Sparse CNN and Deep Reinforcement Learning-Based D2D Scheduling in UAV-Assisted Industrial IoT Networks

Van Dat Tuong, Wonjong Noh, *Senior Member, IEEE,* and Sungrae Cho, *Member, IEEE*

***Abstract*—Unmanned aerial vehicles (UAVs) have been widely applied in wireless communications because of its high flexibility and line-of-sight transmission. In this study, we develop low-complexity and robust device-to-device (D2D) link scheduling in UAV-assisted industrial Internet of Things (IIoT) networks. First, we propose a sparse convolutional neural network (SCNN) model that uses the geographical map of transmission links as input. The model consists of three main blocks: generic feature filtering, speed–accuracy balancing, and deep feature processing. Unlike other state-of-the-art methods, the proposed SCNN directly processes the geographical map collected using a connected UAV. Second, we propose a deep deterministic policy gradient-based reinforcement learning model that processes the output feature map from the SCNN to optimize the D2D scheduling decision and maximize the achievable system rate in the long run. Extensive simulations revealed that the proposed scheme significantly improved the achievable rate over other benchmark comparison schemes, such as transmitters and receivers (T&R) density-based deep learning (DL), ResNet-based DL, VGGNet-based DL, random scheduling, and all-active schemes, respectively. The simulations also demonstrated that the proposed scheme reduces computational complexity. With reduced complexity and nearly optimal performance, the proposed solution can be more efficiently applied to large-scale and dense IIoT networks.**

***Index Terms*—Deep deterministic policy gradient (DDPG)-based reinforcement learning, geographical map, sparse convolutional neural network (SCNN), unmanned aerial vehicle (UAV)-assisted device-to-device (D2D) scheduling, UAV-assisted industrial internet of things (IIoT) networks**

I. INTRODUCTION

WITH the advent of sixth-generation (6G) technology, the industrial Internet of Things (IIoT), a network of smart devices, group of machinery, or various connected sensors with the Internet, is becoming crucial. In IIoT networks, with numerous connections between IIoT devices operating in proximity, each device is connected to the wireless Internet, and the machine-type communication constitutes a major part of the IIoT [1]. Naturally, for 6G IIoT networks with the inherent characteristic of machine-type communications, device-to-device (D2D) communications in representative IIoT services [2], [3], such as content distribution, sensing- and monitoring-based control, cooperative carrying, process automation, and vehicle-to-everything (V2X), have become one of the key enablers [4]–[6]. The D2D communications provide several benefits, including cellular traffic offloading gain, a high data rate, low power consumption, low latency, and reuse gain when cellular and D2D links can use the same radio resources simultaneously [6].

One of the most pressing challenges of using D2D communications over IIoT networks involves determining D2D link scheduling that allows maximum information transmission while efficiently sharing a spectrum between the links within a given area. However, some problems must be addressed before this challenge can be taken up. First, the D2D resource scheduling in IIoT networks requires channel state information (CSI) for all links throughout the network. However, the signal and interference channel gains among D2D transmitters, their receivers, and neighboring users are difficult to acquire because of the significantly high cardinality of channel gains. Moreover, the task of exact estimation and collection of CSI are expensive and consume resources. The transmission of this CSI to the local or global control unit requires considerable power and control overhead. Second, the resource distribution and interference control between D2D transmissions demand high computational complexity. Although most existing D2D schedulers for IIoT networks are algorithmically different, they carry out numerous mathematical calculations to allocate resources using all the required information, such as the channel and interference state of the network, imposing considerable computational complexity. Hence, as the IIoT networks grow, these factors increasingly become critical issues.

Moreover, unmanned aerial vehicles (UAVs) have been widely applied in wireless communications because of the advantage of establishing stable and trusted line-of-sight air-to-ground links for ground users. With their high mobility and low cost, UAVs can quickly and effectively serve users as auxiliary means in remote areas [7]. UAV-assisted mobile edge computing schemes can also improve secure transmission [8]. Specifically, UAV deployments have found widespread use in various IIoT scenarios to handle dynamic changes and service requests effectively. Furthermore, many distributed algorithms have been proposed to solve these problems in D2D IIoT networks, which generally lack a centralized control unit. For example, [9] and [10] proposed approximation- and randomization-based distributed algorithms. However, the

This research was supported by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2023-RS-2022-00156353) supervised by the IITP (Institute for Information Communications) and the National Research Foundation of Korea (NRF) grant (NRF-2020R1F1A1069119). It was also supported by the Chung-Ang University Young Scientist Scholarship in 2021. *(Corresponding authors: Wonjong Noh and Sungrae Cho.)*

V. D. Tuong and S. Cho are with the School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, Republic of Korea. *(email: vdtuong@uclab.re.kr, srcho@cau.ac.kr)*

W. Noh is with the School of Software, Hallym University, Chuncheon 24252, Republic of Korea. *(email: wonjong.noh@hallym.ac.kr)*

available D2D schedulers still lack the simplicity required for practical implementation [11]. Therefore, in this study, we propose a fast- and low-complexity D2D IIoT scheduling scheme based on UAV-assisted system topology information to maximize the system throughput.

A. Related Work

In this subsection, we provide the related work on scheduling tasks in IIoT networks, which can be divided into two groups: conventional D2D scheduling research and machine learning-based D2D scheduling research.

1) Conventional D2D Scheduling in IIoT Networks: Existing approaches to achieve wireless link scheduling that maximizes system throughput are primarily based on non-convex combinatorial mathematical and greedy heuristic optimization. Wu *et al.* [12] proposed a synchronous distributed maximal scheduler, FlashLinQ, which measures the signal-to-interference ratio and determines distributed link scheduling through a yielding procedure of T&R. FlashLinQ provides an overall system architecture, including i) timing and frequency synchronization; ii) peer discovery; iii) link management; and iv) channel-aware distributed power, data rate, and link scheduling. However, the scheduler was based on the greedy heuristic search. Moreover, for neighbor discovery, the scheduler used single tone-based signaling, making it difficult to ensure the minimum information transfer. Naderializadeh *et al.* [13] suggested a new link-scheduling approach, information-theoretic link scheduling (ITLinQ), which first defined a new concept of information-theoretic independent sets (ITISs). These indicate a set of links capable of transmitting simultaneously and are information-theoretically optimal (within a constant gap) to deal with each other's interference as noise. The ITLinQ schedules these links, forming an ITIS at a given time. Shen *et al.* [14] designed a new approach called FPLinQ that coordinates link-scheduling decisions with power control among the interfering links. To determine the link scheduling and power allocation, FPLinQ uses a fractional programming approach. Compared with FlashLinQ and ITLinQ, FPLinQ does not require design parameter tuning. Zeng *et al.* [15] proposed a degrees-of-freedom based distributed link-scheduling method, maximizing the approximated system throughput. The proposed method uses the local CSI from neighboring nodes. However, the CSI is assumed to be accurate over a long channel coherence time. Ibrahim *et al.* [16] presented a distributed link-scheduling algorithm to minimize energy consumption under a throughput constraint. Users send simple control indicators to share the local CSI. The sharing of these CSI indicators could cause collisions. However, the distributed scheduling algorithm successfully identifies the optimal D2D links using certain collision reduction strategies for this local CSI sharing.

