Supervised Learning Based Resource Allocation with Network Slicing

Tianxiang Zhang[1], Yuxin Bian[2], Qianchun Lu[1], Jin Qi[1], Kai Zhang[2], Hong Ji[2], Wanyuan Wang[2], and Weiwei Wu[2]

[1]zte corporation, Nanjing, Jiangsu, P. R. China
[2]School of Computer Science and Engineering, Southeast University, Nanjing, Jiangsu, P. R. China
{zhang.tianxiang, lu.qianchun9, qi.jin}@zte.com.cn, {bianyuxin, kzhang, hong_ji, wywang, weiweiwu}@seu.edu.cn

***Abstract*—With the fast growth of wireless network technologies(e.g., 5G, data center networks) and increasing demand for services with high Quality of Services(QoS), efficient management of network resources becomes more and more important. Network slicing is a effective method for reducing computing time through parallel computing and improve QoS of services. However, determining in which slice a request should be deployed on the premise of ensuring the success rate of transmission is difficult. To this end, we propose a slicing model and a resource allocation scheme in network slices. We make several contributions: i) a slice model that help us reduce the time it takes to calculate routes by parallel computing, and the formulation of resource allocation problem in network slices which aims at maximizing the amount of data transmitted successfully, and ii) a supervised-learning based model to quickly determine in which slices requests should be deployed, that can improve the success rate of transmission. Experimental results show that our proposed approach can achieve a good performance in network slicing environment.**

***Index Terms*—Network Slicing, Resource Allocation, Supervised Learning**

I. Introduction

With the fast growth of network technologies(e.g. 5G, data center networks) and ever-increasing demand for services with high Quality of Service (QoS) demand [1], the management of network resources becomes a more challenging task. Network slicing [2] provides an effective approach to introduce flexibility and higher utilization of network resources in the management of network resources. A slice is a collection of network resources, selected to satisfy the resource requests (e.g, bandwidth for traffic flows) to be provided by the slice [3].

Numerous studies have been done to improve QoS and network resource utilization for traffic scheduling. A helpful way to achieve this target is queue management and scheduling. In general, packets in a queue may have different priorities, e.g., packets in front may have higher priority than other packets [4]. Queue-based scheduling techniques are widely used to achieve the QoS support in Floodlight-controlled SDN networks [5]. In [6], traffics to be transmitted are classified as QoS flows and best flows, and then assigned to queues based on their priorities.

However, the queuing nature of the resource allocation or scheduling strategies above have an implicit drawback. As the feasible routes for flows/requests need to be calculated real-time by checking the remaining available bandwidth over network links, the time for computing a feasible route may be time-consuming as requests in the queue are sequentially processed. It has been noticed that network slicing have an effect of acceleration on routing [7]. Observing this, we introduce a network slicing model in this work to accelerate routing. In the proposed slicing model, different slices share the same network topology but have different bandwidth resources on the links. Flows are partitioned into different slices. As flows/requests are processes in independently in each slice, the computation of routes can be in parallel and this accelerates the routing process.

The main challenge in implementing such an idea is that, when determining which slice a request should be deployed in, it should guarantee a high success rate of transmission as some flows may fail to be transmitted due to resource scarcity. Moreover, if the routes are found by checking the feasibility in all partitioned slices, it would go against the purpose of accelerating the routing. To address this problem, we propose a supervised-learning-based resource allocation model to predict which slice a request should be deployed in, so as to maximize the number of successful transmitted requests. The proposed method employ multi-class multi-instance learning and predicts for a batch of requests at a time, which further enables the acceleration of computation.

The main contributions of this paper are summarized as follows,

- We propose a slicing model that makes different slices share the same network topology, which can help us achieve the purpose of reducing the time it takes to calculate routes for requests by parallel computing.
- We formulate the problem of resource allocation in network slices with the aim of maximizing the number of successfully transmitted requests. Then we propose a graph-convolutional-network based model MCEGCN to extract link-feature representations and determine which slice the request should be deployed in. This model can quickly determine the deployment of requests and improve the success rate of transmission.
- We conduct experimental evaluations in the ring network that is similar to real 5G network. Experimental results

show that our approach can achieve good performance on reducing time to calculate routes and improving the success rate of transmission.

