of Electrical and Electronics Engineering, Kunsan National University, Gunsan 54150, Republic of Korea. [4]School of Computer and Software, Kunsan National University, Gunsan 54150, Republic of Korea. [5]These authors contributed equally: Sujitha Venkatapathy and Thiruvenkadam Srinivasan. ✉email: hgjo@kunsan.ac.kr; ihra@kunsan.ac.kr

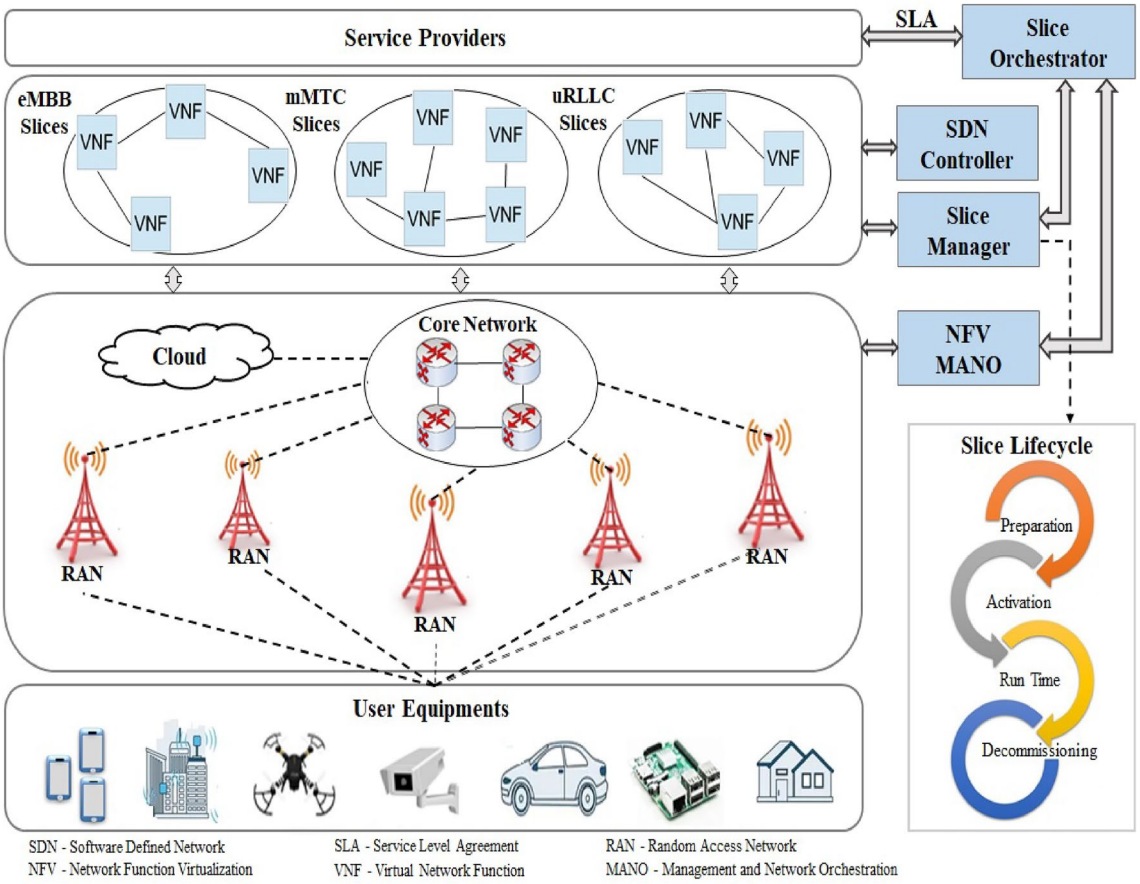

Figure 1. Slice architecture for 5G network.

respect to the underlying actual infrastructure so that individual slices can operate autonomously[5,6]. A study was carried out on the creation of interfaces and protocols for inter-slice communication between RAN and CN[7].

Isolating network slices from one another is an essential requirement for network slicing. This will eventually guarantee that only the VNFs required to run the intended applications are present in each slice. Therefore, related efforts are typically focused on optimizing the deployment of several autonomous virtual networks[8]. For the 5G network as a whole to be able to provide the needed network slice services, each instance of the slice must have all of the relevant VNFs. 5G slicing enables for different degrees of isolation that share network services or functions across numerous slice instances. In the course of the last few decades, a number of different algorithms for embedding VNs have been developed. According to the type of optimization method employed, these VN embedding algorithms can be classified in the cases of heuristic algorithms[9–11], meta-heuristic algorithms[12,13], and accurate algorithms[14,15]. During the node mapping stage, heuristic algorithms primarily rely on node importance ranking techniques to rank the corresponding importance of the substrate nodes and virtual nodes. In both cases, link mapping for the virtual nodes is handled using the K-shortest path technique. Also, some learning-based algorithms are rarely used in the literature when it comes to service provisioning[16] or the creation of slices[17].

As is commonly known, network slicing involves three main components: slice creation, isolation, and management. Many network slicing orchestration frameworks are only focused on a specific network slicing function. It is widely acknowledged that the orchestration framework must still be developed in order to meet the requirements of delivering each service for all three network slicing responsibilities. Furthermore, when VNF managers use a single framework that provides services in all three aspects of slicing, they gain a competitive advantage in providing higher-quality services to their customers. In order to address the issues identified in previous studies, this article provides an orchestration framework that considers all three steps of the network slicing process. The following is a list of the most important contributions that our work has made:

- To the greatest extent of our knowledge, this is the first work to provide a full package for 5G network slicing, covering the three main aspects of network slicing: slice creation, isolation, and management. Coordination between the three components is effectively handled by properly defining the VNF at each component.
- To apply a deep neural network (DNN) training approach for slice creation, the performance dataset is prepared for learning based on the realistic characteristics of diverse use cases.
- To model a slice-aware network orchestration framework for optimal resource allocation based on dual-slice isolation and management strategy (D-SIMS), namely self-sustained resource reservation (SSRR) and master-sliced resource reservation (MSRR). The strategy selection is made based on the degree to which the

underlying physical infrastructure has been sliced. Both of these strategies are capable of handling all three stages of the execution process, which are as follows: (1) slice creation by deep neural network (SC-DNN), (2) slice isolation by hybrid fuzzy-PROMETHEE (SI-FP), and (3) slice management by addressing four responsibilities (SM-4R).

- To prepare strategy specific VNF for addressing four major responsibilities of slice management, such as request report generation (RRG), dynamic resource recovery (DRR), inner-resource sharing (IRS), and priority resource allocation (PRA).
- To make a more informed recommendation regarding the optimal network slicing strategy and PI virtualization. This paper conducts an in-depth examination of two distinct strategies and evaluates them under different physical infrastructure conditions and VSR settings. The optimal strategy for a given system is then adopted.

The literature review for slice creation, isolation and management are carried out independently due to the fact that this work builds the orchestration framework for addressing all three aspects of network slicing.

Slice creation strategies

Machine learning (ML) is a branch of artificial intelligence (AI) in which a machine or system can learn on its own by analyzing the results of previous observations. Growing amounts of data and increased competition have led to the widespread adoption of ML in a wide range of industries and sectors[18,19]. 5G networks are becoming more complicated as the number of newly connected devices and types of services has increased exponentially. ML can be considered for automating various network functionalities such as slice creation, deployment, management, monitoring, fault detection, and security[20,21]. With a focus on ML-based slice creation, a lot of research has been done to show how the ML model can be used with the 5G network. Slicing a 5G network allows for adaptable service delivery to specific use cases. In[22], an effective slice formation mechanism used that supports Key Performance Indicators (KPIs) connected to 5G and ensures resource allocation by modifying the inter and intra-slice allocation methods. Random forest (RF), support vector machine (SVM), k-nearest neighbors (KNN), and decision tree (DT) have been used in[23] to select the best suitable slice in the physical network. According to the results of the study, using RF and KNN to create slices yields more accurate results than the SVM and DT methods. In addition to that, this strategy achieved a high quality of service and service level agreement. In[24], RF shows greater performance while maintaining a high level of accuracy in the classification operation to distribute the slices for the network traffic. In[25] efficient network slicing, ML and hybrid learning models have been used. Both of them significantly improve the network's performance. A deep belief network (DBN) method was used in[26] for slice classification. The slices were classified according to eMBB, mMTC, and uRLLC with regard to the use cases. DNN has been used to build slices that are more efficient and accurate. It uses KPIs to analyze the network traffic and use cases. This method aims to select slices and for slice prediction with resource allocation, slice management, and traffic management in case of network failure.

Slice isolation and management strategies

The authors of[27] developed two node mapping methods with the help of the sequential selection principle and auxiliary graphs, which turned the issue into a traditional graph theory problem. In[28], an actor-critic Reinforcement Learning (RL) model uses five ideal graph characteristics to create the issue environment, which is then changed using a ranking mechanism. For the purpose of virtual network embedding, the network topology attribute and network resource-considered algorithm (VNE-NTANRC) was suggested in[29]. Before embedding any particular VN, the VNE-NTANRC algorithm uses a novel way to rank nodes to determine the order in which all of the substrate and virtual nodes should be ranked. The innovative method for ranking nodes takes into account a total of five significant network topology aspects as well as global network resources. On the basis of the limitations of computational, network, and storage resources[30,31], proposed two fundamental VNE techniques, namely node ranking metric for virtual network embedding (NRM-VNE) and resource consumption ratio for virtual network embedding (RCR-VNE). Authors of[32] suggested a heuristic technique known as information-centric networking-network slice determination and embedding to address challenges associated with network embedding in large-scale networks. In[33], Long Short-Term Memory (LSTM)-based modeling of network slice traffic demands for resource management to tackle the issues of mobility management and non-uniform slice deployment across an identified region. The primary value of the proposed algorithm is that it solves the problem without requiring any background information about the topology of the network or the availability of its resources. In order to distribute the connections and nodes over a network, methods including PROMETHEE-II and SLE are employed[34]. The PROMETHEE-II technique considers a selection of node features, and the SLE method is suggested to collect every possible links for network service request nodes in order to guarantee the shortest path of the network service request. The authors of[35] used an integer linear program (ILP) approach to solve a 5G E2E sharable-VNFs-based multiple linked VNE problem (SVM-VNE). The unique topological features of the 5G network were used to categorize VNFs as either sharable or non-sharable and to determine how much of a difference there would be in the resources needed to support each. By taking advantage of VNFs' inherent capacity for sharing, the suggested work is able to reduce the amount of material resources needed. In[36], the authors examined several metrics for multi-dimensional mapping efficiency and rated their usefulness for heuristic and accurate network service embedding (NSE) approaches. Two heuristics and a mixed integer linear program (MILP) were suggested by the authors based on their study on optimising multidimensional NSE.

In[37], optical network virtualization was implemented, adding an extra layer of complexity due to the optical networks' unique physical characteristics. In addition to providing a user-friendly environment, this unified virtualization technology also provides an abundance of bandwidth resources. In[38], the authors focused solely on the

challenges of modeling resource and energy utilization for IoT applications operating in edge-cloud paradigms. For the purpose of providing a concrete example for the accurate modeling of a service chain, a smart traffic monitoring IP camera system was implemented. The authors then proposed a dynamic energy-aware service function chain (SFC) strategy for edge-to-cloud IoT applications. The goal of the article[39] was to simplify the network topology by integrating data centers and optical networks effectively. This was done by utilizing the parallel transmission properties of optical fiber. The work presented a mathematical model for virtual network embedding with the support of node proximity sensing and path comprehensive evaluation. The authors of the paper[40] developed a model for estimating energy consumption and created an algorithm for energy-aware virtual network migration (EA-VNM). They also developed an enhanced group VN migration algorithm to reduce the time complexity of EA-VNM to a large extent while simultaneously allowing migration to take place in parallel. The most recent contribution made by the authors was a proposal for a prioritized admission control mechanism[41] for concurrent slices, which was based on an approach to infrastructure resource reservation. The reservation takes into account the dynamic nature of slice requests while also being resistant to the unpredictability of slice resource requirements.