2) Machine Learning-Based D2D Scheduling in IIoT Networks: Previously, the mathematical optimization-based conventional techniques required accurate CSI estimation and high computational complexity. Machine learning -based wireless link-scheduling techniques have been proposed to overcome these issues. Neogi *et al.* [17] formulated a multi-player multi-armed bandit (MP-MAB) problem to solve the power and resource allocation problem for D2D users. The problem considers a realistic situation in which the channel gains are unknown at the base station (BS) and are referred to as partial CSI. Cui *et al.* [18] suggested D2D scheduling based on convolutional neural networks (CNNs). The model comprises a neural network architecture that facilitates the spatial learning of geographical locations of D2D nodes and achieves a throughput close to that currently achieved by the state-of-the-art algorithm without using explicit CSI. The results demonstrated that the location information could be efficiently exploited to obtain nearly optimal solutions for wireless networks. Lee *et al.* [19] designed graph-embedding-based link scheduling. A fully connected directed graph is built, where nodes and edges denote D2D pairs and interference links between D2D pairs. Then, the model determines a low-dimensional feature vector for each graph node. The graph-embedding procedures exploits only the distance information for communication and interference links without using the accurate CSI. Then, link scheduling is trained in a supervised manner using the result of the graph embedding. Jamshidiha *et al.* [20] proposed fully unsupervised framework-based link scheduling in a random network that consists of users with the same transmission powers. The proposed approach defines an interference graph and transforms it into a low-dimensional stochastic latent representation using a variational graph autoencoder, which is significantly more scalable than existing graph-based approaches. In addition, using a Gaussian mixture model, this approach clustered users in the representation domain and activated a random user from each cluster in each time slot. Liu *et al.* [21] suggested a novel link scheduling algorithm that jointly optimizes the system throughput and age of information. This study used graphic location information obtained from a global positioning satellite, from which the necessary CSI to schedule links is implicitly determined. The neural network can learn direct link scheduling from the graphic location information without estimating the CSI. In contrast to [18], the objective function of this approach cannot be represented explicitly; thus, directly determining its solution using gradient-based numerical approaches is difficult.

B. Motivations, Contributions, and Paper Organization

To the best of our knowledge, none of the existing work has considered a deep learning (DL) solution that efficiently exploits the architecture of the CNN in learning geographical information for the scheduling problem in wireless D2D networks. In this study, we propose a novel sparse CNN (SCNN)-based DL algorithm for wireless D2D transmissions. Table I summarizes the key features of the proposed approach and key differences between it and the existing approaches. Moreover, the main contributions of this study are summarized as follows.

- We developed a low-complexity and robust D2D link-scheduling scheme that maximizes the achievable system rate. This is accomplished using a novel neural network architecture that employs geographic information from the T&R and an implicit interference estimation among neighboring links. The geographical information

TABLE I
SUMMARY OF KEY FEATURES AND DIFFERENCES (MACHINE LEARNING-BASED DEVICE-TO-DEVICE (D2D) SCHEDULING)

Schemes	Objectives	Machine Learning techniques	Main outcomes/Features	Shortcomings
[17]	Maximize the cumulative sum rate of D2D users	Distributed learning with fairness	D2D users reuse resources of cellular users without hindering communications	No equilibrium is guaranteed for the scheme for multiple D2D users
[22]	Accurately approximate the WMMSE algorithm for the optimal power allocations	Supervised learning with a fully connected neural network	Real-time resource management and rigorous theoretical analysis	Computational complexity of using deep neural networks remains high
[18]	Maximize sum rate and fairness over wireless networks via link scheduling	Unsupervised learning with CNNs and fully connected neural networks	Exploit the location information of T&R instead of the perfect CSI	Computational complexity of using deep neural networks is not optimized
[19]	Maximize throughput of the overall D2D network via wireless link scheduling	Deep learning with KNN neighbor graph	Utilize the distances of communication and interference links instead of the accurate CSI	Need to determine the distances of communication and interference links
[20]	Maximize sum rate for a wireless network via link activation	Unsupervised learning with a KNN interference graph and variational graph autoencoder	User clustering helps reduce the complexity of deep neural networks	Only one user is allowed to be activated in each cluster
[21]	Optimize the age of information and throughput for D2D links	Deep learning with CNNs and fully connected neural networks	Learn mapping from the geographical location for optimal scheduling under a stationary randomized policy	Computational complexity of using deep neural networks remains high
Proposed	Optimize link scheduling to maximize the achievable rate for D2D links	SCNN-based deep deterministic policy gradient	Exploit geographical map input instead of the perfect CSI and optimize computational complexity of using deep neural networks	Effects on cellular users have not been investigated

is collected using a UAV, and instead of requiring expensive CSI, the inter-link interference is estimated as the discriminated feature extracted from the geographical map input using deep neural networks. This approach is advantageous in that it implicitly estimates the interlink interference and reduces communication overhead for updating the CSI.

- First, we propose an SCNN model that takes the geographical map of transmission links as input. The model has three main blocks: generic feature filtering (GFF), speed–accuracy balancing (SAB), and deep feature processing (DFP). In contrast to the state-of-the-art methods, the proposed method directly processes the geographical map of transmission links as input.
- Second, we propose a deep deterministic policy gradient (DDPG)-based reinforcement learning model that processes the output feature map of the SCNN to optimize the wireless scheduling policy and maximize the achievable system rate in the long run.
- Extensive simulations prove that the proposed approach significantly improves the achievable rate and reduces computational complexity. In particular, the proposed scheme provided a nearly optimal system rate and achieved 26.5%, 51.7%, 202.4%, 349.6%, 814.6%, and 1713.0% higher rates compared with the transmitters and receivers (T&R) density-based DL, ResNet-based DL, VGGNet-based DL, random, and all-active schemes, respectively. Furthermore, for deep neural network computations, the proposed approach used 21.9%, 131.3%, and 293.8% lower floating-point operations per second (FLOPS) compared with the density-based DL, ResNet-based DL, and VGGNet-based DL, respectively. These performance gains increase as IIoT networks become larger or denser.

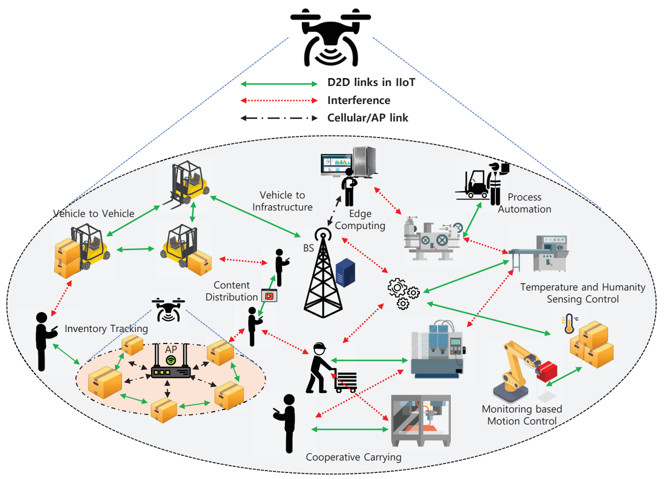


Fig. 1. System model: UAV-assisted D2D communications in an IIoT network.

The remainder of this paper is organized as follows. Section II describes the system model and problem formulation. Sections III and IV present the proposed low-complexity and robust link scheduling algorithm based on DL. Next, the performance evaluation is discussed in Section V. Finally, we present the conclusions in Section VI.

II. SYSTEM MODEL AND PROBLEM FORMULATION

A. Network Model

As illustrated in Fig. 1, we consider a single UAV-assisted D2D system, which is highly complex owing to multiple heterogeneous communication links. Within the scope of this work, we focus on D2D communications and do not consider cellular communications that may exist with aerial or ground BSs. Cellular communications can also be considered when the frequency resource is fully reused, and the difference in the altitude between BSs and user terminals is not significant. The network model consists of N independent D2D links with the corresponding T&R. We let $\mathcal{T} = \{1, \dots, N\}$ and $\mathcal{R} = \{1, \dots, N\}$ denote the sets of T&R, respectively. The transmitter–receiver distance lengths of all links are independent and identically distributed (i.i.d.) and follow a Gaussian distribution.

