drones

Article

Non-Euclidean Graph-Convolution Virtual Network Embedding for Space–Air–Ground Integrated Networks

Ning Chen [1,2], Shigen Shen [3,*], Youxiang Duan [1], Siyu Huang [4], Wei Zhang [5,*] and Lizhuang Tan [5]

1 Qingdao Institute of Software, College of Computer Science and Technology, China University of Petroleum (East China), Qingdao 266580, China
2 State Key Laboratory of Integrated Services Networks, Xidian University, Xi'an 710071, China
3 School of Information Engineering, Huzhou University, Huzhou 313000, China
4 Xiongan Institute of Innovation, Chinese Academy of Sciences, Baoding 071702, China
5 Shandong Provincial Key Laboratory of Computer Networks, Shandong Computer Science Center (National Supercomputer Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan 250013, China
* Correspondence: shigens@zjhu.edu.cn (S.S.); wzhang@sdas.org (W.Z.)

Abstract: For achieving seamless global coverage and real-time communications while providing intelligent applications with increased quality of service (QoS), AI-enabled space–air–ground integrated networks (SAGINs) have attracted widespread attention from all walks of life. However, high-intensity interactions pose fundamental challenges for resource orchestration and security issues. Meanwhile, virtual network embedding (VNE) is applied to the function decoupling of various physical networks due to its flexibility. Inspired by the above, for SAGINs with non-Euclidean structures, we propose a graph-convolution virtual network embedding algorithm. Specifically, based on the excellent decision-making properties of deep reinforcement learning (DRL), we design an orchestration network combined with graph convolution to calculate the embedding probability of nodes. It fuses the information of the neighborhood structure, fully fits the original characteristics of the physical network, and utilizes the specified reward mechanism to guide positive learning. Moreover, by imposing security-level constraints on physical nodes, it restricts resource access. All-around and rigorous experiments are carried out in a simulation environment. Finally, results on long-term average revenue, VNR acceptance ratio, and long-term revenue–cost ratio show that the proposed algorithm outperforms advanced baselines.

Keywords: future internet architecture; space–air–ground integrated network; resource orchestration; virtual network embedding; graph convolution; non-Euclidean structure; deep reinforcement learning

Citation: Chen, N.; Shen, S.; Duan, Y.; Huang, S.; Zhang, W.; Tan, L. Non-Euclidean Graph-Convolution Virtual Network Embedding for Space–Air–Ground Integrated Networks. *Drones* **2023**, *7*, 165. https://doi.org/10.3390/drones7030165

Academic Editor: Emmanouel T. Michailidis

Received: 01 February 2023
Revised: 22 February 2023
Accepted: 25 February 2023
Published: 27 February 2023

1. Introduction

Future sixth-generation (6G) mobile technology is devoted to providing ubiquitous, anytime, anywhere intelligent information services through wireless networks [1,2]. Realizing the coverage of all corners of the earth without blind spots, especially for remote and rural areas, is an indispensable goal pursued by the future 6G networks [3]. However, depending on existing ground infrastructure alone is not enough. As illustrated in Figure 1, the space–air–ground integrated network (SAGIN) is a novel network architecture with a bright future. Through the three-dimensional interconnection, it can provide high-efficiency, stable, and reliable low-latency communication services with seamless global coverage [4,5]. Its network architecture is composed of three parts: (1) a space network, mainly composed of different orbits, such as geosynchronous orbit (GEO), low-earth orbit (LEO), and satellite clusters with different functions, e.g., communication satellites, navigation satellites, remote sensing satellites, etc.; (2) an aerial network, mainly composed of aircraft, high-altitude platforms (HAPs), and unmanned aerial vehicles (UAVs) [6], etc.; and a (3) ground

network, mainly composed of traditional communication networks, e.g., mobile terminal communications, the Internet of things (IoT) [7], wireless sensors [8], ad hoc vehicles [9], etc.

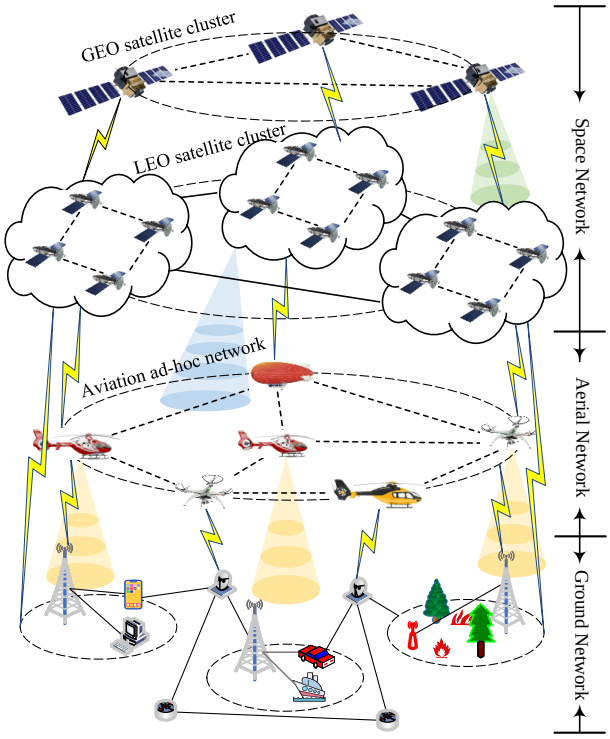


Figure 1. Typical architecture of space–air–ground integrated networks

In a SAGIN, each part has its specific advantages and corresponding limitations. To improve the utilization of limited resources, it is necessary to make full use of the advantages of each part in terms of coverage, flexibility, reliability, and availability [10]. However, since some nodes are always in a high-speed moving state, a SAGIN has heterogeneous and time-varying characteristics among others. Similarly, the consumption of hardware resources such as cache and computing will greatly increase. It has caused many essential problems to be solved, such as resource orchestration (RO) [11], routing scheduling [12], spectrum allocation [13], load balancing [14], etc.

It is worth noting that AI techniques [15], especially deep reinforcement learning (DRL), have long been successful and maturely applied to the field of wireless communications [16,17]. Through AI-related technologies, entities in the network can adjust and optimize adaptively according to the collected relevant feature information. This bright prospect has prompted us to introduce AI technology into SAGINs [18]. However, it requires plentiful data and resources preparation during the training process while providing comfortable intelligent services [19]. Therefore, the RO problem of SAGINs has become the focus of attention. The decision-making of resource scheduling is related to whether a SAGIN can employ the smallest resources to complete the most tasks (improving resource utilization) [20]. Through logical abstraction, network virtualization (NV) can provide intelligent and flexible RO decisions, where virtual network embedding (VNE) is the key to research [4,21]. Without recreating the hardware topology, VNE decouples resources from complex physical networks through slicing and isolation to provide elastic and flexible resource allocation strategies. Inspired by it, we convert the RO of a SAGIN into VNE problems for further research.

1.1. Pose Challenges

In the problem of RO based on VNE, how to effectively fit the complex characteristics of different domain networks and pay attention to the dynamic changes of the physical environment in real time are the keys to improving the success rate of responding to requests [4,22]. The traditional VNE algorithm uses a heuristic search algorithm, through

artificial subjective embedding, which reduces the effectiveness of embedding to a certain extent and introduces serious interference errors. Some related works combine AI technology to adaptively fuse network characteristics [10,23], such as calculating node embedding probability through a fully connected layer or a convolutional layer. However, these methods neglect the non-Euclidean structure (Different from traditional aligned regular data structures, irregular graph structures such as communication networks, social networks, and knowledge graphs do not belong to the category of Euclidean structures. Therefore, they are defined as non-Euclidean structures.) of SAGIN, which makes it difficult to effectively integrate the original features of nodes [24].

In addition, with the development of Internet technologies, network security issues have aroused people's high attention [25,26]. In particular, information resources, privacy resources, etc., if they are not protected, they can easily be maliciously attacked and obtained by criminals. However, in the VNE problem, this aspect is indeed a point that researchers tend to overlook. Especially in SAGINs, the importance of the resources of different nodes in the three-dimensional global architecture is different, and random access brings serious threats to the important resources of the nodes.

1.2. Work Contributions

To solve these problems, we propose a security-aware non-Euclidean graph-convolution VNE algorithm for SAGINs. Specifically, the contributions of this work are:

- To scientifically schedule the resources while maximizing resource utilization, we employ VNE technology to arrange multidomain resources in a SAGIN.
- Based on the excellent decision-making properties of DRL, we propose an orchestration network to calculate the embedding probability of nodes. It mainly utilizes graph convolution to capture the non-Euclidean structural information of the neighborhood, which updates its own features by fusing the feature information of neighboring nodes. In addition, the breadth-first search algorithm is employed for link mapping. Finally, a specific reward mechanism combined with the gradient descent mechanism is employed to guide positive learning.
- In view of the security issues of VNE in SAGINs, we impose security-level constraints on nodes, so that resource access is restricted. Each SAGIN node is subject to the security level, and VN requests can only access resources that meet the security constraints.
- Aiming at the actual characteristics of SAGINs, a simulation environment is modeled. Moreover, we conduct all-around rigorous experiments. In the end, the experimental results on long-term average revenue, VNR acceptance ratio, and long-term revenue–cost ratio show that the proposed algorithm outperforms advanced baselines.

1.3. Paper Organization

The other parts of this paper are organized as follows: In Section 2, we review related works. In Section 3, the problem modeling and definition is stated. In Section 4, the VNE embedding constraints and related indicators are presented. In Section 5, we introduce the flow of the proposed algorithm in detail. The results of the experiment and analysis are shown in Section 6. Finally, in Section 7, we summarize the work done and provide an outlook on future work plans.

