Journal of
Sensor and Actuator Networks

Article

VNF-Enabled 5G Network Orchestration Framework for Slice Creation, Isolation and Management

Thiruvenkadam Srinivasan [1], Sujitha Venkatapathy [2], Han-Gue Jo [3,*] and In-Ho Ra [3,*]

[1] School of Electrical Engineering, Vellore Institute of Technology, Vellore 632014, Tamil Nadu, India; thiruvenkadam.s@vit.ac.in
[2] TIFAC-CORE in Cyber Security, Amrita School of Engineering, Amrita Vishwa Vidyapeetham, Coimbatore 641112, Tamil Nadu, India; v_sujitha@cb.amrita.edu
[3] School of Software, Kunsan National University, Gunsan 54150, Republic of Korea
* Correspondence: hgjo@kunsan.ac.kr (H.-G.J.); ihra@kunsan.ac.kr (I.-H.R.)

Abstract: Network slicing is widely regarded as the most critical technique for allocating network resources to varied user needs in 5G networks. A Software Defined Networking (SDN) and Network Function Virtualization (NFV) are two extensively used strategies for slicing the physical infrastructure according to use cases. The most efficient use of virtual networks is realized by the application of optimal resource allocation algorithms. Numerous research papers on 5G network resource allocation focus on network slicing or on the best resource allocation for the sliced network. This study uses network slicing and optimal resource allocation to achieve performance optimization using requirement-based network slicing. The proposed approach includes three phases: (1) Slice Creation by Machine Learning methods (SCML), (2) Slice Isolation through Resource Allocation (SIRA) of requests via a multi-criteria decision-making approach, and (3) Slice Management through Resource Transfer (SMART). We receive a set of Network Service Requests (NSRs) from users. After receiving the NSRs, the SCML is used to form slices, and SIRA and SMART are used to allocate resources to these slices. Accurately measuring the acceptance ratio and resource efficiency helps to enhance overall performance. The simulation results show that the SMART scheme can dynamically change the resource allocation according to the test conditions. For a range of network situations and Network Service Requests (NSRs), the performance benefit is studied. The findings of the simulation are compared to those of the literature in order to illustrate the usefulness of the proposed work.

Keywords: 5G network; network slicing; resource allocation; machine learning

check for updates

Citation: Srinivasan, T.; Venkatapathy, S.; Jo, H.-G.; Ra, I.-H. VNF-Enabled 5G Network Orchestration Framework for Slice Creation, Isolation and Management. *J. Sens. Actuator Netw.* **2023**, *12*, 65. https://doi.org/10.3390/jsan12050065

Academic Editor: Lei Shu

Received: 24 July 2023
Revised: 6 September 2023
Accepted: 7 September 2023
Published: 13 September 2023

1. Introduction

Mobile traffic on private and commercial 5G networks has increased dramatically in recent years as the number of people using portable devices has grown. A wide range of businesses and industries, including transport, healthcare, and the energy supply chain, expect to gain from 5G networks in a few years. A highly complicated 5G architecture is required to provide high-quality, always-reliable service to a large variety of User Equipments (UEs) [1]. The Next Generation Mobile Networks (NGMN) Alliance has come up with a concept called network slicing to make it easier for 5G networks to allocate resources to user devices with varying performance requirements [2]. In order to partition the physical infrastructure and assure Quality of Service (QoS) for distinct sets of requirements, we use virtual resources, a logical topology, traffic control, node and link provisioning requirements, and security evaluation parameters [3]. The 3rd-Generation Partnership Project (3GPP) created and authorized the 5G system design that allows for network slicing in the initial iteration of 5G normative standards [4]. Radio Access Networks (RANs), Transport Networks (TNs), and Core Networks (CNs) are the three main components of the physical architecture that make up each logical network slice in the scheme [5]. The VNF orchestration issue is described as an Integer Linear Program in [6], and a heuristic strategy is offered that takes into account the costs of deployment, energy, traffic, and propagation delay. VNF insertion in large-scale systems has been addressed by the development of WiNE (Wireless Network Embedding) [7]. In order to solve the VNF placement problem, WiNE considers three distinct scenarios: linear requests, branching requests, and requests with loops. In [8], we find an in-depth analysis of SDN and VNF network slicing, including their foundational concepts, designs, and problems. Providing network slices is difficult for many reasons, such as maximising resource efficiency, maximising acceptance ratio, maximising data confidentiality, maximising latency, and supporting a wide diversity of user needs [9].

Network slicing, as is well known, entails three key responsibilities: slice creation, isolation, and management. Many of the network slicing orchestration frameworks concentrate on one of the single duties of network slicing. It is recognized that there is still a requirement to build the orchestration framework to provide each service for all three network slicing duties. Furthermore, when a single framework provides services in all three elements of slicing, VNF managers have a competitive advantage in offering higher-quality services to their clients. By consideration of the problems previously described in the existing research, this investigation presents an orchestration architecture that takes care of all three network slicing responsibilities. The main objectives and findings of our study are as follows:

- To model a network orchestration framework that combines network slice generation, isolation, and management to provide efficient and simple network slicing.
- To identify the best machine learning approach for slice creation where the performance dataset is prepared based on the realistic characteristics of the use cases.
- To achieve the best possible distribution of physical infrastructure (PI) resources among Network Service Requests (NSRs), it is necessary to develop a mechanism for isolating network slices. The combination of Multi Criteria Decision Making (MCDM) and Shortest Path Algorithms are utilised for the implementation.
- By the development of the inner slicing idea, which makes use of the resource transfer approach for resource allocation, PI slice performance can be improved with the use of a prepared slice management strategy.

The rest of this work is structured as follows: Section 2 presents a literature review; Section 3 summarizes the proposed system model and mathematical background; Section 4 describes the proposed 5G mobile network orchestration framework; Section 5 discusses dynamic slicing of 5G mobile network with case study; Section 6 concludes the work with proposal for future extension.

2. Related Literature

Numerous research has been conducted recently on the issue of network slicing in 5G mobile networks. Due to the fact that this proposed work combines machine learning and optimal resource allocation algorithms in an orchestration framework for dynamic network slicing, this section reviews pertinent research on both approaches.

2.1. Machine Learning Techniques for Network Slice Creation

An in-depth examination of all machine learning technologies is presented in [10–12], along with a list of prospective domains where machine learning techniques may be advantageous. A deep learning strategy is used to classify modulation, according to the authors in [13], which provides a more accurate solution than other conventional methods of classification. Using a Neural Network technique, the previously mentioned problem was subsequently addressed in [14], and the findings demonstrate that as the network size grows, the approach quickly converges to its optimal performance. A hybrid learning algorithm for network slicing was proposed [15], which combines approaches from Machine Learning and Deep Learning to achieve optimal results. In particular, this method optimised the weight function by using Glowworm Swarm Optimisation and the Deer Hunting Optimisation algorithm together. A multi-slice Deep Neural Network (DNN) for dynamic end-to-end slicing was proposed in [16], which incorporates two service level agreements (SLAs), namely, violation rate-based SLA and resource bounds-based SLA, in addition to other factors. To prepare the performance dataset for training, the method made use of a live cellular network with traffic analysis on the gathered data. It was demonstrated in [17] that two reinforcement learning-based algorithms could be used to optimise mode selection in fog-radio access networks (F-RANs). The mode selection is determined by the method, which is based on the immediate reward provided by the surroundings. In the paper [18], the authors discuss the deep learning-based work that has been proposed to improve the 5G RAN portion of the network, as well as how these works can be integrated with the AI-enabled Open-RAN architecture.

2.2. Techniques for Network Slice Isolation and Management

The subject of resource allocation for RAN slicing is the focus of the research in [19–21]. Their research looked into how the heterogeneous cloud radio access network (H-CRAN) architecture allocated both baseband and radio resources, according to the authors of the paper. The issue is set up using the Mixed Integer Linear Programming (MILP) method, and it is solved using the Lagrangian dual strategy, which is not convex. The problem of allocating the resources of sensor network slices in the context of sensor network slices were discussed in [22]. Their solution was to develop a standardized architecture for controlling application acceptance and allocating slice resources. A heuristic technique is used to solve the problem of joint optimization in this case. Radio Access Network (RAN) and core network resource allocation are studied in [23], which present an end-to-end quality of service architecture for 5G networks. Using complex network theory, the authors of [24] propose a method for optimizing resource utilization while simultaneously reducing expenditures on research and development. Two stages are recommended in the technique that has been proposed: (i) creating virtual network functions, and (ii) choosing link paths for the virtual network functions. In addition, the viability of the proposed method is demonstrated by contrasting the simulation outcomes attained through the application of the Simulated Annealing (SA), the LAVA, and the GLL-based VNF placement techniques.

In reference [25], it was proposed that a Network Topology Attribute and Network Resource-Considered method (VNE-NTANRC) may be utilised in combination with a Service Provider (SP) approach to mapping the connections between nodes. This algorithm would consider five different network topology attributes and global network resources. A study published in [26] investigated the topic of radio resource allocation in Fog Radio Access Network (F-RAN) slicing. A Stackelberg game between the global radio resource manager and the local radio resource manager is depicted as a "stacking" Stackelberg game in the context of an organizational hierarchy. Among the several multi-criteria decision-making techniques, the VIKOR method was utilised to rank the nodes in [27]. The process of establishing a virtual network node is identical to that of a physical or logical network node. Floyd's method of link provisioning involves assigning the shortest possible path to the slice request nodes. Using Stochastic Network Calculus (SNC), the delay bound in 5G mobile networks for a specified traffic distribution and resource allocation was determined [28].