We consider a network model operating in a time-slot manner. In each time slot, the scheduling policy $\mathbf{x}$ is characterized by the binary variable $\{0, 1\}$ (i.e., $\mathbf{x} = \{x_i(t) | i = 1, \dots, N, x_i(t) \in \{0, 1\}\}$), where $x_i(t) = 1$ indicates that link i is scheduled, and $x_i(t) = 0$ indicates otherwise. Based on the advantage of flying objects, UAVs can serve ground terminals including D2D equipment, by readily establishing line-of-sight air-to-ground links. Thus, D2D equipment can obtain the scheduling decision through control signals sent from their serving UAVs. We let p_i denote the fixed transmission power of link i each time it is activated. Furthermore, $g_{i,j}(t) \in \mathbb{C}$ denotes the complex channel gain of receiver i from transmitter j, that is, $g_{i,i}(t)$ is the channel gain of link i from transmitter i to its corresponding receiver i. All transmission links are free to reuse frequency bandwidth W. As an important metric for evaluating network performance, the achievable rate $\Phi_i(t)$ of link i at time slot t is computed based on Shannon's formula [23] as follows:

$$\Phi_i(t) = W \log_2 \left(1 + \frac{|g_{i,i}|^2(t) p_i x_i(t)}{\sum\limits_{j \neq i} |g_{i,j}|^2(t) p_j x_j(t) + \sigma^2} \right), \quad (1)$$

where σ^2 is the additive white Gaussian noise power spectral density, assumed to be the same for all receivers.

B. Problem Formulation

Because of severe inter-link interference, the achievable system rate is poor if all transmission links are activated simultaneously. Therefore, the wireless scheduling problem in any given transmission period is determining the subset of links to be activated to maximize the system rate. This study aims to maximize the weighted sum rate of all users over the long term. The weighting coefficients are specified based on link preferences, which are assumed to remain unchanged during a specific period. The corresponding wireless scheduling problem is formulated as follows:

$$\max_{\mathbf{x}} \quad \sum_{t=0}^{T-1} \sum_{i=1}^{N} \gamma^t w_i \Phi_i(t), \quad (2)$$

$$\text{s.t.} \quad x_i(t) \in \{0, 1\}, \forall i = 1, \dots, N, \quad (2a)$$

$$\Phi_i(t) \geq (\Phi_i)_{\min}, \forall i = 1, \dots, N, \quad (2b)$$

where γ is a discounting factor and $w_i \in (0, 1)$ represents a weighting coefficient that reveals the preference of transmission links; that is, the more-preferred links are more likely to be scheduled. Constraint (2a) indicates the binary domain of the scheduling decisions. In constraint (2b), $(\Phi_i)_{\min}$ represents the minimum data rate that link i must achieve to guarantee the quality of service.

The formulated problem is a discrete optimization that is difficult to solve owing to a complicated scheduling policy. It may derive numerous interactions between adjacent links, making the interlink interference unpredictable. The conventional optimization approach often requires perfect CSI to solve this problem, which is impractical for large-scale wireless D2D networks. Another approach is to use machine learning-based optimization, which has the advantage of a rapid decision-making time. Several practical problems remain when considering the state-of-the-art studies [22] and [18], which proposed DL-based solutions for the wireless scheduling problem. First, as [22] used a fully connected neural network that directly takes CSI as input, the overall complexity grows rapidly with the number of transmission links. Second, although [18] addressed the previous problem by processing the geographical map of transmission links as input, direct CSI was still used in the training process. In addition, [18] increased the computational cost because the original geographical map had to be transformed into T&R density grids before inputting it into the deep neural network. The following section introduces the architecture of SCNNs that help process geographical maps to extract discriminative features efficiently and obtain the optimal wireless scheduling policy using an SCNN-based DL framework.

III. Sparse Convolutional Neural Networks

A. Fundamentals of Convolutional Neural Networks

Generally, a CNN is constructed from multiple convolutional, pooling, and fully connected layers. The convolutional process involves a convolutional kernel and input map. Particularly, the feature map is obtained by summing the products of the kernel weights and their corresponding input map values over spatial locations. We let $\upsilon_{x,y}$ denote the convolutional result at the spatial location (x, y). Similar to [24], $\upsilon_{x,y}$ can be mathematically computed as

$$\upsilon_{x,y} = \sum_{i=1}^{a} \sum_{j=1}^{b} \omega_{i,j} \mathbf{X}_{x+i,y+j} + \varpi, \quad (3)$$

where $(a \times b)$ represents the kernel size, $\omega_{i,j}$ denotes the kernel weight at location (i, j), $\mathbf{X}_{x+i,y+j}$ denotes the matching input map value, and ϖ indicates a bias value added to the convolutional result. Subsequently, $\upsilon_{x,y}$ can be applied to the activation functions to obtain the class label. Several activation functions are frequently used in CNNs for fast training and high accuracy: sigmoids, tangents, and rectified linear units (ReLUs) and their variants, such as the randomized ReLU, parametric ReLU, clipped ReLU, and leaky ReLU.

Grouping convolution was proposed as an improvement that allows a deep CNN architecture to work over multiple memory-constrained GPUs in ResNeXt [25], Deep Roots [26], and ShuffleNet [27]. Grouping convolution demonstrates higher classification accuracy and lower complexity than conventional CNN architectures. A convolutional operation is performed in each group by dividing the depth of the input map of multiple layouts into multiple groups to derive

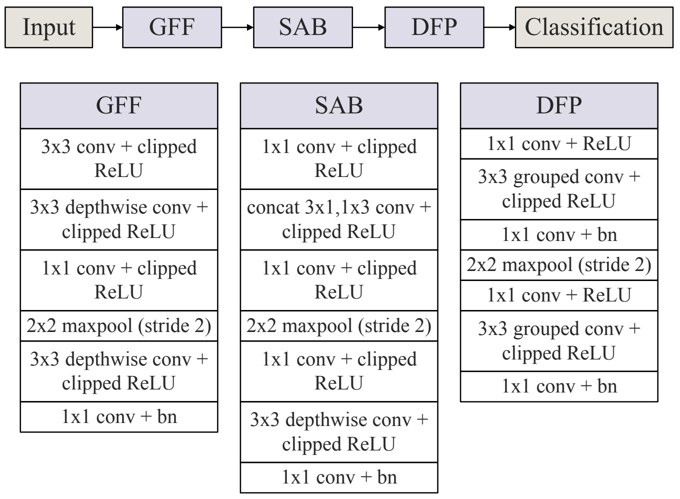


Fig. 2. Illustration of sparse convolutional architecture.

several unconstrained feature maps. Subsequently, the overall feature map was obtained by concatenating these independent feature maps in a grouping convolutional layer. In particular, the filter kernel for all groups must be the same size. The number of filters in each group can vary depending on the neural network architecture. In the simplest scenario, the grouping convolutional layer has a single filter in each group is called the depthwise convolutional layer, which enhances the performance of a CNN in resource-constrained devices [28]. Considering that there are Ω filter groups, according to [24] and [29], the grouping convolution operation can be mathematically expressed as follows:

$$\upsilon_{x,y} = \upsilon_{x,y,1} \oplus \ldots \oplus \upsilon_{x,y,\phi} \oplus \ldots \oplus \upsilon_{x,y,\Omega}, \tag{4}$$

where $\phi \in [1, \Omega]$, $\oplus$ denotes the concatenation operation, and $\upsilon_{x,y,\phi} = \sum_{i=1}^{a} \sum_{j=1}^{b} \omega_{i,j,\phi} \mathbf{X}_{x+i,y+j}$ represents the convolutional result of group ϕ at spatial location (x, y) with kernel weight $\omega_{i,j,\phi}$.

B. Sparse Convolutional Blocks

To achieve a high-speed and accurate classification, we designed the CNN architecture based on three main blocks: GFF, SAB, and DFP, as illustrated in Fig. 2. The input map is sequentially processed in GFF, SAB, and DFP blocks. First, generic features are extracted using the GFF block. Then, the output from the GFF block is processed to balance the speed and accuracy in the SAB block. Subsequently, the output from the SAB block is subjected to a multilayer transition before arriving at the DFP block, where useful deep and distinct features are extracted to improve the classification accuracy.