Motivation

The most recent works associated with network slicing are compiled in Table 1, which provides a summary of the literature on network slicing components such as slice creation (SC), slice isolation (SI), and slice management (SM). The table also includes the techniques, performance measurements, and task criteria used in the solution process. In most studies, researchers have employed the same metrics for performance measurement of the strategies. Different methodologies and aspects of network slicing have been described in the various pieces of

Ref. No.	Approach	Objective	Technology	SC	SI	SM	RR	IS	PP
[23]	Machine learning	Allocate the slices for service Achieve the QoS and SLA	KNN and RF	✓	–	–	–	–	–
[24]	Machine learning	Allocate the secure slice for network traffic	RF	✓	–	–	–	–	–
[25]	Deep learning	Serve diverse services efficiently over a single infrastructure	NN	✓	–	–	–	–	–
[26]	Machine learning & Deep learning	Maximize the slice classification accuracy	DBN and NN	✓	–	✓	✓	–	–
[40]	Deep Reinforcement learning	High efficiency and reduces end-to-end delay	Deep deterministic policy gradient (E-DDPG)	–	✓	✓	–	–	–
[27]	Sequential selection & Auxiliary graphs	Resource consumption Availability gap Blocking probability SLA violation penalty	ILP AI based node mapping K shortest path algorithm	–	✓	–	–	–	–
[29]	Direct node ranking, Stable node ranking	Average revenue to cost ratio Average node utilization Average link utilization	Greedy node mapping SP link mapping	–	✓	–	–	–	–
[30]	3-D resource constraints, Node ranking	VNR acceptance ratio Revenue to cost (R/C) ratio	NRM-VNE RCR-VNE	–	✓	–	–	–	–
[32]	Integer Linear Program	Service quality Resource consumption	Greedy-based ICN-NS-DE ICN-GW mapping ICN-CR generating and mapping	–	✓	–	–	–	–
[35]	Sharable virtual network functions	Slice acceptance ratio Physical resource utilization	ILP formulation Back-tracking coordinated virtual network mapping	–	✓	✓	✓	–	–
[36]	Mixed integer linear program	Slice acceptance ratio Physical resource utilization CPU and memory utilization	Multi-Dimensional heuristic Heuristic baseline greedy VNF bundling	–	✓	✓	✓	–	✓
[37]	Path growing approach, Decomposition approach	Resource usage blocking ratio	Branch-and-bound	–	✓	✓	✓	–	–
[38]	Service Function Chain	Resource usage Energy efficiency	Heuristic resource Energy-efficient SFC embedding strategy	–	✓	✓	✓	–	✓
[39]	Routing and wavelength assignment	Acceptance rate Revenue-over-head ratio	Priority of location - VNE	–	✓	✓	✓	–	✓
[40]	Migration technique	Energy cost CPU and BW utilization	EA-VNM EA-VNM-G	–	✓	✓	✓	✓	–
[41]	Prioritized admission control mechanism	Acceptance rate Response delay Total cost	Max-min optimization Slice resource reservation	–	✓	✓	✓	–	✓
Proposed	MCDM approach, Deep learning, Slice aware resource allocation	Slice creation accuracy Slice acceptance ratio Physical resource efficiency CPU and BW utilization	DNN PROMETHEE-II Fuzzy min-max scaler Dijkstra's SP algorithm	✓	✓	✓	✓	✓	✓

Table 1. Summary of literature.

literature, leading to inconsistencies in the field. According to our research and analysis of the relevant literature, following is the summary and limitations of the aforementioned works on network slicing:

- The majority of the works listed in the table offer approaches for slice creation [18–21], while some works focus solely on slice isolation [22–26] and others cover both slice isolation and slice management [27–33].
- When it comes to resource allocation, the literature [27–29] focusing on slice management only offer solutions for resource recovery (RR), taking into consideration the life-time of VSRs.
- Many previous works have allotted resources to a fixed assignment by making the assumption that every application of a VSR is one of a kind. In order to prevent unsuccessful use cases, the monitoring of the accessible resources within the other slices of data that are of a different kind has been disabled.
- Because of the aforementioned assumption for slice isolation, almost no works have adopted the concept of inner-slicing (IS) in order to supply resources for unsuccessful use case, with the notable exception of [31]. The VNF should be defined into individual slices so as to find a way to allocate resources for different types of failed use cases on a priority basis.
- Furthermore, priority provisioning (PP) is rarely discussed in the literature. There is no provisioning of VNF for the individual slices to make space for resources supporting priority services. It is the responsibility of the VNF in each slice to effectively manage resource contention while making allocation attempts for critical services and unsuccessful use cases.

This led us to conclude that it is important to create an orchestration framework that can handle all three aspects of network slicing. As a result, each part of network slicing is tackled with the most effective methods in the suggested framework. The foundational structure of the new work that is being proposed has been developed to address the shortcomings of the previous works. The accuracy, precision, recall, and F1 score are used to evaluate the classification performance of the proposed DNN training method during the slice generation phase. Acceptance rate and resource effectiveness are two metrics used to evaluate the efficiency of resource allocation.

Paper organization

The rest of this article is organized as follows: "System model" section summarizes the proposed system model; "Mathematical background" section provides the mathematical background; "Proposed framework" section describes the proposed 5G network framework; "Simulation results" section discusses simulation results with case study; "Conclusion" section concludes the work.

System model

Physical infrastructure

A weighted undirected graph is used to describe the topological structure of the physical infrastructure (PI). The graph is represented as $G_P = (N_P, E_P)$, where N_P represents a collection of PI nodes and E_P indicates a collection of physical links. In a graph, every node should have a CPU capacity (C), a link capacity (LC), a security level (SL), a delay rate (DR), and a jitter value (JT). As an example, the parameters of the i^{th} node of a graph is specified as a subset containing $(C_P^i, LC_P^i, SL_P^i, DR_P^i, JT_P^i)$, $i \in N_P$. Each edge in a graph is also given a subset of information about its associated nodes, denoted by $E_P^i(BW_P^{ij}, L_P^{ij})$, where BW_P^{ij} is the bandwidth of the link connecting node i and j in PI, L_P^{ij} is the length of the link connecting node i and j in PI.

Virtual service request

Each VSR needs to meet requirements for the number of nodes, CPU power, bandwidth, security, the lifespan, and types of user devices. Every VSR is modeled as an undirected graph $G_V = (N_V, E_V)$ where N_V represents the nodes of the VSR and E_V represents the edges of the VSR. Each node of a VSR should have $(C_V^m, LC_V^m, SR_V^m, LT_V^m, DT_V^m)$, $m \in N_V$ where $C_V^m, LC_V^m, SR_V^m, LT_V^m$ and $DT_V^m$are the required CPU capacity, link capacity, security requirement, life-time and device type of m^{th} node, respectively. Similarly, we can denote the bandwidth requirement for the link between nodes m and n as BW_V^{mn}. We assume that the VSRs are received during a single transmission window[42]. The VSRs make use of the physical infrastructure at regular intervals in order to assign nodes and establish connections. The LT_V of the previous request is used to determine how the physical infrastructure's available resources should be adjusted whenever a new VSR is received.

Mathematical background

Variables

π_i^m : A binary variable with a value equals 1 if the mapping of virtual network node m to physical network node i was successful, and 0 otherwise.

f_{ij}^{mn} : A binary variable whose value is 1 if the physical network path (i, j) accommodates the virtual network link (m, n) successfully and whose value is 0 if the link mapping is unsuccessful.

l^{mn}: A variable that will indicate the shortest path available on the virtual link (m, n).

Objective

$$Minimize, \sum_{m \in N_V} \sum_{i \in N_P} \pi_i^m (C_V^m + LC_V^m) + \sum_{(m,n) \in E_V} \sum_{(i,j) \in E_P} f_{ij}^{mn} . BW_V^{mn}(l^{mn}) \tag{1}$$

Constrains

Node processing constraints

$$\pi_i^m \times C_V^m \leq C_P^i, \quad \forall m \in N_V, \forall i \in N_P \tag{2}$$

$$\pi_i^m \times LC_V^m \leq LC_P^i, \quad \forall m \in N_V, \forall i \in N_P \tag{3}$$

$$\pi_i^m \times SR_V^m \leq SL_P^i, \quad \forall m \in N_V, \forall i \in N_P \tag{4}$$

Link associated constraints

$$f_{ij}^{mn} \times BW_V^{mn} \leq BW_P^{ij}, \forall m, n \in E_V, \forall i, j \in E_P \tag{5}$$

$$\sum_{(i,j) \in E_P} f_{ij}^{mn} - \sum_{(j,i) \in E_V} f_{ji}^{mn} = \begin{cases} BW_V^{mn}, & \pi_i^m = 1 \\ -BW_V^{mn}, & \pi_i^n = 1 \\ 0, & otherwise \end{cases} \tag{6}$$

$\forall i, j \in E_P, \forall m, n \in E_V$

Variable constraints

$$\sum_{i \in N_P} \pi_i^m = 1, \forall i \in N_P \tag{7}$$

$$\sum_{m \in N_V} \pi_i^m = 1, \forall m \in N_V \tag{8}$$

$$\pi_i^m \in (0, 1), \forall i \in N_P, \forall m \in N_V \tag{9}$$

$$f_{ij}^{mn} \in (0, 1), \forall i, j \in E_P, \forall m, n \in E_V \tag{10}$$

Statements: The objective function (1) seeks to minimize the cost of the D-SIMS to accommodate the virtual request nodes. The (2) constraint ensures that each PI node $i \in N_P$ has enough CPU capacity to meet the needs of each virtual node $m \in N_V$. The (3) constraint guarantees that the requested link capacity of a virtual node $m \in N_V$ can be met by the link capacity of a PI node $i \in N_P$. Security level of the virtual node $m \in N_V$ can be satisfied by the security measures of the PI node $i \in N_P$, according to constraint (4). The bandwidth of the PI node $i \in N_P$ must meet the bandwidth requirements of the virtual node $m \in N_V$ according to constraint (5). The connectivity constraint is represented by Eq. (6). Mapping virtual links (m, n) to their physical links (i, j) is carried out in link mapping stage. If the outflow from the source PI node $i \in N_P$ is 1, but the inflow from node $j \in N_P$ is 0, the result is $\sum_{(i,j) \in E_P} f_{ij}^{mn} - \sum_{(j,i) \in E_P} f_{ji}^{mn} = 1$; in the destination PI node $i \in N_P$, the outflow value is 0, while the inflow value is 1, then $\sum_{(i,j) \in E_P} f_{ij}^{mn} - \sum_{(j,i) \in E_P} f_{ji}^{mn} = -1$; in the other PI node the outflow and inflow value is 1, then $\sum_{(i,j) \in E_P} f_{ij}^{mn} - \sum_{(j,i) \in E_P} f_{ji}^{mn} = 0$. Constraints (7) and (8) ensure that a PI node can only host a virtual node from the same VSR, and that a virtual node can only be mapped onto a PI node. π_i^m and f_{mn}^{ij} are assured to be two binary variables according to constrains (9) and (10).

