Adaptive Digital Twin and Communication-Efficient Federated Learning Network Slicing for 5G-enabled Internet of Things

Daniel Ayepah-Mensah
School of Computer Science and Engineering,
University of Electronic Science
and Technology of China, UESTC
Chengdu, China
Email:ayeps2000@gmail.com

Guolin Sun
School of Computer Science and Engineering,
University of Electronic Science
and Technology of China, UESTC
Chengdu, China
Email: gulin.sun@uestc.edu.cn

Yu Pang
School of Electro-optical Engineering,
Chongqing University of Posts and Telecommunications, CUPT,
Chongqing, China
Email: pangyu@cqupt.edu.cn

Wei Jiang
Department of Electrical and Information
Technology,
Technische University, TU,
Kaiserslautern, Germany
Email: wei.jiang@dfki.de

***Abstract*—Network slicing enables industrial Internet of Things (IIoT) networks with multiservice and differentiated resource requirements to meet increasing demands through efficient use and management of network resources. Typically, the network slice orchestrator relies on demand forecasts for each slice to make informed decisions and maximize resource utilization. The new generation of Industry 4.0 has introduced digital twins to map physical systems to digital models for accurate decision-making. First, graph-attention networks are used to build a digital twin environment for network slices for real-time traffic analysis, monitoring, and demand forecasting. Based on the predictions, we formulate the resource allocation problem as a federated multi-agent reinforcement learning problem and use a Deep Deterministic Policy Gradient to determine the resource allocation policy while preserving the privacy of the slices. We show that our proposed approaches can improve the accuracy of demand prediction for network slices and reduce the communication overhead of dynamic network slicing.**

***Index Terms*—Demand forecasting, Digital twins, Network Slicing, Resource allocation, Federated learning;**

I. Introduction

The explosive growth of the Internet of Things (IoT) poses significant hurdles for network operators. As a result of the rapid development of communication and sensor technologies paving the way to realize the IoT, it has become more challenging to support urgent and reliable communications with their quality of service (QoS) requirements [1]. As a core component of the 5G architecture, Network Slicing supports various application scenarios and customized services. The physical infrastructure can be divided into logical network instances, called slices [2]. However, IoT networks are characterized by considerable differences among service types [3]. Hence dynamic module for resource allocation should meet individual user requirements. Dynamic network slicing algorithms developed in recent years take advantage of the large amounts of data flowing through the network, which contain information relevant to resource allocation decisions. By leveraging the data generated by the network, network slices can predict and exploit the upcoming behavior of a system with many different actors. This will enable slices to allocate resources accurately to their users. To this end, accurately predicting the demand of a given network slice is critical to increasing the resource utilization of network slices.

Some works study the impact of demand forecasting for network slicing and dynamic resource allocation for IoT 5G services, while others analyze slicing as a radio access network (RAN) sharing issue. In [4], a framework is proposed to implement a capacity prediction algorithm that considers guaranteed and best-effort traffic. The authors in [5] proposed an AutoRegressive Integrated Moving Average (ARIMA) based traffic analysis to prevent slice-level agreement violation. In addition, various Deep Q-Network (DQN) algorithms are used to address the problem of demand-based resource allocation, where an agent learns how to perform optimal actions in an environment by observing state transitions and providing feedback [6]. As a result of the increasing number of IoT devices and slices within the network, the network cannot handle the workload associated with large networks resulting in scalability issues. In addition, for dynamic network slicing to be successful, resource-sharing strategies must be collaborative and autonomous. There is also a need for network slices to share information with the infrastructure providers (InP). This includes user numbers, resource requirements, and any other sensitive information. However, network slices may not want to share because of privacy concerns. Data such as these are stored in a centralized database. Accessing them

from the centralized InP would be computationally prohibitive due to the amount of processing and optimization required as well privacy issues associated with a centralized data store. Finally, many works have examined demand forecasting for network slices but primarily focused on modeling a single sequence. This is constrained to only account for the time series dependence on traffic networks and ignores the underlying spatiotemporal topological traffic information of the IoT devices in the slice. Hence, the forecasting model for network slicing should be able to capture the spatial-temporal evolution of traffic generated by the slices.

This paper proposes an intelligent digital twin (DT) framework as a scalable solution for real-time data-driven monitoring and accurate prediction of IoT demand in 5G-based IoT network slices to address the issues above. As an emerging digitization method, DT provides a viable alternative to capture the dynamic and complex network environment [7]. It creates a virtual environment in digital space through software definition. By utilizing a graph learning model algorithm, we can fabricate DTs, i.e., virtual representations of IoT devices with similar structural properties to the physical network. We propose a Graph-Attention-Network (GAT) based DT that can capture the temporal characteristics and accurately monitor and predict the traffic demand of each slice. Based on the demand predictions, we formulate the resource allocation problem as a decentralized multiagent reinforcement learning (RL) problem to find the optimal resource allocation policy under slice service demand uncertainty. Furthermore, we rely on federated learning (FL), a decentralized machine learning technique, to train deep learning models without compromising privacy. The outcome is a digital-twin multiagent federated learning (DT-MAFL) network slicing. The remainder of this article is organized as follows: In Section II, we describe the system model and propose the dynamic resource allocation problem. Section III describes the construction of the DT for each slice. Section IV presents the design of a federated multiagent learning system for dynamic slicing. Section V provides a detailed simulation to validate our proposed slicing framework. Section VI concludes the paper.