The rest of this paper is organized as follows. Section II reviews the related work on resource allocation in different types of network technologies. Section III describes the problem formulation and model. We then present our solution design in Section IV. Section V presents the simulation results. Finally, we conclude this paper in Section VI.

II. Related Works

Network slicing and the managing of network resources have received consistent and considerable attention in recent years.

Virtualization of network resources has been seen as one of the main evolution trends in the fifth generation (5G) cellular networks and data center networks, which improve the Quality of Service (Qos), Quality of Experience (QoE) and network resource utilization [8]. Bari et al. [9] prospse the virtualized infrastructure which is a network infrastructure where some or all of the elements (e.g., servers, links, switches, topology) are virtualized by the data center network. The cloud computing platform consists of single or multiple virtualized infrastructures, which rely on virtualization techniques to partition available resources and share them among users.

For the limitation of network resources, the resource allocation scheme can be implemented to improve communication reliability and network utilization. Parsaeefard et al. [10] propose a combined resource provisioning and resource allocation mechanism aiming to maximize the total rate of virtualized networks based on their channel state information. An iterative slice provisioning algorithm has been proposed to adjust minimum slice requirements based on channel state information but without considering global resource utilization of the network as well as slice priority. A centralized joint power and resource allocation scheme for prioritized multi-tier cellular networks has been proposed in [11]. The authors put forward the scheme to admit users with higher priority level in order to maximize the number of served users. The priority is only considered at the user's level, and different priorities among slices are ignored.

In this work, we propose a slicing model that enable the scheduling of requests among different partitioned slices, which accelerates the routing by reducing computing and meanwhile maximizes the successful rate.

III. Modeling and Problem Formulation

In this section, we introduce the proposed slicing model and the corresponding resource allocation problem.

A. Problem Setup

We formulate a communication network as a weighted undirected graph $G = (V, E, C)$ where V is the set of nodes which represent the routers or switches in the network, E is the set of edges which represent links connecting the nodes and C is the set of weight vectors of edges which denote metrics of links in the network. Assume that there are m nodes $V = \{v_1, v_2, ..., v_m\}$ in the graph, and we denote the edge between v_s and v_t by e_{st}. We use $C_{st} = (d_{st}, b_{st}, a_{st})$ to denote the weight of edge e_{st}, where d_{st}, b_{st}, a_{st} represents the delay, bandwidth and available bandwidth respectively, to reflect the features of Qos. Fig. 1 shows a simple example of communication network which contains six nodes and seven edges. There are three edges $\{e_{12}, e_{13}, e_{14}\}$ which start from node v_1, and the weight of edge e_{12} is $C_{12} = (d_{12}, b_{12}, a_{12})$.

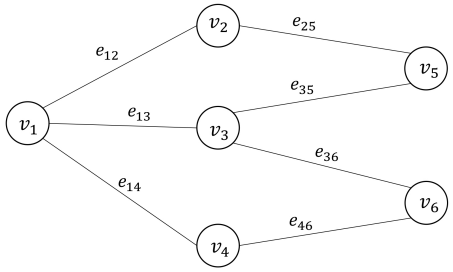


Fig. 1. A diagram of a communication network.

There are n requests $F = \{f_1, f_2, ..., f_n\}$ which need to be deployed in the network. Each request $f_l = ((S_l, D_l), d_l)$ has a source-destination pair (S_l, D_l) which indicates that the request should be transmitted from the the source node S_l to destination node D_l, and the required data size d_l. Each request f_l has a routing path, which is generated by the routing strategy. In this work, we assume that each path is generated by the shortest path with minimum delay. Note that a feasible shortest route depends on the current state of the network especially each edge's current available bandwidth. We define the shortest route for request f_l as P_l, which is the minimum transmission delay path among all paths from S_l to D_l that satisfy the bandwidth constraints, i.e., the available bandwidth of each edge in path P_l should be no less than request f_l's data size d_l,

$$a_{st} \geq d_l, \forall e_{st} \in P_l. \tag{1}$$

B. Slicing Model

After determining the transmission route of a request, the corresponding bandwidth resource is allocated to serve the request and we need to update the remaining available bandwidth of edges on the shortest route. Thus, to deploy a batch of requests, the shortest routes must be calculated sequentially to avoid deployment failure of subsequent request. For example, there are two requests f_1, f_2 should be deployed in the network in Fig. 2 where $f_1 = ((v_1, v_3), 80)$ and $f_2 = ((v_1, v_3), 60)$. If calculating their shortest route at the same time, both of them will be transmitted in the path $v_1 \rightarrow v_3 \rightarrow v_2$, which makes one of the two requests fail to transmit due to insufficient available bandwidth.