Upper-tier First with Latency-bound Overprovisioning Prevention (UFLOP) is a method created to optimise capacity and traffic sharing in two-tier 5G slicing networks while meeting renters' latency constraints [29]. Incoming traffic is analysed in accordance with the services offered to tenants and the assets needed to supply resources, and any necessary adjustments are then suggested by the method. [30] describes a technique for service provisioning in RAN slicing that guarantees not only that QoS criteria, but also those for bandwidth and compute capacity, are met. This technique is being utilised to decrease the bandwidth required to provide services. It was suggested by [31] to solve the issues of scalability, monitoring, and aggregation of slice-level measured data that arise while tracking the efficiency of individual parts of a network.

2.3. Motivation

Table 1 summarizes the literature on network slicing components, such as slice creation, isolation, and management, and includes all related works. In addition, to aid comprehension, the table highlights the methodology, performance metrics, and work parameters used. The vast majority of papers use the same parameters and metrics to evaluate performance. It has been found that the literature differs depending on the methodology that was used as well as the network slicing components that were discussed. The majority of the works in the table provide solutions for slice isolation; however, a few of the works addressed slice creation on its own, and a few of the papers addressed slice management in addition to slice isolation. As a result, we determined that introducing an orchestration framework that addresses all three components of network slicing would be significant. In the proposed framework, individual components of network slicing are addressed using the most appropriate techniques. For this study, we made use of a machine learning technique called Random Forest to carry out the slice creation. After slice creation, we employed a PROMETHEE-II-based resource allocation scheme to provide better resource allocation to the requested service. Then, we use resource management to make sure that resources are used effectively, and also, we give priority to uRLLC services. The proposed framework's performance is evaluated based on the component it addresses. Precision, F1 Score, and recall are used to measure performance for slice creation via machine learning, whereas resource efficiency and acceptance rate are used to measure performance for slice isolation and management.

Table 1. The literature Survey.

Ref. No.	Methodology Adopted	Components	Performance Metrics	Considered Attributes
[14]	Reinforcement Learning	Slice creation	Learning Time and Revenue	Number of Resources in base station, base station capacity, Bandwidth, distribution
[15]	Machine Learning and Deep Learning	Slice creation	Accuracy	Device type, packet loss rate, speed, duration, bandwidth, jitter, delay rate
[16]	Deep Learning	Slice creation and isolation	Resource Efficiency and Violation Rate	Traffic, PRB usage and CPU load
[17]	Reinforcement Learning	Slice isolation	Power consumption	Number of sensors, traffic and monitoring devices
[32]	Deep Reinforcement Learning	Slice isolation	CPU and bandwidth utilization	CPU capacity, bandwidth, path length and number of requests
[19]	Dynamic Programming	Slice isolation	Bandwidth and fairness	Data rate, transmission power, link capacity and bandwidth
[22]	Dynamic Greedy Technique	Slice isolation	Residual energy consumption	Transmission power, distance, path loss and bandwidth
[24]	Complex Network Theory	Slice isolation	Resource Efficiency and Acceptance Ratio	CPU and bandwidth
[25]	VNE-NTANRC algorithm	Slice isolation	Resource Efficiency and Acceptance Ratio	CPU and bandwidth
[26]	Stackelberg game approach	Slice isolation	Power consumption and delay	Subchannels, devices, data rate and traffic
[27]	VIKOR approach	Slice isolation	Resource Efficiency and Acceptance ratio	CPU and bandwidth
[28]	Stochastic Network Principles	Slice isolation	Resource Efficiency	Delay rate, traffic rate, capacity and service type
[29]	UFLOP mechanism	Slice creation and isolation	Provisioning ratio and traffic allocation ratio	CPU, bandwidth and delay time
[30]	SABA scheme	Slice isolation and management	Bandwidth consumption	Delay rate, bandwidth, link capacity and service type
[31]	FlexRAN controller	Slice isolation and management	RAM and CPU consumption	CPU, Memory and bandwidth utilization

Table 1. *Cont.*

Ref. No.	Methodology Adopted	Components	Performance Metrics	Considered Attributes
Proposed	Machine Learning for Slice Creation, PROMETHEE for Slice Isolation and Resource Transfer for Slice Management	Slice creation, isolation and management	Slice creation: Precision, Recall, and F1 Score Slice isolation and management: Resource Efficiency, Acceptance Rate	CPU capacity, link capacity, bandwidth, jitter, delay rate, and closeness centrality

3. Network Model and Mathematical Foundations

3.1. Physical Infrastructure (PI) Model

PI's physical structure is illustrated by a weighted undirected graph. The weighted undirected graph is denoted as $G^{PI} = (N^{PI}, E^{PI})$, where N^{PI} represents a group of physical nodes and E^{PI} represents a group of physical links. There is a unique set of properties for each graph node and link. Every individual node of a graph should be represented with CPU Capacity (CPU), Link Capacity (LC), Security Level (SL), Delay Rate (DR) and Jitter (JT). As an example, the parameters of the ith node of a graph should be specified as a subset containing $(CPU_i^{PI}, LC_i^{PI}, SL_i^{PI}, DR_i^{PI}, JT_i^{PI}), i \in N^{PI}$. The values for a node's relevant metrics are developed to match the true features of a 5G mobile network's core nodes. As with nodes, each link of a graph is specified with the subset reflecting the associated network features, $E_i^{PI}(BW_{ij}^{PI}, L_{ij}^{PI})$, where BW_{ij}^{PI} is the bandwidth between the link i and j of PI and L_{ij}^{PI} is the length of the link among the nodes i and j of PI.

3.2. Network Service Request (NSR) Model

In this analysis, it is assumed that the NSR parameters including the number of nodes in each NSR, CPU capacity, bandwidth, security level, delay, and user device type are assigned. Every NSR is represented by an undirected graph represented as, $G^{NSR} = (N^{NSR}, E^{NSR})$, where N^{NSR} denoted the nodes of the graph and E^{NSR} indicates the edges between them. Each node in NSR should be specified as a subset containing $(CPU_i^{NSR}, LC_i^{NSR}, SR_i^{NSR}, LT_i^{NSR}, DT_i^{NSR})$, $i \in N^{NSR}$ where $CPU_i^{NSR}, LC_i^{NSR}, SR_i^{NSR}$, LT_i^{NSR} and DT_i^{NSR} are the required CPU Capacity, Link Capacity, Security Level, Life Time and Device Type of the i^{th} node, respectively. The data transfer rate between nodes i and j is calculated using the formula BW_{ij}^{NSR}. We assume that the NSRs arrive during a single transmission window [33]. Nodes and connections for the NSR are allotted and set up at the beginning of each time period using the physical infrastructure. Every time a new NSR comes in, the available resources in the underlying physical infrastructure is adjusted based on the LT^{NSR} of the prior request.

3.3. Problem Description

It makes sense that efficient network slicing maximises the use of physical network resources while simultaneously lowering the cost of slice provisioning [27]. With these limitations in mind, we may formulate the issue of minimising the cost of slice provisioning as an integer linear programming model, as illustrated below.

$$\begin{aligned} min, \sum_{n^k \in N^{NSR}} \sum_{n^i \in N^{PI}} x_i^k (1 + SL(n^i)) CPU(n^k) \\ + \sum_{e^m \in E^{NSR}} \sum_{e^n \in E^{PI}} a_n^m BW(e^m) \end{aligned} \tag{1}$$

$$subject\ to, \sum_k x_i^k = 1,\ \forall i \in N^{NSR} \tag{2}$$

$$\sum_i x_i^k \leq 1,\ \forall i \in N^{PI} \tag{3}$$

$$x_i^k CPU(n^k) \leq CPU(n^i), \forall i \in N^{PI}, \forall k \in N^{NSR} \tag{4}$$

$$x_i^k SR(n^k) \leq SL(n^i), \forall i \in N^{PI}, \forall k \in N^{NSR} \tag{5}$$

$$\sum_{e^m \in E^{NSR}} a_n^m BW(e^m) \leq BW(e^n), \forall e^n \in E^{PI} \tag{6}$$

x_i^k is a binary variable. If the k-th node of NSR is served onto the i-th node of PI, then $x_i^k = 1$; otherwise, $x_i^k = 0$. a_n^m is a binary variable. If the m-th link of NSR is served to the n-th link of PI, then $a_n^m = 1$; otherwise, $a_n^m = 0$. In order to guarantee the integrity of the nodes, network providers employ various additional measures. To enhance resource allocation and slice generation accuracy at nodes, we additionally take into account Link Capacity, Delay Rate, and Jitter in addition to CPU Capacity and bandwidth. Constraint (2) specifies that each request node must be associated with a physical node. The third constraint guarantees that a given physical node will only ever host a single node from the same request. Limitation (4) shows the CPU Capacity. The security constraints of all nodes are guaranteed by the fifth constraint. Constraint (6) ensures that the bandwidth requested by services delivered over a physical link does not exceed the available bandwidth.

3.4. Network Node Attributes

Nodes in both the physical infrastructure and the NSR have varying parameter values; therefore, they are seen as complicated networks. According to the core principles of complex network theory, examining the significance of the nodes is required. Every node in a complex network is affected by four critical features of this problem. The significance of nodes in a network is evaluated in light of the values assigned to each node's set of associated parameters. The formulas for the elements are derived from the basic complex network theory.

3.4.1. Node Capacity Element (NCE)

The capacity of a node is set by the amount of CPUs that are available at any given time. If a node has more CPUs, it may be capable of providing service to more NSRs. Because of this, a node with many CPUs should be given preference when provisioning resources. As soon as the service of an NSR is completed, the value of the element is updated instantaneously in the system. It is shown in the following equation that the NCE of node *i* at the given time *t* is as follows

$$NCE_i^t = CPU(N_i^{PI}) \tag{7}$$

3.4.2. Node Topology Element (NTE) and Node Bandwidth Element (NBE)

Both NTE and NBE are determined by the physical node's number of nearby connections. NTE denotes the total number of neighbouring connections to a node, whereas NBE denotes the total bandwidth of nearby links. A node with a higher NTE and NBE is prioritized for provisioning in the network. The value of each element is updated as per the life time of each NSR. Listed below are the equations for updating the factors at time *t*.

$$NTE_i^t = \sum_{j=1}^{n} a_{ij,available}, \quad i \neq j \tag{8}$$

$$NBE_i^t = \sum_{j=1}^{n} BW_{ij}^{PI}, \quad i \neq j \tag{9}$$

where n is the number of nodes in the network, and if the i^{th} node is connected to the j^{th} node and both are unserved, then $a_{ij,available}$ will be 1; otherwise, it will be 0.