The GFF block directly processes the input map using two depthwise convolutional layers, two pointwise convolutional layers, one regular convolutional layer, and one max-pooling layer. First, the generic features of the input map are extracted using a 3×3 kernel. Then, the features are processed with a 3×3 depthwise kernel for feature map compression. After a pointwise convolutional stage, a 2×2 max-pooling layer with a stride of (1,2) is deployed to reduce the computational cost in the subsequent layers. Finally, a 3×3 depthwise kernel followed by a pointwise convolutional layer with batch normalization is applied to further filter the low-cost features. Batch normalization tends to reduce the internal covariate shift. In the GFF block, the clipped ReLU is used for all convolutional layers, except for the last pointwise layer, as a low-cost activation function.

The SAB block handles the output feature map of the GFF block to balance the speed–accuracy tradeoff. First, a pointwise convolutional layer is deployed to expand the feature map for better feature extraction. Next, grouping convolutional layers with asymmetrical kernels, 3×1 and 1×3, are employed to generate independent feature maps that are combined and then processed with a pointwise convolutional layer. Afterward, the feature map is connected to a 2×2 max-pooling layer with a stride of (1,2) for computational reduction. The feature map is expanded again with a pointwise convolutional layer. Finally, a 3×3 depthwise kernel and a pointwise convolutional layer with batch normalization are applied to extract discriminative features. Similar to the GFF block, the clipped ReLU is applied for all convolutional layers except the last pointwise layer. In the SAB block, although grouping and depthwise convolutional layers help extract more discriminative features to improve the classification accuracy, the deployed pointwise convolutional layers slightly increase the computational cost. Moreover, depthwise convolutional kernels with many free parameters may lead to suboptimal performance when used in deep layers [30]. Therefore, a DFP block must be implemented to process the feature map efficiently in deep layers.

To reduce the computational cost of the convolutional operations, we transform the feature map before it goes to the DFP block using multilayer transition. First, the feature map is processed using a pointwise convolutional layer and a 3×3 depthwise kernel. Afterward, another pointwise convolutional layer and a 2×2 max-pooling layer with a stride of (1,2) are employed to reduce the dimension of the feature map for complexity reduction. The feature map arrives at the DFP block, where it is processed using the grouping convolutional layers. After extracting features with a pointwise convolutional layer, more features are independently generated using 3×3 grouping convolutional layers. These features are linearly combined using a pointwise convolutional layer. Subsequently, a 2×2 max-pooling layer with a stride of (1,2) is applied to reduce the computational complexity. The final output feature map is delivered with more discriminative features using two pointwise and 3×3 grouping convolutional layers. It can be observed that deeper DFP block layers result in more extracted discriminative features.

IV. SCNN-Based Deep Learning for Optimal Wireless Scheduling

A. Efficient Feature Extraction

By employing high-resolution and high-speed cameras, UAVs can regularly capture and collect geographical information for network coverage areas, containing the location of T&R of all D2D links. For simplicity, we assume that the geographical map is ready to be used at the UAVs, where each geographical map is a grid image containing node pairs of transmitters and their corresponding receivers. Using the geographical map as input, a feature map can be achieved using convolutional layers connected for classification. Thus, we can accurately specify the optimal scheduling decision using only the geographical map input. Generally, deploying deep hierarchical network layers improves classification accuracy because more layers help extract discriminative features more efficiently [31]. However, it also increases network complexity and operational costs.

The proposed algorithm employs SCNNs for low-complexity and robust feature extraction. The network judiciously performs volume convolution that combines two types of convolutional layers, depthwise and regular, to achieve higher classification accuracy while making the network more lightweight. The proposed network leverages SCNNs with three primary blocks: GFF, SAB, and DFP. Depthwise convolutional layers are deployed in the SAB block to enhance the relevant feature output of the GFF block and balance the accuracy and time complexity. Furthermore, the DFP block employs a cascade structure of regular convolutional layers to mine more discriminative features from the SAB block. The aim is to prevent the loss of essential details in the geographical map. In addition, the problem of interblock feature loss can be addressed while allowing inter-block feature sharing by deploying skip connections between the subblocks of SAB and DFP.

B. Wireless Scheduling Decision Training

The output feature map can determine the optimal wireless scheduling decision using a fully connected network with numerous parameters for accurate classification. To optimize the computational complexity, we developed a DDPG-based model that properly learns the optimal wireless scheduling policy. Notably, it is difficult to determine a reinforcement learning-based model that delivers the best performance. We select a DDPG-based model for the following reasons. First, DDPG is an off-policy algorithm that tends to be more cost-effective than other on-policy models, such as Proximal Policy Optimization and SARSA, regarding deployment in real-world scenarios. Indeed, DDPG encourages the agent to explore more in the action space. Second, DDPG is a policy gradient method that can significantly reduce unnecessary explorations of a deep Q-Network model. Thus, it is more appropriate for large IIoT systems, where the scheduling problem for multiple D2D links is highly complex. A reinforcement learning agent is employed at the UAV to iteratively interact with the network environment and collect training data regarding experiences. The communication system is modeled as a Markov decision process model, where the optimal scheduling policy is trained toward maximizing the long-term reward (e.g., achievable rate). As illustrated in Fig. 3, the detailed implementation consists of a D2D network with N links, a replay memory $\mathbb{M}$ with size S, and deep neural networks including SCNN layers, primary and target actor networks, $\mathbb{A}$ and $\mathbb{A}^{'}$, with respective weights, $\theta_{\mathbb{A}}$ and $\theta_{\mathbb{A}'}$, (initially $\theta_{\mathbb{A}} = \theta_{\mathbb{A}'}$), and primary and target critic networks, $\mathbb{C}$ and $\mathbb{C}^{'}$, with respective weights, $\theta_{\mathbb{C}}$ and $\theta_{\mathbb{C}'}$, (initially $\theta_{\mathbb{C}} = \theta_{\mathbb{C}'}$). The algorithm is coordinated by the UAV agent, which enables control of the wireless scheduling action of all D2D links. The replay buffer is initialized as empty and stores the maximum S experiences of the agent. When the buffer runs out of memory, the least recent experience is replaced by the most recent experience. Additionally, the weights of the primary actor and critic networks, $\theta_{\mathbb{A}}$ and $\theta_{\mathbb{C}}$, are randomly initialized and copied to those of the target actor and critic networks, $\theta_{\mathbb{A}'}$ and $\theta_{\mathbb{C}'}$, respectively.

Because changes in the topology of the geographical map can frequently occur, the training process should be conducted for all geographical map inputs to ensure the high performance of the scheduling output. In this study, we assume that all changes are captured in the input data and available for training. Furthermore, the SCNN model can regularly continue training with newly captured geographical map input if detected. The training process comprises multiple epochs, each including T time steps. At the beginning of each epoch, the geographical map input of D2D links is randomly initialized. Notably, the geographical map of the transmission links is preprocessed by sparse convolutional layers to extract the feature map before sending it to the actor and critic networks. At time t, the feature map is inputted into the actor and critic networks as state S_t. Then, the UAV agent selects a scheduling decision for all D2D links, A_t, based on the output of the primary actor network, $\mathbb{A}(S_t;\theta_{\mathbb{A}})$, and ϵ-greedy strategy that promotes exploration first (random A_t) and gradually enhances exploitation ($A_t = \mathbb{A}(S_t;\theta_{\mathbb{A}})$) to improve the training speed and convergence. The exploration probability decayed at a rate of 0.9999. For exploitation, additional noise is used to restrict the sub-optimal scheduling action such that the scheduling action is obtained as $A_t = \mathbb{A}(S_t;\theta_{\mathbb{A}}) + \varphi_t$, where φ_t could be Gaussian white noise or Ornstein–Uhlenbeck noise [32]. When the selected scheduling action is applied, the UAV agent receives a reward, R_t, and the system evolves to the next state, S_{t+1}. Based on this observation, the step reward was calculated in terms of the achievable sum rate as $R_t = \sum_{i=1}^{N} w_i \Phi_i(t)$. Additionally, experience tuples $\langle S_t, A_t, R_t, S_{t+1}\rangle$ are saved in the replay buffer, from which random mini-batches of experiences are sampled to update the weights of the actor and critic networks. n particular, a loss function is formulated between the estimated and target Q-values as the output of the primary and target critic networks [33] as follows:

$$L(\theta_{\mathbb{C}}) = \frac{1}{\Lambda}\sum_{i=1}^{\Lambda}\left(R_{t,i}+\gamma\mathbb{C}^{'}\left(S_{t+1,i}, A_{t+1,i};\theta_{\mathbb{C}'}\right)-\mathbb{C}(S_{t,i}, A_{t,i};\theta_{\mathbb{C}})\right)^2, \tag{5}$$

where Λ denotes the minibatch size, and $\langle S_{t,i}, A_{t,i}, R_{t,i}, S_{t+1,i}\rangle$ represents the ith experience sample collected at time t. To minimize the loss, we updated the weight coefficients of the primary critic network using the Adam optimizer [34]. Similar to [32], the policy gradient is applied to update the weight coefficients of the primary actor network, as follows:

$$\begin{aligned}\nabla_{\theta_{\mathbb{A}}} &= \mathbb{E}\left[\nabla_{\theta_{\mathbb{A}}}\mathbb{C}(S_t, A_t;\theta_{\mathbb{C}})\right]\\ &= \mathbb{E}\left[\nabla_{A_t}\mathbb{C}(S_t, A_t;\theta_{\mathbb{C}}) \times \nabla_{\theta_{\mathbb{A}}}\mathbb{A}(S_t;\theta_{\mathbb{A}})\right].\end{aligned} \tag{6}$$

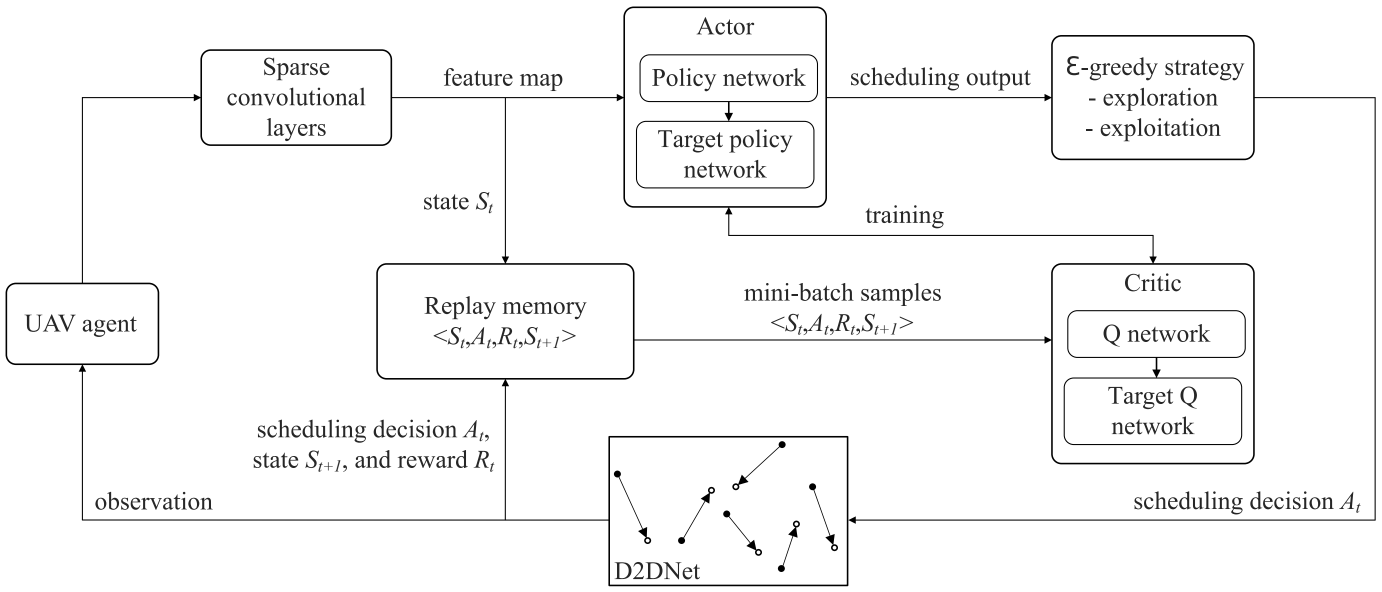

Sparse convolutional layers
feature map
Actor
Policy network
Target policy network
scheduling output
Ɛ-greedy strategy - exploration - exploitation
state S_t
training
UAV agent
Replay memory $<S_t,A_t,R_t,S_{t+1}>$
mini-batch samples $<S_t,A_t,R_t,S_{t+1}>$
Critic
Q network
Target Q network
scheduling decision A_t, state S_{t+1}, and reward R_t
observation
D2DNet
scheduling decision A_t

Fig. 3. Deep learning algorithm for training the optimal scheduling policy

For stability, the weight coefficients of the target networks are gradually updated after every G steps based on a low learning rate, η, as $\theta_{\mathbb{A}'} = \eta\theta_{\mathbb{A}} + (1-\eta)\theta_{\mathbb{A}'}$ and $\theta_{\mathbb{C}'} = \eta\theta_{\mathbb{C}} + (1-\eta)\theta_{\mathbb{C}'}$.

V. Performance Evaluation

A. Simulation Setup

Owing to the limitations of collecting data in a real-world smart industry environment, we investigated and verified the performance of the proposed scheme using simulations, where an IIoT network model was generated based on practical settings and the D2D link lengths were uniformly distributed between 10 and 100 m. Specifically, we consider a D2D network in which the locations of all transmitters are uniformly distributed within a circular area with a radius of r = 500 m. The locations of the receivers are distributed within the area and satisfy the distance constraint from the corresponding transmitters. The UAV is assumed to have a projection at the center point. By learning the optimal scheduling policy based on the geographical map of transmission links, the UAV plays a vital role in this network model, in which all transmission links observe the scheduling decision. Some parameters were derived from a state-of-the-art study [18] to build the network model. For example, network settings comprised a channel bandwidth of 5 MHz at a 2.4 GHz frequency, transmit power of 40 dBm, and background noise of -169 dBm/Hz. The maximum number of epochs for the training process was 1000, with 200 steps each. The learning rate was initially set at 0.01 and gradually reduced to obtain better convergence. All simulations were conducted using a server powered by an Intel Core i7-10700 2.90 GHz CPU and Nvidia GTX 1650 GPU, with a memory of 1 TB. To evaluate the performance of the proposed solution, we implemented the existing schemes for comparison.

- *T&R density-based DL [18]*: This scheme trained the wireless scheduling decision using the T&R density grid maps constructed from the geographical map of the transmission links.
- *VGGNet-based DL*: This scheme trained the wireless scheduling decision using VGG convolutional networks [35] instead of the proposed SCNNs.
- *ResNet-based DL*: This scheme trained the wireless scheduling decision using residual convolutional networks [36] instead of the proposed SCNNs.
- *Random*: The D2D links were randomly scheduled.
- *All-active*: All D2D links were activated.

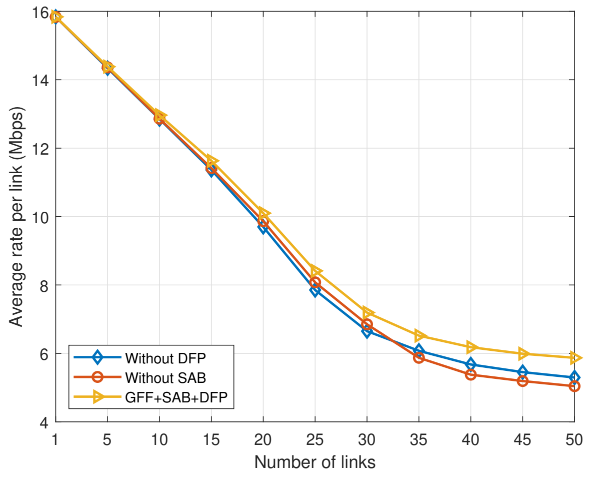


Fig. 4. Influence of the main blocks on the achievable rate.