As a consequence, it is time-consuming to calculate routes for a set of numerous requests F. Let T_F denote the total calculation time,

$$T_F = \sum_{l=1}^{n} T_l, \tag{2}$$

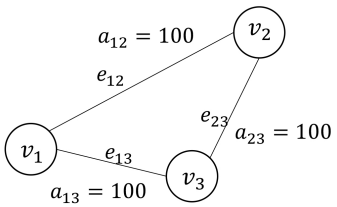


Fig. 2. A network with 3 nodes and 3 edges. $d_{12} = 3, d_{23} = 4, d_{13} = 10; a_{12} = a_{13} = a_{23} = 100$

where T_l is the time took to calculate the route for request f_l.

To solve the above problem, we introduce the slicing model which uses the Network Slicing method to reduce the time consumption. Specifically, we slice the network into different pieces and use parallel computing to calculate the routes of requests on different slices at the same time. In this case, the time it takes to calculate routes for a set of requests is the maximum time consumed among all slices. Assume that the network is sliced into K slices, and the total time took to calculate routes for a set of requests is defined as,

$$T_F^s = max\left(T^1, T^2, ..., T^K\right), \tag{3}$$

where T^h is the time took to calculate routes for all requests in slice s_h. In this case, the time T_F^s can be approximately reduced to $\frac{1}{K} \cdot T_F$ if n requests are equally partitioned and served by K network slices.

We consider the co-existence of multiple slices on the same physical network. Different slices can share requests which need resources to be served. Each slice is a partition of the virtual network. In detail, we slice the network into K network slices $S = \{s_1, s_2, s_3, ..., s_K\}$. For each network slice s_i, we also formulate it as a weighted undirected graph $G_i = (V_i, E_i, C_i)$ which has the same network topology as the original graph $G = (V, E, C)$, i.e., $V_i = V, E_i = E$. Additionally, the delay of each edge e_{st} in E_i is also d_{st}.

However, the bandwidths and available bandwidths of all edges in different slices can be different, which depend on the specific segmentation. Fig. 3 illustrates the slicing method in our slicing model.

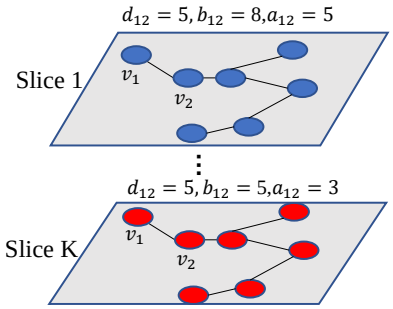


Fig. 3. The network slices in our slicing model. Different slices have the same network topology and delay, while their bandwidths and available bandwidths can be different.

In addition, in order to achieve the purpose of accelerating parallel computing, a large number of services should be avoided from transmitting in the same slice. To achieve this target, we should ensure that the bandwidths allocated by different slices is as uniform as possible during the initial slicing.

C. Resource Allocation in Network Slices

Although Network slicing can reduce the time consumption on calculating shortest routes, a request may fail to transmit due to the variation of edges' available bandwidth, when the available bandwidth of the network can support a feasible routing path to serve the request. To address this problem, we design an appropriate resource allocation scheme to determine which slice a request should be deployed in as well as the order of transmission within each network slice, which aims to improve the success rate of transmission and the total amount of transmitted data.

Assume that there are m requests $F_l = \{f_1^l, f_2^l, ...f_m^l\}$ deployed in slice s_l, where the requests will be transmitted in the order of $(f_1^l, f_2^l, ...f_m^l)$ and f_q^l is the q-th request will be transmitted in slice s_l. Denote by $F_l^s \subseteq F_l$ the set of requests that are successfully transmitted in slice s_l.