2. Related Work

2.1. AI-Driven Space–Air–Ground Integrated Network

Networks with a single spatial dimension, such as terrestrial networks, cannot achieve full coverage and satisfactory quality of service (QoS). For example, technologies such as autonomous driving, intelligent in-vehicle applications, disaster monitoring, edge computing [27], etc., cannot be achieved alone by ground-based networks. Therefore, SAGINs have attracted widespread attention from all walks of life. Works [4,28] summarize and review SAGIN's structure from different angles, analysis the existing works, and propose future development trends and prospects. One point worthy of attention is that they all

advocate the combined utilization of AI and SAGIN to provide intelligent scalability and high QoS applications. To supplement the terrestrial network and improve the quality of experience (QoE), Kato et al. [18] proposed to combine AI technology to optimize a SAGIN and improve the performance of the network. In response to link errors, a series of concurrent data uploading, and computing offloading problems caused by frequent requests for intelligent services in SAGIN, Gu et al. [14] proposed a coding storage and computing system that was combined with AI (CSC-AI). This framework could effectively accelerate the distributed AI calculation in the SAGIN and realize reliable retrieval services in massive data. In response to the problem that the enhanced Internet of vehicles in edge computing relies heavily on the connectivity of infrastructure, which makes remote areas prone to failures, Yu et al. [29] proposed a SAGIN framework (EC-SAGIN) that could support edge computing in combination with deep learning, which used AI-driven algorithms to make intelligent decisions. In addition, there have also been a large number of technologies that apply AI-driven SAGINs to various fields for this bright prospect [10,30].

AI technology has brought many driving forces to the development of SAGINs, and it has significantly increased the scalability of services and the QoS of users. This promising prospect has prompted many hardworking researchers to devote themselves to this research career. Similarly, this is also the focus of our work and research.

2.2. Virtual Network Embedding Algorithm

Because of its flexibility, the VNE algorithm can overcome the resistance of the network to the dynamic variation of the architecture, so it is widely applied in the RO problem of physical networks [21]. Unlike other works, Chowdhury et al. [31] treated node embedding and link embedding as a whole process and obtained the mapping algorithm by relaxing the constraints. Taking the resource weight of nodes as heuristic information, Cheng et al. [32] proposed a VNE algorithm based on NodeRank, which preferentially allocated physical nodes with sufficient resources. In our previous work [33], based on a genetic algorithm to map the nodes and links, the node mapping was separated from the link mapping process. However, such traditional search algorithms utilize a heuristic search through artificial subjective embedding, which bring serious interference error to the prediction process to a certain extent [18]. In addition, AI technology can adaptively fit the original characteristics of the data, bringing promising developments to tasks such as prediction and classification. To dynamically adapt to the underlying structure of the physical network, Yao et al. [34] proposed an algorithm combining reinforcement learning to assist continuous decision (CDRL) to dynamically update the embedded decision.

Considering the advantages of the DRL paradigm for interacting with the environment, some of our previous works [10,23] utilized convolutional neural networks as the agent to perform VNE (Conv-VNE). They captured the local neighborhood relationship through the movement of the convolution kernel but ignored the graph structure relationship of the nodes in the physical network. Moreover, this discrete convolution had difficulties maintaining the translation invariance in the non-Euclidean structure of the SAGIN [24]. Therefore, this paper uses graph convolution [35] to fully capture the graph structure information of the nodes in a SAGIN and update its characteristics by fusing the neighborhood information. It makes full use of the rich node information in the network and obtains more accurate embedding probabilities, further improving the embedding success rate of virtual networks over previous work.

2.3. Security-Aware VNE Algorithm

There are few studies on the security of VNE, which have gradually attracted people's attention in recent years. Aiming at the security vulnerabilities in the embedding process, Lin et al. [36] manually formulated security constraints and solved the problem in a physically isolated manner. Zhang et al. [37] solved the security problem of the embedding process by adding a virtual layer to the physical network and evaluating the importance of physical nodes through information entropy by the Technique for Order Preference by

Similarity to an Ideal Solution (TOPSIS) method. However, the above methods lacked a sufficient consideration of node security, did not take security attributes as important information of the embedding process, and only embedded them through subjectively formulated conditions, which limited the full consideration of security issues. Therefore, in this work, we carefully impose security-level constraints on all network nodes and utilize them as important training information in the embedding process, so that users can only access resources that meet the constraints. Compared with previous VNE works, this work further improves the security of resource allocation.

3. Problem Modeling and Definition

In this section, the modeling process of a SAGIN and the definition of the VNE problem are given in detail.

3.1. Network Modeling

Table 1 summarizes the notations used in the VNE-based SAGIN's RO problem. Specifically, VNE comes from the user's request for resource requirements to the physical network of a SAGIN, i.e., the virtual network request (VNR). The physical network is modeled as a weighted undirected graph $\boldsymbol{G}^S = (\boldsymbol{N}^S, \boldsymbol{E}^S, \boldsymbol{R}^S, \boldsymbol{D}^S, \boldsymbol{L}^S, \boldsymbol{C}^S)$, mainly composed of six parts: physical node $\boldsymbol{N}^S$, physical link $\boldsymbol{E}^S$, required resource $\boldsymbol{R}^S$ (mainly including computing resources $\boldsymbol{CPU}^S$ and bandwidth $\boldsymbol{BW}^S$), the links' delay constraints $\boldsymbol{D}^S$, the nodes' location of each domain $\boldsymbol{L}^S$, and the node security level of each domain $\boldsymbol{C}^S$. Moreover, in the SAGIN, the elements in the three-domain networks have similar characteristics. In VNs, the primary research entity is the virtual network, which is modeled as a weighted undirected graph $\boldsymbol{G}^V = (\boldsymbol{N}^V, \boldsymbol{E}^V, \boldsymbol{R}^V, \boldsymbol{D}^V, \boldsymbol{C}^V)$, mainly composed of five parts: virtual node $\boldsymbol{N}^V$, virtual link $\boldsymbol{E}^V$, required resource $\boldsymbol{R}^S = \{\boldsymbol{CPU}^V, \boldsymbol{BW}^V\}$, the links' delay constraints $\boldsymbol{D}^V$, and the nodes' security $\boldsymbol{C}^S$. In addition, we use lowercase bold letters to indicate specific elements. For example, $\boldsymbol{n}_i^V$ represents the ith node in VN, and $\boldsymbol{e}_k^V$ represents the kth link in VN. If the start node of $\boldsymbol{e}_k^V$ is $\boldsymbol{n}_i^V$ and the end node is $\boldsymbol{n}_j^V$, it can also be expressed as $\boldsymbol{n}_i^V - \boldsymbol{n}_j^V$.

Table 1. The notations of the SAGIN and VNs.

Notation		Definition
	$\boldsymbol{G}^S$	SAGIN
$\boldsymbol{N}^S$	$\boldsymbol{N}_S^S$	Nodes of the space network
	$\boldsymbol{N}_A^S$	Nodes of the aerial network
	$\boldsymbol{N}_G^S$	Nodes of the ground network
$\boldsymbol{E}^S$	$\boldsymbol{E}_S^S$	Links of the space network
	$\boldsymbol{E}_A^S$	Links of the aerial network
	$\boldsymbol{E}_G^S$	Links of the ground network
	$\boldsymbol{E}_{S-A}^S$	Interlinks of the space and aerial networks
	$\boldsymbol{E}_{S-G}^S$	Interlinks of the space and ground networks
	$\boldsymbol{E}_{A-G}^S$	Interlinks of the aerial and ground networks
$\boldsymbol{R}^S$	$\boldsymbol{CPU}^S(\boldsymbol{N}^S)$	Computing resources of the SAGIN nodes
	$\boldsymbol{BW}^S(\boldsymbol{E}^S)$	Bandwidth resources of the SAGIN links
$\boldsymbol{D}^S$	$\boldsymbol{D}^S(\boldsymbol{E}^S)$	Delay constraint of the SAGIN links
$\boldsymbol{L}^S$	$\boldsymbol{L}^S(\boldsymbol{N}^S)$	Location of the SAGIN nodes
$\boldsymbol{C}^S$	$\boldsymbol{C}^S(\boldsymbol{N}^S)$	Security level of the SAGIN nodes
	$\boldsymbol{G}^V$	VNs
	$\boldsymbol{N}^V$	Virtual nodes of the VNs
	$\boldsymbol{E}^V$	Virtual links of the VNs

Table 1. *Cont.*

	Notation	Definition
$\boldsymbol{R}^V$	$\boldsymbol{CPU}^V(\boldsymbol{N}^V)$ $\boldsymbol{BW}^V(\boldsymbol{E}^V)$	Computing resources of the virtual nodes Bandwidth resources of the virtual links
$\boldsymbol{D}^V$	$\boldsymbol{D}^V(\boldsymbol{E}^V)$	Delay constraint of the virtual links
$\boldsymbol{C}^V$	$\boldsymbol{C}^V(\boldsymbol{N}^V)$	Security level of the virtual nodes

3.2. Problem Definition

An example of SAGIN multidomain RO based on VNE is shown in Figure 2. The virtual nodes are linked by virtual links, and the elements in the VNs have similar resource attributes as the SAGIN element. Therefore, how to efficiently allocate physical resources in the SAGIN to different VNEs according to different VNRs to embed different VNs is the considered problem of VNE in the SAGIN, i.e., resource orchestration. Correspondingly, the VNE process is defined as:

$$\boldsymbol{G}^{S'} \rightarrow \boldsymbol{G}^{V_i}, 1 \leq i \leq |\text{VNR}|, \tag{1}$$

where $\boldsymbol{G}^{S'}$ represents a subset of SAGIN. $\boldsymbol{G}^{V_i}$ represents the virtual network generated by the ith VNR request. |VNR| indicates the number of virtual network requests.

At the same time, it has also been proven to be NP-hard [38]. It should be noted that multiple VNs generated by multiple VNEs can coexist in the same SAGIN, i.e., share physical resources, which come from frequent VNRs. Based on the existing physical resources in a SAGIN, how to allocate the required resource $\boldsymbol{R}^S$ for different VNRs under the premise of the least consumption of $\boldsymbol{R}^V$ is the foothold of our research.