II. System Model

This section describes the RAN slicing system model for 5G-enabled IoT networks. As shown in Fig. 1, the wireless spectrum resources provided by InP are abstracted to constitute a shared resource pool that consists of a certain number of available resource blocks (RBs). These resources can be leased and shared between different network slices. To meet the expected service requirements, the a controller in the cloud is assumed to manage the shared resource pool and allocates resources to slices according to their characteristics [1].

A. Network Model

We consider an orthogonal frequency division multiplexing (OFDM) system in which a single BS b are assumed to be shared by $\mathcal{M} = \{1, 2, 3, 4 \dots M\}$ network slices in a downlink setup. Each slice has a set of IoT devices, denoted by $\mathcal{U}$, and can be represented by $\mathcal{U} = \{1, 2, 3, 4 \dots \mathcal{U}_{\mathcal{M}}\}$. In our network model, time is divided into slots denoted by t. Each slice is characterized by a demand $d_m = \{r_u, r_u, \dots, r_u\}$ $(u \in \{1, 2, \dots \mathcal{U}\})$, which is the determinant factor for allocating resources w_m to slices. In this paper, Shannon theory is used to define the transmission rate for IoT devies, i.e.,

$$r_u = w_{u,t}^m \log_2 \left(1 + \frac{p_{u,t} \left|h_{u,t}\right|^2}{N_0}\right) \tag{1}$$

where $w_{u,t}^m$ is the amount of resource assigned to user u and $p_{u,t}$ is the transmit power on base station at the t-th TTI. $h_{u,t}$ is the channel coefficient, which contains path loss, shadowing effect Θ_u and Rayleigh fast fading $g_{u,}$ between IoT device u and the base station. We define as $h_{u,t} = 10^{-\mathrm{PL}^*(u,)/20} \sqrt{\vartheta_{u,} \Theta_{u,} g_{u,}}$ where (PL) is the Path Loss and is defined as $(\mathrm{dB}) = 20 \log_{10}(d) + 20 \log_{10}(f) - 27.55$ where f (in MHz) and d (in meters) representing IoT device to base station channel frequency and distance respectively [8]. Moreover, N_0 is the power of additive white Gaussian noise (AWGN). The average delay τ_u experienced by IoT device u in slice m is $\tau_u = \frac{1}{r_b - \lambda_u}$ where λ_u is the packet arrival rate. Based on IoT device requirements, different slices can be created to serve different types of IoT device traffic. A summary of commonly used notations has been provided in Table I.

B. Utility Model

We introduce a utility function to quantify the QoS needs of IoT devices in a slice [8]. We define QoS utility as the satisfaction of user u on either data rate or delay. The utility function of a slice represents the user preference in the network

TABLE I: Summary Of Main Notations

Notation	Definition
$t \in T$	Time slots
$m \in \mathcal{M}$	Number of slices
$u \in \mathcal{U}$	Number of users
$r_{u,t}^m$	Data rate by user
λ_u	Packet arrival rate
d_m	Demand of slice m
$p_{u,t}$	Transmit power on base station
$h_{u,t}$	Channel coefficient
Θ_u	Shadowing effect
g_u	Rayleigh fast fading
$\mathbf{w}_m$	Amount of RBs for slice m
φ_m	Number of RBs needed by slice m
τ_u	The average delay experienced by user u
$U_u(\cdot)$	QoS satisfaction
Ω_m^t	Resource utilization for a slice
$\mathcal{D}$	Data samples for slice
$\mathcal{G}$	Graph
$\boldsymbol{q}$	Weight vector
W^z	Weight matrix
$\boldsymbol{A}$	Adjacency matrix
β	hyper-parameter
$\mathbf{M}^1, \mathbf{M}^2$	Dynamic Filters
θ	Learning parameter for slice
s, a, $\mathcal{R}$	State, action and reward of a slice
π, y, $\nabla_\theta J$	Policy , Target and gradient policy and for slice m
η	Learning rate
γ	Discount factor