3.4.3. Node Closeness Centrality Element (NCCE)

Using NCE, NTE, and NBE, we can examine the local information for every node. However, the relevance of a node in a network can be assessed by the weight it is assigned when determining the shortest path between any two other nodes. NCCE's shortest route information is used to determine a node's global importance. That is why proximity to other nodes increases a node's centrality. The following is the formula to calculate NCCE at a given time t:

$$NCCE_i^t = \{ \sum_{j=1}^{nNodes} L_{i,j} \}^{-1}, \quad i \neq j \tag{10}$$

where $L_{i,j}$ is the shortest path among the vertices i and j.

4. Proposed Strategy

As seen in Figure 1, the planned work is divided into three phases, each with its own set of responsibilities. Phase I focuses on the slice creation of physical infrastructure as well as NSRs. Phase II is responsible for Slice Isolation which ensures that each NSR achieves the optimal allocation of resources in PI. Phase III combines the findings of Phases I and II and incorporates slice management through resource transfer. The following subsections describe the procedures involved in each phase.

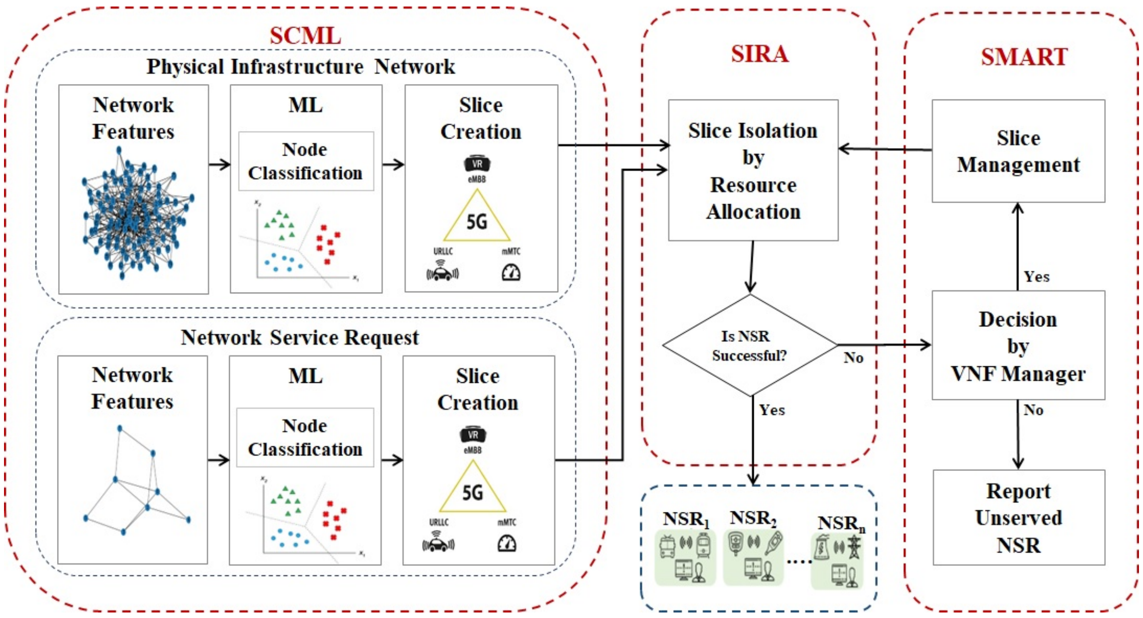

Figure 1. Proposed 5G mobile network orchestration for network slicing.

4.1. Slice Creation by Machine Learning Approaches

The field of artificial intelligence has made extensive use of machine learning in a variety of research applications that concentrate on system characterization. Machine learning can play a vital role in the implementation of a resource provisioning mechanism for network slicing. Machine learning algorithms can be classified into multiple categories, such as supervised, unsupervised, and reinforcement learning. A supervised learning approach employs a labelled data set for model training. The system knows both the input data and the desired output data to be produced in a supervised manner. Different machine learning algorithms such as K-Nearest Neighbour, Support Vector Machine, Naive Bayes and Random Forest are used to perform classification [34–36]. Figure 2 shows the workflow of supervised machine-learning-based classification. As shown in Figure 2, supervised learning algorithms divide the nodes into the eMBB, mMTC, and uRLLC slices after receiving KPIs from the network nodes. Each node's KPIs are contained in a single database, which must be split into two separate datasets for training and validation. Classification of slices necessitates training the ML algorithm with a collection of characteristics from multiple services. Since the predictive model can provide results for new data after training, this allows for new known input values (KPIs) and new unknown slices to be assigned. Therefore, once the ML algorithm has been trained, the predictive model may correctly forecast or classify the requests for service. The ML algorithm is used to classify network slices in Algorithm 1. These three types of slices will be sorted using different methods of Supervised Machine Learning (SM) classification.

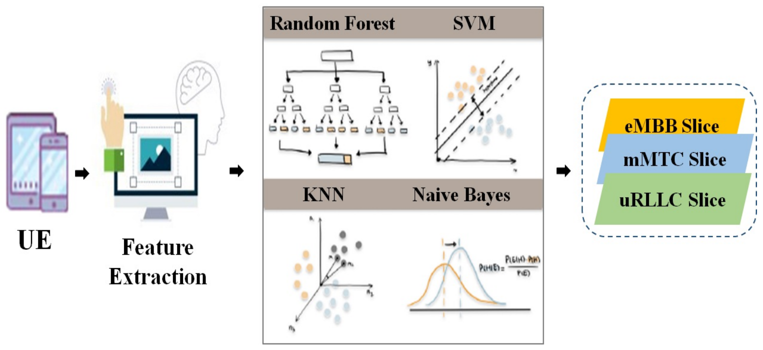

Figure 2. NSR placement process with NIRA and SPA.

Algorithm 1 Slice classification in 5G

Input: G^{PI}
Output: $eMBB, mMTC, uRLLC$

```
 1: Represent eMBB, mMTC, uRLLC as an empty vector of length vs.
 2: Represent node, k = 0
 3: Receive the node characteristics of $N^{PI}$
 4: Do the supervised ML Classification()
 5: while $k \leq v$ do
 6:     $X_{Predict}$ of k
 7:     $Y_{Predict}$=Predict the slice of $X_{Predict}$
 8:     if $Y_{Predict}$=eMBB then
 9:         eMBB ← node k
10:     else if $Y_{Predict}$=mMTC then
11:         mMTC ← node k
12:     else
13:         uRLLC ← node k
14:     end if
15:     k = k + 1
16: end while
17: return eMBB, mMTC, uRLLC slices
```

4.1.1. K-Nearest Neighbour (KNN)

KNN is a straightforward method for storing all available cases and grouping them according to their degree of similarity. A data sample with an unknown distribution is classified using the class of KNN. If the majority of the sample's nearest neighbours belongs to the same class, the sample is classified as belonging to that class. KNN supports a wide variety of distance functions, including Euclidean, Minkowski, and Manhattan. In this scheme, the distance is calculated using the Euclidean distance function. The Euclidean distance function is

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \tag{11}$$

where x and y are defined as a pair of samples.

4.1.2. Support Vector Machine (SVM)

Classification problems can be solved using SVM and it classifies points into one of two disjoint half spaces. Both linear and nonlinear data can be classified using SVM. To perform the classification, we need to identify the hyperplane that separates the two classes by the optimal (maximum) margin. The SVM hyperplane is denoted as

$$h_{w,b}(x) = g(w^2 x + b) \tag{12}$$

where w is a weight vector, x is an input vector and b is a bias vector.

4.1.3. Naive Bayes

A Naive Bayes classifier is a simple probabilistic classifier model and it is based on the Bayes theorem. A Naive Bayes classifier assumes that the presence or absence of one feature is unrelated to the other features existing in a class [37]. With the help of a set of node features that are supplied at the time of training, this method is able to correctly categorize the nodes of the network. The probability of Naive Bayes is

$$P(c|x) = \frac{P(x|c)P(c)}{(P(x)} \tag{13}$$

where $P(c|x)$ is the posterior probability of class, $P(x|c)$ is the likelihood probability of class, $P(c)$ is the class prior probability and $P(x)$ is the predictor prior probability.

4.1.4. Random Forest (RF)

In a Random Forest, each decision is represented by a single tree. Each decision tree is built on a randomly selected set of features. All new datasets are classified according to the majority of trees, and all unknown data samples are assigned to one of these classes. Using a group of classifiers $h_1(x), h_2(x) \ldots, h_n(x)$, a training set is randomly selected from the distribution of random vectors X and Y. The margin function is defined as follows:

$$mg(x,y) = av_k I(h_k(X) = Y) - max_{j \neq y} av_k I(h_k(X) = j) \tag{14}$$

where I is the indicator function. The margin is the difference between the average number of votes for the right class at X, Y and the average vote for any other class. The greater the margin, the more confident and accurate the classification.

Slice classification is performed using the supervised machine learning algorithms discussed above in the proposed network slicing process. Along with the appropriate slices serving as the target, the performance dataset that contains the KPIs of the network nodes is prepared. The effectiveness of the machine learning algorithms that have been implemented is analyzed in order to determine whether or not the method is appropriate for the dataset that has been prepared before it is proposed for the slice isolation and management procedures.