Our objective is to design a resource allocation scheme to maximize the number of total transmitted requests $|F^s|$ and the total amount of data transmitted among all the network slices, i.e.,

$$\max \sum_{s_l \in S} \sum_{f_l \in F_l^s} d_l. \tag{4}$$

IV. Methodology

To address the resource allocation problem above, we propose a supervised learning algorithm named MCEGCN (multi-output classification edge-based graph convolutional network) to determine/predict which slice a request should be deployed in. By improving the accuracy of this model's prediction, the success rate of request transmissions can also be improved.

A. Supervised Learning Based Resource Allocation

We introduce the MCEGCN learning model which employs multi-class multi-instance classification method [12]. The labels of samples in MCEGCN model have more than two classes. Thus, a single estimator should handle several joint classification tasks. The MCEGCN model is a generalization of the multilabel classification model in [12], which only considers binary attributes, as well as a generalization of the multi-class classification model in [13], where only one property is considered.

The main idea of MCEGCN is to specify one output layer for each request, which allows multiple variable classifications. Specifically, we extend a single estimator to be able to estimate n target functions $(func_1, func_2, ..., func_n)$, each corresponding to a request. Furthermore, each target function (e.g., $func_l$) predicts K-dimensional responses $(y_1, y_2, ..., y_K)$ where y_k specifies the probability of allocating to slice k (for request f_l).

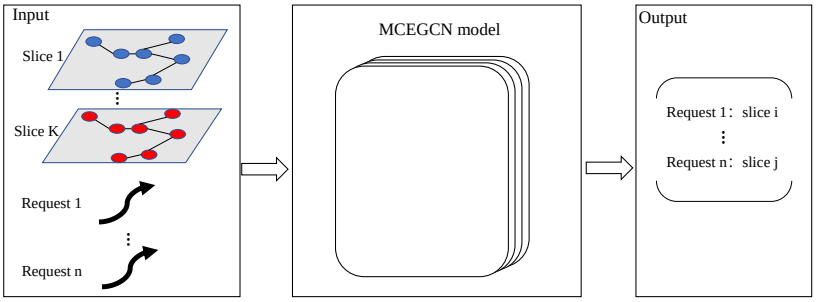


Fig. 4. The framework of the resource allocation method

In our work, the target of the classification task is to predict which slice a request should be deployed in. Each sample contains the input information, K slices $(G_1, G_2, ..., G_K)$, a batch containing n requests $(f_1, f_2, ..., f_n)$, and the output label $Y = (y_1, y_2, ..., y_K)$ which specifies the slice to be allocated. The slice attribution is a property consisting of n classes corresponding to K slices. Recall that, for each slice G_k, it contains the set of nodes, the set of edges, and the set of weights of edges, i.e., $G_k = (V_k, E_k, C_k)$ and $C_{st} = (d_{st}, b_{st}, a_{st})$. And for each request f_l, it involves the information of source-destination pair (S_l, D_l) and data size d_l.

The framework of our resource allocation method is illustrated in Fig. 4. When the MCEGCN model receives the input, the trained model will predict the target slices to be allocated for n requests. Specifically, we construct n output layers corresponding to n requests. In order to train our model to achieve better results, we proposed a Sample Generation Algorithm to provide valid samples, which will be described in Section IV-C.

The MCEGCN model shown in Fig. 5 consists of two edge-based graph convolutional neural network (EGCN) layers which can further take full advantage of the graph structural properties of network slices and will be detailed in Section IV-B. In this model, we use Cross-entropy as loss function, i.e.,

$$J = -\sum_{k=1}^{K} y_k log(p_k) \tag{5}$$

where K is the number of classifications, y_k is the label ($y_k = 1$ means that the current flow is allocated to slice k, otherwise, $y_k = 0$), and p_k is the predicted probability that the request should be allocated to slice k.