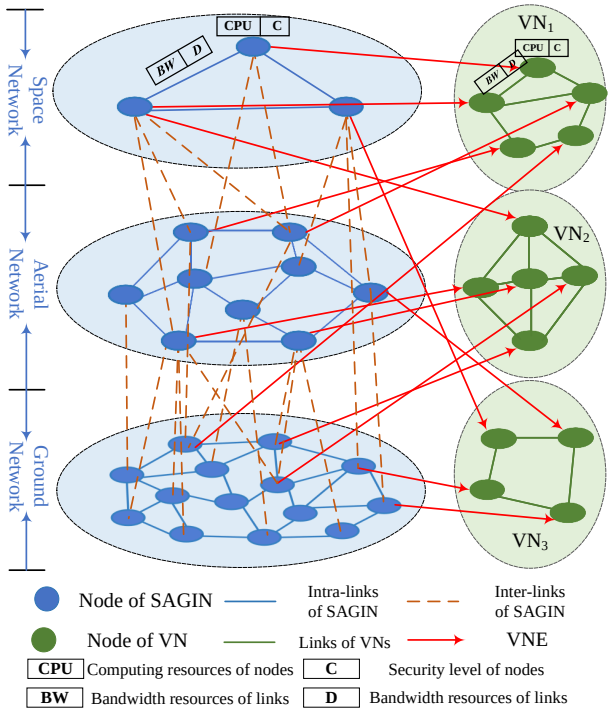


Figure 2. The VNE schematic diagram of a SAGIN. In the SAGIN, each node has ***CPU*** resources and security level ***C***, and each link has ***BW*** resources and ***D*** constraints. A VN is derived from the resource mapping strategies of the SAGIN based on VNR under the premise of satisfying relevant constraints. The ***CPU*** and ***C*** on the nodes of VNs represent the required computing resources and security level, respectively, and the ***BW*** and ***D*** on the links indicate the required bandwidth resources and the maximum delay limit, respectively.

4. VNE Constraints and Related Indicators

The embedding constraints and evaluation indicators of VNE are presented in this section.

4.1. Embedded Constraints

In the process of VNE, the corresponding resources are mapped from the SAGIN to VNs according to each VNR. In this process, the following constraints must be met:

(1) ***Nodes' available resources in the SAGIN***: For the node $\boldsymbol{n}_j^S$ in the SAGIN, its ***CPU*** resources should be enough to be allocated to all virtual nodes that want to embed it. Specifically, it is expressed as:

$$\boldsymbol{CPU}^S\left(\boldsymbol{n}_j^S\right) - \sum_{i=1}^{|\,\mathrm{VNR}\,|} \sum_{\forall\left(\boldsymbol{n}_j^S \to \boldsymbol{n}_k^{V_i}\right)} \boldsymbol{CPU}^{V_i}\left(\boldsymbol{n}_k^{V_i}\right) \geq 0, where\ 1 \leq k \leq |\boldsymbol{N}^{V_i}|, \tag{2}$$

where $|\boldsymbol{N}^{V_i}|$ represents the number of virtual nodes in $\boldsymbol{N}^{V_i}$.

(2) ***Links' available resources in SAGIN***: For the link $\boldsymbol{e}_k^S$ in the SAGIN, its ***BW*** resources should be enough to be allocated to all virtual links that want to embed it. Specifically, it is expressed as:

$$\boldsymbol{BW}^S\left(\boldsymbol{e}_k^S\right) - \sum_{i=1}^{|\,\mathrm{VNR}\,|} \sum_{\forall\left(\boldsymbol{e}_k^S \to \boldsymbol{e}_l^{V_i}\right)} \boldsymbol{BW}^{V_i}\left(\boldsymbol{e}_l^{V_i}\right) \geq 0, where\ 1 \leq l \leq |\boldsymbol{E}^{V_i}|, \tag{3}$$

where $|\boldsymbol{E}^{V_i}|$ represents the number of virtual links in $\boldsymbol{E}^{V_i}$.

(3) ***Nodes' security level constraints in the SAGIN***: For the node $\boldsymbol{n}_j^S$ in the SAGIN, the security level ***C*** attribute restricts the mapping of nodes. It requires that the virtual node $\boldsymbol{n}_k^{V_i}$ be embedded in the physical node $\boldsymbol{n}_j^S$ with a higher security level. In this way, security requirements are imposed on the node, and the security of resource scheduling is ensured. The specific implementation is through the following equation:

$$\boldsymbol{C}^S\left(\boldsymbol{n}_j^S\right) \geq max\left\{\boldsymbol{C}^{V_i}\left(\boldsymbol{n}_k^{V_i}\right) \middle| \boldsymbol{n}_j^S \to \boldsymbol{n}_k^{V_i}, 1 \leq i \leq |\,\mathrm{VNR}\,|, 1 \leq k \leq |\boldsymbol{N}^{V_i}|\right\}. \tag{4}$$

(4) ***Paths' available resources in the SAGIN***: For any loop-free path $\boldsymbol{p}^S$ in the SAGIN, the available ***BW*** resource is the minimum ***BW*** of all links in that path, which is specifically expressed as:

$$\boldsymbol{BW}^S\left(\boldsymbol{p}^S\right) = min_{\boldsymbol{e}_k^S \in \boldsymbol{p}^S} \boldsymbol{BW}\left(\boldsymbol{e}_k^S\right). \tag{5}$$

(5) ***Links' delay limits in the SAGIN***: Links in different domains of the SAGIN have different delay characteristics, and the delay of interlinks is generally greater than that of intralinks. For the link $\boldsymbol{e}_k^S$ in the SAGIN, its ***D*** should be less than that of all virtual links that want to embed it. Specifically, it is expressed as:

$$\boldsymbol{D}^S\left(\boldsymbol{e}_k^S\right) \leq min\left\{\boldsymbol{D}^{V_i}\left(\boldsymbol{e}_l^{V_i}\right) \middle| \boldsymbol{e}_k^S \to \boldsymbol{e}_l^{V_i}, 1 \leq i \leq |\,\mathrm{VNR}\,|, 1 \leq l \leq |\boldsymbol{E}^{V_i}|\right\}. \tag{6}$$

4.2. Resource Consumption Indicators

As discussed above, the main purpose of VNE in the SAGIN is to respond to as many VNRs as possible and allocate physical resources reasonably and efficiently. That is, under the premise of the least physical resource loss (loss cost), it should respond to as many VNRs and generate as much revenue as possible. Therefore, we can utilize some performance indicators to measure the resource consumption of the current VNR successfully mapped to the VN, and the following are specifically used:

(1) ***Cost***: This indicator indicates the amount of SAGIN resources consumed to respond to the VNR and successfully map the VN. It is not only related to the resources of ***CPU*** and ***BW*** requested in the VNR but also related to factors such as the number of link hops. For example, the longer the mapped links, the higher the cost. Therefore, it can be expressed as:

$$\boldsymbol{Cost}\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right) = \alpha \sum_{\boldsymbol{n}^V \in \boldsymbol{N}^V} \boldsymbol{CPU}^V\left(\boldsymbol{n}^V\right) + \beta \sum_{\boldsymbol{e}^V \in \boldsymbol{E}^V} hops(\boldsymbol{e}^V) \times \boldsymbol{BW}^V\left(\boldsymbol{e}^V\right), \tag{7}$$

where α and β indicate ***CPU*** and ***BW***, respectively, accounting for the important proportions. We assumed that they were equally important in this work, so we set $\alpha = 1$ and $\beta = 1$. *hops* denotes the number of hops of the links.

(2) ***Revenue***: This indicator indicates the quantity of resources successfully mapped to the VN in response to the VNR, i.e., the quantity of resources requested in the VNR. It is expressed as:

$$\boldsymbol{Revenue}\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right) = \alpha \sum_{\boldsymbol{n}^V \in \boldsymbol{N}^V} \boldsymbol{CPU}^V\left(\boldsymbol{n}^V\right) + \beta \sum_{\boldsymbol{e}^V \in \boldsymbol{E}^V} \boldsymbol{BW}^V\left(\boldsymbol{e}^V\right). \tag{8}$$

4.3. Evaluation Indicators

To comprehensively evaluate the performance of the proposed VNE-based multi-domain RO algorithm in the SAGIN, we employed the following indicators for evaluation:

(1) ***Long-term average revenue***: It is mainly expressed as the integral of long-term revenue averaged over time. In this work, according to the enlightenment of the integral definition, we discretized it into a summation form:

$$\boldsymbol{L\_R} = \lim_{T \to +\infty} \frac{\int_{t=0}^{T} \boldsymbol{Revenue}_t\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right)}{T} = \lim_{T \to +\infty} \frac{\sum_{t=0}^{T} \boldsymbol{Revenue}_t\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right)}{T}. \tag{9}$$

(2) ***VNR acceptance ratio***: It is mainly expressed as the ratio of the number of successfully mapped VNRs to the number of all VNRs:

$$\boldsymbol{ACC\_R} = \lim_{T \to +\infty} \frac{\int_{t=0}^{T} \boldsymbol{G}_{acc_t}^V}{\int_{t=0}^{T} \boldsymbol{G}_t^V} = \lim_{T \to +\infty} \frac{\sum_{t=0}^{T} \boldsymbol{G}_{acc_t}^V}{\sum_{t=0}^{T} \boldsymbol{G}_t^V}. \tag{10}$$

(3) ***Long-term revenue–cost ratio***: It is mainly expressed as the ratio of long-term revenue to long-term cost:

$$\boldsymbol{L\_R/L\_C} = \lim_{T \to +\infty} \frac{\int_{t=0}^{T} \boldsymbol{Revenue}_t\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right)}{\int_{t=0}^{T} \boldsymbol{Cost}_t\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right)} = \lim_{T \to +\infty} \frac{\sum_{t=0}^{T} \boldsymbol{Revenue}_t\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right)}{\sum_{t=0}^{T} \boldsymbol{Cost}_t\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right)}. \tag{11}$$

It should be noted that the long-term average revenue implies the actual quantity of resources requested by the VNR. Similar to Equation (9), the long-run average cost is defined as the integral of the long-run cost over time, as shown in Equation (12). It implies the total quantity of resources consumed by the SAGIN to allocate resources in the whole process of VNE. Therefore, the long-term average revenue–cost ratio is equivalent to the ***resource utilization*** of the network, which is an intuitive and meaningful indicator.

$$\boldsymbol{L\_C} = \lim_{T \to +\infty} \frac{\int_{t=0}^{T} \boldsymbol{Cost}_t\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right)}{T} = \lim_{T \to +\infty} \frac{\sum_{t=0}^{T} \boldsymbol{Cost}_t\left(\boldsymbol{G}^S \to \boldsymbol{G}^V\right)}{T}. \tag{12}$$