4.2. Slice Isolation by Resource Allocation (SIRA) Scheme

This phase is responsible for allocating the nodes and links of the NSR slices in the PI slices. According to SIRA, nodes in each slice of NSR and PI are prioritized first based on node information. Based on complex network theory, each node of the network is influenced by four elements (14)–(17); hence, the multi-criteria decision-making (MCDM) technique is used to prepare the node ranking (NR) array. An established MCDM [38] approach, PROMETHEE-II, can be used in this proposed study to create a node ranking system called Node Centric SIRA (NC-SIRA). Dijikstra's shortest path technique, referred to as Link Centric SIRA (LC-SIRA), is used to connect the assigned nodes. The VNFs are prepared for both the NC-SIRA and the LC-SIRA for the provisioning of nodes and links for NSRs in PI.

4.2.1. NC-SIRA-based Decision Making

Prior to receiving the NSR, the node ranking for slices of physical infrastructure should be created. Upon receipt of the NSR, the node ranking for the slices of logical structure is prepared. NCE, NTE, NBE, and NCCE all play an important part in the process of determining the value of each node in the network that is connected to it. As a result, it is imperative that the ranking must be constructed without compromising the intricacies of the nodes' configurations. It is necessary to use PROMETHEE-II in order to create an array of nodes for node ranking based on each node's four characteristics. This technique takes into account the element's individual preferences and disinterestedness. The PROMETHEE-II method for allocating resources for nodes and that consists of the following phases [39,40]:

(a) Making an evaluation table: Each node in the evaluation table has its factor values saved in the evaluation table. In order to prepare Table 2, the factors of each node are evaluated with their corresponding weight parameters. It is assumed that $N = n_1, n_2, \ldots, n_i$ and $F = f_1, f_2, f_3, f_4$ are the sets of nodes and factors, respectively.

Table 2. Evaluation table.

	f_1	f_2	f_3	f_4
n_1	$f_1(NCE^t_{n1})$	$f_2(NTE^t_{n1})$	$f_3(NBE^t_{n1})$	$f_4(NCCE^t_{n1})$
n_2	$f_1(NCE^t_{n2})$	$f_2(NTE^t_{n2})$	$f_3(NBE^t_{n2})$	$f_4(NCCE^t_{n2})$
.	.	.	.	.
.	.	.	.	.
n_i	$f_1(NCE^t_{ni})$	$f_2(NTE^t_{ni})$	$f_3(NBE^t_{ni})$	$f_4(NCCE^t_{ni})$

(b) Function of preferences planning: All nodes in a network are compared to each other in a pairwise comparison for each factor in the network.

$$d_k(n_i, n_j) = f_k(n_i) - f_k(n_j) \tag{15}$$

where the preference function is introduced to convert the $d_k(n_i, n_j)$ unified value as below

$$P_k(a, b) = G_k[d_k(n_i, n_j)] \tag{16}$$

(c) Calculating a World Preference Index:

$$\pi(n, m) = \sum_{k=1}^{i} P_k(a, b) * w_k \tag{17}$$

where w_k is a weight function of factor k which is assumed as greater than 0 and the sum of the weights is equal to 1.

(d) Positive and negative outranking flows are calculated: The following two factors are calculated to locate each node with respect to all other nodes.

$$\phi^+(a) = \frac{1}{i-1} \sum_{x \in A} \pi(a, x) \tag{18}$$

$$\phi^-(a) = \frac{1}{i-1} \sum_{x \in A} \pi(x, a) \tag{19}$$

where $\phi^+(a)$ is obtained by ranking the nodes according to the nonincreasing values of their respective positive flow values, whereas $\phi^-(a)$ is obtained by ranking the nodes according to the nondecreasing values of their respective negative flow values.

(e) Net flow calculation:

$$\phi(a) = \phi^{+}(a) - \phi^{-}(a) \tag{20}$$

The highest value of $\phi(a)$ refers to the best node in the node ranking (NR). After implementing NC-SIRA-based node allocation, the algorithm described in [40] is used to return the details of the NSR nodes that have been assigned in PI. Every time a new NSR is received, this procedure is carried out. The revised values of unsuccessful NSR (usNSR) and successful NSR (sNSR) are obtained at the conclusion of the execution.

4.2.2. LC-SIRA-Based Decision Making

NSR nodes in the physical infrastructure must be routed in the most efficient manner possible. The efficiency of the PI resources can be improved by providing the shortest route for NSR nodes. There are some situations in which the shortest route can be linked to existing NSRs. To do this, we need to first find all of the connections between NSR nodes and then organise them in a static link array with the lengths of the paths between them taken into account. This array can be used to connect NSRs in PI to each other. Connection building is expected to correspond to each of the most direct routes, one after the other. When it comes to link provisioning, the link array always uses the least time-consuming option among the available methods.

4.2.3. Resource Allocation and Performance Assessment

In this analysis, we assume that all significant NSR nodes have been automatically segmented as eMBB, mMTC, or uRLLC based on their corresponding data. An NSR must be allowed to provide any kind of slice request at any time. To provide the best feasible optimal provisioning for the NSR, the combined NC-SIRA and LC-SIRA are applied. All NSRs which received between 0 and T_{max} are assigned a node and connected to one another through this combined process. The lifetime of the received NSR is used to update the rank array and path array of the physical infrastructure to identify the best possible provisioning for the next arrival of NSR. This leads to better performance and effectiveness of NSR allocation under changing conditions in physical infrastructure. The proposed process, depicted in Figure 3, begins with the allocation of resources for NSRs for the longest possible time, denoted by the parameter (T_{max}). Three key measures used to analyze the efficacy of slice provisioning are the success rate, the resource efficiency and the execution time.

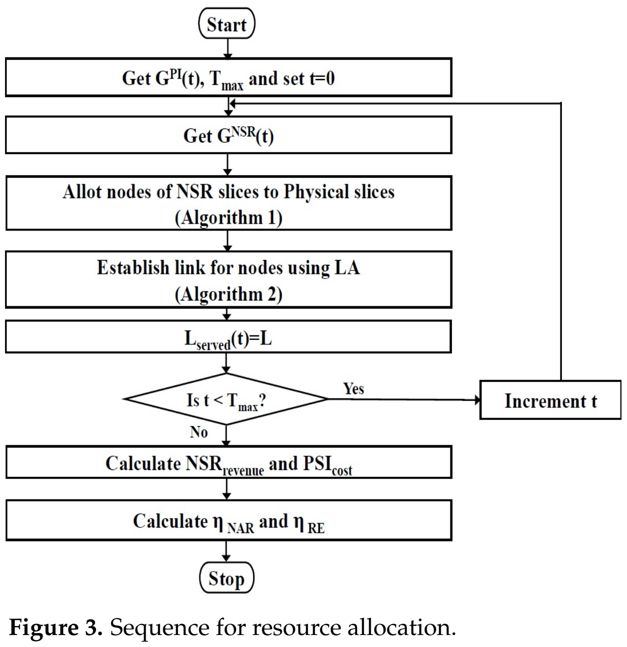
Start
Get $G^{PI}(t)$, T_{max} and set t=0
Get $G^{NSR}(t)$
Allot nodes of NSR slices to Physical slices (Algorithm 1)
Establish link for nodes using LA (Algorithm 2)
$L_{served}(t)$=L
Is $t < T_{max}$?
Yes
No
Increment t
Calculate $NSR_{revenue}$ and PSI_{cost}
Calculate η_{NAR} and η_{RE}
Stop

Figure 3. Sequence for resource allocation.

In supervised machine learning, algorithms are trained on a labelled dataset. This study investigates the classification of 5G network slices using a number of different classification algorithms such as KNN, SVM, Naive Bayes, and RF. In order to evaluate the performance of these algorithms, a dataset is divided into training and testing procedures. Precision, recall, and F1 Score of the obtained results are used to assess the most appropriate machine learning technique for the prepared performance dataset [34]. Regarding the assessment of resource allocation, the performance of every network must be measured in order to optimise the benefits to both the utility and its customers. Customers want utilities to provide them with consistent and high-quality service. Meanwhile, network service providers aim to maximize earnings while providing the highest degree of customer satisfaction. The network's performance is evaluated in this study by taking into consideration the benefits gained by all stockholders. When determining the outcomes, performance metrics such as the NSR acceptance ratio and resource efficiency are taken into account.

(i) NSR Acceptance Ratio (NAR): With this metric, we can see how well the chosen network slicing strategy is working with the underlying physical infrastructure. It is determined by dividing the number of successfully completed NSRs by the number of unsuccessful NSRs over a certain period of time (T_{max}), which is stated as

$$\eta_{NAR} = \frac{sNSR}{TNSR} \tag{21}$$

where sNSR and TNSR refer to successful and total NSR, respectively.

(ii) Resource Efficiency (RE): This indicator is found by dividing the revenue produced by the physical infrastructure by the amount of resources spent on building it. The amount of CPU power available on the nodes and the amount of link bandwidth needed by NSRs can be used to figure out how much revenue will be made. The expected cost of the investment is based on the actual infrastructure that will be needed for the goal. The following are the mathematical expressions for calculating RE:

$$\eta_{RE} = \frac{\sum_{t=0}^{T_{max}} NSR_{revenue}(t)}{\sum_{t=0}^{T_{max}} PSI_{cost}(t)} \tag{22}$$

$$NSR_{revenue}(t) = \sum_{i \in N^{NSR}} CPU_i^{NSR} + \sum_{m \in E^{NSR}} BW_m^{NSR} \tag{23}$$

$$PSI_{cost}(t) = \sum_{i \in N^{NSR}} CPU_i^{NSR} + \sum_{m \in E^{NSR}} \sum_{n \in E^{PI}} P_m^n * BW_m^{NSR} \tag{24}$$

where $NSR_{revenue}(t)$ is the revenue of NSR served at time 't', $PSI_{cost}(t)$ is the utilised physical infrastructure for serving the NSR received at time 't', and P_m^n is the number of physical infrastructure edges in path 'n' for mapping the virtual link 'm'.