B. Edge-based Graph Convolutional Networks

To predict which slice a request should be allocated to, one simple solution is to employ deep neural network. However, this may fail in completing the prediction task as it cannot exploit the graph nature of the links. To this end, we adopt a graph convolutional network (GCN) framework to represent the link status information in network slices with different remaining bandwidths. Unlike ordinary GCN [14] which takes nodes as input, we design a novel edge-based graph convolutional neural network (EGCN) to extract the

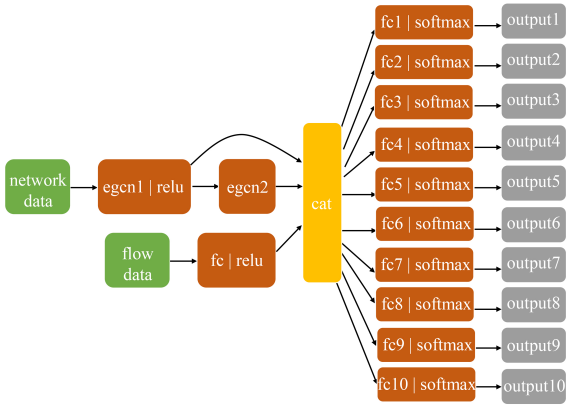


Fig. 5. A diagram of network structure of MCEGCN

spatial correlations of the edge features. Both the feature matrix and adjacency matrix in our problem are defined on edges instead of nodes. Fig. 6 is an example showing the difference of adjacency matrix that is defined on nodes and edges, respectively. In detail, Fig. 6(a) and Fig. 6(c) illustrate the node-based graph and the corresponding adjacency matrix on nodes, while Fig. 6(b) and Fig. 6(d) demonstrate the edge-based graph and the corresponding adjacency matrix on edges, respectively.

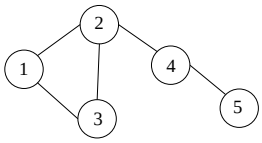
We define EGCN as $f(w, V)$ where $w \in \mathbb{R}^{|E| \times |Z|}$ is the state matrix of given network slices, and V is the adjacency matrix of the edges. Formally, we adopt the formulation proposed by Kipf and Welling [14] to model our EGCN with the following layer-wise formulation:

$$H^{(l+1)} = f(H^l, A) = \sigma(\tilde{M}^{-\frac{1}{2}} \tilde{V} \tilde{M}^{-\frac{1}{2}} H^{(l)} \Phi^{(l)}) \tag{6}$$

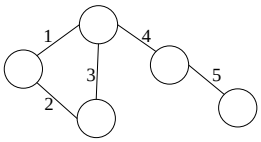
where $\tilde{V} = V + I_N$ and V is the adjacency matrix defined on the edges, I_N is an identity matrix to account for the effect of a node itself and M is a diagonal edge-degree matrix utilized to normalize $\tilde{V}$. $H^{(l)}$ and $H^{(l+1)}$ are the outputs of the l-th and the $(l+1)$-th layers, respectively. $\Phi^{(l)}$ is a trainable weight matrix for the l-th layer, and σ is the activation function.

C. Sample Generation

For the training of MCEGCN model, we simulate the deployment process to generate numerous valid training samples.



(a) Graph coded by node



(b) Graph coded by edge

$$\begin{pmatrix} 0 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

(c) Adjacency matrix on nodes

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

(d) Adjacency matrix on edges

Fig. 6. An example of adjacency matrix defined on nodes

To obtain multi-class multi-instance samples, we generate many batches that contain n requests and simulate the deployment process to check whether a batch could be successfully allocated. If requests in a batch are all successfully transmitted, this batch will be selected as a valid sample.

The proposed generation algorithm is shown in Algorithm 1. In detail, we first sort requests in the non-increasing order of their data size, and the request with larger data size will be deployed first. After that, we deploy the sorted requests in k slices by polling method. For example, we deploy the largest request f_1 in slice s_1, and then deploy the second request f_2 in slice s_2 and so on. When the current request cannot be deployed in the selected slice, we try to deploy in the next slice until it is successfully deployed. If all requests in a batch are successfully transmitted, we take it as a valid sample which contains the corresponding slice (label) for each request. Otherwise, that batch would be discarded.

V. Experimental Results

In this section we conduct experiments to evaluate our algorithm.

A. Experiment Setup

We first introduce the network topology, compared methods, and parameters of the MCEGCN model.