Theorem 1. *The improvement of the long-term average revenue* ***L_R*** *and VNR acceptance ratio* ***ACC_R*** *promotes the improvement of the SAGIN's resource utilization* ***L_R/L_C*** *and then reaches a maximum value.*

Proof of Theorem 1. According to Equations (7) and (9), $\boldsymbol{L\_R}$ is further expressed as:

$$\boldsymbol{L\_R} = \lim_{T \to +\infty} \frac{\sum_{\boldsymbol{n}^V \in \boldsymbol{N}^V} \boldsymbol{CPU}^V(\boldsymbol{n}^V) + \sum_{\boldsymbol{e}^V \in \boldsymbol{E}^V} \boldsymbol{BW}^V(\boldsymbol{e}^V)}{T}, \tag{13}$$

According to Equations (13) and (10), during a certain period, the T and the number of VNRs $\boldsymbol{G}_t^V$ are constant. Therefore, it can be inferred that the improvement of $\boldsymbol{L\_R}$ and $\boldsymbol{ACC\_R}$ comes from the better utilization of the SAGIN's node and link resources, that is, the increase in the number of successfully addressed VNRs. In turn, it means that the resource fragmentation of the SAGIN during the RO process is significantly reduced. For example, for a VNR_i requesting 10 CPU resources, a physical node with 20 CPU resources generates more fragmented resources than a physical node with 40 CPU resources. The more fragmented resources, the more physical resources are wasted, which leads to a low $\boldsymbol{L\_R}$ and $\boldsymbol{ACC\_R}$.

Therefore, after the above inference, the increase of the values of $\boldsymbol{L\_R}$ and $\boldsymbol{ACC\_R}$ reflect the reduction of the resource fragmentation phenomenon of the SAGIN, which promotes the improvement of resource utilization. Finally, all three reach the optimal value. □

5. The Proposed Algorithm

In this section, we introduce in detail how the proposed novel VNE algorithm arranges resources for the SAGIN. The first step that needs to be done is data preprocessing, which extracts features from the SAGIN to construct a feature matrix. Afterward, the embedding process is carried out, which mainly includes node embedding and link embedding. Based on the excellent decision-making properties of DRL, we propose an orchestration network to calculate the embedding probability of the nodes. Furthermore, we utilize a breadth-first search (BFS) to obtain the embedded links.

5.1. Data Preprocessing

The purpose of AI-driven algorithms is to provide intelligent decision-making in a real environment, which requires batch data to undergo many epochs of training to extract potentially more accurate and rich information. Therefore, we need to extract the underlying features of the SAGIN in each epoch. In addition, the difference between different periods reflects the time-varying characteristics of the SAGIN.

In this work, we extracted five types of network attributes for each node of each domain in the SAGIN to construct a feature matrix, which served as the state input of the subsequent DRL-based orchestration network. These attributes were: $\boldsymbol{CPU}^S$, $\boldsymbol{C}^S$, $\boldsymbol{SUM\_BW}^S$, $\boldsymbol{SUM\_D}^S$, $\boldsymbol{A\_Dis}^S$, the first two of which were introduced before, and the other three are described as follows:

- $\boldsymbol{SUM\_BW}^S$ represents the sum of the bandwidth of the links between the nodes connected to the current node of interest in all domains of the SAGIN. It should be noted that these links not only include interlinks but also intralinks. In this way, we can take into account the multidomain characteristics of the SAGIN at the same time, not only paying attention to the local characteristics within the domain but also the global characteristics between the domains. This can well describe the comprehensive information of the SAGIN. Let $\boldsymbol{e}_{ij}^S$ denote the link connected to $\boldsymbol{n}_i^S$, then this attribute is specifically expressed as:

$$\boldsymbol{SUM\_BW}^S\left(\boldsymbol{n}_i^S\right) = \sum_{\forall \boldsymbol{e}_{ij}^S \in \boldsymbol{E}^S} \boldsymbol{BW}^S\left(\boldsymbol{e}_{ij}^S\right). \tag{14}$$

- $\boldsymbol{SUM\_D}^S$ represents the sum of the delay constraint of the links between nodes connected to the current node of interest in all domains of the SAGIN. Similarly, it can be expressed as:

$$\boldsymbol{SUM\_D}^S\left(\boldsymbol{n}_i^S\right) = \sum_{\forall \boldsymbol{e}_{ij}^S \in \boldsymbol{E}^S} \boldsymbol{D}^S\left(\boldsymbol{e}_{ij}^S\right). \tag{15}$$

  It is worth noting that the larger its value, the more serious the delay phenomenon around the current node.
- $\boldsymbol{A\_Dis}^S$ represents the average distance to nonembedded nodes, which is mainly determined by the distance between all connected points and the number of hops of the links, and is specifically expressed as:

$$\boldsymbol{A\_Dis}^S\left(\boldsymbol{n}_j^S\right) = \frac{\sum_{\forall \boldsymbol{e}_{ij}^S \in \boldsymbol{E}^S} \|\boldsymbol{L}^S\left(\boldsymbol{n}_i^S\right) - \boldsymbol{L}^S\left(\boldsymbol{n}_j^S\right)\|_2}{hops\left(\boldsymbol{e}_{ij}^S\right) + 1}, \tag{16}$$

  where $\boldsymbol{n}_j^s$ indicates the nonembedded node and $\|$ indicates the Euclidean distance (The Euclidean distance between two points $\boldsymbol{x} = (x_1, x_2, x_3)$ and $\boldsymbol{y} = (y_1, y_2, y_3)$ in a three-dimensional space is $E_{dis}(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2 + (x_3 - y_3)^2}$.).

When the node $\boldsymbol{n}^V$ of VNs is embedded in the physical node $\boldsymbol{n}^S$ in the SAGIN, the larger $\boldsymbol{SUM\_BW}^S$ is, the more bandwidth can be provided. The larger $\boldsymbol{SUM\_D}^S$ is, the more serious the delay effect. The larger $\boldsymbol{A\_Dis}^S$ is, the more resources are consumed in the embedding process and the more serious the delay effect is.

To speed up the solution speed of the subsequent depth model's gradient backpropagation and the model's convergence speed, the extracted features need to be normalized:

$$X' = Normalization(X). \tag{17}$$

Therefore, we can obtain the normalized attribute features $\boldsymbol{CPU}'^S$, $\boldsymbol{C}'^S$, $\boldsymbol{SUM\_BW}'^S$, $\boldsymbol{SUM\_D}'^S$, $\boldsymbol{A\_Dis}'^S$. Combining these attributes, the obtained feature matrix $\boldsymbol{A}^S$ of the preprocessed network attributes is:

$$\begin{aligned} \boldsymbol{A}^S &= \left\{\boldsymbol{CPU}'^S, \boldsymbol{C}'^S, \boldsymbol{SUM\_BW}'^S, \boldsymbol{SUM\_D}'^S, \boldsymbol{A\_Dis}'^S\right\} \\ &= \begin{bmatrix} \boldsymbol{CPU}'^S\left(\boldsymbol{n}_1^S\right) & \cdots & \boldsymbol{CPU}'^S\left(\boldsymbol{n}_n^S\right) \\ \boldsymbol{C}'^S\left(\boldsymbol{n}_1^S\right) & \cdots & \boldsymbol{C}'^S\left(\boldsymbol{n}_n^S\right) \\ \boldsymbol{SUM\_BW}'^S\left(\boldsymbol{n}_1^S\right) & \cdots & \boldsymbol{SUM\_BW}'^S\left(\boldsymbol{n}_n^S\right) \\ \boldsymbol{SUM\_D}'^S\left(\boldsymbol{n}_1^S\right) & \cdots & \boldsymbol{SUM\_D}'^S\left(\boldsymbol{n}_n^S\right) \\ \boldsymbol{A\_Dis}'^S\left(\boldsymbol{n}_1^S\right) & \cdots & \boldsymbol{A\_Dis}'^S\left(\boldsymbol{n}_n^S\right) \end{bmatrix}, \end{aligned} \tag{18}$$

where we assume that there are n nodes in the SAGIN. $\boldsymbol{A}^S$ is used as the input of the node embedding model, whose columns are the network attributes of the nodes. In addition, we extract an $\boldsymbol{A}^S$ in each iteration epoch to capture the time-varying characteristics of the SAGIN through its dynamic variation.

5.2. Node Embedding

Traditional VNE works employ heuristic search algorithms, which lead to unsatisfactory embedding effects, and some existing works ignore the graph structure relationship of nodes in the physical network. Therefore, we utilized the graph convolution to fully capture the graph structure information of the nodes in the SAGIN and update its own characteristics by fusing the neighborhood information, which makes full use of the rich node information in the network and obtains more accurate embedding probabilities.