4.3. Slice Management through Resource Transfer (SMART)

The VNF Manager (VNF-M) is the core of the SMART scheme since it is responsible for making the ultimate determination of the success of NSRs after thoroughly checking the available resources. VNF-M is responsible for four essential responsibilities related to NSRs and physical resources, such as dynamic provisioning, inner slicing, reporting, and priority provisioning as shown in Figure 4. In order to carry out the related duties, VNFs are defined individually.

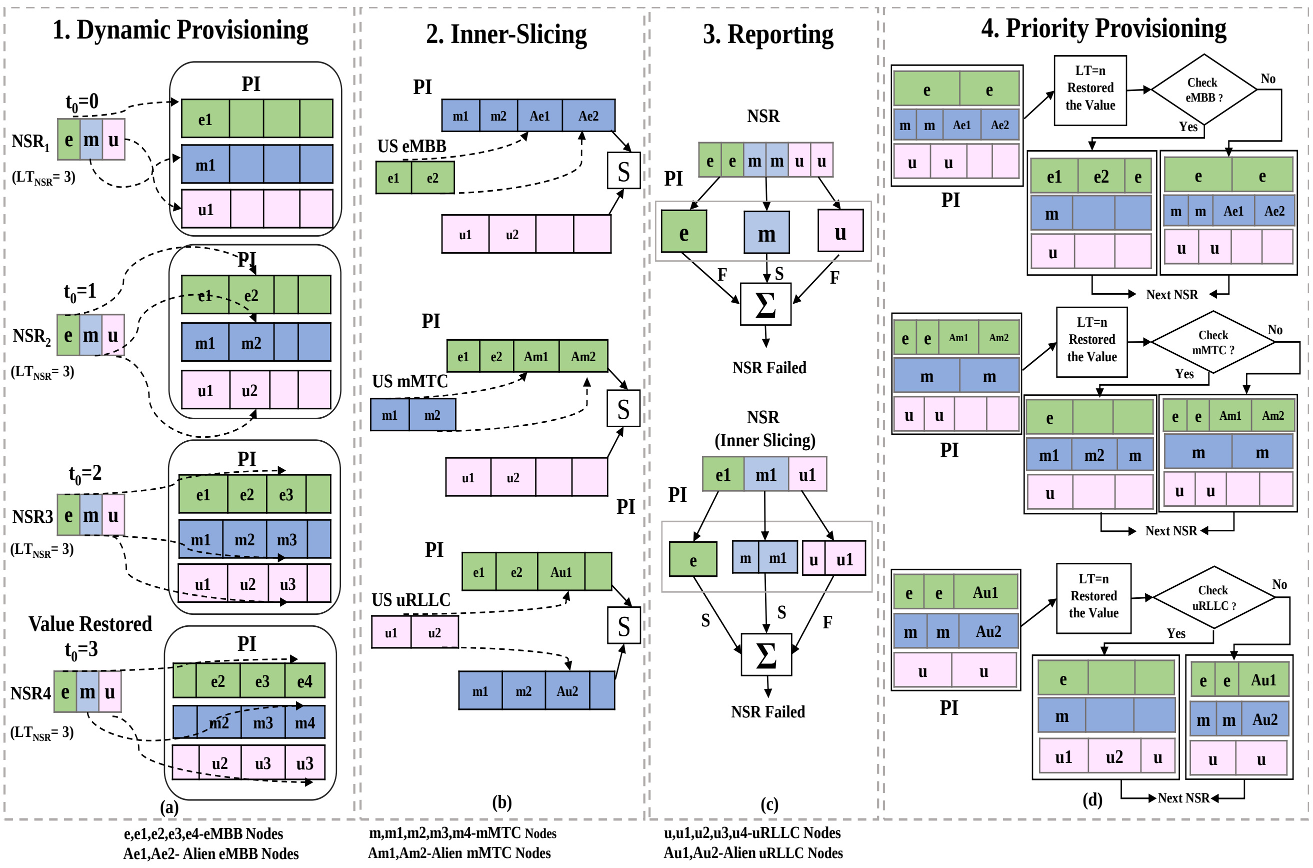

Figure 4. Workflow diagram for slice management through resource transfer (SMART).

4.3.1. VNF for Dynamic Provisioning

The life time of the NSRs that are currently in service is being checked on a continual basis. As soon as an NSR's life cycle is over, VNF-M releases the physical resources that have been occupied by the NSR, enabling the next generation of NSRs to profit from the resources. For example, at time $t = 0$, the NSR_1 with a life time of 3 consumes the resources of PI and the occupied resources (i.e., e1, m1, and u1) are withdrawn at time $t = 3$ according to its life cycle, as shown in Figure 3a.

4.3.2. VNF for Inner Slicing

This procedure begins when one of the slices (i.e., eMBB, uRLLC, or mMTC) of the current NSR fails from provisioning. VNF-M searches for the potential of delivering service to the failed nodes in the remaining two slices until it finds the resources in the associated slice of PI which are treated as "Alien Nodes". In other words, Alien Nodes are nodes of unsuccessful NSR slices that have been provisioned in two unassociated slices which are referred to as "Alien Slices". The concept of inner slicing and Alien Nodes is depicted in Figure 3b. As per the figure, the first part assumes that the eMBB nodes of the NSR slice are unsuccessful and the remaining uRLLC and mMTC slices are successful. This situation is realized by VNF-M which attempts to distribute resources from the mMTC and uRLLC slices. To begin, the VNF-M tries to assign resources from the mMTC slice, but if there are not enough resources available, it searches for resources for Alien Nodes in uRLLC. The provisioned eMBB nodes are becoming the Alien Nodes of the mMTC slice. The Alien Nodes are denoted as Ae1 and Ae2 which are allocated in mMTC. The eMBB nodes use the resources of mMTC and uRLLC until they find the space in the eMBB slice of PI. The same inner slicing is applicable in case of uRLLC or mMTC becomes unsuccessful. An improvement in resource efficiency and acceptance ratio is achieved by transferring available resources from successful slices to the failing slice.

4.3.3. VNF for Reporting

When resource provisioning via inner slicing fails or when more than one slice of an NSR fails, VNF-M reports the received NSRs as unsuccessful. According to Figure 3c, the first half assumes that eMBB and uRLLC slices fail, but mMTC succeeds. By reporting the NSR failure immediately, VNF-M avoids any more inner slicing operations in the future. NSR fails at inner slicing as shown in the second section of Figure 3c. VNF-M realizes that there are not enough resources in both Alien Slices when attempting inner slicing.

4.3.4. VNF for Priority Provisioning

In the next provisioning cycle, the Alien Nodes in each slice are given the highest priority because they are the most difficult to reach. When new NSRs seek resource provisioning at physical resources, priority service is given to Alien Nodes in each slice when the request is received. As illustrated in Figure 3d, when VNF-M detects that there are Alien Nodes in the mMTC slice, it first searches for the release of Alien Nodes (i.e., Ae1 and Ae2) supplied at the Alien Slice before receiving the new NSR when it goes to the next time cycle. If the resources required by the Alien Nodes are not available, the nodes are either kept in the same slice or are differently assigned to the appropriate slice, depending on the situation. Essentially, the goal of this endeavour is to provide priority service to Alien Nodes while simultaneously maintaining the Alien Slice in order to preserve resources for their linked NSR slice nodes for the following cycle.

5. Simulation Results and Discussion

5.1. Simulation Parameters

The proposed work is divided into three phases of execution, which are as follows: (i) slice creation using machine learning techniques (SCML), (ii) slice isolation using resource allocation (SIRA), and (iii) slice management using resource transfer (SMART). The effectiveness of the approaches used in the various phases is assessed through the use of the parameters associated with each approach. Table 3 lists the variables considered for the test case, including the variety of physical infrastructure resources and the NSR for the test scenario [24].

The scale-free network model proposed in [41] is used to create the graphs for the physical infrastructure and NSR. The proposed research used Python to construct the simulation platform.

5.2. Slice Creation through Machine Learning Techniques

The proposed work creates slices for both the PI and the NSR concurrently. The performance datasets for physical infrastructure and NSR have been prepared separately due to their distinct characteristics. For the purpose of preparing the performance dataset, several key indicators are identified, including CPU Capacity, Link Capacity, Bandwidth, Jitter, Delay Rate, and Closeness Centrality [15]. The datasets are classified into three categories based on their fields: eMBB, mMTC, and uRLLC. A dataset with 1000 samples is currently being assembled. Training and testing are conducted using a variety of supervised machine learning techniques, including KNN, Naive Bayes, SVM, and Random Forest. A 9:1 training-to-testing ratio is proposed for training. Precision, Recall, and F1 Score are used to compare the efficiency of different training methods.

The machine learning techniques are implemented using the Python library. The results for PI and NSR using various machine learning techniques are shown in Table 4. As illustrated in the table, all techniques capable of achieving greater than 90% training accuracy fall into one of three categories. When the overall percentage accuracy of the performance dataset is considered, Random Forest training gives better results than other techniques. Figure 5 illustrates how these four supervised machine learning techniques perform under different training-to-testing ratios, which is further evidence of their effectiveness. From the figure, it is clear that the Random Forest technique has a better overall accuracy rate than other techniques.

Table 3. Simulation parameters.