Performance metrics

We define a set of performance criteria to evaluate the proposed techniques[43], which include the classification accuracy (A), precision (P), recall (R), F1 score (FS), resource efficiency (RE), resource utilization ($CPU_{utilized}$ and $BW_{utilized}$) and acceptance ratio (AR).

Classification

Accuracy, precision, recall, and F1 score metrics are used to evaluate the classification performance of the predicted model. The metrics are calculated with the help of the following four indicators: (1) True positives (*TP*), (2) False positives (*FP*), (3) False negatives (*FN*), (4) True negatives (*TN*)[44]. The process of determining a classification accuracy, precision, recall and F1 score is denoted as,

$$A = \frac{TP + TN}{TP + FP + FN + TN} \tag{11}$$

$$P = \frac{TP}{TP + FP} \tag{12}$$

$$R = \frac{TP}{TP + FN} \tag{13}$$

$$FS = 2 * \frac{P * R}{P + R} \tag{14}$$

Resource utilization
Total CPU capacity, link capacity and bandwidth can be computed by summing the utilized quantities of each successful VSR (SV). The total utilized CPU capacity and bandwidth of each successful VSR are stored in TC_k, TLC_k and TBW_k, respectively, after the VSR has been successfully provisioned in PI.

$$CPU_{utilized} = \sum_{k=1}^{SV} TC_k \tag{15}$$

$$LC_{utilized} = \sum_{k=1}^{SV} TLC_k \tag{16}$$

$$BW_{utilized} = \sum_{k=1}^{SV} TBW_k \tag{17}$$

Resource efficiency
Resource efficiency can be measured using the revenue-to-cost ratio. The total of the node capacity, link capacity and link bandwidth needs of an VSR can be used to calculate the revenues from adopting an VSR. A network's overall cost is equal to the total cost of its node capacity, link capacity, and link bandwidth. Resource efficiency is represented as,

Modified the below equation

$$RE = \frac{\sum_{n=1}^{SV} TC_n + \sum_{n=1}^{SV} TLC_n + \sum_{n=1}^{SV} TBW_n}{\sum_{n=1}^{SV} TC_n + \sum_{n=1}^{SV} TBW_n * (L_{SV})} \tag{18}$$

where L_{SV} represents the mapping length of each VSR's shortest path which has been retrieved at the end of link provisioning.

Acceptance ratio
To calculate this metric, we divide the number of successful VSRs by the total number of VSRs that occurred within the chosen transmission time interval T_{max}.

$$AR = \lim_{0 \to T_{max}} \frac{SV}{TV} \tag{19}$$

where TV denotes total VSR.

Proposed framework

For the purpose of network slicing, the proposed framework suggests two different alternative strategies, which are referred to as SSRR and MSRR of D-SIMS. The VNF manager chooses one of these strategies based on the manner in which the PI is sliced. When the PI is virtualized as three slices, the VNF manager chooses the SSRR strategy. On the other hand, the MSRR strategy is chosen when the PI is virtualized as four slices. As can be seen in Fig. 2, both approaches are broken down into the same set of three distinct phases, which are labeled as slice creation, isolation, and management. Each of these phases is in charge of its own specific group of responsibilities, and together they make up the overall process. However, the architecture's phases are interdependent and must be carried out in sequence. In the slice creation phase, the slices are generated for the PI and received by the VSR, which gets the initial state of PI as the input at $t = 0$ and gets the information about updated states of PI from the slice management phase while getting the next VSR. The created slices of PI and VSR are transferred to the slice isolation phase, where a VNF placement method is applied to allocate resources. The slice management phase is commenced when any of the VSRs fails, where the effective management approaches are applied to make unsuccessful VSRs successful. This phase also incorporates resource sharing for slice management after

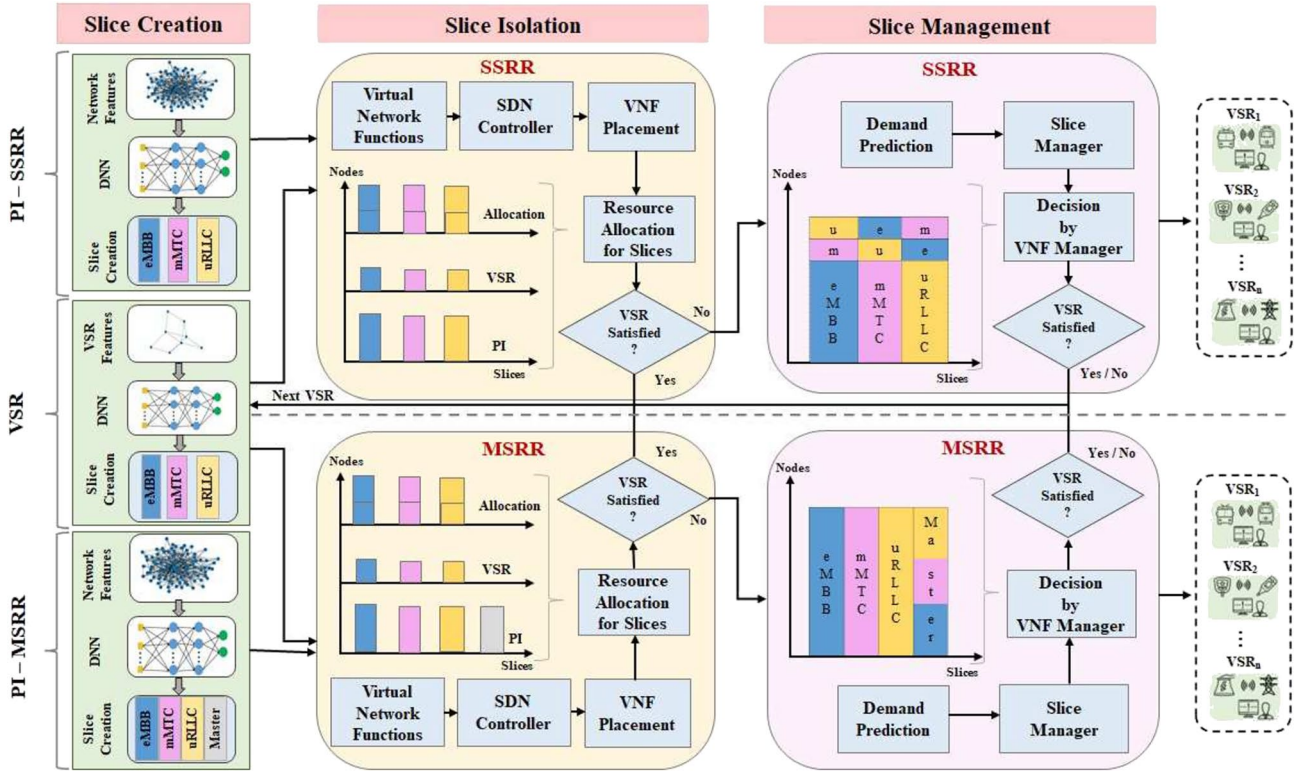

Figure 2. Proposed architecture for D-SIMS.

combining the results of the previous two phases. The following subsections describe the processes involved in each phase of the strategies.

DNN for slice creation

DNN is a kind of ML in which the system makes use of several layers of nodes to extract higher-level functions from lower-level data. The construction of DNN is built on layer connections via nodes known as neurons, analogous to the biological neurons in the brain. DNN has three layers, namely the input layer, two or more hidden layers, and the output layer. As an input, the KPI of the user devices is provided, and for each of the two hidden layers, 15 neurons are taken into consideration. An associated connection weight, w_j, is assigned to each interconnection between two neurons. This weight is fitted throughout the learning phase of the process. In order to determine its output, each neuron first computes a weighted sum of its inputs. In addition, the bias value is added along with the cumulative input. Weights are optimized with the assistance of a variety of methods, including stochastic gradient descent (SGD), limited-memory broyden-fletcher goldfarh shanno (LBFGS), and adaptive moment estimation (ADAM). In order to get the intended outcome for the neuron, the activation function applies itself to the cumulative input value that is generated via the additive function. In this situation, the sigmoid activation function, which is a continuous, nonlinear, and straightforward derivative function, is utilized.

The output neurons of the hidden layers are calculated using,

$$a_i = \sum_{j=1}^{n} (w_j.x_j) + b \tag{20}$$

where x_j is the KPIs of node, b is the bias, and w_j is the weight of connection links. The output of the i^{th} node in the hidden layer is,

$$y_i = f(a_i) \tag{21}$$

where f is the sigmoid activation function. Final output of all neurons in the output layer $y^{\wedge}$ is denoted as,

$$y^{\wedge} = f(\sum_{i=0}^{m} w_i y_i^H) \tag{22}$$

Due to differences in how they virtualize the underlying physical infrastructure, the proposed work creates slices independently for each of the two D-SIMS. The SSRR strategy has been sliced into three parts: eMBB, mMTC, and uRLLC, while the MSRR strategy has been sliced into four parts: eMBB, mMTC, uRLLC, and a master slice. A performance dataset with input and label is prepared independently for each approach. Slices in SSRR are made using each node's KPIs. Since the master slice nodes in MSRR must be shared across all possible user requests

during critical conditions, the rules for these nodes are designed to preserve those with the highest KPI values. The remaining PI nodes are divided into three groups based on their KPI values. Classification is performed using deep neural network. Accuracy is calculated to measure the performance of classification. Based on classification, three groups are formed in SSRR and four groups in case of MSRR.

Slice isolation of D-SIMS

We comprehend that allocating resources to the VSR nodes in the PI involves two sequential operations, namely, node provisioning and link provisioning. To ensure the throughput, CPU capacity, and VSR security service level agreements (SLAs) are met, it is important to validate the allotted nodes and links. Additionally, the distribution of resources should maximize revenue and efficient resource utilization. In order to meet those requirements, this work proposes a hybrid fuzzy-PROMETHEE method to perform node provisioning and Dijkstra's algorithm for link provisioning.

Node mapping through fuzzy-PROMETHEE

Most studies have found that feature extraction from individual nodes is the most effective way for analysing network nodes. In order to determine a node's ranking, we take into account factors like its CPU capacity (23), degree (24), bandwidth (25), and closeness centrality (26).

$$CN^i = C^i \quad ; \quad \forall i \in N_P, N_V \tag{23}$$

$$DN^i = \sum_j E^{ij} \quad ; \quad i \neq j \quad \& \quad \forall j \in N_P, N_V \tag{24}$$

$$BN^i = \sum_j BW^{ij} \quad ; \quad i \neq j \quad \& \quad \forall j \in N_P, N_V \tag{25}$$

$$LN^i = \left\{ \sum_j L^{i,j} \right\}^{-1} \quad ; \quad i \neq j \quad \& \quad \forall j \in N_P, N_V \tag{26}$$

where C^i is the CPU capacity of node i in the network, E^{ij} is a binary value that is 1 if nodes i and j are connected and else 0, Shortest path length between nodes i and j is denoted by $L^{i,j}$ where BW^{ij} is the bandwidth of the connection between nodes i and j.