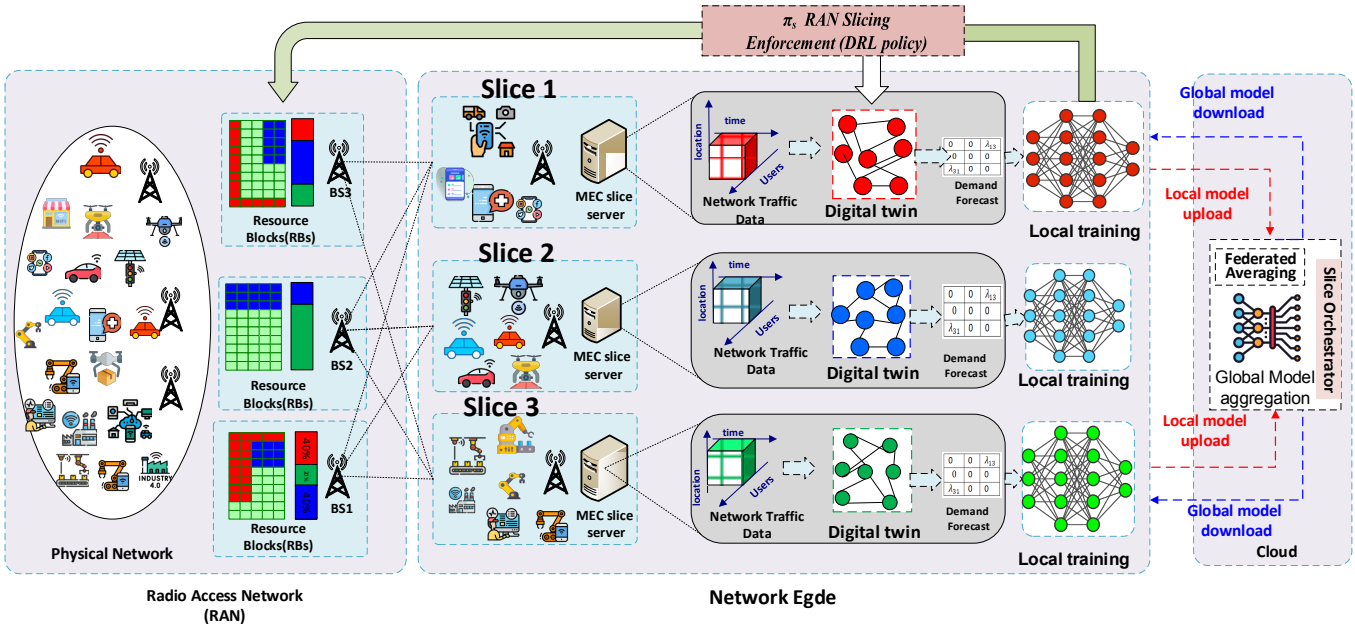


Fig. 1: DT-MAFL dynamic network slicing architecture.

by mapping the achievable rate and maximum delay with the level of UE satisfaction. Here, we use a sigmoid function to express users' satisfaction in terms of rate and delay. In this article, we adopt a framework for satisfaction function $U(\cdot)$ that can meet different optimization goals for both delay and rate constraint as $U(\tau_u)$ and $U(r_u)$, respectively. Hence, we propose a unified utility function for both delay and rate constraint users.

1) Rate-constrained slice: For the slice m, the average QoS satisfaction of a rate-constrained user is defined as;

$$U_u(r_u) = \frac{1}{1 + e^{-\phi(r_u - r_m^{min})}}, \tag{2}$$

where r_m^{min} denotes the minimum rate requirement of the user in slice m, ϕ specifies the steepness of the satisfactory curve.

2) Delay-constrained slice: The average satisfaction on delay in slice m is defined as

$$U_u(\tau_u) = \frac{1}{1 + e^{-\phi(\tau_u^{max} - \tau_u)}}, \tag{3}$$

where the maximum tolerant delay is denoted as τ_u^{max}, which is required to satisfy the upper bound delay for the IoT device u. Essentially, a slice utility U_m can be defined as a summation of individual utilities of the users who make up that slice. The slice utility is defined as:

$$U_m = \sum_{u \in \mathcal{U}_n} U_u(.) \tag{4}$$

Furthermore, the average utilization of a slice is defined as:

$$\Omega_m = \frac{\mathbf{w}_m}{\varphi_m} \tag{5}$$

where $\mathbf{w}_m = \sum_{u \in \mathcal{U}_m} w_{u,t}^m$ is the amount of RBs occupied by slice m and φ_m is the number of RBs needed by slice m which is the demand of the slice.

C. Federated Learing Model

Each slice m has a set of data samples and trains its local model using stochastic gradient descent (SGD) [9]. We formulate a learning problem for dynamic slicng where each slice has the task to optimize the global loss function $F(\theta)$ by minimizing the weighted average of the local loss function $F_m(\theta)$ as

$$\min \left\{ F(\theta) \triangleq \sum_{m \in \mathcal{M}} \frac{|\mathcal{D}_m|}{|\mathcal{D}|} F_m(\theta) \right\} \tag{6}$$

Taking our proposed slicing model into account, we propose that the centralized cloud, i.e. the InP, only performs the aggregation of the model and network slices autonomously perform dynamic network slicing.

D. Problem Formulation

In our proposed dynamic slicing framework, our goal is to optimize the resource utilization of a slice and ensure the the QoS requirement of IoT devices in the slice are met. We formulate the optimization problem as maxizing the QoS utility of the slice as:

$$\begin{aligned} &\arg\max_{b} \{\Omega_m + U_m\} \\ &\text{s.t. } \mathbf{w}_m \leq \kappa, \qquad \mathbf{w}_m > 0 \end{aligned} \tag{7}$$

where κ is the maximum amount that can be allocated to a slice. $\mathbf{w}_m \leq \kappa$ indicates that the amount of the resources allocated to the slice should not exceed the total resources available on the base station, and $\mathbf{w}_m > 0$ is the constraints that ensure that of at least a slice always has resources. In dynamic network slicing, not only the current service demand needs to be considered, but the future demands of the slice

also need to be considered. Hence we proposed a DT model which monitors the real-time behavior and accurately predicts each slice's demand.

III. Adaptive DT based demand Forecast for RAN slices

In this section, we design a dynamic graph-based DT to capture the dynamics and spatial dependency of traffic in a slice. During the resource allocation stage of a slice, the graph-based DT is composed of four steps: ***i:*** **Feature Extraction**: a CNN-based feature extraction net is used to convert the raw spatial-temporal data into the feature matrix representations. ***ii:*** **Graph Construction:** We adopt a direct optimization approach to learn dynamic graph structures, which generate a graph adjacency matrix that represent the DT. ***iii:*** **Interaction Modeling:** Captures the dynamic interaction of UEs and the structural information within the nodes (UEs and gNodeB) of the slice. ***iv:*** **Prediction Model:** integrates each node's individual sequential information to predict the slice's traffic demands. The network topology of the slice at each time interval can be represented as a sequence of graph snapshots $\mathcal{G} = \{\mathcal{G}_1, \cdots, \mathcal{G}_T\}$, where T is the number of time steps.