B. Data Rate Performance

Fig. 4 presents the achievable rate when replacing the main blocks (e.g., SAB and DFP) with their equivalent regular convolutional layers. We consider the achievable rate per link, which is the average of 200 consecutive steps in one epoch.

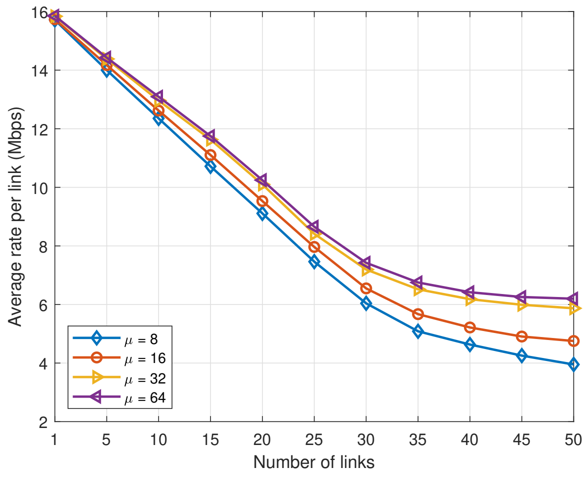


Fig. 5. Influence of the convolutional layer group sizes on the achievable rate.

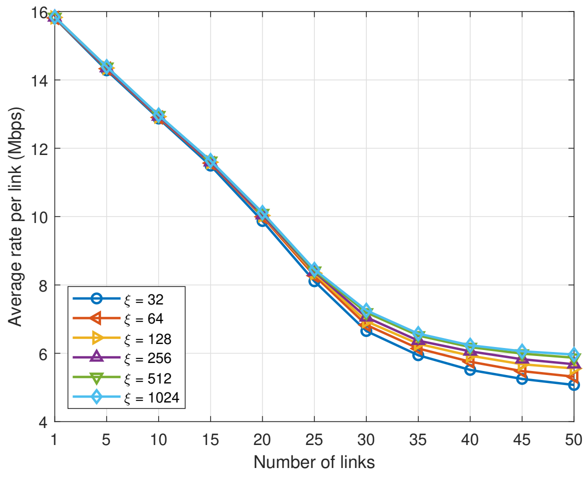


Fig. 6. Infuence of the frame sizes on the achievable rate.

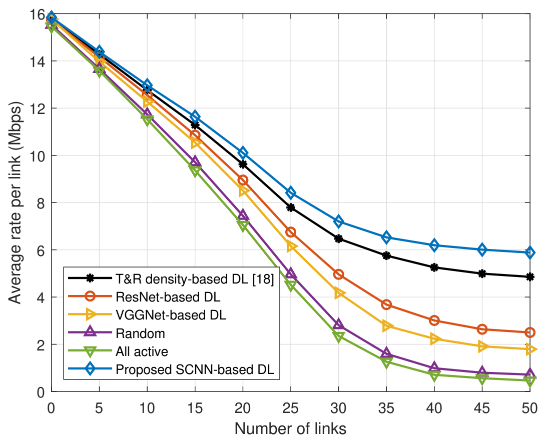
Moreover, the GFF block was not considered because of its generic feature processing, which resulted in an insignificant contribution to the overall performance. As observed in the figure, when the number of links increases, the achievable rate per link decreases in all settings. Specifically, for the proposed scheme, which combines GFF, SAB, and DFP blocks, the achievable rate drops from 15.84 to 7.19 and 5.87 Mbps when the number of links increases from 1 to 30 and 50, respectively. This outcome can be explained by interlink interference, which is more severe when the link distribution is denser. More importantly, when replacing the SAB and DFP blocks, the achievable rate decreased faster. Starting from a similar achievable rate of 15.84 Mbps for the one-link setting in all architectures, when the number of links is 30, the achievable rate per link is 6.65 and 6.85 Mbps, respectively, in architectures with SAB only and DFP only, 8.17% and 5.01% lower than that in the architecture with both SAB and DFP. When the number of links is 50, the rate is 5.29 and 5.04 Mbps in architectures with SAB only and DFP only, respectively, 10.96% and 16.47% lower than that in the architecture with both SAB and DFP.

Fig. 5 presents the achievable rate for various convolutional layer group sizes, μ. It is observed that the achievable rate decreased with an increase in the number of groups. When the number of links is 30, the average rate per link with $\mu = 64$ is 7.42 Mbps, 3.2%, 13.3%, and 22.8% greater than rates with $\mu = 32$, $\mu = 16$, and $\mu = 8$, respectively. Similarly, when the number of links is 50, the rate is 6.20 Mbps with $\mu = 64$, 5.62%, 30.53%, and 56.96% greater than rates with $\mu = 32$, $\mu = 16$, and $\mu = 8$, respectively. We can see that increasing the number of groups from 32 to 64 increases the achievable rate by approximately 6%; however, this requires more trainable weights. Therefore, the number of groups that can maintain the performance–complexity tradeoff in practical scenarios must be carefully selected. Within the scope of this research, we used a group size of $\mu = 32$.

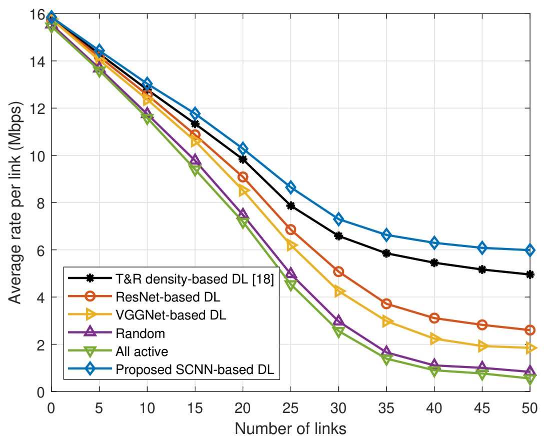
Fig. 6 plots the achievable rate for various frame sizes, ξ. It is observed that a greater frame size provides a better achievable rate. However, this gap is not apparent when the number of links is small. For example, when the number of links is less than 20, the average rate per link is similar for various frame sizes, such as $\xi = 32, \dots, 1024$. When the number of links is 30, the achievable rate with $\xi = 1024$ is 7.256 Mbps, 0.86%, 2.85%, 4.59%, 6.24%, and 9.08% greater than rates with $\xi = 512$, $\xi = 256$, $\xi = 128$, $\xi = 64$, and $\xi = 32$, respectively. For a greater number of links, $N = 50$, the achievable rate with $\xi = 1024$ reaches 5.963 Mbps, 1.58%, 4.98%, 7.33%, 12.19%, and 17.52% better than rates with $\xi = 512$, $\xi = 256$, $\xi = 128$, $\xi = 64$, and $\xi = 32$, respectively. This frame size effect is because the larger input frame sizes result in a higher resolution that helps extract more distinct features and yields better performance. Despite the better performance, larger input frame sizes increase the computational and memory costs. Therefore, the input frame size should be judiciously selected to balance the performance–complexity tradeoff. In this study, we used a frame size of $\xi = 512$.

We conducted a performance comparison between the proposed solution and state-of-the-art schemes in terms of the achievable data rate. All schemes considered the same 2000 layouts for each setting of the link length distribution (e.g., independent lengths between 10 and 100 m), and the lengths were fixed at 50 m to ensure accurate comparisons.

Fig. 7-(a) and (b) present the achievable rates corresponding to different link length settings by varying the number of links. Generally, in both settings, the average rate per link decreases when the number of links increases because the numerous links generate severe interlink interference. Moreover, compared with the density-based DL, ResNet-based DL, VGGNet-based DL, random, and all-active schemes, the proposed solution significantly improves the achievable rate. For instance, in Fig. 7-(a), when 50 links exist, and the link lengths are distributed between 10 and 100 m, the proposed scheme achieved the average rate per link of 5.88 Mbps, 21.0%, 135.3%, 228.1%, 722.1%, and 1165.9% better compared with density-based DL, ResNet-based DL, VGGNet-based DL, random, and all-active schemes, respectively. Fig. 7-



(a) Link length distribution between 10 and 100 m.