These four variables have different relative importance's and need be normalised before being ranked. In order to meet this demand, a min-max scalar based on fuzzy rules was developed for each of the four parameters. When applied to the parameters, the common membership function can be written as,

$$\mu_{Y,i} = \begin{cases} 1 & if\,(Y_i \geq Y_{max}) \\ \frac{Y_{max} - Y_i}{Y_{max} - Y_{min}} & if\,(Y_{min} \leq Y_i < Y_{max}) \\ 0 & if\,(Y_i < Y_{min}) \end{cases} \tag{27}$$

Where, Y_i refers to any parameter of a node 'i'. The lower and upper bounds (Y_{min} and Y_{max}) of the respective node parameters are identified from the network information. For instance, when calculating a node's bandwidth parameter (BN_i), the value is calculated by first finding by the node with the lowest available bandwidth in the network BN_{min}, and then the value is finding the node with the largest available bandwidth BN_{max}. The network characteristics are also used to derive minimum and maximum values for the other node properties. After computing the membership values of every node in both the PI and VSR networks, the PROMETHEE-based multi-criteria decision making[45] is immediately put into effect, and the ranking of the nodes is done. In this approach, the parameters of each node are considered separately, based on their interests and individual preferences. The steps of the PROMETHEE-II method are as follows:

Step 1: Preparation of evaluation table. Each node in the evaluation table has its fuzzy membership values saved in the evaluation Table 2. Assumed here $N = [n_1, n_2, ..., n_l]$ and $X = [X_{CN}, X_{DN}, X_{BN}, X_{LN}]$ are the sets of nodes and membership values, respectively.

	X_{CN}	X_{DN}	X_{BN}	X_{LN}
n_1	$X_{CN,1}$	$X_{DN,1}$	$X_{BN,1}$	$X_{LN,1}$
n_2	$X_{CN,2}$	$X_{DN,2}$	$X_{BN,2}$	$X_{LN,2}$
.	.	.	.	.
.	.	.	.	.
n_l	$X_{CN,l}$	$X_{DN,l}$	$X_{BN,l}$	$X_{LN,l}$

Table 2. Evaluation table.

Step 2: Preparation of preference function. In a network, every node is compared to every other node in the network using a pairwise comparison for every parameter in the network. For example, the pairwise comparison of node degree parameter is done by,

$$P_{CN,ij} = X_{CN,i} - X_{CN,j} \tag{28}$$

Step 3: Building a global preference index.

$$\pi(i,j) = \sum_{k=1}^{4} P_{CN,ij} * w_k \tag{29}$$

where w_k is an assumed nonzero weight function of component k and the total weights add up to 1.

Step 4: Outranking flows, all positive and negative, are computed. Calculations are performed using the following two factors in order to determine the location of each node in relation to all of the other nodes.

$$\phi^+(i) = \frac{1}{l-1} \sum_{x \in n_i} \pi(i,x) \tag{30}$$

$$\phi^-(i) = \frac{1}{l-1} \sum_{x \in n_i} \pi(x,i) \tag{31}$$

where $\phi^+(i)$ is obtained by ranking nodes by the absolute value of the positive flow to each one and $\phi^-(i)$ is determined by ranking the nodes by the absolute value of the negative flow to each one.

Step 5: Determination of net flow.

$$\phi(i) = \phi^+(i) - \phi^-(i) \tag{32}$$

Require: Physical and Virtual nodes $G_P(t), G_V(t)$, life time $t = 0$
Ensure: Physical nodes $G_P(t)$, Successful Virtual nodes SV

```
Calculate membership values μ_CN, μ_DN, μ_BN, μ_LN for all nodes of G_P(t), G_V(t)
Prepare NR_P(t) using PROMETHEE-II
Prepare NR_V(t) using PROMETHEE-II
Set nodeCount=0
for each node j of NR_V(t) do
    if (NR_V(t) ≠ empty) then
        for every node k of NR_P(t) do
            if (Is constraints (2) to (4) satisfied?) then
                Allocate N_P(t,k) to N_V(t,j)
                Update G_P(t)
                Increment nodeCount
            else
                Increment j
            end if
        end for
    end if
end for
if (N_V(t) == nodeCount) then
    Increment SV
end if
Calculate TC_SV
return G_P(t)
```

Algorithm 1. Node mapping through hybrid fuzzy-PROMETHEE

The most value of $\phi(i)$ has been considered as the best node in the node ranking (NR). Algorithm 1 provides a step-by-step breakdown of the hybrid fuzzy-PROMETHEE[46] procedure applied for mapping nodes. The algorithm checks for the violation of the constraint while it is attempting to map nodes. However, node mapping is only attempted if the constraints are satisfied. At the end of execution, the algorithm returns the nodes that were allocated in PI for VSR. This algorithm is carried out each time a new VSR is received for the purpose of node provisioning.

Link mapping strategy
The primary objective of this strategy is to locate the shortest path that the VSR nodes in the physical infrastructure can take to reach one another. When it comes to VSR, it is abundantly clear that there are situations in which the shortest and most direct route is not the one that should be taken. Under certain conditions, a connection on the shortest route that has been chosen can be linked to the VSRs that have already been provided. Due to this, it is absolutely necessary to find all of the connections between VSR nodes and then put them in Dijkstra's shortest path array (DSPA)[47] in a way that doesn't go down in terms of the total length of the individual paths that connect each node. This array has the potential to be utilized in the process of establishing connections for VSRs. The shortest paths will likely be referred to in order as they are built during the process of establishing connections. When no links are being provisioned at the present, the DSPA will select the path with the shortest overall length. After the successful provisioning of links, it is able to calculate the total bandwidth utilized by the VSR (TBW_{SV}) and shorest path length L_{SV}.

Resource provisioning strategy
Assigning a node and establishing a link between the nodes of the VSRs received between 0 and T_{max} is the essence of this process comprises. In the proposed work, each VSR is modelled as a set of eMBB, mMTC, and uRLLC use cases in order to build a realistic dynamic environment. As a result, slicing the VSR and the PI prior to allocating resources is an absolute requirement. The proposed orchestration architecture comprises a slice creation step that feeds information into the resource allocation phase. This means that during the resource allocation phase, it is guaranteed that all necessary VSR nodes are grouped into eMBB, mMTC, or uRLLC nodes. The proposed resource distribution has been modeled by mapping the VSR slices to the relevant sliced PI. A VSR should be able to obtain all kinds of slice in any quantity without exceeding the maximum number of nodes per request. The whole process involved in the resource allocation phase is presented below in the form of Algorithm 2.

Require: Physical nodes G_P, maximum lifetime T_{max}, t=0
Ensure: Virtual Network Embedding
$G_P(0) = G_P$
for $t < T_{max}$ **do**
  Get VSR(t)
  Call Algorithm 1 for node mapping
  Apply DSPA method for link mapping
  for each path (DSPA) **do**
    if (DSPA $\neq$ empty) & (constraints (5)-(6) satisfied) **then**
      do link mapping in $G_P(t)$
      break
    else
      increment p
    end if
  end for
end for
$Calculate\ AR, RE, CPU_{utilized}, BW_{utilized}$

Algorithm 2. Resource provisioning

Slice management of D-SIMS

This slice management phase is the final phase of the proposed orchestration framework, which has been initiated according to the input received from the slice isolation phase. D-SIMS offers two strategies, namely self-sustained resource reservation (SSRR) and master slice resource reservation (MSRR). The VNF manager determines the best way to handle unsuccessful use cases in the self-organized slices, which are regarded as the immediate reserves available for service provisioning, in accordance with the SSRR strategy.

When using the master slice as the immediate reserve for service provisioning, the VNF manager accommodates unsuccessful use cases in accordance with the MSRR strategy. Both the slice management strategies find the best possible way to accommodate resources for the unsuccessful VSR from the available resources. The strategy between the two is selected by the VNF manager according to the virtualization of the PI. The VNF manager handles four responsibilities (4R) to manage the resource allocation, such as inner-resource sharing (IRS), priority resource allocation (PRA), dynamic resource recovery (DRR), and request report generation (RRG). Figure 3 describes all four responsibilities of a VNF manager working under two different strategies.

Inner-resource sharing
In this case, the unsuccessful VSR nodes are provided for resource allocation in the 'alien-slice'. The term 'alien-slice' is defined as the slice which accommodates its resources for the use case of different kinds referred as 'alien-nodes'. An Alien-slice provides a possible kind of 'quarantine' for the alien-nodes for a short span of time. Alien-nodes can complete their 'quarantine' whenever their associated slice has sufficient resources. The way that

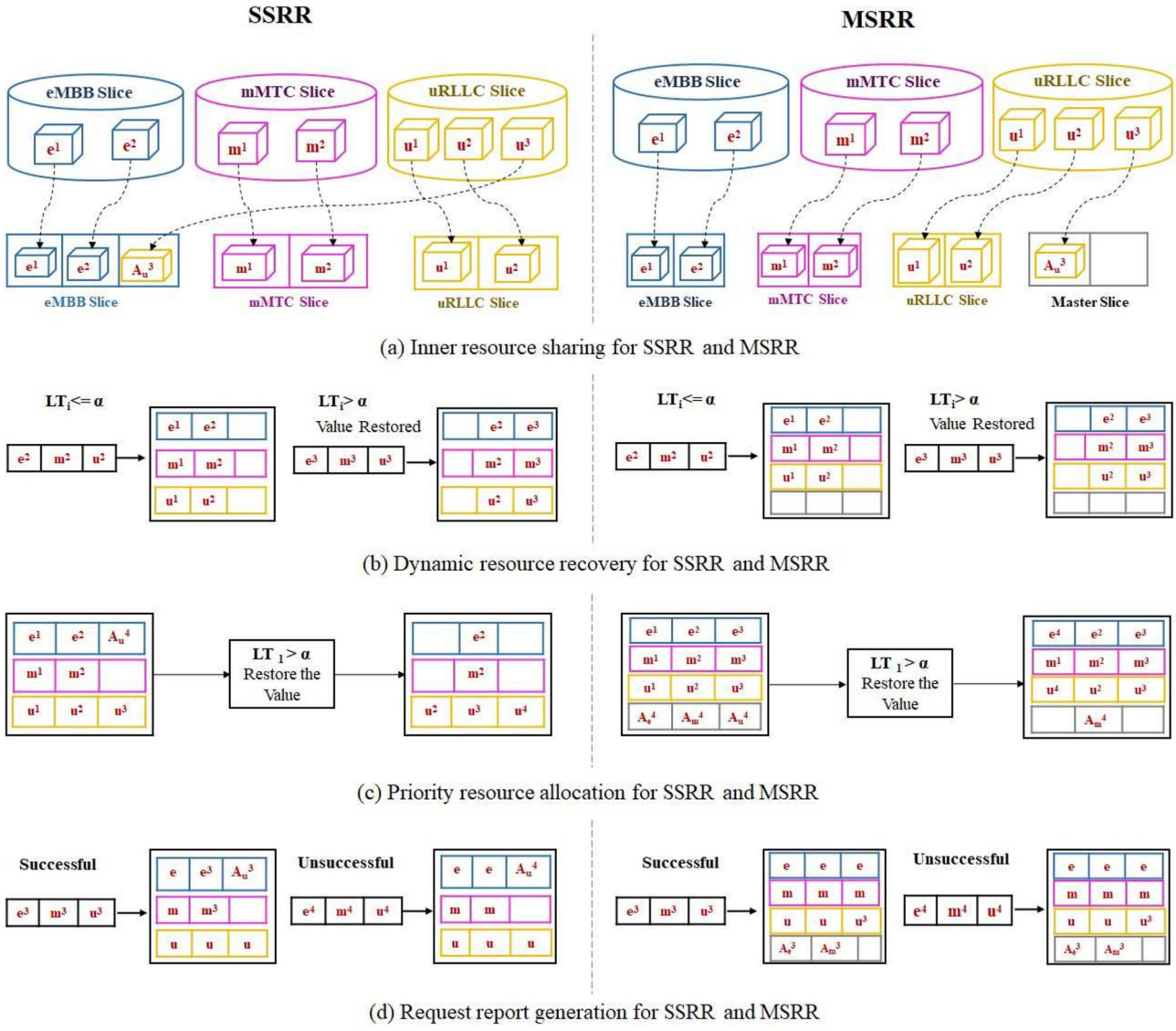

Figure 3. Four responsibilities of VNF manager in D-SIMS.