	Physical Infrastructure Network	
Definitions	Descriptions	Range
N^{PI}	Total number of PI nodes	100, 200, 300
CPU^{PI}	CPU capacity of each PI node	U(20,50)
LC^{PI}	Link capacity of each PI node	U(20,50)
SL^{PI}	Security level of a PI node	(0–1)
BW^{PI}	Bandwidth of each PI links	U(20,50)
DR^{PI}	The rate of delay for each PI node	(0–1)
JT^{PI}	The jitter for each PI node	(0–1)
	Network Service Request	
Definitions	Descriptions	Range
T^{NSR}	The total number of NSRs	U(5,35)
N^{NSR}	Nodes count in each NSR	20
CPU^{NSR}	CPU requirement of NSR node	U(5,25)
LC^{NSR}	LC requirement of NSR node	U(5,25)
BW^{NSR}	Bandwidth requirement of NSR node	U(5,25)
SR^{NSR}	Security level of a NSR node	(0–0.5)
LT^{NSR}	Life time of each NSR	T(10,35)

Table 4. Classificationand accuracy reports.

ML Approach	**90 % of Training Set and 10 % of Testing Set**									
	eMBB			**mMTC**			**uRLLC**			**Overall**
	Precision	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**	**Precision**	**Recall**	**F1 Score**	**(%)**
KNN	0.93	0.95	0.97	0.97	0.92	0.94	0.94	0.97	0.96	**95**
Naive Bayes	0.96	0.95	0.95	1.00	0.98	0.99	0.92	0.96	0.94	**96.3**
SVM	0.96	0.95	0.95	1.00	1.00	1.00	0.95	0.96	0.95	**97**
Random Forest	**0.94**	**1.00**	**0.97**	**1.00**	**1.00**	**1.00**	**1.00**	**0.93**	**0.97**	**98.3**

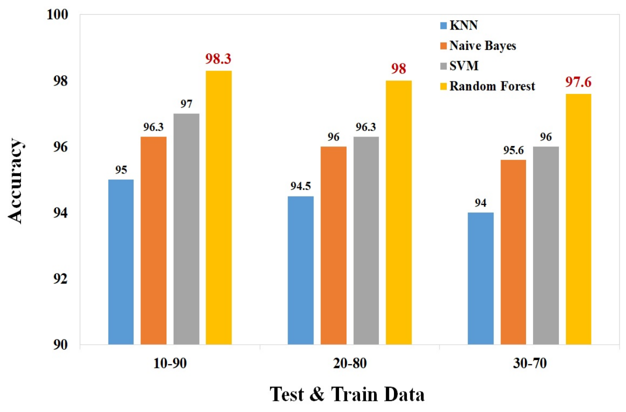

Figure 5. Slice classification accuracy for different ML approaches.

5.3. SIRA Scheme

This phase focuses on the allocation of resources for slice isolation, as shown in Figure 3. This system is evaluated by considering clusters of 100, 200, and 300 NSR nodes in different scenarios with 10, 20, and 30 NSR nodes, respectively. Table 2 shows the range of values from which network features are randomly selected. In order to better measure performance, the lengths of the graph's edges are fixed at one unit each. Node-generated values are classified using the Random Forest technique and made available as slices in PI and NSR to allocate resources. By assigning the available resources of PI slices to NSR slices in an appropriate manner, the SIRA scheme guarantees that the required NSRs will have access to the necessary nodes and links. The obtained resource efficiency under different system operating conditions is shown in Figure 6. Figure 6 summarises this generalisation about NSRs and resource efficiency by demonstrating how an increase in the number of NSRs reduces the network's resource efficiency while an increase in the number of nodes in the physical infrastructure raises efficiency. The resource efficiency calculation shows that adding more nodes to a network makes the shortest way shorter, which makes better use of the resources that are available.

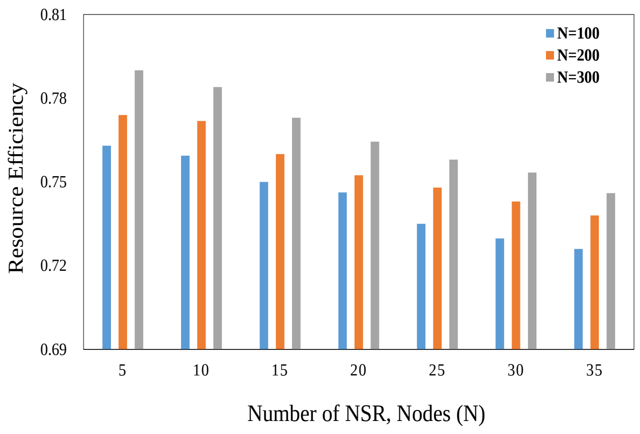

Figure 6. Resource efficiency for SIRA.

Additionally, more powerful and numerous nodes should be made available to accommodate the growing number of NSRs. Using the suggested method on an infrastructure that includes 100 nodes, efficiencies of 0.759, 0.746, and 0.733 are achieved for NSRs of 10, 20, and 30, respectively, as shown in Figure 7. The number of PI nodes increases to 200 and 300, and efficiency with that resource rises as well, achieving a maximum of 0.79 under 5 NSRs; a total amount of 9429 CPUs are utilised. Likewise, in the same operational state, bandwidth utilisation varies between a high of 29,662 and a low of 4300.

Comparisons are made between the proposed algorithm's performance and that of other algorithms in the literature, including those from the CN [24], SA [42], LAVA [43], GLL [44], VIKOR [27]. The algorithms are run under similar operating conditions to facilitate comparison. Figure 7 depicts the algorithms' resource efficiency results. SIRA outperforms all other resource allocation methods in a wide range of operating conditions and NSRs, as seen in Figure 7.

When compared to the MCDM-based node ranking algorithm, the VIKOR-CNSP method of resource allocation provides significantly better performance. The performance of NSR implementation, the SA-based approach, and LAVA were all superior to that of GLL. An efficiency of 0.753 is observed for 300 PI nodes and 30 NSRs. The suggested method for figuring out the acceptance rate is tested with NSRs ranging from 5 to 35. The expected lifetimes of the NSRs are arbitrarily chosen. The outcomes of the algorithms' effective execution across a range of physical infrastructure circumstances are depicted in Figure 8. Even though there are more actual nodes, the acceptance ratio drops as the overall NSR rises.

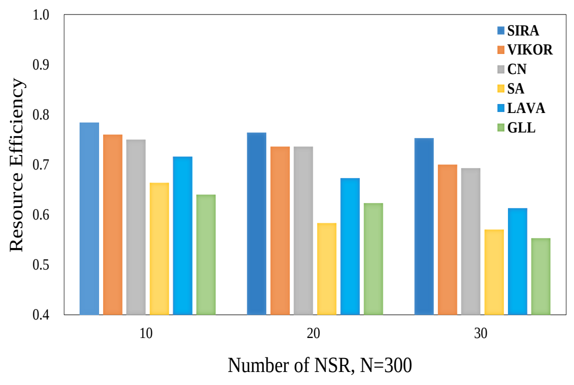

Figure 7. Resource efficiency of SIRA with other algorithms.

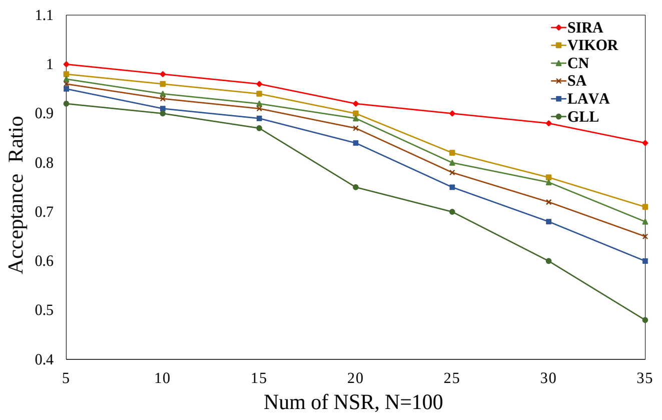

Figure 8. Acceptance ratio of SIRA with other algorithms.

Because more nodes are present, a greater percentage of the population accepts it. As NSR lifespans are shortened, the acceptance ratio also rises. The maximum acceptance ratio that can be attained with the suggested method is 0.98 when just 10 NSRs are used and 300 PI nodes are provided. A minimal acceptance ratio of 0.84 is achieved when 35 NSRs and 100 accessible nodes are used. It has been shown that the proposed algorithm outperforms previous approaches under numerous operational conditions.

VIKOR-CNSP is the second-best algorithm, with results that are comparable to those of the SIRA algorithm. For 35 NSRs under available nodes, the lowest acceptance ratios while employing the LAVA and GLL techniques to allocate network resources are 0.48 and 0.60, respectively. Algorithms are evaluated not only by their resource efficiency and acceptance rate but also by how quickly they can be executed. This determines how quickly each algorithm can complete a task. As shown in Figure 9, the execution times are compared under a variety of physical infrastructure conditions as well as different NSRs.

The algorithm execution times increase proportionally as the number of NSRs and the available physical infrastructure nodes increases. Under reduced NSRs and reduced PI, the three algorithms listed above complete their tasks in the shortest amount of time. When comparing the algorithms, SIRA's resource allocation algorithm takes the shortest amount of time compared to the other two under all working circumstances. Serving NSRs in the 5–35 range with a 100-node physical infrastructure takes less than 10 ms. Also, the response time for the same amount of NSRs used in 300 PI nodes is not higher than the 10 ms. The execution time of the VIKOR approach has not been included in the above figure for comparison because the execution times of the VIKOR and SIRA procedures are nearly identical. However, all the compared approaches solely deal with slice resource allocation, whereas the suggested SIRA approach also deals with slice formation via machine learning.