In this work, we designed a DRL-based orchestration network composed of a three-layer architecture, including graph convolution network (GCN) layers, a probability layer, and a filter layer as shown in Figure 3, which is used to calculate the embedding probability

of nodes. Among them, the GCN layers can well capture the spatial structure information of the node and fully fit the characteristics of the node by fusing its information and that of its adjacent points. The specific operation process is expressed as:

$$\boldsymbol{G}^S = \sigma\left(\tilde{\boldsymbol{J}}^{-\frac{1}{2}}\tilde{\boldsymbol{H}}\tilde{\boldsymbol{J}}^{-\frac{1}{2}}\boldsymbol{A}^S\boldsymbol{W}^S\right), \tag{19}$$

where $\boldsymbol{G}^S$ is the output of the GCN layer, σ is the ReLU activation function, and $\boldsymbol{W}^S$ is the weight matrix. $\tilde{\boldsymbol{H}} = \boldsymbol{H} + \boldsymbol{I}$, where $\boldsymbol{I}$ is the identity matrix, $\boldsymbol{H}$ is the adjacency matrix of nodes, and its elements are expressed as:

$$\boldsymbol{H}_{ij} = \begin{cases} 1, & \boldsymbol{n}_i^S - \boldsymbol{n}_j^S \in \boldsymbol{E}^S, \\ 0, & \boldsymbol{n}_i^S - \boldsymbol{n}_j^S \textit{ is not exist}. \end{cases} \tag{20}$$

Therefore, $\boldsymbol{H}$ is expressed as:

$$\boldsymbol{H} = \begin{bmatrix} \boldsymbol{H}_{11} & \boldsymbol{H}_{12} & \dots & \boldsymbol{H}_{1n} \\ \vdots & \vdots & \ddots & \vdots \\ \boldsymbol{H}_{n1} & \boldsymbol{H}_{n2} & \dots & \boldsymbol{H}_{nn} \end{bmatrix}. \tag{21}$$

Accordingly, $\tilde{\boldsymbol{J}}$ is the degree matrix, and its elements can be expressed as:

$$\tilde{\boldsymbol{J}}_{ii} = \sum_j \tilde{\boldsymbol{H}}_{ij}. \tag{22}$$

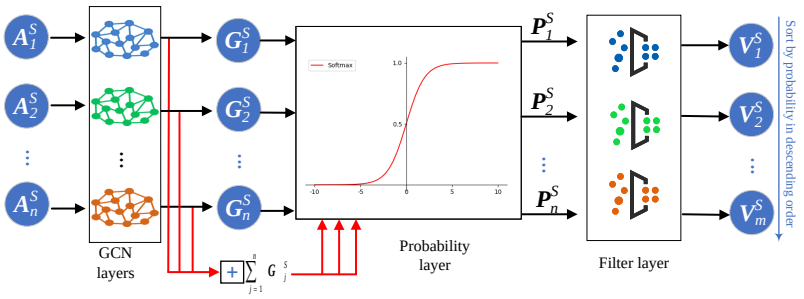


Figure 3. The proposed DRL-based orchestration network. It is composed of a three-layer architecture: GCN layers; probability layer; and filter layer

Theorem 2. *Traditional discrete convolution cannot maintain translation invariance on data with a non-Euclidean structure. Combined with the theoretical basis of spectral graphs (Spectral graph theory is the product of the combination of graph theory and linear algebra. It studies the properties of graphs by analyzing the eigenvalues and eigenvectors of certain matrices of the graph. The Laplace matrix and Fourier transformation are the core and basic concepts in spectral graph theory.), graph convolutional networks can effectively extract spatial features from topological data structures with non-Euclidean structures.*

Proof of Theorem 2. Based on the above notation, the degree matrix of the graph is the adjacency matrix $\boldsymbol{H}$, and the degree matrix is $\boldsymbol{J}$. Then, its Laplace matrix is defined as:

$$\boldsymbol{L} = \boldsymbol{J} - \boldsymbol{H}. \tag{23}$$

Its normalized Laplacian matrix is defined as:

$$\boldsymbol{\mathcal{L}} = \boldsymbol{J}^{-\frac{1}{2}}\boldsymbol{L}\boldsymbol{J}^{-\frac{1}{2}} = \boldsymbol{I} - \boldsymbol{J}^{-\frac{1}{2}}\boldsymbol{H}\boldsymbol{J}^{-\frac{1}{2}} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T, \tag{24}$$

where $\boldsymbol{U}$ is the Fourier orthonormal basis obtained by eigendecomposition (Suppose $\boldsymbol{X}$ has n eigenvectors $\boldsymbol{V} = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_n\}$, corresponding to eigenvalues $\boldsymbol{\Lambda} = \{\lambda_1, \cdots, \lambda_n\}^T$. Then, the eigendecomposition of $\boldsymbol{X}$ is recorded as: $\boldsymbol{X} = \boldsymbol{V}diag(\boldsymbol{\Lambda})\boldsymbol{V}^{-1}$.) of $\boldsymbol{\mathcal{L}}$, and $\boldsymbol{\Lambda}$ is the corresponding eigenvalue matrix (diagonal matrix).

The convolved signal in the SAGIN is $\boldsymbol{a} \in \mathcal{R}^n$, which is the feature vector extracted by a node in $\boldsymbol{A}$. For $\boldsymbol{a}$, its Fourier transform and inverse transform are:

$$\tilde{\boldsymbol{a}} = \boldsymbol{U}^T \boldsymbol{a}, \tag{25}$$

$$\boldsymbol{a} = \boldsymbol{U}^T \tilde{\boldsymbol{a}}. \tag{26}$$

To transform the graph signal g_θ into the spectral domain, it is combined with a filter to perform a product operation in the Fourier domain to complete the graph convolution operation [39] as follows:

$$g_\theta * \boldsymbol{a} = g_\theta(\boldsymbol{L})\boldsymbol{a} = \boldsymbol{U} g_\theta \boldsymbol{U}^T \boldsymbol{a}, \tag{27}$$

where g_θ can be thought of as a function of the $\boldsymbol{\mathcal{L}}$ eigenvectors, i.e., $g_\theta(\boldsymbol{\Lambda})$:

$$g_\theta(\boldsymbol{\Lambda}) = \begin{bmatrix} \theta_1 & & \\ & \ddots & \\ & & \theta_n \end{bmatrix}, \tag{28}$$

where θ_i is an arbitrary parameter similar to the weights in a neural network.

However, for the spectral convolution of graphs, there exist the following problems: (1) the computational complexity of matrix multiplication using $\boldsymbol{U}$ is $O(n^2)$; (2) computing the eigendecomposition of the Laplacian matrix of a large graph requires a huge amount of computation. Especially for large-scale graph data in a SAGIN, it is computationally expensive. Therefore, an approximation to $g_\theta(\boldsymbol{\Lambda})$ is obtained using a Kth-order truncation of the Chebyshev polynomial $T_k(\boldsymbol{x})$ [40]:

$$g'_\theta(\boldsymbol{\Lambda}) \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{\boldsymbol{\Lambda}}), \tag{29}$$

where $\tilde{\boldsymbol{\Lambda}} = \frac{2}{\lambda_{max}}\boldsymbol{\Lambda} - \boldsymbol{I}$ is the eigenvector matrix scaled according to the largest eigenvalue (spectral radius) of $\boldsymbol{\mathcal{L}}$. $\theta' \in \mathcal{R}^K$ is the Chebyshev vector. Chebyshev polynomials are defined using the following recursion:

$$T_k(x) = \begin{cases} 1, & k = 0, \\ x, & k = 1, \\ 2xT_{k-1}(x) - T_{k-2}(x), & k \geq 2. \end{cases} \tag{30}$$

At this point, using the approximate g'_θ to replace the original g_θ, we get:

$$g'_\theta * \boldsymbol{a} \approx \boldsymbol{U} \sum_{k=0}^{K} \theta'_k T_k(\tilde{\boldsymbol{\Lambda}}) \boldsymbol{U}^T \boldsymbol{a} = \sum_{k=0}^{K} \theta'_k \boldsymbol{U} T_k(\tilde{\boldsymbol{\Lambda}}) \boldsymbol{U}^T \boldsymbol{a}, \tag{31}$$

where $T_k(\tilde{\boldsymbol{\Lambda}})$ is a kth order polynomial of $\tilde{\boldsymbol{\Lambda}}$ with $\boldsymbol{U}\tilde{\boldsymbol{\Lambda}}^k\boldsymbol{U}^T = (\boldsymbol{U}\tilde{\boldsymbol{\Lambda}}\boldsymbol{U}^T)^k = \tilde{\boldsymbol{\mathcal{L}}}$ and $\tilde{\boldsymbol{\mathcal{L}}} = \frac{2}{\lambda_{max}}\boldsymbol{\mathcal{L}} - \boldsymbol{I}$. At this point, Equation (31) can be derived as:

$$g'_\theta * \boldsymbol{a} \approx \sum_{k=0}^{K} \theta'_k T_k(\tilde{\boldsymbol{\mathcal{L}}}) \boldsymbol{a}. \tag{32}$$

An approximate spectral graph convolution can establish Kth-order neighbor dependencies but still requires Kth-order computations on $\boldsymbol{\mathcal{L}}$. To reduce the cost, Kipf et al. [35] proposed to use $K = 1$ and $\lambda_{max} = 2$ to further simplify the calculation. Although only

first-order neighbor dependencies can be established, stacked multilayer graph convolution can still establish *K*th-order neighbor dependencies. More advantageously, it is not restricted by the Chebyshev polynomial. Then, Equation (32) can be simplified to:

$$g'_\theta * \boldsymbol{a} \approx \theta'_0 \boldsymbol{a} + \theta'_1(\boldsymbol{\mathcal{L}} - \boldsymbol{I}) = \theta'_0 \boldsymbol{a} - \theta'_1 \boldsymbol{J}^{-\frac{1}{2}} \boldsymbol{H} \boldsymbol{J}^{-\frac{1}{2}} \boldsymbol{a}. \tag{33}$$

To suppress the number of parameters and prevent overfitting, we let $\theta = \theta'_0 = -\theta'_1$, then Equation (33) can be further derived as:

$$g_\theta * \boldsymbol{a} \approx \theta(\boldsymbol{I} + \boldsymbol{J}^{-\frac{1}{2}} \boldsymbol{H} \boldsymbol{J}^{-\frac{1}{2}}) \boldsymbol{a}. \tag{34}$$

In addition, to prevent the occurrence of a gradient explosion or vanishing phenomenon due to the instability of operator $\boldsymbol{I} + \boldsymbol{J}^{-\frac{1}{2}} \boldsymbol{H} \boldsymbol{J}^{-\frac{1}{2}}$ when the number of layers is deep, we let $\tilde{\boldsymbol{H}} = \boldsymbol{H} + \boldsymbol{I}$ and $\tilde{\boldsymbol{J}}_{ii} = \sum_j \tilde{\boldsymbol{H}}_{ij}$. Then, this operator is derived as:

$$\boldsymbol{I} + \boldsymbol{J}^{-\frac{1}{2}} \boldsymbol{H} \boldsymbol{J}^{-\frac{1}{2}} = \tilde{\boldsymbol{J}}^{-\frac{1}{2}} \tilde{\boldsymbol{H}} \tilde{\boldsymbol{J}}^{-\frac{1}{2}}. \tag{35}$$

When we batch-process the data of multiple groups of nodes, let $\boldsymbol{A}^S$ be the graph signal matrix and $\boldsymbol{W}^S$ be the parameter matrix of θ; then, we obtain the convolution operation of the graph convolution layer proposed in Equation (19):

$$\boldsymbol{G}^S = \sigma\left(\tilde{\boldsymbol{J}}^{-\frac{1}{2}} \tilde{\boldsymbol{H}} \tilde{\boldsymbol{J}}^{-\frac{1}{2}} \boldsymbol{A}^S \boldsymbol{W}^S\right), \tag{36}$$

where $\boldsymbol{G}^S$ is the output of the GCN layer and σ is the ReLU activation function.