Network topology. We use the ring network similar to real 5G network as the experimental environment. This ring network consists of three kinds of rings, namely, the core layer, the convergence layer and the access layer. The bandwidth and delay of links in three kinds of layers are different, e.g., core layer, convergence layer, and access layer. Core layer has the largest bandwidth of the network, which is the destination of most services in the network. The bandwidth of the convergence layer is smaller than that of the core layer, and convergence layer connects core layer and access layer. Access layer has the smallest bandwidth of the network,

Algorithm 1 Sample Generation

Require: Initial state of network slices $S = \{s_1, s_2, s_3, ..., s_K\}$ and a batch of requests $F = \{f_1, f_2, ..., f_n\}$
Ensure: Decision of the batch requests will be transmitted in which slices

```
Initialize: current slice s_cur ← s_1, choices of slices c ← {}
Sort all requests in the non-decreasing order of data size, f_1 ≥ f_2 ≥ ... ≥ f_n
for l ← 1 to n do
    Calculate the shortest path p_l for f_l if it will be transmitted in s_cur;
    if Shortest path p_l exists then
        Update the network state of current slice s_cur : a_st = a_st − f_{l,cur}, ∀e_st ∈ p_l (f_{l,cur} denotes cost by transmitting f_l in the shortest path)
        Update the choice of slices c ← c ∪ {s_cur}
        s_cur ← nextslice
    else
        s_cur ← nextslice
        Go to step 4
    end if
    if Shortest path does not exist when f_l transmit in any slices then
        Drop this sample and Regenerate sample by other batch of requests
    end if
end for
return choices of slices c
```

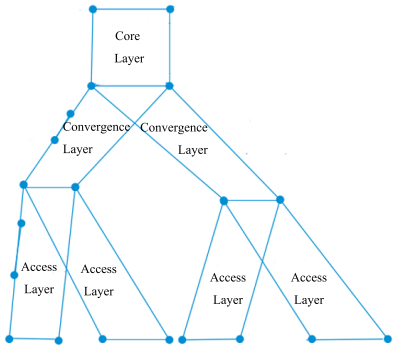


Fig. 7. An example of network topology of ring network. Core layer is a ring network composed of the core equipment, and it is the destination of most of the services in the network; Convergence layer connects access layer and core layer, and aggregate each access layer network; Access layer is composed of users and the terminals.

which is composed of users and terminals. An example of the network topology of the ring network is shown in Fig. 7. In this experiment, we use a ring network with 18 nodes and 26 links. The batch size of requests should be handled at a time is set as $n = 10$ and the number of network slices is set to $K = 3$. The initial available bandwidth of the link and the size of the request are generated by uniform distribution, following $U(\gamma b_{st}, b_{st})$ and $U\left(0.2\gamma b_{st}^{min}, \gamma b_{st}^{min}\right)$ respectively, where γ is a a coefficient between 0 and 1(the values of γ are set to $0.9, 0.8, 0.7, 0.6$ corresponding to network utilization rates of $10\%, 20\%, 30\%, 40\%$ respectively) and b_{st}^{min} is the minimum bandwidth of edge in the network.

Compared methods. For fair comparison, we compare our algorithm with the following three methods, Random strategy, Polling strategy and Multilayer perceptron (MLP) model. The Random strategy randomly selects a slice among three slices for each request. The polling strategy allocates the requests into slices in a round-robin manner. The MLP model is a trained multi-output model which employs multi-layer

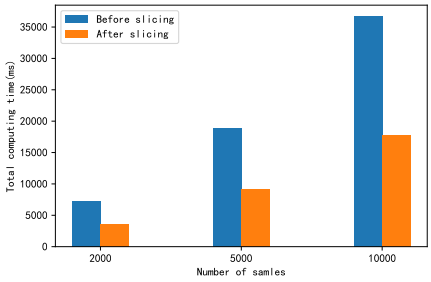
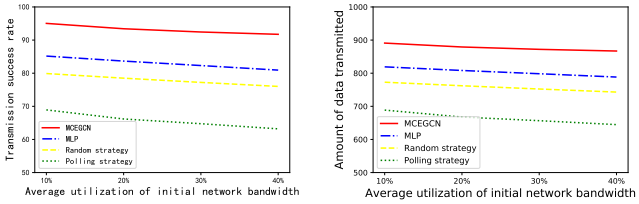


Fig. 8. Results of computing time.



(a) Success rate of deployment (b) Transmission data

Fig. 9. Results over different average utilization when the initial network bandwidth changes.

perceptron for feature representation. The three methods have the same advantage that requests could be easily determined to which slice it should be deployed without testing the available remaining resources in K slices.