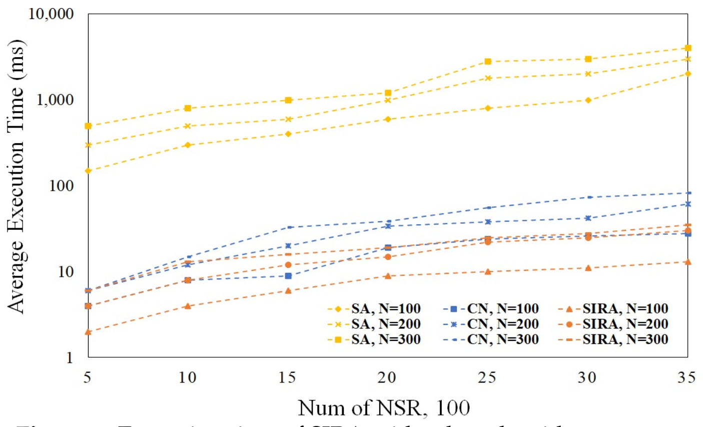

Figure 9. Execution time of SIRA with other algorithms.

5.4. SMART Scheme

In order to improve the performance of SIRA scheme-based resource allocation, our SMART scheme is being implemented in its current form. As previously stated, this scheme is responsible for four different tasks depending on the information it receives. With this scheme, the unsuccessful NSRs are given the greatest amount of attention to improve resource efficiency and acceptance ratios overall. The SMART scheme seeks to determine the most effective method of turning unsuccessful NSRs into successful ones. This function starts the node allocation process for unsuccessful NSRs when only one slice of an NSR is unsuccessful among the three different types of NSR slices that are available. It is necessary to treat the nodes in the unallocated slice of NSR as Alien Nodes, and the process of searching for node provisioning in the other two PI slices is known as inner slicing. The Alien Nodes are kept in unassociated slices until they are able to locate the provisioning in their associated primary interface slice. SMART scheme performance is demonstrated by executing the scheme under identical operating conditions, with nodes of PI ranging from 100 to 300 and NSRs ranging from 5 to 35. Figure 10 depicts the resource efficiency obtained under various conditions by using SIRA and SMART, as well as the results of the study.

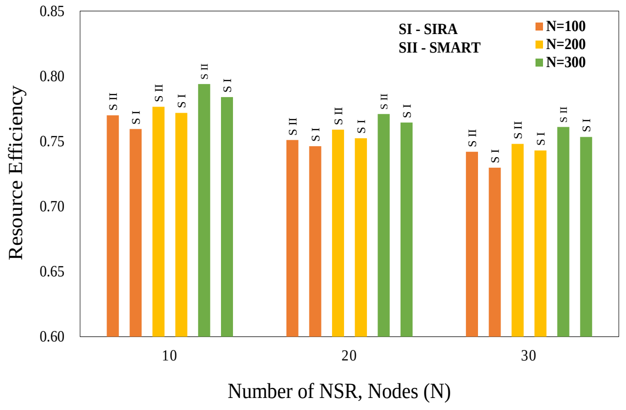

Figure 10. Resource efficiency of SMART with SIRA.

On the basis of the graph, it is clear that the SMART scheme contributes to a reasonable increase in resource efficiency. To serve 20 NSRs, it provides service to 142/126/132 nodes in the eMBB, mMTC, and uRLLC networks. SIRA scheme, on the other hand, could serve 142 eMBB nodes, 126 mMTC nodes, and 132 uRLLC nodes with the same number of NSRs. The SMART scheme achieves a greater improvement in serving eMBB, mMTC, and uRLLC nodes than the SIRA scheme. This results in a physical infrastructure that can serve a greater number of NSRs than any other resource allocation method. It can also be justified by comparing the acceptance ratio and execution time of two schemes, which are depicted in Figures 11 and 12, respectively, to see how they compare.

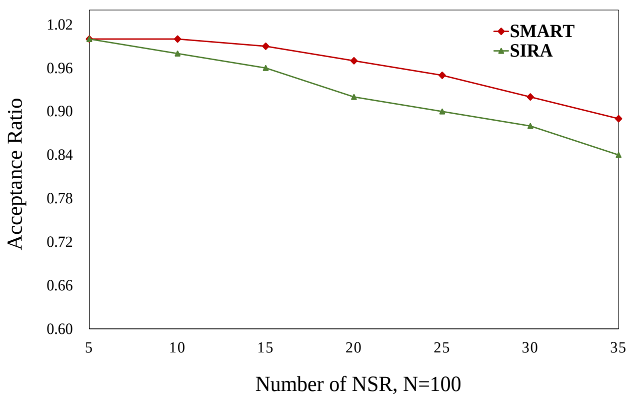

Figure 11. Acceptance ratio of SMART with SIRA.

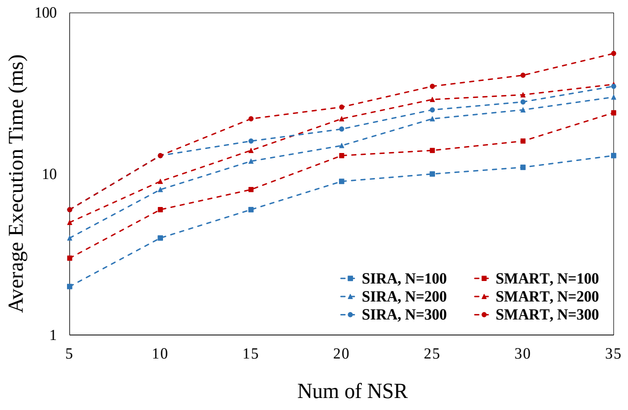

Figure 12. Execution time of SMART with SIRA.

According to the figures, the SMART scheme generates the optimal solution more quickly than any other algorithm while maintaining a higher acceptance ratio. Also, the figures demonstrate that the acceptance ratio of the SMART scheme has increased significantly as a result of the scheme's execution taking a comparatively short amount of time. With 10 NSRs and 300 reachable PI nodes, the proposed method can achieve a maximum acceptance ratio of 1. A minimal acceptance ratio of 0.89 is achieved when 35 NSRs and 100 accessible nodes are used. Moreover, the amount of CPU utilised and BW consumed by SMART is increased compared to SIRA schemes as shown in Figure 13. This current scheme can handle a maximum of 10,568 CPUs and a minimum of 3058 CPUs, and for bandwidth, a maximum of 31,628 and a minimum of 8340 under the PI with 100 and 300 nodes.

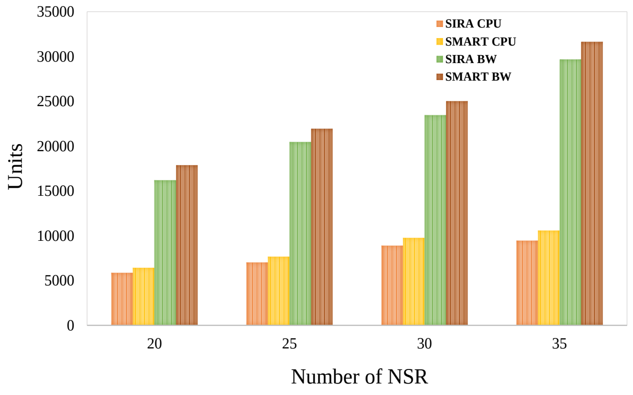

Figure 13. CPU and bandwidth utilization of SMART with SIRA.

6. Conclusions

The proposed research focused on three key aspects of resource allocation in 5G mobile networks: slice creation, slice isolation, and slice management. The prepared performance dataset addressed slice creation using machine learning techniques and discovered that slice creation using the Random Forest technique produced comparatively better accuracy than other techniques. Slice isolation was achieved using the SIRA scheme, which uses NC-SIRA and LC-SIRA for node allocation and link establishment. For node allocation, NC-SIRA used a PROMETHEE-II-based MCDM technique, while LC-SIRA used Dijkstra's algorithm. In addition, slice management was handled by the SMART scheme, which is responsible for four key aspects of resource allocation: dynamic slicing, reporting, inner slicing, and priority provisioning. The main goal of SMART is to turn a failed NSR slice into a success by using an inner slicing approach based on Alien Node provisioning concepts. The proposed schemes are put to the test under various network operating conditions with varying PI and NSRs. Furthermore, the life time of the NSRs is taken into account, allowing for effective dynamic slicing and priority provisioning. Also, the NSRs' security concerns and minimal SLA have been properly addressed by incorporating constraints into the resource allocation process. Finally, the effectiveness of the proposed scheme is supported by the literature findings. The scope of the proposed work can be enhanced to address further SLA and security issues. It can also be extended to address the mobility and energy management of user equipment.

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

Funding: This research was funded by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1A2C2014333).

Data Availability Statement: This study did not report any data.