To sum up, at each layer we obtain and fuse the first-order neighbor information of each node. In the SAGIN, through multilayer graph convolution stacking, we can effectively fuse the information of *K*th-order neighbors and then comprehensively consider the local and global information of the nodes. □

The probability layer was mainly used to map the aggregated features obtained through the GCN layers to the embedding probability of the nodes in the SAGIN, which was specifically expressed as:

$$\boldsymbol{P}_i^S = \frac{\boldsymbol{G}_i^S}{\sum_{j=1}^n \boldsymbol{G}_j^S}, \tag{37}$$

where $\boldsymbol{P}_i^S$ indicates the embedding probability of node i. For each SAGIN node, after calculating the embedding probability, we used the cross-entropy loss to calculate its loss, which jointly guided the learning of the model.

$$Loss_i = -\boldsymbol{G}_i^S log(\boldsymbol{P}_i^S). \tag{38}$$

The filtering layer was mainly used to filter out the nodes that did not meet the conditions according to the relevant constraints in Section 4.1 and finally sort the nodes that met the constraints in descending order according to their embedding probability.

5.3. Link Embedding

After node embedding, we obtained the nodes in the SAGIN that satisfied both the constraints of $\boldsymbol{CPU}$ resource and the $\boldsymbol{C}$ security level and also obtained the embedding probability ranking of these nodes. In addition, according to these nodes, we adopted the BFS strategy to embed links under the condition of satisfying the $\boldsymbol{BW}$ and $\boldsymbol{D}$ constraints in Section 4.1. To be clear, the orchestration network was the agent modeled by DRL. Node embedding and link embedding were two-phase actions adopted by the agent.

To further clarify the adopted DRL algorithm, it was necessary to model it as a Markov decision process (MDP). The state was the extracted feature matrix as shown in

Equation (18). The agent was the proposed orchestration network. The actions taken were node embedding and link embedding. Therefore, the resource allocation of the SAGIN was modeled as the following MDP process:

$$MDP =< \boldsymbol{A}^S, \text{orchestration network}, \{node\ embedding, link\ embedding\}, Reward > . \quad (39)$$

In addition, the time complexity of the proposed algorithm needed to be analyzed. Let n represent the number of nodes of the SAGIN. Then, the complexity of extracting $\boldsymbol{A}^S$ during data preprocessing was $O(n)$. When embedding nodes, the time complexity through the proposed orchestration network was $O(n^2)$. In addition, $\boldsymbol{A}^S$ needed to be re-extracted when each VNR arrived, so the time complexity was $O(|VNR| \times n^2)$. Therefore, the overall complexity of the algorithm was $O(n + n^2 + |VNR| \times n^2)$.

5.4. Training and Testing

In the training process, to better guide decision-making, since $\boldsymbol{L\_R}/\boldsymbol{L\_C}$ could well reflect the resource utilization of the underlying physical network nodes and links, we used that indicator as the reward mechanism of the agent. If the current decision-making reward was large, it implied that the current decision-making could bring benefits. On the contrary, it was necessary to adjust the mapping decision. In addition, the cross-entropy loss combined with the reward mechanism was used for backpropagation to guide the embedding training process of the orchestration network. We repeated this process until all VNRs requests were allocated or the SAGIN was unable to satisfy subsequent VNRs. It should be noted that in a VNR, only all requested nodes, links, and corresponding attributes completed the mapping, and the current VNR was considered to be completed. The specific training flow is presented in Algorithm 1. The test process of the network only used the forward propagation process and did not perform gradient backpropagation and parameter update. The other processes were similar to the training process and are not repeated here.

Theorem 3. *In the orchestration network, the loss function and reward mechanism jointly guide positive learning.*

Proof of Theorem 3. In this work, the update of the gradient settings were combined with the reward mechanism as follows:

$$Loss := Loss - \mu \cdot Loss' \cdot Reward, \quad (40)$$

where μ is the learning rate of the orchestration network. In a neural network, the learning rate guides the direction and step size of the loss reduction. If μ is too large, the optimal point is missed. If μ is too small, the optimization time cost is high. Accordingly, its value selection is more critical; after verification, we set it to 0.005. $Loss'$ is the derivative of $Loss$. It should be noted that the MDP process promotes the agent to optimize towards the direction of high reward. In addition, the parameter optimization mechanism of the neural network promotes the optimization of the orchestration network in the direction of gradient descent. Therefore, in the gradient descent optimization process of the model, the reward mechanism jointly participates in the optimization and guides the model to learn in the direction of high revenue and low loss. □

Algorithm 1: The training flow

Input: G^S; G^V.
Output: The parameters of orchestration network.

Initialization: The parameters of the orchestration network; minibatch size: $bc = 100$; the number of training VNRs: $N_{train} = 1000$; the number of iteration: $t = \lceil \frac{N_{train}}{bc} \rceil$; the current iteration number: *iter*; the number of epochs: $e = 200$.
repeat
  Extract the attributes of the SAGIN to construct a feature matrix;
  for $iter = 1, 2, \cdots, t$ **do**
    Calculate the node embedding probability through the forward propagation process of the orchestration network;
    Extract the attributes of VNs in the minibatch VNRs to construct the feature matrix;
    for $i = 1, 2, \cdots, bc$ **do**
      if $\boldsymbol{n}^V$ *of* VNR_i *is embedded* **then**
        Link embedding through BFS strategy;
      else
        $\boldsymbol{VNR}_i$ embedding fails;
      end
    end
    if $\boldsymbol{n}^V$ *successfully embedded and* $\boldsymbol{e}^V$ *successfully embedded* **then**
      Calculate the embedded reward and gradient;
      Backpropagate to update network parameters;
    end
  end
until *Cycle iterates e times*;

6. Experiment

In this section, to evaluate our proposed algorithm, simulation experiments of a real environment were carried out. In addition, we conducted a comprehensive analysis of the scientific research from various angles.

6.1. Simulation Environment and Dataset Configuration

The detailed configuration information of the simulation environment is recorded in Table 2, we randomly generated 100 SAGIN nodes, which were divided into $\boldsymbol{N}_S^S$, $\boldsymbol{N}_A^S$, and $\boldsymbol{N}_G^S$ according to a 1:3:6 ratio. Moreover, according to a connection probability of 12.3%, 581 links were randomly generated, including interlinks and intralinks.

In the allocation of nodes' computing resources, the computing of the space network nodes was randomly allocated 20 TFLOPS to 40 TFLOPS, the computing of the aerial network nodes was randomly allocated 20 TFLOPS to 40 TFLOPS, and the computing of the ground network nodes was randomly allocated 50 TFLOPS to 100 TFLOPS. The security level (Sl) of all nodes was randomly assigned from zero to five. In the allocation of link bandwidth resources, all interlinks and intralinks were randomly allocated 50 Mbps to 100 Mbps. In the allocation of link delay, the intralinks in the space network was randomly allocated 20 ms to 40 ms, the intralinks in the aerial network were randomly allocated 10 ms to 30 ms, and the intralinks in the ground network were randomly allocated 1 ms to 20 ms. In addition, the delay of all interlinks was randomly assigned to 40 ms to 60 ms.

Moreover, 2000 VNRs were randomly generated for model learning, where the first 1000 were for the training set and the last 1000 were for the testing set. The specific configuration of the VNRs is recorded in Table 3. The computing of the virtual nodes was randomly assigned 1 TFLOPS to 20 TFLOPS, the security of the virtual nodes was randomly assigned to an Sl of zero to five, the bandwidth of the virtual links was randomly

assigned 1 Mbps to 20 Mbps, and the delay of the virtual links was randomly assigned 1 ms to 50 ms. It needs to be stated that the number of requested nodes in each VNR was randomly selected between 2 and 10.

Table 2. Detailed configuration information of the simulation environment.

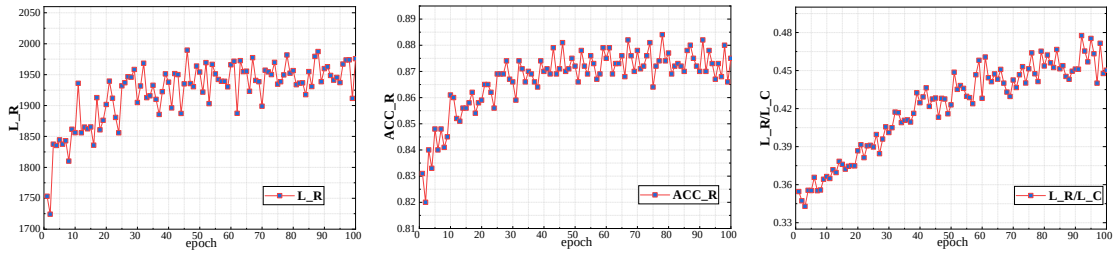
Simulation Environment	Configuration Information
Number of $\boldsymbol{N}_S^S$	10
Number of $\boldsymbol{N}_A^S$	30
Number of $\boldsymbol{N}_G^S$	60
Number of $\boldsymbol{E}^S$	581
$\boldsymbol{CPU}_S^S$	20–40 TFLOPS
$\boldsymbol{CPU}_A^S$	20–40 TFLOPS
$\boldsymbol{CPU}_G^S$	50–100 TFLOPS
$\boldsymbol{C}^S$	1–5 Sl
$\boldsymbol{BW}^S$	50–100 Mbps
$\boldsymbol{D}^S(\boldsymbol{E}_S^S)$	20–40 ms
$\boldsymbol{D}^S(\boldsymbol{E}_A^S)$	10–30 ms
$\boldsymbol{D}^S(\boldsymbol{E}_G^S)$	1–20 ms
$\boldsymbol{D}^S(\boldsymbol{E}_{S-A}^S)$	40–60 ms
$\boldsymbol{D}^S(\boldsymbol{E}_{S-G}^S)$	40–60 ms
$\boldsymbol{D}^S(\boldsymbol{E}_{A-G}^S)$	40–60 ms

Table 3. Detailed configuration information of the VNRs.