To construct the DT, the network topology of IoT devices in the slice is represented as a graph. The graph can be defined as $\mathcal{G} = \{\mathcal{V}, \mathcal{E}\}$ with N nodes, where $\mathcal{V}$ and $\mathcal{E}$ are the set of nodes and edges, respectively. The network traffic data generated by the slice is defined as $d \in \mathbb{R}^{Z \times T \times V}$ where Z is the input channels and T time intervals, and V is the spatial data associated with the channel [10]. The task of traffic forecasting is to learn a mapping function $h(\cdot)$, which takes historical traffic data d and graph $\mathcal{G}$ as inputs to forecast future time intervals traffic data:

$$\hat{d}_m = h(d_m, \mathcal{G}, \psi), \tag{8}$$

where ψ is the learnable parameters. However, the Slice traffic data is defined as multivariate and temporal data and no nodes. Therefore, the proposed DT is used to build a graph that comprehensively models the interaction between IoT devices in a slice.

A. Graph construction.

The first stage of the proposed slice DT is a graph learning layer that learns a graph adjacency matrix adaptively to capture the hidden relationships among time series data for the slice. The adjacency matrix can be seen as DT, representing IoT devices' topology and spatial behavior in a slice. Firstly, CNN-based feature extraction is used to convert the raw spatial-temporal data into the feature matrix representation $\mathbf{B}^m$. Based on the feature matrix, a unique graph structure is generated for the slice, consistent with the dynamic property of traffic data generated by BS. Inspired by [11], the underlying adjacency matrix $\mathbf{A}$ is dynamically generated with:

$$\boldsymbol{A} = \mathrm{ReLU}\left(\tanh\left(\beta\left(\mathbf{M}_m^1 \mathbf{M}_m^{2\,\mathrm{T}} - \mathbf{M}_m^2 \mathbf{M}_m^1\right)\right)\right) \tag{9}$$

where β is a hyper-parameter which is a saturation rate of the activation function and $\mathbf{M}^1 = \mathbf{M}^2 = \tanh\left(\beta(\mathbf{E}_m \mathbf{B}^m\right)$ is a dynamic filter with $\mathbf{B}^m$ as its input.

B. Interaction Modeling for slices with GAT

The GAT layer can model the relationships between nodes and frequencies [12]. The GAT generates the attention weight that reflects the channel quality of traffic generated by IoT devices in a slice. The input to the graph layer is a set of node features $\{\boldsymbol{x}_1 \ldots, \boldsymbol{x}_N\}$ embedded in the adjacency matrix $\boldsymbol{A}$ and the output is a new set of node features $\{\boldsymbol{x}'_1, \ldots, \boldsymbol{x}'_N\}, \boldsymbol{x}'_z \in \mathbb{R}^H$, where H is the number of features for each node. We compute the attention coefficient of node $z \in \mathcal{N}_z$ to node v as follows [12]:

$$\alpha_{zv} = \frac{\exp\left(\mathrm{LeakyReLU}\left(\boldsymbol{q}^T\left[W^z \boldsymbol{x}_z \| W^z \boldsymbol{x}_v\right]\right)\right)}{\sum_{w \in N_v} \exp\left(\mathrm{LeakyReLU}\left(\boldsymbol{q}^T\left[W^z \boldsymbol{x}_u \| W^z \boldsymbol{x}_v\right]\right)\right)} \tag{10}$$

where $\mathcal{N}_v$ is the set of immediate neighbors and $W^z \in \mathbb{R}^{F \times D}$ is the weight matrix for the graph. $\boldsymbol{q} \in \mathbb{R}^{2D}$ is a weight vector of each node in the graph. Note that A_{zv} is the weight of link (u, v) in the current snapshot $\mathcal{G}$. α_{zv} indicates the importance of node features in the traffic data. The attention coefficients are then used to compute the final output features for every node:

$$\boldsymbol{x}'_z = \sigma\left(\sum_{u \in \mathcal{N}_v} \alpha_{zv} W^s \boldsymbol{x}_v\right) \tag{11}$$

where $\sigma(\cdot)$ is applied component-wise. The GAT layer is able to dynamically learn the relationships between different channels and timestamps and abstract more meaningful information from the traffic channel of the slice.

C. Prediction Model

Finally, a two-layer fully connected neural network is used to obtain the final traffic prediction as follows:

$$\hat{d}_m = \mathrm{softmax}\left(\boldsymbol{x}'_z \psi + \boldsymbol{b}\right) \tag{12}$$

where ψ is a learnable matrix and $\boldsymbol{b}$ is the bais. We define a cross-entropy loss to train the model as:

$$\mathrm{Loss}_{\text{forecasting}}\,(t) = \frac{1}{T}\sum_{t=1}^{T}\left|d_m^{(t)} - \hat{d}_m^{(t)}\right| \tag{13}$$

where $d_m^{(t)}$ and $\hat{d}_m^{(t)}$ are the actual demand and the predicted demand of the slice m at timestamp T.