alien-slice is taken into account depends on the PI virtualization and strategy that have been chosen. Attempts to prioritize the allocation of resources for unsuccessful use cases necessitate defining the VNF in terms of individual slices. VNF takes on the responsibility of effectively managing resource contention while attempting to allocate resources for critical services and unsuccessful use cases as defined. According to the SSRR strategy, alien-slices are those that can accommodate foreign nodes. In contrast, for MSRR, a master-slice is viewed as a universal alien-slice that can serve any of the three possible use cases by treating them as alien-nodes. As shown in Fig. 3a, when slice 'uRLLC' lacks resources, use case 'u^3' transforms into an alien node 'A_u^3' and obtains the reserved resource from an alien slice under SSRR, while under MSRR, use case 'u^3' obtains the reserved resource from the master slice, which itself functions as an alien slice. The allocation of resources for the alien-nodes in the alien-slices is done by using the following equation:

$$SSRR, IRS = \begin{cases} F_e, & R_{e,V} \Rightarrow R_{m,P} \| R_{u,P} \\ F_m, & R_{m,V} \Rightarrow R_{e,P} \| R_{u,P} \\ F_u, & R_{u,V} \Rightarrow R_{e,P} \| R_{m,P} \\ Failed, & Otherwise \end{cases} \tag{33}$$

$$MSRR, IRS = \begin{cases} F_e, & R_{e,V} \Rightarrow R_{ms,P} \\ F_m, & R_{m,V} \Rightarrow R_{ms,P} \\ F_u, & R_{u,V} \Rightarrow R_{ms,P} \\ Failed, & Otherwise \end{cases} \tag{34}$$

where, F_e, F_m, and F_u refer to the failed eMBB, mMTC, and uRLLC nodes, respectively. $R_{e,V}, R_{m,V}, andR_{u,V}$ represent the requested resource of eMBB, mMTC, and uRLLC virtual nodes. $R_{e,P}, R_{m,P}, andR_{u,P}$ show the available resources of eMBB, mMTC and uRLLC nodes in PI. $R_{e,ms}$ denotes available resources of master slice in PI.

Dynamic resource recovery

In this particular scenario, the VNF manager keeps a close eye on the duration of life for each of the individual use cases that have been provisioned at the PI in the past. When it is determined that the use cases do not require the use of PI resources, those resources that were occupied by the use cases are released back into availability. It has been realized that the recovery would benefit from more dynamics if life-time was granted to individual use cases rather than VSRs. Also, in practice, the intended life-span of each individual use case will be different. According to Fig. 3b, for both strategies, the occupied resources of the use cases 'e^1', 'm^1', and 'u^1' are released when it is realized that they have reached the end of their life-time and no longer require service. For the VSR with a life-time LT, the dynamic recovery of the resources is performed by the following equations:

During the time interval $0 < t < LT$,

$$\sum_{i=1}^{n} R_P^i = \sum_{i=1}^{n} R_P^i - \sum_{i=1}^{n} R_V^i \tag{35}$$

when $t > LT$,

$$\sum_{i=1}^{n} R_P^i = \sum_{i=1}^{n} R_P^i + \sum_{i=1}^{n} R_V^i \tag{36}$$

where, R_P^i represents the resources of PI nodes and R_V^i represents the required resources of virtual request.

Priority resource allocation

This case manages two kinds of priority provisioning that are alien-centric priority service (ACPS) and latency-centric priority service (LCPS). The ACPS holds that because the quarantine period for the alien-nodes in the alien-slices cannot be extended, serving the alien-nodes must come before serving the new VSR. LCPS, on the other hand, thinks that when it comes to provisioning for nodes, uRLLC services should be given top priority over all other use cases. When the SSRR strategy determines that the resources used by the use cases e^1, m^1, and u^1 have expired, the corresponding alien-nodes A_u^3 in the alien slice are relocated to the appropriate slice (see Fig. 3c). To a similar extent, the MSRR method brings back the master slice's foreign nodes as needed during the course of the use cases' lifetime. The equation that corresponds to the addressing of two different kinds of priority service provisioning is as follows,

$$PRA = \begin{cases} Restoration, & \sum_{i=1}^{n} A_e^i \Rightarrow R_{e,P}^i \| \\ & \sum_{i=1}^{n} A_m^i \Rightarrow R_{m,P}^i \| \\ & \sum_{i=1}^{n} A_u^i \Rightarrow R_{u,P}^i \\ No, & Otherwise \end{cases} \tag{37}$$

where, Restoration indicates that the value of any alien-nodes are restored if possible when $LT > t$ for any of the previous requests. LT indicates life-time of the previous request node and t denotes dynamic value of life-time. A_e^i, A_m^i and A_u^i represents the alien-nodes of eMBB, mMTC and uRLLC nodes.

Request report generation

As this is the case, the VNF manager needs to send the final report of the VSR, regardless of whether it is successful or not. The VNF manager should, at the very least, attempt to assign resources in order to avoid unsuccessful situations. The unsuccessful VSRs will only be realized by the strategies if the PI does not have sufficient resources. According to Fig. 3d, the report is unsuccessful for both strategies because it is discovered that the newly arrived use cases 'e^4', 'm^4', and 'u^4' cannot be provisioned in the physical resources due to resource inadequacy. The following equations are used to compile the information for the report.

$$RRG = \begin{cases} 1, & R_{e,V} \le R_{e,P} \& R_{m,V} \le R_{m,P} \& R_{u,V} \le R_{u,P} \\ 0, & Otherwise \end{cases} \tag{38}$$

$$SSRR - RRG = \begin{cases} 1, & R_{e,V} \le R_{e,P} \| R_{m,P} \| R_{u,P} \& \\ & R_{m,V} \le R_{m,P} \| R_{e,P} \| R_{u,P} \& \\ & R_{u,V} \le R_{u,P} \| R_{m,P} \| R_{e,P} \\ 0, & Otherwise \end{cases} \tag{39}$$

$$MSRR - RRG = \begin{cases} 1, & R_{e,V} \le R_{e,P} \| R_{ms,P} \& \\ & R_{m,V} \le R_{m,P} \| R_{ms,P} \& \\ & R_{u,V} \le R_{u,P} \| R_{ms,P} \\ 0, & Otherwise \end{cases} \tag{40}$$

where, the value '1' indicates that resource allocated for virtual request node successfully and 0 represents resource allocation is failed.

Simulation results

The suggested dual slice isolation and management strategies, called SSRR and MSRR, are tested separately under a variety of set parameter conditions to get a full understanding of how well they work. Both of these strategies handle all three phases of the execution process, which are as follows: (1) slice creation by deep neural network (SC-DNN), (2) slice isolation by hybrid fuzzy-PROMETHEE (SI-FP), and (3) slice management by four responsibilities (SM-4R). Through the use of the parameters associated with each approach, the efficacy of the approaches used in various phases is evaluated. Table 3 includes a list of the variables taken into account for the test case, including the variety of physical network resources and the VSR for the test scenario. The datasets are classified into three or four categories based on their methods (SSRR and MSRR) : eMBB, mMTC, uRLLC and master slice. The two dataset with 1000 samples are separately generated (SSRR one dataset with three class and MSRR with four class). Based on the parameters provided in the reference study[5], we consider the range for attributes in the dataset. Training and testing are conducted using deep neural network with ration of 90:10, 80:20, 70:30. The graphs for the PI and VSR are made using the scale-free network model suggested in[48]. The proposed work makes use of Python to construct the simulation platform.

D-SIMS through SSRR

According to this strategy, it is assumed that the provided physical infrastructure is virtualized into three slices. In accordance with the characteristics of the use case, the SC-DNN is used to generate three slices for both the PI and the VSR. To account for the dissimilar nature of VSR and PI, we have constructed these performance datasets independently. The CPU capacity, link capacity, bandwidth, jitter, and delay rate are the key parameters identified for use in building the performance dataset[26,49]. The datasets are categorized as eMBB, mMTC, and uRLLC depending on the specific fields they represent. An effort is underway to compile a dataset containing information on 1000 population. DNNs are trained and evaluated using a variety of weight optimization techniques, including ADAM, SGD, and LBFGS. The training-testing ratio is 9:1. DNN use the hyper parameter values for our model such as learning rate 0.001, Batch size 32, Number of layers 4, Activation function ReLU. To evaluate the relative effectiveness of various optimization techniques, we use metrics like precision, recall, and F1 score. DNN methods are implemented using the scikit-learn python library. The precision, recall, and F1 score outcomes for PI using DNN with various weight optimizations are displayed in Fig. 4a–c. As shown in the figure, there are three major categories of approaches being evaluated, where each achieves performance levels of almost 90% in terms of precision, recall, and F1-score. As measured by the overall percentage of correct predictions in the performance dataset, ADAM is superior to competing methods. Figure 5 displays the results of these three supervised DNN methods at different training-to-testing ratios, providing yet another illustration of their usefulness. Overall, the figure demonstrates that the ADAM method achieves a higher rate of accuracy than competing methods.

In the SI-FP phase, hybrid fuzzy-PROMETHEE is used to map VSR nodes in PI and their corresponding links. At this stage, as shown in Fig. 6, priority is given to allocating resources for isolating slices. This scheme is evaluated by assuming that PI is equipped with 100, 200, and 300 nodes, with request nodes ranging from 100 to 800. The values for the network features are drawn at random from the range shown in Table 3. Each link in the graph has a fixed length of 1 unit so that performance can be more accurately measured. Node-generated values from the preceding phase are made available as slices in PI and VSR for use in resource allocation. The SI-FP system ensures that the needed VSRs have access to the necessary nodes and links by assigning the resources of PI slices to VSR slices in an optimal manner. If the SI-FP scheme fails to allocate the necessary resources for VSR, the SM-4R phase begins to support the resource allocation process. SM-4R is in charge of four tasks that assess

Parameters	Descriptions	Values
Physical Infrastructure Network		
N_P	Total count of PI nodes	100,200,300
C_P	CPU capacity of a PI node in unit	U[20, 50]
LC_P	Link capacity of a PI node in unit	U[20, 50]
SL_P	Security level of a PI node in real number	(0–1)
ine BW_P	Bandwidth of each PI link in unit	U[20, 50]
DR_P	Delay for each PI node in range	(0–1)
JT_P	Jitter for each PI node in range	(0–1)
Virtual service request		
N_V	The distribution of nodes for each VSR	20
T_V	Total count of VSRs arrived in the time frame	U[5, 40]
ine C_V	CPU requirement for each VSR node in unit	U[5, 25]
LC_V	Link capacity requirement for each VSR node in unit	U[5, 25]
SR_V	Security level of a VSR node in real number	(0–0.5)
BW_V	Bandwidth requirement of each VSR link in unit	U[5, 25]
LT_V	Time duration of each VSR node in unit	U[10, 35]

Table 3. Test case parameters.