VNR	Configuration Information
Total number	2000
Number (training)	1000
Number (testing)	1000
$\boldsymbol{CPU}^V(\boldsymbol{N}^V)$	1–20 TFLOPS
$\boldsymbol{C}^V(\boldsymbol{N}^V)$	0–5 Sl
$\boldsymbol{BW}^V(\boldsymbol{E}^V)$	1–20 Mbps
$\boldsymbol{D}^V(\boldsymbol{E}^V)$	1–50 ms

6.2. Training Performance

With the advancement of the orchestration network training process, the variation process of the evaluation indicators ***L_R***, ***ACC_R***, and ***L_R/L_C*** is shown in Figure 4. We can find that with the iteration of the training epoch, it gradually fit the characteristics of the underlying network and eventually converged quickly. Therefore, the convergence of the training process of the model was effective.



(a) (b) (c)

Figure 4. The variation process of (**a**) ***L_R***, (**b**) ***ACC_R***, and (**c**) ***L_R/L_C*** in the training process of the orchestration network.

6.3. Flexible Decision-Making

We limited the delays to 1–50 ms, 1–40 ms, 1–30 ms, and 1–20 ms, respectively, and generated corresponding VNRs to verify the flexibility of the orchestration network's decision-making for changes in demand. From a theoretical point of view, as the demand delay decreases (50 ms $\rightarrow$ 20 ms), the number of SAGIN links that can meet the conditions decreases, so the number of successfully mapped VNRs also decreases accordingly. According to Equation (9), $\boldsymbol{L\_R}$ should be reduced; according to Equation (10), $\boldsymbol{ACC\_R}$ should be reduced; according to Equation (11), $\boldsymbol{L\_R/L\_C}$ is jointly determined by $\boldsymbol{L\_R}$ and $\boldsymbol{L\_C}$, so its value fluctuates little. As a result, the experimental results in Table 4 were in line with the theoretical reality, and the orchestration network had a satisfactory flexibility when dealing with different VNR decisions.

Table 4. Average performance under VNR delay demand changes.

D^V	L_R	ACC_R	L_R/L_C
1–50 ms	1937.1272 ± 0.0001	0.874	0.3752 ± 0.0001
1–40 ms	1884.2228 ± 0.0001	0.848	0.3811 ± 0.0001
1–30 ms	1725.0045 ± 0.0001	0.794	0.3673 ± 0.0001
1–20 ms	1405.9707 ± 0.0001	0.687	0.4013 ± 0.0001

6.4. Comparison with Other Related Works

The related works in Table 5 were selected for comparative experiments to verify the advantages of the proposed algorithm. The source code of all baselines was provided by the original authors, and each was run in the same experimental environment to eliminate the influence of other factors and improve rigor. In total, 1000 VNRs were utilized to verify the test performance of our proposed algorithm and other related works. The comparative experimental results are displayed in Figure 5.

Table 5. The baselines used in comparative experiments.

Baselines	Algorithm Basis	Optimization Goal	Description
NodeRank [32]	Heuristic	Improve embedding revenue	Heuristic search algorithm, embedding based on the ranking of node resource weights
RCR-VNE [41]	Heuristic	Improve embedding revenue and VNR acceptance ratio	A heuristic algorithm based on multidimensional available resource constraints.
CDRL [34]	RL	Improve resources utilization	A reinforcement-learning-based VNE algorithm
Conv-VNE [10]	RL+DL	Improve embedding revenue and VNR acceptance ratio	A VNE algorithm based on reinforcement learning and deep learning (convolutional neural network)
SGT-VNE [24]	DRL	Improve embedding revenue and reduce embedding cost	A single-domain VNE algorithm based on spectral graph theory applied to a vehicle fog computing network

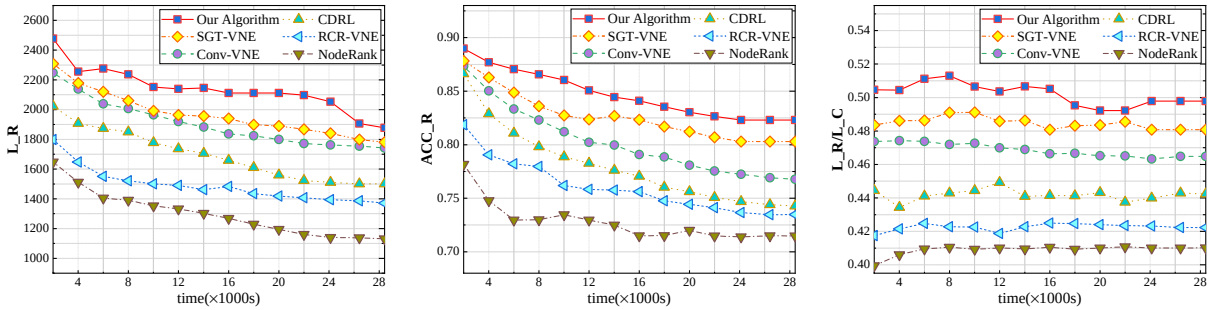


Figure 5. The comparative experimental results on (**a**) ***L_R***, (**b**) ***ACC_R***, and (**c**) ***L_R*/*L_C***.

From a theoretical point of view, as time goes by, the continuous embedding of VNRs that meet the demand leads to the continuous reduction of node and link resources in the SAGIN, so the number of subsequent VNRs embeddings is reduced, which leads to a decreasing trend of ***L_R*** and ***ACC_R***. Since ***L_R*/*L_C*** is jointly determined by ***L_R*** and ***L_C***, it causes little fluctuation. In fact, our experimental results confirmed the theory in a practical scenario.

Furthermore, NodeRank and RCR-VNE initially performed well due to their greedy allocation based on node resource weights, but they performed poorly in the long run. Based on the characteristics of the interaction between the incentive mechanism of RL and the environment, CDRL significantly improved in various indicators. Furthermore, Conv-VNE utilized convolutional layers instead of traditional Q-table learning to better focus on local information. SGT-VNE only considered the resource allocation modeling of a single physical domain but did not fully consider the cross-physical domain scenarios. However, it took into account the modeling of non-Euclidean graph structures of physical networks, so the performance was further improved. In the long run, our proposed algorithm outperformed the other baselines. This was mainly due to the combination of the DRL paradigm, which effectively interacted with the environment while combining the graph convolution to better focus on the global topology and neighborhood information of the SAGIN. To sum up, the above experiments scientifically proved the effectiveness of the RO algorithm for security-aware non-Euclidean graph-convolution VNE in SAGIN.

7. Conclusions and Outlook

In this work, to make the RO strategy of a SAGIN more flexible, intelligent, accurate, and safe, combined with AI and VNE, we proposed a non-Euclidean graph-convolution VNE algorithm. The proposed DRL-based orchestration network integrated the node itself and neighborhood information to calculate the embedding probability of nodes, which satisfactorily fit the original characteristics of physical network nodes. Moreover, we ensured resource security was guaranteed by imposing security-level constraints on the network. Under the combined action of additional constraints, resource allocation strategies were more reasonable and reliable. Finally, we conducted scientific and rigorous experiments in a simulation environment that fit a real SAGIN, which meaningfully proved the effectiveness of the proposed algorithm.

In future work plans, we expect to pay a better attention to the dynamic properties of the SAGIN to further improve the flexibility of resource allocation decisions. In particular, the high-intensity maneuverability of UAVs will be our focus. In addition, how to more effectively explore the specific characteristics of different networks in the SAGIN is also a challenge that we will focus on in the future, which will significantly improve the performance of the VNE algorithm.

Author Contributions: Conceptualization, N.C. and Y.D.; methodology, N.C.; software, N.C.; validation, N.C.; investigation, N.C.; data curation, N.C.; writing—original draft preparation, N.C.; writing—review and editing, N.C., S.S., Y.D., S.H., W.Z. and L.T.; visualization, N.C.; supervision, S.S., Y.D., S.H., W.Z. and L.T.; project administration, S.S., Y.D., S.H., W.Z. and L.T. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the Natural Science Foundation of Shandong Province under grant ZR2022LZH015, ZR2020MF006, ZR2019LZH013, and ZR2020LZH010, partially supported by the Pilot International Cooperation Project for Integrated Innovation of Science, Education and Industry of Qilu University of Technology (Shandong Academy of Sciences) under grant 2022GH007, partially supported by the Jinan Scientific Research Leader Studio Project under grant 2021GXRC091, partially supported by the One Belt One Road Innovative Talent Exchange with Foreign Experts under grant DL2022024004L, partially supported by the Zhejiang Provincial Natural Science Foundation of China under grant LZ22F020002, partially supported by the Industry-university Research Innovation Foundation of Ministry of Education of China under grant 2021FNA01001, partially supported by the Major Scientific and Technological Projects of CNPC under grant ZD2019-183-006, partially supported by the Open Foundation of State Key Laboratory of Integrated Services Networks (Xidian University) under grant ISN23-09.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to privacy.