(b) Link lengths fixed at 50 m.

Fig. 7. Achievable rate versus the number of links.

(b) illustrates the rate comparison when all link lengths are fixed at 50 m, and the achievable rate is similar to that when the link length distribution is between 10 and 100 m.

Fig. 8-(a) and (b) plot the achievable rates corresponding to different link length settings by varying the network coverage sizes (i.e., the network coverage radius r). For both settings, the achievable rate decreases when the network coverage area increases. In Fig. 8-(a), when the link lengths are distributed between 10 and 100 m and the network coverage radius is 100 m, the achievable rate reaches approximately 6.93, 6.66, 6.35, 6.25, 4.39, and 4.17 Mbps in the proposed, density-based DL, ResNet-based DL, VGGNet-based DL, random, and all-active schemes, respectively. In comparison, under the same link length distribution when the network coverage radius is 1000 m, the achievable rate is degraded by 26.5%, 51.7%, 202.4%, 349.6%, 814.6%, and 1713.0% in the proposed, density-based DL, ResNet-based DL, VGGNet-based DL, random, and all-active schemes, respectively. A similar decrease is observed in Fig.8-(b) when all link lengths are fixed at 50 m. The reason for this rate reduction trend is the severe inter-link interference that generates a stronger effect on the larger network while maintaining the link density.

C. Computational Complexity Analysis

To evaluate the performance of the proposed solution further, we estimated and compared the complexity of DL schemes in terms of FLOPS and inference time, as listed in Table II. Based on the recorded inference time, the proposed solution provides a shorter inference time than the ResNet-based DL and VGG-based DL schemes. Compared with the density-based DL scheme, the inference time is slightly longer (i.e., 0.138 compared to 0.135 ms). However, the proposed solution requires less computational complexity than the density-based DL scheme (i.e., 32 MFLOPS in the proposed scheme compared to 39 MFLOPS in the density-based DL scheme). This result is because the proposed training model directly processes the geographical map and delivers the scheduling decision for all D2D links, whereas the density-based DL scheme separates training for each link based on the processed data of the T&R grids. Thus, the proposed scheme saves more FLOPS to complete the training process; however, it slightly increases the inference time to deliver the scheduling decision.

TABLE II
COMPUTATIONAL COMPLEXITY OF DEEP LEARNING SCHEMES
(COVERAGE SIZE: 500 M, N = 50, LINK LENGTHS: 10 TO 100 M)

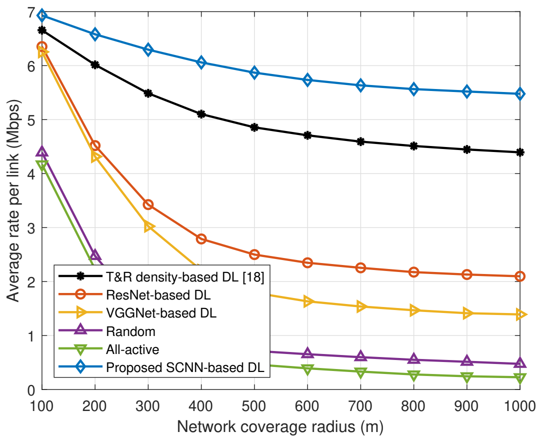
Scheme/Approach	Proposed	Density-based	ResNet-based	VGG-based
Inference time (ms)	0.138	0.135	0.156	0.143
FLOPS (MFLOPS)	32	39	74	126

VI. CONCLUSIONS

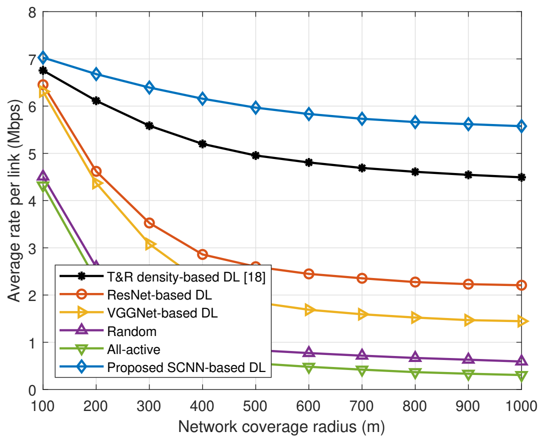
We investigated the wireless scheduling problem in D2D networks. The proposed solution considers only the geographical map of the transmission links, which can be collected using a connected UAV. This new approach is advantageous because it implicitly estimates the inter-link interference and reduces the communication overhead for updating the CSI. We first developed an SCNN-based novel DL solution that efficiently and effectively extracts distinct features from the input map. Subsequently, we developed a DDPG-based reinforcement learning algorithm that optimizes the wireless scheduling decision and maximizes the achievable rate in the long run using the output feature map from the SCNN. Extensive simulations revealed that the proposed solution provided significantly enhanced performance in terms of the achievable rate and computational complexity compared with the density-based DL, ResNet-based DL, VGGNet-based DL, random, and all-active schemes. Furthermore, the performance gains using the proposed scheme increase as the link coverage or density increases; thus, the proposed scheme can be efficiently applied to large-scale and dense IIoT networks.

REFERENCES

[1] S. S. Sarma, R. Hazra, and A. Mukherjee, "Symbiosis between D2D communication and industrial IoT for industry 5.0 in 5G mm-wave cellular network: An interference management approach," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 8, pp. 5527–5536, 2022.



(a) Link length distribution between 10 and 100 m.



(b) Link lengths fixed at 50 m.

Fig. 8. Achievable rate versus network coverage sizes while maintaining the link density.

[2] P. Gupta, S. Fong, X. Wang, S. ElAzzouni, R. Prakash, F. Ulupinar, and X. Zhang, "Enhancing 5G URLLC for industrial IoT with device-to-device communication," in *2021 17th International Symposium on Wireless Communication Systems (ISWCS)*, 2021, pp. 1–6.

[3] F. Jameel, M. U. Sheikh, R. Jäntti, M. I. Ashraf, and J. Torsner, "Efficient mode selection for D2D communication in industrial IoT networks," in *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*, 2020, pp. 1–6.

[4] W. Wang, C. Fang, L. T. Yang, H. Duan, and P. Xu, "A fairness-based collaborative communication ecosystem over sustainable D2D fogs in a 5G industrial IoT," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 11, pp. 7860–7870, 2021.

[5] J. Tang, H. Tang, X. Zhang, K. Cumanan, G. Chen, K.-K. Wong, and J. A. Chambers, "Energy minimization in D2D-assisted cache-enabled Internet of Things: A deep reinforcement learning approach," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 8, pp. 5412–5423, 2020.

[6] U. Saleem, S. Jangsher, H. K. Qureshi, and S. A. Hassan, "Joint subcarrier and power allocation in the energy-harvesting-aided D2D communication," *IEEE Transactions on Industrial Informatics*, vol. 14, no. 6, pp. 2608–2617, 2018.

[7] W. Lu, Y. Ding, Y. Gao, Y. Chen, N. Zhao, Z. Ding, and A. Nallanathan, "Secure NOMA-based UAV-MEC network towards a flying eavesdropper," *IEEE Transaction on Communications*, vol. 70, no. 5, pp. 3364–3376, 2022.

[8] W. Lu, Y. Mo, Y. Feng, Y. Gao, N. Zhao, Y. Wu, and A. Nallanathan, "Secure transmission for multi-UAV-assisted mobile edge computing based on reinforcement learning," *IEEE Transactions on Network Science and Engineering*, 2022. [Online]. Available: http://dx.doi.org/10.1109/TNSE.2022.3185130

[9] P. Chaporkar, K. Kar, X. Luo, and S. Sarkar, "Throughput and fairness guarantees through maximal scheduling in wireless networks," *IEEE Transactions on Information Theory*, vol. 54, no. 2, pp. 572–594, 2008.