Settings in the MCEGCN model. There are 10 output layers in the MCEGCN model, each corresponding to the prediction for requests in a batch with size n. We generate 8000 and 2000 samples in training set and test set respectively.

B. Result Analysis

Fig. 8 shows the computing time consumed by calculating routes before and after slicing. When the number of samples is 2000, 5000, and 10000, the computing time after slicing is much less than that before slicing. The ratio of calculation time before and after slicing is about 2.1 : 1.

As shown in Fig. 9, we evaluate the success rate of deployment and total transmission over different utilization rate when initial network bandwidth changes, which reflects the performance of the methods in different network congestion situations. A successful deployment is achieved if a request can be successfully deployed in the specified slice. It can be seen that, our method has higher success rate of deployment as well as larger transmission data than other three methods. Compared with the MLP model which performs best among the three baseline methods, the MCEGCN model gains about 10% higher success rate and about 10% more transmitted data. When the initial bandwidth utilization is higher, that is, the available bandwidth is scarce, the MCEGCN model still performs much better than the other methods.

VI. Conclusion

In this paper, we present a novel approach for resource allocation in the network with network slicing, including a slicing model and a MCEGCN model determining which slice the requests should be deployed in. The proposed approach can accelerate the time in route computation and meanwhile guarantee a high success rate of transmission. By performing experimental evaluations in a network environment similar to real 5G network, we show that the proposed method outperforms the benchmark methods.

VII. Acknowledgment

The work is supported in part by the national key research and development program of China under grant No. 2018AAA0101204, a grant from ZTE Corporation, and National Natural Science Foundation of China under Grant No. 61672154, 61672370, 61972086.

References

[1] Andrews J G, Buzzi S, Choi W, et al. What will 5G be?[J]. IEEE Journal on selected areas in communications, 2014, 32(6): 1065-1082.
[2] Einsiedler H J, Gavras A, Sellstedt P, et al. System design for 5G converged networks[C]//2015 European Conference on Networks and Communications (EuCNC). IEEE, 2015: 391-396.
[3] Alliance N. 5G white paper[J]. Next generation mobile networks, white paper, 2015, 1.
[4] Karakus M, Durresi A. Quality of service (QoS) in software defined networking (SDN): A survey[J]. Journal of Network and Computer Applications, 2017, 80: 200-218.
[5] Wallner R, Cannistra R. An SDN approach: quality of service using big switch's floodlight open-source controller[J]. Proceedings of the Asia-Pacific Advanced Network, 2013, 35(14-19): 10.7125.
[6] Xu C, Chen B, Qian H. Quality of service guaranteed resource management dynamically in software defined network[J]. Journal of Communications, 2015, 10(11): 843-850.
[7] Xu C, Gamage S, Lu H, et al. vturbo: Accelerating virtual machine i/o processing using designated turbo-sliced core[C]//Presented as part of the 2013 USENIX Annual Technical Conference (USENIXATC 13). 2013: 243-254.
[8] Zhu K, Hossain E. Virtualization of 5G cellular networks as a hierarchical combinatorial auction[J]. IEEE Transactions on Mobile Computing, 2015, 15(10): 2640-2654.
[9] Bari M F, Boutaba R, Esteves R, et al. Data center network virtualization: A survey[J]. IEEE Communications Surveys & Tutorials, 2012, 15(2): 909-928.
[10] Parsaeefard S, Jumba V, Derakhshani M, et al. Joint resource provisioning and admission control in wireless virtualized networks[C]//2015 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2015: 2020-2025.
[11] Monemi M, Rasti M, Hossain E. Low-complexity SINR feasibility checking and joint power and admission control in prioritized multitier cellular networks[J]. IEEE Transactions on Wireless Communications, 2015, 15(3): 2421-2434.
[12] Tsoumakas G, Katakis I. Multi-label classification: An overview[J]. International Journal of Data Warehousing and Mining (IJDWM), 2007, 3(3): 1-13.
[13] Aly M. Survey on multiclass classification methods[J]. Neural Netw, 2005, 19: 1-9.
[14] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks[J]. arXiv preprint arXiv:1609.02907, 2016.