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

References

1. Saad, W.; Bennis, M.; Chen, M. A vision of 6G wireless systems: Applications, trends, technologies, and open research problems. *IEEE Netw.* **2019**, *34*, 134–142. [CrossRef]
2. Duan, Y.; Chen, N.; Bashir, A.K.; Alshehri, M.D.; Liu, L.; Zhang, P.; Yu, K. A Web Knowledge-Driven Multimodal Retrieval Method in Computational Social Systems: Unsupervised and Robust Graph Convolutional Hashing. *IEEE Trans. Comput. Soc. Syst.* **2022**, 1–11 . [CrossRef]
3. Zhang, P.; Chen, N.; Shen, S.; Yu, S.; Wu, S.; Kumar, N. Future Quantum Communications and Networking: A Review and Vision. *IEEE Wirel. Commun.* **2022**, 1–8. [CrossRef]
4. Zhou, Z.; Feng, J.; Zhang, C.; Chang, Z.; Zhang, Y.; Huq, K.M.S. SAGECELL: Software-Defined Space-Air-Ground Integrated Moving Cells. *IEEE Commun. Mag.* **2018**, *56*, 92–99. [CrossRef]
5. Zhang, P.; Li, Y.; Kumar, N.; Chen, N.; Hsu, C.H.; Barnawi, A. Distributed Deep Reinforcement Learning Assisted Resource Allocation Algorithm for Space-Air-Ground Integrated Networks. *IEEE Trans. Netw. Serv. Manag.* **2022**, 1. [CrossRef]
6. Wang, Y.; Su, Z.; Luan, T.H.; Li, R.; Zhang, K. Federated learning with fair incentives and robust aggregation for UAV-aided crowdsensing. *IEEE Trans. Netw. Sci. Eng.* **2021**, *9*, 3179–3196. [CrossRef]
7. Ding, C.; Zhou, A.; Ma, X.; Zhang, N.; Hsu, C.H.; Wang, S. Towards Diversified IoT Services in Mobile Edge Computing. *IEEE Trans. Cloud Comput.* **2021**, 1. [CrossRef]
8. Shen, S.; Zhou, H.; Feng, S.; Huang, L.; Liu, J.; Yu, S.; Cao, Q. HSIRD: A model for characterizing dynamics of malware diffusion in heterogeneous WSNs. *J. Netw. Comput. Appl.* **2019**, *146*, 102420. [CrossRef]
9. Duan, Y.; Chen, N.; Shen, S.; Zhang, P.; Qu, Y.; Yu, S. FDSA-STG: Fully Dynamic Self-Attention Spatio-Temporal Graph Networks for Intelligent Traffic Flow Prediction. *IEEE Trans. Veh. Technol.* **2022**, *71*, 9250–9260. [CrossRef]
10. Zhang, P.; Wang, C.; Kumar, N.; Liu, L. Space-Air-Ground Integrated Multi-Domain Network Resource Orchestration Based on Virtual Network Architecture: A DRL Method. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 2798–2808. [CrossRef]
11. Sun, W.; Lian, S.; Zhang, H.; Zhang, Y. Lightweight Digital Twin and Federated Learning with Distributed Incentive in Air-Ground 6G Networks. *IEEE Trans. Netw. Sci. Eng.* **2022**, 1–13. [CrossRef]
12. Guo, Y.; Li, Q.; Li, Y.; Zhang, N.; Wang, S. Service Coordination in the Space-Air-Ground Integrated Network. *IEEE Netw.* **2021**, *35*, 168–173.
13. Sheng, J.; Cai, X.; Li, Q.; Wu, C.; Ai, B.; Wang, Y.; Kadoch, M.; Yu, P. Space-air-ground integrated network development and applications in high-speed railways: A survey. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 10066–10085 .
14. Gu, S.; Zhang, Q.; Xiang, W. Coded storage-and-computation: A new paradigm to enhancing intelligent services in space-air-ground integrated networks. *IEEE Wirel. Commun.* **2020**, *27*, 44–51.
15. Duan, Y.; Chen, N.; Zhang, P.; Kumar, N.; Chang, L.; Wen, W. MS2GAH: Multi-label semantic supervised graph attention hashing for robust cross-modal retrieval. *Pattern Recognit.* **2022**, *128*, 108676. [CrossRef]
16. Zhan, K.; Chen, N.; Santhosh Kumar, S.V.N.; Kibalya, G.; Zhang, P.; Zhang, H. Edge computing network resource allocation based on virtual network embedding. *Int. J. Commun. Syst.* **2022**, e5344. [CrossRef]
17. Liu, J.; Wang, X.; Shen, S.; Fang, Z.; Yu, S.; Yue, G.; Li, M. Intelligent Jamming Defense Using DNN Stackelberg Game in Sensor Edge Cloud. *IEEE Internet Things J.* **2022**, *9*, 4356–4370. [CrossRef]

18. Kato, N.; Fadlullah, Z.M.; Tang, F.; Mao, B.; Tani, S.; Okamura, A.; Liu, J. Optimizing space-air-ground integrated networks by artificial intelligence. *IEEE Wirel. Commun.* **2019**, *26*, 140–147. [CrossRef]
19. Liao, S.; Wu, J.; Li, J.; Bashir, A.K.; Mumtaz, S.; Jolfaei, A.; Kvedaraite, N. Cognitive Popularity Based AI Service Sharing for Software-Defined Information-Centric Networks. *IEEE Trans. Netw. Sci. Eng.* **2020**, *7*, 2126–2136. . TNSE.2020.2993457. [CrossRef]
20. Liao, H.; Zhou, Z.; Zhao, X.; Wang, Y. Learning-Based Queue-Aware Task Offloading and Resource Allocation for Space–Air–Ground-Integrated Power IoT. *IEEE Internet Things J.* **2021**, *8*, 5250–5263.
21. Fischer, A.; Botero, J.F.; Beck, M.T.; de Meer, H.; Hesselbach, X. Virtual Network Embedding: A Survey. *IEEE Commun. Surv. Tutorials* **2013**, *15*, 1888–1906. [CrossRef]
22. Zhang, W.; Wang, D.; Yu, S.; He, H.; Wang, Y. Repeatable Multi-Dimensional Virtual Network Embedding in Cloud Service Platform. *IEEE Trans. Serv. Comput.* **2021**, *15*, 3499–3512. [CrossRef]
23. Zhang, P.; Wang, C.; Jiang, C.; Benslimane, A. Security-Aware Virtual Network Embedding Algorithm Based on Reinforcement Learning. *IEEE Trans. Netw. Sci. Eng.* **2021**, *8*, 1095–1105. [CrossRef]
24. Chen, N.; Zhang, P.; Kumar, N.; Hsu, C.H.; Abualigah, L.; Zhu, H. Spectral graph theory-based virtual network embedding for vehicular fog computing: A deep reinforcement learning architecture. *Knowl.-Based Syst.* **2022**, *257*, 109931. [CrossRef]
25. Shafi, M.; Jha, R.K. Artificial Dust Based Attack Modelling: A Threat to the Security of Next Generation WCN. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 4001–4016. [CrossRef]
26. Zhang, Y.; Li, C.; Chen, N.; Zhang, P. Intelligent Requests Orchestration for Microservice Management Based on Blockchain in Software Defined Networking: A Security Guarantee. In Proceedings of the 2022 IEEE International Conference on Communications Workshops (ICC Workshops), Seoul, Republic of Korea, 16–20 May 2022; pp. 254–259. [CrossRef]
27. Song, H.; Gu, B.; Son, K.; Choi, W. Joint Optimization of Edge Computing Server Deployment and User Offloading Associations in Wireless Edge Network via a Genetic Algorithm. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 2535–2548. [CrossRef]
28. Liu, J.; Shi, Y.; Fadlullah, Z.M.; Kato, N. Space-Air-Ground Integrated Network: A Survey. *IEEE Commun. Surv. Tutorials* **2018**, *20*, 2714–2741. [CrossRef]
29. Yu, S.; Gong, X.; Shi, Q.; Wang, X.; Chen, X. EC-SAGINs: Edge Computing-enhanced Space-Air-Ground Integrated Networks for Internet of Vehicles. *IEEE Internet Things J.* **2021**, *9*, 5742–5754. [CrossRef]
30. Xiang, W.; Huang, T.; Wan, W. Machine learning based optimization for vehicle-to-infrastructure communications. *Future Gener. Comput. Syst.* **2019**, *94*, 488–495. [CrossRef]
31. Chowdhury, N.M.K.; Rahman, M.R.; Boutaba, R. Virtual network embedding with coordinated node and link mapping. In Proceedings of the IEEE INFOCOM 2009, Rio de Janeiro, Brazil, 19–25 April 2009; IEEE: Piscataway, NJ, USA, 2009; pp. 783–791.
32. Cheng, X.; Su, S.; Zhang, Z.; Wang, H.; Yang, F.; Luo, Y.; Wang, J. Virtual network embedding through topology-aware node ranking. *ACM SIGCOMM Comput. Commun. Rev.* **2011**, *41*, 38–47. [CrossRef]
33. Zhang, P.; Yao, H.; Li, M.; Liu, Y. Virtual network embedding based on modified genetic algorithm. *Peer-Netw. Appl.* **2019**, *12*, 481–492. [CrossRef]
34. Yao, H.; Ma, S.; Wang, J.; Zhang, P.; Jiang, C.; Guo, S. A Continuous-Decision Virtual Network Embedding Scheme Relying on Reinforcement Learning. *IEEE Trans. Netw. Serv. Manag.* **2020**, *17*, 864–875. [CrossRef]
35. Kipf, T.N.; Welling, M. Semi-supervised classification with graph convolutional networks. *arXiv* **2016**, arXiv:1609.02907.
36. Liu, S.; Cai, Z.; Xu, H.; Xu, M. Towards security-aware virtual network embedding. *Comput. Netw.* **2015**, *91*, 151–163. [CrossRef]
37. Zhang, P.; Li, H.; Ni, Y.; Gong, F.; Li, M.; Wang, F. Security aware virtual network embedding algorithm using information entropy TOPSIS. *J. Netw. Syst. Manag.* **2020**, *28*, 35–57. [CrossRef]
38. Zhang, P.; Chen, N.; Li, S.; Choo, K.K.R.; Jiang, C. Multi-Domain Virtual Network Embedding Algorithm based on Horizontal Federated Learning. *arXiv* **2022**, arXiv:2205.14665.
39. Henaff, M.; Bruna, J.; LeCun, Y. Deep convolutional networks on graph-structured data. *arXiv* **2015**, arXiv:1506.05163.
40. Simonovsky, M.; Komodakis, N. Dynamic edge-conditioned filters in convolutional neural networks on graphs. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017; pp. 3693–3702.
41. Zhang, P.; Yao, H.; Liu, Y. Virtual Network Embedding Based on Computing, Network, and Storage Resource Constraints. *IEEE Internet Things J.* **2018**, *5*, 3298–3304. [CrossRef]