IV. Multi-Agent Federated Learning Dynamic Network Slicing

The limited communication capability of wireless networks makes DQN solutions unsuitable for large networks. We propose a multi-agent federated dynamic slicing algorithm to ensure privacy and scalability. The proposed algorithm can determine the near-optimal slicing policy that meets the QoS satisfaction requirements. According to the prediction result of the DT forecast, the dynamic slicing algorithm aims to maximize each slice's average resource utilization and users' satisfaction. Specifically, we model FL with multiple slices as an MDP and design a multi-agent federated reinforcement learning algorithm to explore the optimization solution for IoT slices.

A. Problem Transformation

To maximize the long-term return for the provider while accounting for the real-time arrivals of IoT network slices, we transform the problem into a Markov decision process (MDP) [6]. An MDP is defined by a tuple $\langle t, \mathcal{S}, \mathcal{A}, R\rangle$ where t is an decision epoch, $\mathcal{S}$ is the system's state space, $\mathcal{A}$ is the action space and R is the reward function. The state, action, and reward can be defined as follows.

a) State: At the beginning of each timeslot, the state of each slice is defines as $s_m(t) = \left[d_m(t), \hat{d}_m(t-1)\right]$, where $d_m^{(t)}$ and $\hat{d}_m^{(t)}$ are the actual demand and the predicted demand of the slice m.

b) Actions: The action taken by the slice m at time t is represented by $a_t^m = \{w_m\}$, where w_m represents the increase in the percentage of bandwidth for the m-th slice. When $w_m > 0, w_m$ percentage of bandwidth will be added to the rsource of the slice and when $a_m < 0, |a_m|$ percentage bandwidth of the slice will be reduced.

c) Reward: After the state transition, the each slice would gain rewards w.r.t. current state s_m^t and actions a_m^t. The reward function of the of the slice is defined as:

$$\mathcal{R}_m^t(s,a) = \left\{\Lambda \cdot \Omega_m^t(s,a) + \mu \cdot U_m(s,a)\right\} \tag{14}$$

where Λ and μ are the importance of the algorithm places on the resource utilization and utility, respectively [8]. The ultimate objective of the agent in the multi-agant FL system is to find the optimal slicing strategy (policy) π^*.

B. MAFL Network slicing

Network slices collaborate to find the optimal resource allocation in a decentralized manner. The role of the slice orchestrator is to coordinate all slices to achieve a global optimal resource allocation. In this allocation process, network slices do not need to share or exchange their proprietary information, including resource availability and traffic dynamics, with each other or with the slice orchestrator. the MAFL Network slicing, consist of 1) Local Policy Iteration 2) Global Policy Aggregation. The local policy iteration is performed by the slices and the central InP controller performs the for global policy aggregation. The aggregation period for the global model is referred to as τ.

1) Local Policy Iteration: Taking advantage of Deep Deterministic Policy Gradient (DDPG) [6], an actor's network guides the update of policy parameters θ_t of the network slcing policy π_{θ_l} based on the evaluated values from the critic's network. The policy gradient is defined as follows:

$$\nabla_{\theta\pi} \approx \frac{1}{n}\sum_m \nabla_a Q\left(s_m, a \mid \theta^Q\right) \nabla_{\theta^\pi}\pi\left(s_m \mid \theta^\pi\right). \tag{15}$$

We update the critic by minimizing the loss:

$$\mathcal{L}\left(\theta^Q\right) = \frac{1}{n}\sum_m \left(y_m - Q\left(s_m, a_m \mid \theta_Q\right)\right)^2 \tag{16}$$

where $y_m = \mathcal{R}_m + \gamma Q'\left(s_{i+1}, \pi'\left(s_{i+1} \mid \theta^{\pi'}\right)\right)$. The target parameters in both actor and critic networks are updated as follows:

$$\begin{aligned}\theta_i^{\pi'} &\leftarrow \nu\theta_i^{\pi} + (1-\nu)\theta_i^{\pi'} \\ \theta_i^{Q'} &\leftarrow \nu\theta_i^{Q} + (1-\nu)\theta_i^{Q'}\end{aligned} \tag{17}$$

In the t-th iteration, each slice agent performs model update via the stochastic gradient descent (SGD) algorithm based on its local data with the following expression:

$$\theta_t^{(m)} \leftarrow \theta_{t-1}^{(m)} - \eta\nabla f_m\left(\theta_{t-1}^{(m)}\right) \tag{18}$$

where η is the learning rate, and where $f_m(\theta) = \frac{1}{|D_m|}\sum \mathcal{L}_m\left(\theta^m\right)$ is the loss function for the agent m, $\theta_{t-1}^{(m)}$ is the local model of the slice at the start of the t-th iteration. The update local model $\theta_t^{(m)}$ is uploaded to the slice orchestrator for global model aggregation.

2) Global Policy Aggregation: The slice orchestrator updates the global network slicing model by aggregating all local model updates from IoT slices as follows:

$$\theta \leftarrow \sum_{m\in\mathcal{M}} \frac{|\mathcal{D}_m|}{|\mathcal{D}|}\theta_t^{(m)} \tag{19}$$

where θ denotes the aggregated model at the cloud server. After that, θ is broadcasted to the slices, which can be