[10] B. Li and A. Eryilmaz, "Exploring the throughput boundaries of randomized schedulers in wireless networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 4, pp. 1112–1124, 2012.

[11] J. Fayek, M. Aoude, M. Raad, and R. Raad, "Device-to-Device communication in 5G: Towards efficient scheduling," *International Journal of Digital Information and Wireless Communications*, vol. 8, no. 3, pp. 144–149, 2018.

[12] X. Wu, S. Tavildar, S. Shakkottai, T. Richardson, J. Li, R. Laroia, and A. Jovicic, "FlashLinQ: A synchronous distributed scheduler for peer-to-peer ad hoc networks," *IEEE/ACM Transactions on Networking*, vol. 21, no. 4, pp. 1215–1228, 2013.

[13] N. Naderializadeh and A. S. Avestimehr, "ITLinQ: A new approach for spectrum sharing in Device-to-Device communication systems," in *IEEE International Symposium on Information Theory*, 2014, pp. 1573–1577.

[14] K. Shen and W. Yu, "FPLinQ: A cooperative spectrum sharing strategy for Device-to-Device communications," in *2017 IEEE International Symposium on Information Theory (ISIT)*, 2017, pp. 2323–2327.

[15] H. Zeng, Y. Shi, Y. T. Hou, W. Lou, H. D. Sherali, R. Zhu, and S. F. Midkiff, "A scheduling algorithm for MIMO DoF allocation in multi-hop networks," *IEEE Transactions on Mobile Computing*, vol. 15, no. 2, pp. 264–277, 2016.

[16] R. Ibrahim, M. Assaad, B. Sayrac, and A. Gati, "Distributed vs. centralized scheduling in D2D-enabled cellular networks," *arXiv preprint arXiv:1806.02081*, 2018. [Online]. Available: https://arxiv.org/abs/1806.02081

[17] A. Neogi, P. Chaporkar, and A. Karandikar, "Multi-player multi-armed bandit based resource allocation for D2D communications," *arXiv preprint arXiv:1812.11837*, 2018. [Online]. Available: http://arxiv.org/abs/1812.11837

[18] W. Cui, K. Shen, and W. Yu, "Spatial deep learning for wireless scheduling," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 6, pp. 1248–1261, 2019.

[19] M. Lee, G. Yu, and G. Y. Li, "Graph embedding-based wireless link scheduling with few training samples," *IEEE Transactions on Wireless Communications*, vol. 20, no. 4, pp. 2282–2294, 2021.

[20] S. Jamshidiha, V. Pourahmadi, A. Mohammadi, and M. Bennis, "Link activation using variational graph autoencoders," *IEEE Communications Letters*, vol. 25, no. 7, pp. 2358–2361, 2021.

[21] Z. Liu, Z. Chen, L. Luo, M. Hua, W. Li, and B. Xia, "Age of information-based scheduling for wireless Device-to-Device communications using deep learning," in *2021 IEEE Wireless Communications and Networking Conference (WCNC)*, 2021, pp. 1–6.

[22] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu, and N. D. Sidiropoulos, "Learning to optimize: Training deep neural networks for interference management," *IEEE Transactions on Signal Processing*, vol. 66, no. 20, pp. 5438–5453, 2018.

[23] L. Lundheim, "On shannon and "shannon's formula"," *Telektronikk*, vol. 98, no. 1, pp. 20–29, 2002.

[24] G. B. Tunze, T. Huynh-The, J.-M. Lee, and D.-S. Kim, "Sparsely connected CNN for efficient automatic modulation recognition," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 15 557–15 568, 2020.

[25] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1492–1500.

[26] Y. Ioannou, D. Robertson, R. Cipolla, and A. Criminisi, "Deep Roots: Improving CNN efficiency with hierarchical filter groups," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1231–1240.

[27] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6848–6856.

[28] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861*, 2017. [Online]. Available: https://arxiv.org/abs/1704.04861

[29] Z. Zhang, J. Li, W. Shao, Z. Peng, R. Zhang, X. Wang, and P. Luo, "Differentiable learning-to-group channels via groupable convolutional

neural networks," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2019, pp. 3542–3551.

[30] Y. Guo, Y. Li, L. Wang, and T. Rosing, "Depthwise convolution is all you need for learning multiple visual domains," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 01, 2019, pp. 8368–8375.

[31] R. C. Gonzalez, "Deep convolutional neural networks [lecture notes]," *IEEE Signal Processing Magazine*, vol. 35, no. 6, pp. 79–87, 2018.

[32] V. D. Tuong, T. P. Truong, T.-V. Nguyen, W. Noh, and S. Cho, "Partial computation offloading in NOMA-assisted mobile edge computing systems using deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 8, no. 17, pp. 13 196–13 208, 2021.

[33] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015. [Online]. Available: https://arxiv.org/pdf/1509.02971.pdf

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014. [Online]. Available: https://arxiv.org/abs/1412.6980

[35] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. [Online]. Available: https://arxiv.org/abs/1409.1556

[36] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

Van Dat Tuong received the B.S. degree in mechatronics from Hanoi University of Science and Technology, Hanoi, Vietnam, in 2012, and the M.S. degree in system software, from Chung-Ang University, Seoul, South Korea, in 2021, where he is currently pursuing the Ph.D. degree in big data. From 2012 to 2018, he was a Software Engineer with the Viettel Software Center, Viettel Telecom and the Samsung Vietnam Mobile Center (SVMC), Samsung Electronics Vietnam, Hanoi, Vietnam.

His research interests include machine learning, game theory, convex optimization, and their applications in wireless networking and ubiquitous computing.

Wonjong Noh received the B.S., M.S., and Ph.D. degrees from the Department of Electronics Engineering, Korea University, Seoul, South Korea, in 1998, 2000, and 2005, respectively. From 2005 to 2007, he conducted the Postdoctoral Research with Purdue University, West Lafayette, IN, USA, and the University of California at Irvine, Irvine, CA, USA. From 2008 to 2015, he was a Principal Research Engineer with the Samsung Advanced Institute of Technology, Samsung Electronics, South Korea. After that, he worked as an Assistant Professor with the Department of Electronics and Communication Engineering, Gyeonggi University of Science and Technology, South Korea, and since 2019, he is working as an Associate Professor with the School of Software, Hallym University, South Korea.

He received the Government Postdoctoral Fellowship from the Ministry of Information and Communication, South Korea, in 2005. He was also a recipient of the Samsung Best Paper Gold Award in 2010, the Samsung Patent Bronze Award in 2011, and the Samsung Technology Award in 2013. His current research interests include fundamental analysis and evaluations on machine learning-based 5G and 6G wireless communications and networks.

Sungrae Cho is a professor with the school of computer sciences and engineering, Chung-Ang University (CAU), Seoul. Prior to joining CAU, he was an assistant professor with the department of computer sciences, Georgia Southern University, Statesboro, GA, USA, from 2003 to 2006, and a senior member of technical staff with the Samsung Advanced Institute of Technology (SAIT), Kiheung, South Korea, in 2003. From 1994 to 1996, he was a research staff member with electronics and telecommunications research institute (ETRI), Daejeon, South Korea. From 2012 to 2013, he held a visiting professorship with the national institute of standards and technology (NIST), Gaithersburg, MD, USA. He received the B.S. and M.S. degrees in electronics engineering from Korea University, Seoul, South Korea, in 1992 and 1994, respectively, and the Ph.D. degree in electrical and computer engineering from the Georgia Institute of Technology, Atlanta, GA, USA, in 2002.

His current research interests include wireless networking, ubiquitous computing, and ICT convergence. He has been a subject editor of IET Electronics Letter since 2018, and was an area editor of Ad Hoc Networks Journal (Elsevier) from 2012 to 2017. He has served numerous international conferences as an organizing committee chair, such as IEEE SECON, ICOIN, ICTC, ICUFN, TridentCom, and the IEEE MASS, and as a program committee member, such as IEEE ICC, GLOBECOM, VTC, MobiApps, SENSORNETS, and WINSYS.