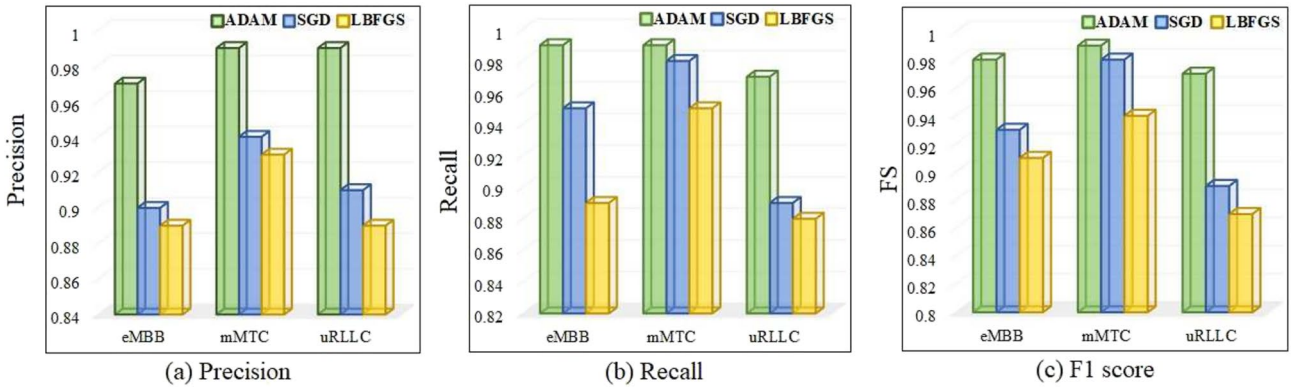

Figure 4. Performance measures for classification in SSRR.

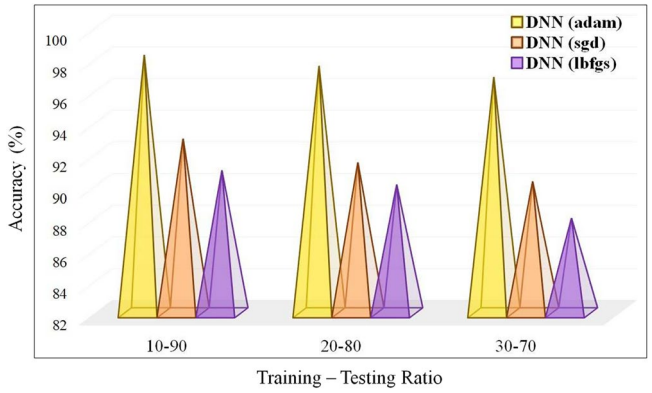

Figure 5. SSRR classification accuracy.

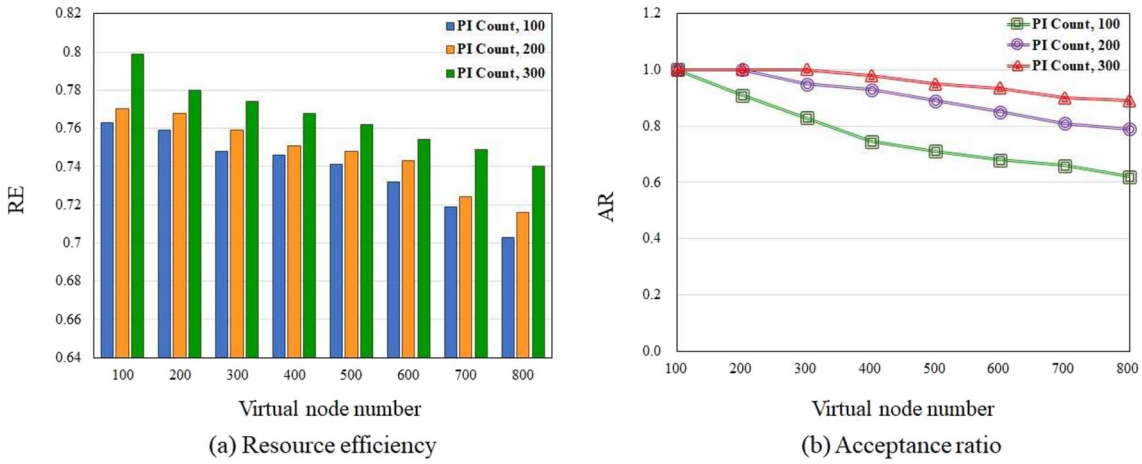

Figure 6. Resource allocation measures of SSRR.

the viability of VSR resource allocation and, ultimately, the VSR's success or failure. Both resource efficiency and acceptance ratio are enhanced by the synergistic use of SI-FP and SM-4R.

In Fig. 6a, we can see the aggregated findings concerning VSRs and resource efficiency, which show that resource efficiency decreases with an increase in the number of nodes by the VSR, while efficiency rises as more nodes are added to the physical infrastructure. Shorter paths between nodes in a larger network means more efficient use of resources, as predicted by the resource efficiency equation. When a 100-node PI is used to allocate resources to 800 virtual request nodes, resource efficiency is recorded as 0.7. Resource efficiency improves as

well with increasing in node counts, reaching an optimal value of 0.8 when 100 virtual nodes are requested at the PI equipped with 300 nodes.

In terms of the acceptance ratio, it can be seen in Fig. 6b that it rises whenever the number of nodes in the PI network expands. This is the case even though more nodes are present. There are more nodes now, therefore a larger percentage of the population is on get on with it. With a shorter use case's lifespan comes a higher acceptance rate. When servicing 800 use cases in PI with 300 available nodes, the proposed technique obtains an acceptance ratio of 0.9. In addition, as can be seen in Fig. 7a, it may accommodate ranging from a minimum of 1494 CPUs up to a maximum of 12,540 CPUs within an infrastructure of 100 to 300 nodes. As can be seen in Fig. 7b, the bandwidth requirements for the same operational state range from a high of 37,158 to a low of 4250.

D-SIMS through MSRR

This strategy makes the assumption that the physical infrastructure is virtualized into four slices, including the master slice, eMBB, mMTC, and uRLLC. In contrast, the VSR is divided into three slices based on the traits of use cases. Slices are produced for both the PI and the VSR using SC-DNN. Similar to the previous approach, we have independently built these performance datasets using key parameters due to the differences between VSR and physical infrastructure. The nodes with the highest KPI values are kept inside the master slice for PI because they must act as an alien slice for all three types of use cases. eMBB, mMTC, and uRLLC are the three remaining PI nodes, which are divided among them based on the particular fields they stand for. Similar to the previous approach, a work is being done to create a dataset with data on 1000 population. The three DNN techniques are used to measure the performance of slice classification. The results of slice creation measures such as precision, recall, and F1 score for PI using various DNN techniques with a 9:1 training to testing ratio are shown in Fig. 8a–c.

The accuracy measure for various training and testing ratios is shown in Fig. 9. It is clear from both Figs. 8 and 9 that the ADAM method outperforms the other two methods in terms of accuracy, precision, recall, and F1 score. After the slices in PI and VSR have been created, the SI-FP phase can then continue. The creation of the PI slices takes place at the very beginning, before the first VSR is received, and these slices do not undergo any changes until the end of the total time frame.

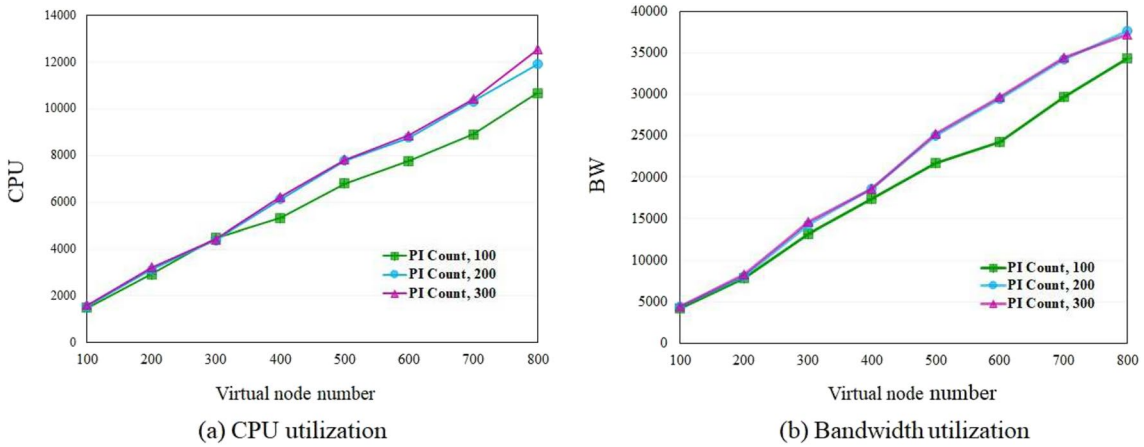

Figure 7. Resource utilization of SSRR.

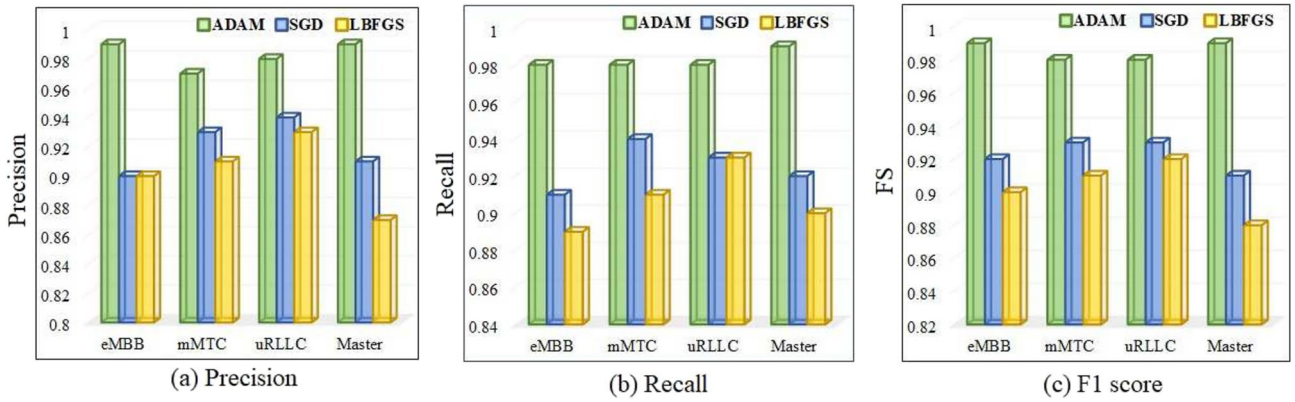

Figure 8. Performance measures for classification in MSRR.

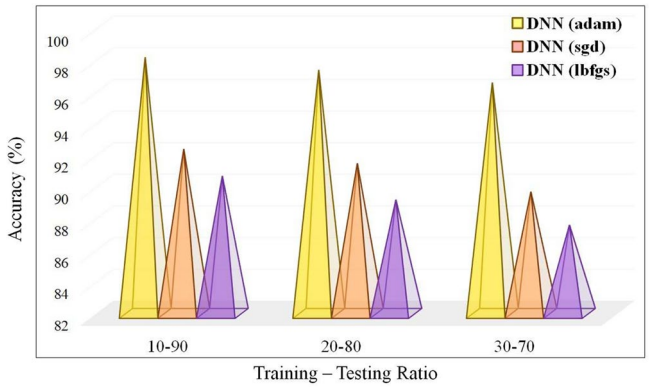

Figure 9. MSRR classification accuracy.