Algorithm 1: Multi-Agent Federated Learning Dynamic Network Slicing

Initialize all slices with the same model, i.e., $\theta^{(m)} = \theta$;
for $t = 1, 2, \ldots, T$ **do**
  ▷ *Local Policy Iteration*;
  for *each slice* $m \in \mathcal{M}$ *in parallel* **do**
    Obtain the current observation $d, \hat{d}$ from the DT;
    Select a random action a_m^t with probability ϵ;
    Otherwise, select action a_m^t satisfying $a_t = {}_{a_t\in A}\,\text{argmax} Q\left(s_t, a_t\right)$;
    Update the resource fraction $\mathbf{w}_m$ and observe reward R_m and new state s^{t+1} ;
    Store experience $\left\langle s_m^t, a_m^t, \mathcal{R}_m, s^{t+1}\right\rangle$ and sample mini-batch $\mathcal{D}$;
    Update the local model as $\theta_t^{(m)}$ according to (18);
    Update $\theta_t^{(m)}$ as θ.;
    Update the main actor network using the sampled policy gradient according to (15) and target network based on (17)
  ▷ *Global Policy Aggregation*;
  if $\text{mod}\,(t, \tau) = 0$ **then**
    Collect the most updated model from the slices;
    Obtain θ by performing model aggregation according to (19);
    Broadcast θ to to all slices;

expressed $\theta_t^{(m)} \leftarrow \theta$. The global policy θ is used in the next training period. Our proposed algorithm, summarized in Algorithm 1, can find the optimal secure resource allocation strategy without sharing information and experience. The communication cost is analyzed as the total number of messages transmitted between the client and server. Considering the forward and backward propagation costs, the entire network requires $O\left(lk^2\right)$ communication messages for an iteration where l is the number of layers, and k is the number of neurons.

V. Numerical Evaluations

A. Simulation Environment Settings

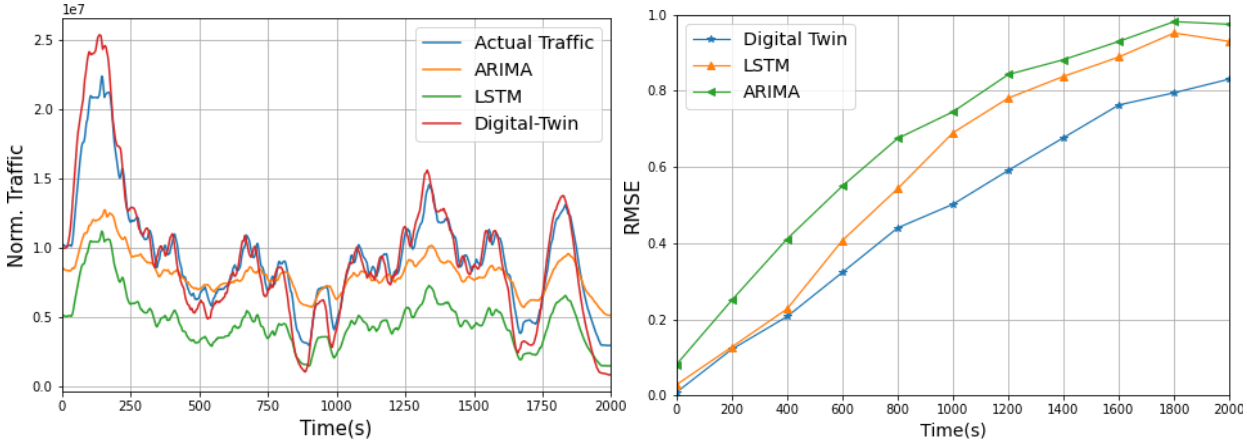
The system consists of a base station with a coverage radius of 500 meters and a carrier frequency of 2 GHz. The average number of IoT devices connected to a slice is 100, which are uniformly distributed. We set the number of slices $|\mathcal{M}|$ to 6. We set the bandwidth at 10 MHz and divide it into 50 RBs. The power spectral density of additive white Gaussian noise (AWGN) and interference thresholds are -174 dBm/Hz and -101.2 dBm, respectively. The Path Loss (PL) is defined as PL $(\mathrm{dB}) = 20\log_{10}(\mathbf{d}) + 20\log_{10}(f) - 27.55$ where f (in MHz) and $\mathbf{d}$ (in meters) represent IoT device to base station channel frequency and distance respectively [13]. Our DQN model consists of 6 hidden layers with 64 neurons in each hidden layer. We set the discount factor for both the actor and critic neural networks at 0.95 and the learning rate at 0.1. The replay buffer size is 1000, and the mini-batch size for sampling is 64. Furthermore, we adopt Adamoptimizer for optimizing the loss function [14]. TensorFlow 2.0 is used to implement the DQN, and FL [15]. We compare our proposed DT with two different prediction models, i.e., a vanilla Long short-term memory (LSTM) model [16] and ARIMA model [5]. Furthermore, our proposed dynamic slicing, i.e., DT-MAFL slicing, is compared with the following benchmarks: (1) Federated Learning-based slicing (FL), which performs resource allocation and sends updates to the slice orchestrator. (2) Multi-agent DQN (MADQN), which models the resource allocation problem as a distributed problem and deploys DQN agents on each slice to allocate resources independently [6] and (3) Netshare, which is a centralized controller with a global network view and performs resource allocation decisions for all slices [2]. The summary of the simulation parameters used can be found in Table II.