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

References

1. 5G PPP Architecture Working Group. *View on 5G Architecture: Version 2.0*; 5G PPP Association: Brussels, Belgium, 2017.
2. Alliance, N. Description of network slicing concept. *NGMN 5G P* **2016**, *1*, 1–11.
3. Zhang, H.; Liu, N.; Chu, X.; Long, K.; Aghvami, A.H.; Leung, V.C. Network slicing based 5G and future mobile networks: Mobility, resource management, and challenges. *IEEE Commun. Mag.* **2017**, *55*, 138–145. [CrossRef]
4. Santos, J.F.; van de Belt, J.; Liu, W.; Kotzsch, V.; Fettweis, G.; Seskar, I.; Pollin, S.; Moerman, I.; DaSilva, L.A.; Marquez-Barja, J. Orchestration Next-Generation Services through End-to-End Networks Slicing. 2018. Available online: https://orca-project.eu/wp-content/uploads/sites/4/2018/10/orchestrating_e2e_network_slices_Final.pdf (accessed on 23 July 2023).
5. Nojima, D.; Katsumata, Y.; Shimojo, T.; Morihiro, Y.; Asai, T.; Yamada, A.; Iwashina, S. Resource isolation in RAN part while utilizing ordinary scheduling algorithm for network slicing. In Proceedings of the 2018 IEEE 87th Vehicular Technology Conference (VTC Spring) Porto, Portugal, 3–6 June 2018; IEEE: Piscataway, NJ, USA, 2018; pp. 1–5.
6. Bari, F.; Chowdhury, S.R.; Ahmed, R.; Boutaba, R.; Duarte, O.C.M.B. Orchestrating virtualized network functions. *IEEE Trans. Netw. Serv. Manag.* **2016**, *13*, 725–739. [CrossRef]
7. Riggio, R.; Bradai, A.; Harutyunyan, D.; Rasheed, T.; Ahmed, T. Scheduling wireless virtual networks functions. *IEEE Trans. Netw. Serv. Manag.* **2016**, *13*, 240–252. [CrossRef]
8. Ordonez-Lucena, J.; Ameigeiras, P.; Lopez, D.; Ramos-Munoz, J.J.; Lorca, J.; Folgueira, J. Network slicing for 5G with SDN/NFV: Concepts, architectures, and challenges. *IEEE Commun. Mag.* **2017**, *55*, 80–87. [CrossRef]
9. Subedi, P.; Alsadoon, A.; Prasad, P.; Rehman, S.; Giweli, N.; Imran, M.; Arif, S. Network slicing: A next generation 5G perspective. *EURASIP J. Wirel. Commun. Netw.* **2021**, *2021*, 102. [CrossRef]
10. Jiang, C.; Zhang, H.; Ren, Y.; Han, Z.; Chen, K.C.; Hanzo, L. Machine learning paradigms for next-generation wireless networks. *IEEE Wirel. Commun.* **2016**, *24*, 98–105. [CrossRef]
11. Pavan, M.; Reddy, B.R. Machine Learning based Prediction for Channel Stability in a 5G Network. In Proceedings of the 2022 IEEE North Karnataka Subsection Flagship International Conference (NKCon), Vijaypur, India, 20–21 November 2022; pp. 1–4. [CrossRef]

12. Islam Arif, M.A.; Kabir, S.; Hussain Khan, M.F.; Kumar Dey, S.; Rahman, M.M. Machine Learning and Deep Learning Based Network Slicing Models for 5G Network. In Proceedings of the 2022 25th International Conference on Computer and Information Technology (ICCIT), Cox's Bazar, Bangladesh, 17–19 December 2022; pp. 96–101. [CrossRef]
13. O'shea, T.; Hoydis, J. An introduction to deep learning for the physical layer. *IEEE Trans. Cogn. Commun. Netw.* **2017**, *3*, 563–575. [CrossRef]
14. Bega, D.; Gramaglia, M.; Banchs, A.; Sciancalepore, V.; Costa-Pérez, X. A machine learning approach to 5G infrastructure market optimization. *IEEE Trans. Mob. Comput.* **2019**, *19*, 498–512. [CrossRef]
15. Abidi, M.H.; Alkhalefah, H.; Moiduddin, K.; Alazab, M.; Mohammed, M.K.; Ameen, W.; Gadekallu, T.R. Optimal 5G network slicing using machine learning and deep learning concepts. *Comput. Stand. Interfaces* **2021**, *76*, 103518. [CrossRef]
16. Chergui, H.; Verikoukis, C. Offline SLA-constrained deep learning for 5G networks reliable and dynamic end-to-end slicing. *IEEE J. Sel. Areas Commun.* **2019**, *38*, 350–360. [CrossRef]
17. 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]
18. Brik, B.; Boutiba, K.; Ksentini, A. Deep Learning for B5G Open Radio Access Network: Evolution, Survey, Case Studies, and Challenges. *IEEE Open J. Commun. Soc.* **2022**, *3*, 228–250. [CrossRef]
19. Lee, Y.L.; Loo, J.; Chuah, T.C.; Wang, L.C. Dynamic Network Slicing for Multitenant Heterogeneous Cloud Radio Access Networks. *IEEE Trans. Wirel. Commun.* **2018**, *17*, 2146–2161. [CrossRef]
20. Srinivasan, T.; Wang, X.; Kim, H.J.; Ra, I.H. Performance enhancement for microgrids under the demand uncertainties with the presence of multiple DGs through stochastic ranking algorithm. *J. Electr. Eng. Technol.* **2021**, *16*, 223–238. [CrossRef]
21. Prasad, J.; Senthil, M.; Yadav, A.; Gupta, P.; Anusha, K.S. *A Comparative Study of Machine Learning Algorithms for Gas Leak Detection*; Springer: Singapore, 2020; pp. 81–90. [CrossRef]
22. Delgado, C.; Canales, M.; Ortín, J.; Gállego, J.R.; Redondi, A.; Bousnina, S.; Cesana, M. Joint Application Admission Control and Network Slicing in Virtual Sensor Networks. *IEEE Internet Things J.* **2018**, *5*, 28–43. [CrossRef]
23. Ye, Q.; Li, J.; Qu, K.; Zhuang, W.; Shen, X.S.; Li, X. End-to-End Quality of Service in 5G Networks: Examining the Effectiveness of a Network Slicing Framework. *IEEE Veh. Technol. Mag.* **2018**, *13*, 65–74. [CrossRef]
24. 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* **2018**, *6*, 19691–19701. [CrossRef]
25. 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.* **2017**, *5*, 108–120. [CrossRef]
26. Sun, Y.; Peng, M.; Mao, S.; Yan, S. Hierarchical radio resource allocation for network slicing in fog radio access networks. *IEEE Trans. Veh. Technol.* **2019**, *68*, 3866–3881. [CrossRef]
27. Li, X.; Guo, C.; Gupta, L.; Jain, R. Efficient and secure 5G core network slice provisioning based on VIKOR approach. *IEEE Access* **2019**, *7*, 150517–150529. [CrossRef]
28. Xu, Q.; Wang, J.; Wu, K. Learning-based dynamic resource provisioning for network slicing with ensured end-to-end performance bound. *IEEE Trans. Netw. Sci. Eng.* **2018**, *7*, 28–41. [CrossRef]
29. Chien, H.T.; Lin, Y.D.; Lai, C.L.; Wang, C.T. End-to-end slicing with optimised communication and computing resource allocation in multi-tenant 5G systems. *IEEE Trans. Veh. Technol.* **2019**, *69*, 2079–2091. [CrossRef]
30. Sun, Y.; Qin, S.; Feng, G.; Zhang, L.; Imran, M.A. Service provisioning framework for RAN slicing: User admissibility, slice association and bandwidth allocation. *IEEE Trans. Mob. Comput.* **2020**, *20*, 3409–3422. [CrossRef]
31. Mekki, M.; Arora, S.; Ksentini, A. A Scalable Monitoring Framework for Network Slicing in 5G and Beyond Mobile Networks. *IEEE Trans. Netw. Serv. Manag.* **2021**, *19*, 413–423. [CrossRef]
32. Troia, S.; Vanegas, A.F.R.; Zorello, L.M.M.; Maier, G. Admission Control and Virtual Network Embedding in 5G Networks: A Deep Reinforcement-Learning Approach. *IEEE Access* **2022**, *10*, 15860–15875. [CrossRef]
33. Gao, L.; Li, P.; Pan, Z.; Liu, N.; You, X. Virtualization framework and VCG based resource block allocation scheme for LTE virtualization. In Proceedings of the 2016 IEEE 83rd Vehicular Technology Conference (VTC Spring), IEEE, Nanjing, China, 15–18 May 2016; pp. 1–6.
34. Gupta, R.K.; Misra, R. Machine learning-based slice allocation algorithms in 5G networks. In Proceedings of the 2019 International Conference on Advances in Computing, Communication and Control (ICAC3), Mumbai, India, 20–21 December 2019; IEEE:Piscataway, NJ, USA, 2020; pp. 1–4.
35. Archanaa, R.; Athulya, V.; Rajasundari, T.; Kiran, M.V.K. A comparative performance analysis on network traffic classification using supervised learning algorithms. In Proceedings of the 2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS), Coimbatore, India, 6–7 January 2017; pp. 1–5. [CrossRef]
36. Balachandran, A.; Amritha, P.P. VPN Network Traffic Classification Using Entropy Estimation and Time-Related Features. In *IOT with Smart Systems*; Senjyu, T., Mahalle, P., Perumal, T., Joshi, A., Eds.; Springer: Berlin/Heidelberg, Germany, 2022; pp. 509–520.
37. Nair, M.R.; Ramya, G.; Sivakumar, P.B. Usage and analysis of Twitter during 2015 Chennai flood towards disaster management. *Procedia Comput. Sci.* **2017**, *115*, 350–358. [CrossRef]
38. Brans, J.P.; Mareschal, B.; Figueira, J.; Greco, S. *Multiple Criteria Decision Analysis: State of the Art Surveys*; Springer Science+ Business Media, Inc.: New York, NY, USA, 2005; pp. 163–196.
39. Greco, S.; Figueira, J.; Ehrgott, M. *Multiple Criteria Decision Analysis*; Springer: Berlin/Heidelberg, Germany, 2016; Volume 37.

40. 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* **2023**, *23*, 1556. [CrossRef]
41. Barabási, A.L.; Albert, R. Emergence of scaling in random networks. *Science* **1999**, *286*, 509–512. [CrossRef]
42. Aarts, E.; Korst, J. *Simulated Annealing and Boltzmann Machines: A Stochastic Approach to Combinatorial Optimization and Neural Computing*; John Wiley & Sons, Inc.: Hoboken, NJ, USA, 1989.
43. Yu, C.; Hou, W.; Guan, Y.; Zong, Y.; Guo, P. Virtual 5G network embedding in a heterogeneous and multi-domain network infrastructure. *China Commun.* **2016**, *13*, 29–43. [CrossRef]
44. Mijumbi, R.; Serrat, J.; Gorricho, J.L.; Bouten, N.; De Turck, F.; Davy, S. Design and evaluation of algorithms for mapping and scheduling of virtual network functions. In Proceedings of the 2015 1st IEEE Conference on Network Softwarization (NetSoft), London, UK, 13–17 April 2015; IEEE:Piscataway, NJ, USA, 2015; pp. 1–9.