The slice for the VSR is generated as soon as the new VSR arrives and submits a request to the PI for resource allocation. The effectiveness of MSRR is evaluated using PI equipped with 100, 200, and 300 nodes, much like the prior strategy. The values for the network features are chosen at random from the range shown in Table 3, and the VSRs are received with a random number of node requests. The SM-4R phase works with the SI-FP to find the best way to turn the unsuccessful VSR into a successful one in order to increase resource efficiency and acceptance rate. When determining the best way to allocate resources, all three performance criteria are taken into account. The results are displayed in Figs. 10 and 11, and they are broken down into categories such as resource efficiency, acceptance ratio, and CPU and BW consumption, respectively. Resource efficiency of 0.8, 0.7, and 0.6 under the PI equipped with 100, 200, and 300 nodes for the total request of 800 nodes are clearly seen in Fig. 10a. When the PI is equipped with 100 and 300 nodes and the total number of requested nodes is 100 and 800, respectively, the acceptance ratio is recorded as having a minimum of 0.7 and a maximum of 0.9, as shown in Fig. 10b. Furthermore, as shown in Fig. 11a, it may accommodate between 100 and 300 nodes, each with a minimum of 1481 CPUs and a maximum of 12112 CPUs. Bandwidth utilization varies between a maximum of 29,662 and a minimum of 4006 in Fig. 11b for the same operational state.

Discussions

Through a series of comparisons with the aforementioned literature, this section demonstrates the efficacy of the MSRR and SSRR of D-SIM strategies, in particular with respect to VNE-NTANRC[29], SVM-VNE[35] and SCE-DIVS[38]. We assumed that all of the proposed algorithms would have access to the dynamic provisioning service for network requests. Evaluating how well and widely those requests are accepted when the PI has 300 nodes and is responsible for managing a wide variety of network requests is important. Figure 12 shows the algorithms' relative performance in a number of different settings. It is clear from Fig. 12a that when all other methods are combined, the VNE-NTANRC approach has the lowest throughput. The SVM-VNE and SCE-DIVS algorithms outperformed the VNE-NTANRC algorithm. In addition, both D-SIM techniques exhibit superior performance in comparison to other evaluated algorithms. In terms of optimal resource allocation, the SSRR method is marginally superior to the MSRR technique. The least resource efficiency obtained by the two methods

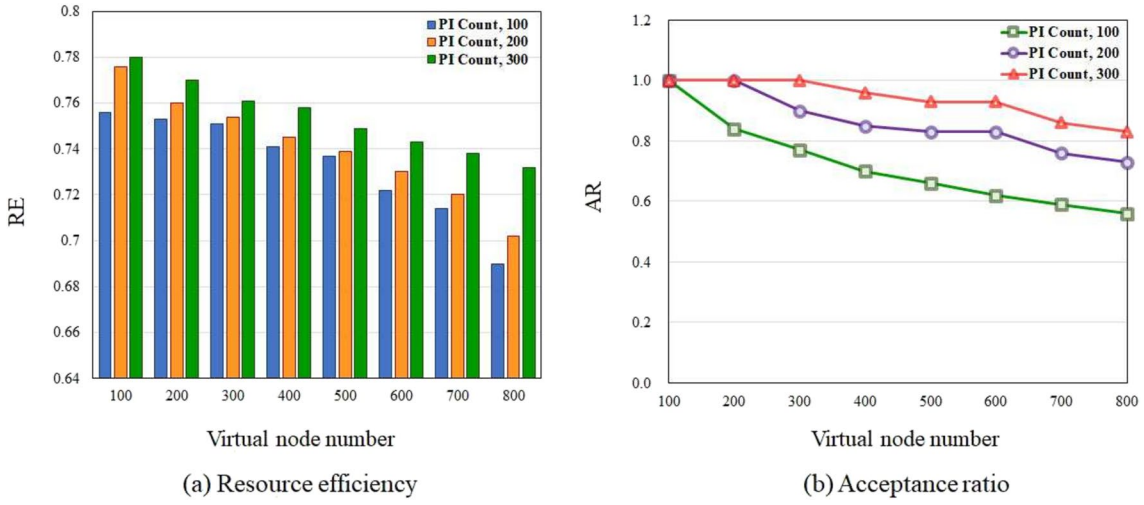

Figure 10. Resource allocation performance measures of MSRR.

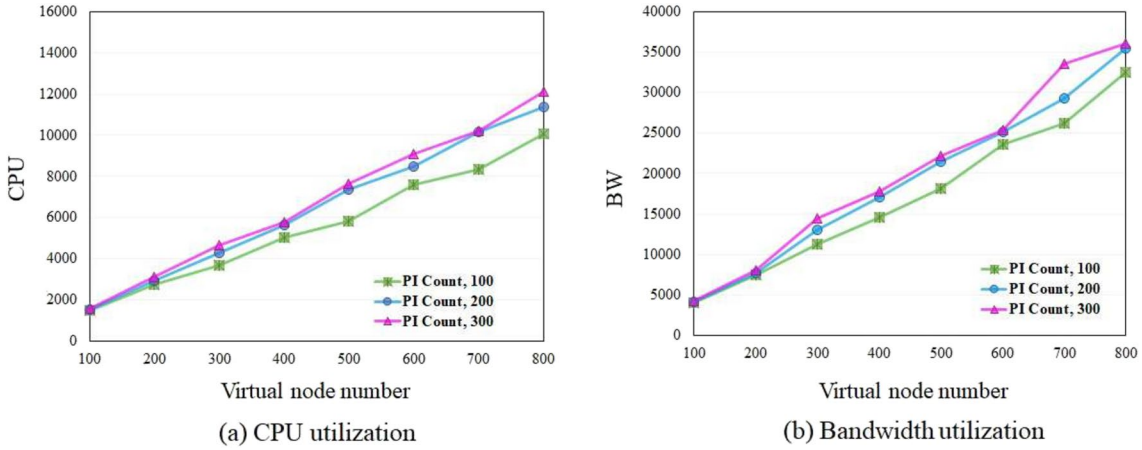

Figure 11. Resource utilization of MSRR.

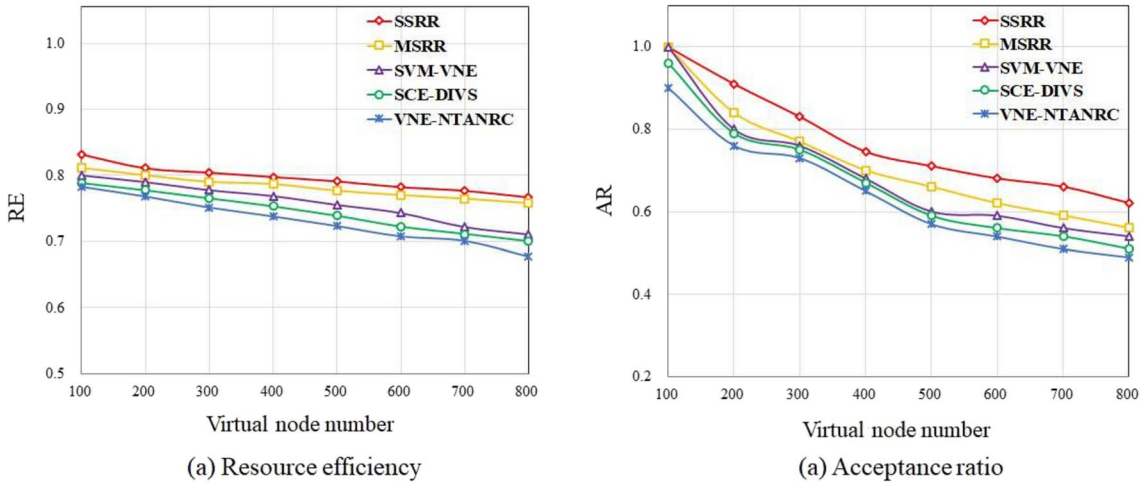

Figure 12. Comparison results of SSRR and MSRR with existing methods.

in the PI outfitted with 300 nodes and 800 VSR is 0.73 for MSRR and 0.74 for SSRR, with SSRR producing a 10% increase in resource efficiency over VNE-NTANRC. The adoption rate of the offered strategies is compared in similar conditions. Figure 12b illustrates the outcomes of the algorithms' successful execution in the context of a 300-node physical infrastructure. The acceptance rate decreases as the total number of node requests rises.

Conversely, when VSR life-spans are lowered, acceptance ratios tend to increase as well. When adopting the SSRR technique, it is possible to achieve a maximum acceptance ratio of 1 and a minimum acceptance ratio of 0.62 when serving a total of 100 and 800 requested nodes. Other algorithms, such as MSRR, generate a lower acceptance rate than SSRR. The VNE-NTANRC, which serves 800 request nodes, has the lowest overall acceptance rate of 0.5. The effectiveness of the provided methods is validated by contrasting the time required to perform the task with other metrics, such as resource efficiency and acceptance rate.

Execution duration's for provisioning requests of different node counts are compared in Fig. 13. In general, the time needed to execute the algorithmic procedures scales linearly with the sum of the request nodes and the number of accessible nodes in the physical infrastructure. The SSRR algorithm for allocating resources takes significantly longer than the MSRR technique to finish in all operational situations. The MSRR takes less than ten milliseconds to service nodes with values between 100 and 300 when the underlying physical infrastructure has 100 nodes. In addition, the PI with 300 nodes needs a response time somewhat longer than 10 ms, despite servicing the same number of customers. Nonetheless, under PI with 300 nodes, the allocation procedure can be finished by either strategy in less than 100 ms.

Conclusion

The proposed work provides D-SIMS called SSRR and MSRR for effective network slicing. The way the physical infrastructure is sliced has been modeled as influencing the choice strategies. In addition, the orchestration frameworks of both techniques are ready to solve the three network slicing components, namely slice creation, isolation, and management. With the aid of a performance dataset that included the crucial KPIs of the nodes, deep neural networks were used to create slices for PI and VSRs. Isolating slices has been accomplished using

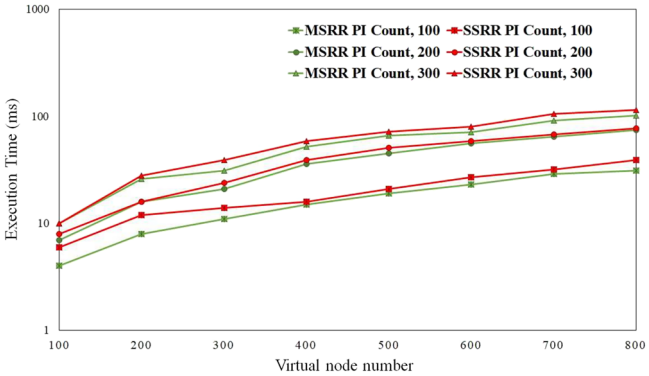

Figure 13. Execution time of MSRR and SSRR.

both the Hybrid Fuzzy PROMETHEE for node mapping and Dijkstra's approach for link mapping. In order to improve resource usage, slice management was combined with the slice isolation phase. Four VNF functions were constructed to handle the VNF manager's four responsibilities. The performance of the suggested techniques has been investigated under various PI and VSR conditions. The findings obtained were compared to those found in the previous research to ascertain the efficacy of the proposed method. It has been determined that both the SSRR and MSRR strategies of D-SIM outperformed previous work. More specifically, the framework based on the SSRR method provides better performance than MSRR. In the worst-case scenario, when provisioning 800 request nodes in a PI of 100 nodes, SSRR achieved a maximum resource efficiency of 0.7 and an acceptance ratio of 0.6. As a result, it is therefore suggested that under the three slice virtualization of PI, proper management strategies will lead to increased performance. The current work can be expanded in the future to cover the mobility and energy management of several use cases.