TABLE II: Simulation parameters

Parameters	Values
Cell radius	500m
Bandwidth	10 MHz
Number of RBs	50
Bandwidth of per RB	180kHz
The number of BSs	1
The number of slice tenants	6
Transmitting power	30dBm
Channel gain	path loss model
Noise Spectral Density	-174 dBm/Hz
User location	Uniform distribution
Mini-batch size	64
Experience replay buffer size	1000
Discount factor	0.5
Learning rate	0.1

B. Forecasting Performance

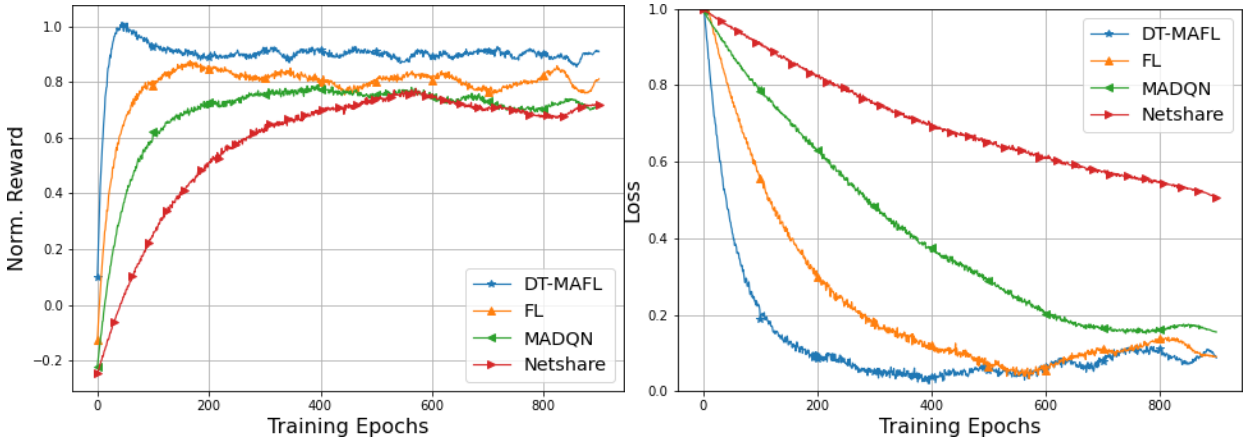
In Fig. 2, we show the qualitative comparison of the prediction in terms of the average traffic load (Mbs) of each slice on the BS over a period of time. The results show that the DT models can follow the wave of traffic relatively well. The LSTM and ARIMA models cannot follow it, with the ARIMA model having the worse pattern. The GAT network in our proposed DT model encodes spatial features in graph learning, which can track the spatial correlations in the 5G traffic data generated by the IoT devices in a slice. The ARIMA model increases linearly and leads to overfitting of the dataset as the number of data increases. In Fig. 2a, the Root Mean Square



(a) Demand forecast

(b) Prediction errors on RMSE

Fig. 2: *Forecasting Performance by various models.*



(a) Reward (b) Loss

Fig. 3: *Convergence performance.*

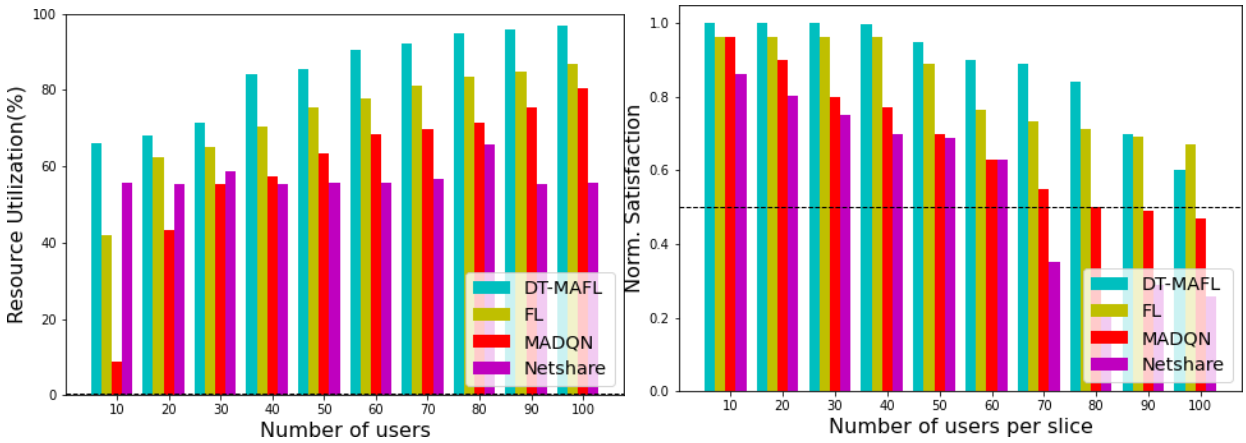
Error (RMSE) measures the performance of the predictive models. It is helpful to estimate which model carries the larger error point based on RMSE because it magnifies the relative difference between the errors. As a measure of accuracy, the RMSE is a good indicator. Our DT model methods achieve low error rates, highlighting the importance of modeling spatial correlations in traffic forecasting.

C. Convergence Performance

Fig. 3 illustrates the reward and loss results of our proposed dynamic slicing system. The reward is numerical feedback that evaluates the resource allocation of the proposed framework in equation (14). Fig. 3a shows that our proposed DT-MAFL slicing framework has the highest reward and achieves stability quickly. The FL model can achieve relatively good performance, but the resource allocation is unstable without an accurate prediction model. The MADQN converges slower than FL and DT-MAFL due to the communication overhead. At the same time, the optimal model cannot adapt to the dynamic conditions of the IoT devices in the slice. The loss curve further proves the convergence performance of the proposed scheme in Fig. 3b. The proposed DT-MAFL has the lowest loss, indicating stable convergence.

D. Performance on Slice Satisfaction and Resource Utilization

This section analyzes the system's performance regarding resource utilization and QoS satisfaction. In Fig. 4a, we compare the resource utilization of the algorithms with the increasing number of users in each slice. From Fig. 4a, it can be seen that the proposed DT-MAFL achieves higher resource allocation than FL, MADQN, and Netshare. With DT-MAFL, the prediction model can accurately predict the traffic load for the next time. Therefore, it can adapt to the dynamic changes of slice requests in a 5G network and increase resource utilization. As a result, the QoS requirements of the IoT devices in the slices are satisfied, as shown in Fig. 4b. Netshare is the most inefficient due to its lack of learning ability. This is because the QoS satisfaction values do not reach



(a) Resource Utilization (b) QoS Satisfaction

Fig. 4: *Resource Utilization and QoS satisfaction performance.*

the threshold when the number of users in the slice increases. In Fig. 4b, we can also see that the QoS satisfaction of FL and MADQN decreases rapidly when the number of network users increases. The QoS satisfaction of MADQN in high load scenarios also does not reach the QoS threshold of the slices.

VI. Conclusion

This paper examines the traffic characteristics of IoT network slices. We developed a DT based on GAT for real-time traffic analysis, monitoring, and demand forecasting of the slice. Based on the predictions, we propose a federated multi-agent reinforcement learning method for dynamic network slicing. The proposed allocation framework enables collaboration between slices while maintaining the privacy of the slices. We show through simulations that our proposed approaches can improve the accuracy of demand prediction for network slices and reduce the communication overhead of dynamic network slicing.

Acknowledgment

This work is supported in part by the Natural Science Foundation of China under Grant No.61806040, Grant No. 61771098, Grant No. 61971079, and Grant No. U21A20447; in part by the fund from the Department of Science and Technology of Sichuan Province under Grant No. 2020YFQ0025; in part by the fund from Intelligent Terminal Key Laboratory of Sichuan Province under Grant No. SCITLAB-1018, and by the Science and Technology Research Program of Chongqing Municipal Education Commission (KJZD-k202000604).

References

[1] D. Wu, Z. Zhang, S. Wu, J. Yang, and R. Wang, "Biologically inspired resource allocation for network slices in 5g-enabled internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9266–9279, 2019.

[2] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "Nvs: A substrate for virtualizing wireless resources in cellular networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, pp. 1333–1346, 2012.

[3] F. Zhang, G. Han, L. Liu, M. Martínez-García, and Y. Peng, "Joint optimization of cooperative edge caching and radio resource allocation in 5g-enabled massive iot networks," *IEEE Internet of Things Journal*, vol. 8, no. 18, pp. 14 156–14 170, 2021.

[4] M. Yan, G. Feng, J. Zhou, Y. Sun, and Y.-C. Liang, "Intelligent resource scheduling for 5g radio access network slicing," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 8, pp. 7691–7703, 2019.

[5] O. U. Akgul, I. Malanchini, and A. Capone, "Anticipatory resource allocation and trading in a sliced network," in *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7.

[6] Q. Liu, T. Han, N. Zhang, and Y. Wang, "Deepslicing: Deep reinforcement learning assisted resource allocation for network slicing," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–6.

[7] F. Pires, A. Cachada, J. Barbosa, A. P. Moreira, and P. Leitão, "Digital twin in industry 4.0: Technologies, applications and challenges," in *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, vol. 1, 2019, pp. 721–726.

[8] G. Sun, G. O. Boateng, D. Ayepah-Mensah, G. Liu, and J. Wei, "Autonomous resource slicing for virtualized vehicular networks with d2d communications based on deep reinforcement learning," *IEEE Systems Journal*, vol. 14, no. 4, pp. 4694–4705, 2020.

[9] C. T. Dinh, N. H. Tran, M. N. H. Nguyen, C. S. Hong, W. Bao, A. Y. Zomaya, and V. Gramoli, "Federated learning over wireless networks: Convergence analysis and resource allocation," *IEEE/ACM Transactions on Networking*, vol. 29, no. 1, pp. 398–409, 2021.

[10] S. Huang, B. Guo, and Y. Liu, "5g-oriented optical underlay network slicing technology and challenges," *IEEE Communications Magazine*, vol. 58, no. 2, pp. 13–19, 2020.

[11] Z. Wu, S. Pan, G. Long, J. Jiang, X. Chang, and C. Zhang, "Connecting the dots: Multivariate time series forecasting with graph neural networks." New York, NY, USA: Association for Computing Machinery, 2020, p. 753–763.

[12] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," *arXiv preprint arXiv:1710.10903*, 2017.

[13] M. Jiang, M. Condoluci, and T. Mahmoodi, "Network slicing management and prioritization in 5g mobile systems," in *European Wireless 2016; 22th European Wireless Conference*, 2016, pp. 1–6.

[14] J. Pomerat, A. Segev, and R. Datta, "On neural network activation functions and optimizers in relation to polynomial regression," in *2019 IEEE International Conference on Big Data (Big Data)*, 2019, pp. 6183–6185.

[15] M. Abadi, A. Agarwal, P. Barham, and E. Brevdo, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: https://www.tensorflow.org/

[16] J. Mei, X. Wang, and K. Zheng, "Intelligent network slicing for v2x services toward 5g," *IEEE Network*, vol. 33, no. 6, pp. 196–204, 2019.