Data availability

The datsets used and analysed during the current study available from the corresponding author on reasonable request.

Received: 23 March 2024; Accepted: 29 July 2024
Published online: 11 August 2024

References

1. Henry, S., Alsohaily, A. & Sousa, E. S. 5G is real: Evaluating the compliance of the 3G pp 5G new radio system with the ITU IMT-2020 requirements. *IEEE Access* **8**, 42828–42840 (2020).
2. Foukas, X., Patounas, G., Elmokashfi, A. & Marina, M. K. Network slicing in 5g: Survey and challenges. *IEEE Commun. Mag.* **55**, 94–100 (2017).
3. Afolabi, I., Taleb, T., Samdanis, K., Ksentini, A. & Flinck, H. Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. *IEEE Commun. Surv. Tutor.* **20**, 2429–2453 (2018).
4. Chang, C.-Y. & Nikaein, N. Ran runtime slicing system for flexible and dynamic service execution environment. *IEEE Access* **6**, 34018–34042 (2018).
5. Guan, W., Wen, X., Wang, L., Lu, Z. & Shen, Y. A service-oriented deployment policy of end-to-end network slicing based on complex network theory. *IEEE Access* **6**, 19691–19701 (2018).
6. Min, Z. *et al.* A novel 5g digital twin approach for traffic prediction and elastic network slice management. In *2024 16th International Conference on Communication Systems & NETworkS (COMSNETS)* 497–505 (IEEE, 2024).
7. Ferrus, R., Sallent, O., Pérez-Romero, J. & Agusti, R. On 5g radio access network slicing: Radio interface protocol features and configuration. *IEEE Commun. Mag.* **56**, 184–192 (2018).
8. Kotulski, Z. *et al.* Towards constructive approach to end-to-end slice isolation in 5g networks. *EURASIP J. Inf. Secur.* **2018**, 1–23 (2018).
9. Li, X., Guo, C., Gupta, L. & Jain, R. Efficient and secure 5g core network slice provisioning based on Vikor approach. *IEEE Access* **7**, 150517–150529. https://doi.org/10.1109/ACCESS.2019.2947454 (2019).
10. Ogino, N., Kitahara, T., Arakawa, S. & Murata, M. Virtual network embedding with multiple priority classes sharing substrate resources. *Comput. Netw.* **112**, 52–66 (2017).
11. Liao, J., Feng, M., Qing, S., Li, T. & Wang, J. Live: Learning and inference for virtual network embedding. *J. Netw. Syst. Manag.* **24**, 227–256 (2016).
12. Zhang, Z. *et al.* A unified enhanced particle swarm optimization-based virtual network embedding algorithm. *Int. J. Commun. Syst.* **26**, 1054–1073 (2013).
13. Shahin, A. A. Memetic multi-objective particle swarm optimization-based energy-aware virtual network embedding (2015). arXiv preprint arXiv:1504.06855
14. Melo, M., Sargento, S., Killat, U., Timm-Giel, A. & Carapinha, J. Optimal virtual network embedding: Node-link formulation. *IEEE Trans. Netw. Serv. Manag.* **10**, 356–368 (2013).
15. Pathak, I. & Vidyarthi, D. P. An optimal virtual network mapping model based on dynamic threshold. *Wirel. Pers. Commun.* **83**, 2381–2401 (2015).
16. Xu, Q., Wang, J. & Wu, K. Learning-based dynamic resource provisioning for network slicing with ensured end-to-end performance bound. *IEEE Trans. Network Sci. Eng.* **7**, 28–41. https://doi.org/10.1109/TNSE.2018.2876918 (2020).
17. Butt, M. M., Pantelidou, A. & Kovács, I. Z. Ml-assisted UE positioning: Performance analysis and 5g architecture enhancements. *CoRR* (2021). arXiv:2108.11365
18. Alazab, M. *et al.* Deep learning for cyber security applications: A comprehensive survey (2021).

19. Wu, Z.-X., You, Y.-Z., Liu, C.-C. & Chou, L.-D. Machine learning based 5g network slicing management and classification. In *2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)* 371–375 (IEEE, 2024).
20. Bega, D., Gramaglia, M., Banchs, A., Sciancalepore, V. & Costa-Pérez, X. A machine learning approach to 5g infrastructure market optimization. *IEEE Trans. Mob. Comput.* **19**, 498–512 (2019).
21. Archanaa, R., Athulya, V., Rajasundari, T. & Kiran, M. V. K. A comparative performance analysis on network traffic classification using supervised learning algorithms. In *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)* 1–5 (2017). https://doi.org/10.1109/ICACCS.2017.8014634
22. Patro, S., Rath, H. K. & Panigrahi, B. Dynamic KPI-aware network slicing for 5g+ networks. In *2024 IEEE 13th International Conference on Communication Systems and Network Technologies (CSNT)* 94–99 (IEEE, 2024).
23. Gupta, R. K. & Misra, R. Machine learning-based slice allocation algorithms in 5g networks. In *2019 International Conference on Advances in Computing, Communication and Control (ICAC3)* 1–4 (IEEE, 2019).
24. Immadisetti, M. K. N., Murukessan, A. & Srinivas, M. Automate allocation of secure slice in future mobile networks using machine learning. In *2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)* 1–7 (IEEE, 2021).
25. Thantharate, A., Paropkari, R., Walunj, V., Beard, C. & Kankariya, P. Secure5g: A deep learning framework towards a secure network slicing in 5g and beyond. In *2020 10th Annual Computing and Communication Workshop and Conference (CCWC)* 0852–0857 (IEEE, 2020).
26. Abidi, M. H. *et al.* Optimal 5g network slicing using machine learning and deep learning concepts. *Comput. Stand. Interfaces* **76**, 103518 (2021).
27. Jiang, H., Wang, Y., Gong, L. & Zhu, Z. Availability-aware survivable virtual network embedding in optical datacenter networks. *J. Opt. Commun. Netw.* **7**, 1160–1171 (2015).
28. Javadpour, A., Ja'fari, F., Taleb, T. & Benzaïd, C. Enhancing 5g network slicing: Slice isolation via actor-critic reinforcement learning with optimal graph features. In *GLOBECOM 2023—2023 IEEE Global Communications Conference* 31–37 (IEEE, 2023).
29. Cao, H., Yang, L. & Zhu, H. Novel node-ranking approach and multiple topology attributes-based embedding algorithm for single-domain virtual network embedding. *IEEE Internet Things J.* **5**, 108–120. https://doi.org/10.1109/JIOT.2017.2773489 (2018).
30. Zhang, P., Yao, H. & Liu, Y. Virtual network embedding based on computing, network, and storage resource constraints. *IEEE Internet Things J.* **5**, 3298–3304. https://doi.org/10.1109/JIOT.2017.2726120 (2018).
31. Wang, Y. & Ye, C. Individualized resource allocation for 5g network slicing based on knapsack stragegy. In *2023 International Conference on Computer Science and Automation Technology (CSAT)* 383–387 (IEEE, 2023).
32. Liu, J., Zhao, B., Shao, M., Yang, Q. & Simon, G. Provisioning optimization for determining and embedding 5g end-to-end information centric network slice. *IEEE Trans. Netw. Serv. Manag.* **18**, 273–285. https://doi.org/10.1109/TNSM.2020.3045051 (2021).
33. Tariq, M. A. *et al.* Network slice traffic demand prediction for slice mobility management. In *2024 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)* 281–285 (IEEE, 2024).
34. Venkatapathy, S., Srinivasan, T., Jo, H.-G. & Ra, I.-H. Optimal resource allocation for 5g network slice requests based on combined promethee-II and SLE strategy. *Sensors* **23**, 1556 (2023).
35. Mei, C., Liu, J., Li, J., Zhang, L. & Shao, M. 5g network slices embedding with sharable virtual network functions. *J. Commun. Netw.* **22**, 415–427. https://doi.org/10.1109/JCN.2020.000026 (2020).
36. Pentelas, A., Papathanail, G., Fotoglou, I. & Papadimitriou, P. Network service embedding across multiple resource dimensions. *IEEE Trans. Netw. Serv. Manag.* **18**, 209–223. https://doi.org/10.1109/TNSM.2020.3044614 (2021).
37. Wang, Y. & Hu, Q. A path growing approach to optical virtual network embedding in slice networks. *J. Lightw. Technol.* **39**, 2253–2262. https://doi.org/10.1109/JLT.2020.3047713 (2021).
38. Thanh, N. H. *et al.* Energy-aware service function chain embedding in edge–cloud environments for IOT applications. *IEEE Internet Things J.* **8**, 13465–13486. https://doi.org/10.1109/JIOT.2021.3064986 (2021).
39. Fan, W., Xiao, F., Chen, X., Cui, L. & Yu, S. Efficient virtual network embedding of cloud-based data center networks into optical networks. *IEEE Trans. Parallel Distrib. Syst.* **32**, 2793–2808. https://doi.org/10.1109/TPDS.2021.3075296 (2021).
40. Zhang, Z., Cao, H., Su, S. & Li, W. Energy aware virtual network migration. *IEEE Trans. Cloud Comput.* **10**, 1173–1189. https://doi.org/10.1109/TCC.2020.2976966 (2022).
41. Luu, Q.-T., Kerboeuf, S. & Kieffer, M. Admission control and resource reservation for prioritized slice requests with guaranteed SLA under uncertainties. *IEEE Trans. Netw. Serv. Manag.*https://doi.org/10.1109/TNSM.2022.3160352 *(2022).*
42. Gao, L., Li, P., Pan, Z., Liu, N. & You, X. Virtualization framework and VCG based resource block allocation scheme for LTE virtualization. In *2016 IEEE 83rd Vehicular Technology Conference (VTC Spring)* 1–6 (IEEE, 2016).
43. Balachandran, A. & Amritha, P. P. VPN network traffic classification using entropy estimation and time-related features. In *IOT with Smart Systems* (eds Senjyu, T. *et al.*) 509–520 (Springer, 2022).
44. Prasad, J., Senthil, M., Yadav, A., Gupta, P. & K S. A. *A Comparative Study of Machine Learning Algorithms for Gas Leak Detection* 81–90 (2020).
45. Kylili, A., Christoforou, E., Fokaides, P. A. & Polycarpou, P. Multicriteria analysis for the selection of the most appropriate energy crops: The case of cyprus. *Int. J. Sustain. Energy* **35**, 47–58 (2016).
46. Gul, M., Celik, E., Gumus, A. T. & Guneri, A. F. A fuzzy logic based Promethee method for material selection problems. *Beni-Suef Univ. J. Basic Appl. Sci.* **7**, 68–79 (2018).
47. Makariye, N. Towards shortest path computation using Dijkstra algorithm. In *2017 International Conference on IoT and Application (ICIOT)* 1–3 (IEEE, 2017).
48. Barabási, A.-L. & Albert, R. Emergence of scaling in random networks. *Science* **286**, 509–512 (1999).
49. Pakzad, F., Portmann, M. & Hayward, J. Link capacity estimation in wireless software defined networks. In *2015 International Telecommunication Networks and Applications Conference (ITNAC)* 208–213 (IEEE, 2015).

Acknowledgements

This research was supported by "Regional Innovation Strategy (RIS)" through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (MOE) (2023RIS-008).

Author contributions

Writing (original draft) and methodology: S.V. and T.S.; writing (review and editing): H.-G.J. and I.-H.R. All authors have read and agreed to the published version of this manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to H.-G.J. or I.-H.R.