WGAN-based Oversampling for QoS-Aware M2M Network Power Allocation

Junxiu Zhou
Department of Computer Science
Northern Kentucky University
Highland Heights, USA

Yangyang Tao
Department of Computer Science
Northern Kentucky University
Highland Heights, USA

Abstract- Internet of Things (IoT) in the new era of 5G and 6G. Pursuing an autonomous strategy for optimal power allocation remains a significant research topic in M2M communication networks. Recent advances in machine learning offer a viable solution to this challenge. However, the main issue in the power allocation problem when adopting machine learning is the lack of real datasets in the fluctuating network environment. This study delves into the oversampling of the grid-like structure of channel information in M2M communication by leveraging the WGAN. We conducted extensive experiments to assess the benefits of oversampling of the dataset from the simulation environment. Results indicate that the generated dataset is good enough to improve the performance of the machine learning training. Moreover, when comparing the use of solely the original data to a dataset that includes oversampled data, it's evident that the oversampling procedure is beneficial.

Keywords— Generative Adversarial Network (GAN), Wasserstein GAN, power allocation, optimization, machine-to-machine communications.

I. INTRODUCTION

Currently, the M2M communication network is fundamental for enabling the Internet of Things (IoT). It establishes the concept of autonomous data transfer that can be used in numerous applications, Like Industry 5.0, Web 4.0, Augmented Reality (AR), and Virtual Reality (VR). For now, this network is based on a cellular network but aims to evolve towards the fifth or even sixth-generation network communication system (5G, 6G). A lot of new phases are being brought up right now. The vision for the future is a symbiotic world. Where smart devices are embedded into our daily lives, computing is ubiquitous. To address the rising needs of this industry, academic, and society sectors, Energy Harvesting-backed Cognitive Industrial Machine-to-Machine (M2M) networks are seen as a potential driving force in these domains. In this paper, we focus on one of the significant issues in M2M communication, Energy Efficiency (device power allocation). Resolving the power allocation for those end devices that have rigorous battery resource restrictions is a crucial problem.

In recent M2M power allocation optimization research, many ML algorithms have been leveraged to address the QoS-conscious power management challenges [1]. Lately, more deep models, particularly Q-learning, Generative Adversarial Networks, and LSTM have been employed to optimize power resources in M2M networks [1-4]. Yet, most of these studies predominantly relied on simulated datasets of a small scale. However, As small-scale datasets are not able to fully depict the dynamic environment in the M2M network, this study delves into the oversampling of the dataset. The purpose is to synthesize more datasets to simulate the dynamic M2M network. So that we can explore superior ML approaches in M2M network power allocation optimization.

In this work, we used WGAN to conduct the oversampling of the dataset based on a limited dataset of grid-like structure of channel information from s simulated M2M communication environment. The contribution of this work is twofold: First, this is the first work that utilizes the WGAN to conduct the oversampling in M2M communication; secondly, we tested the performance of a Feedforward neural network (FNN) in QoS-ware power allocation problem with the generated data which achieved promising improvement in the training and prediction processes.

The rest of this paper is organized as follows. In Section II, the related works of this paper are introduced. In Section III, the power management system model is described. Then, in Section IV, WGAN and oversampling are described in detail. Next, in Section V, the numerical experiments are conducted, and experimental results are reported and discussed. Finally, the conclusion is put in Section VI.

II. RELATED WORK

There are lots of works that adopted machine learning approaches to improve the power allocation in M2M. The work in [1] is the first work to adopt an FNN to optimize the power allocation problem. Specifically, they used FNN to approximate the power allocation performance of the iterative-based WMMSE method [6]. Later, Zappone *et al.* [2] proposed to use FNN to maximize the *global energy efficiency* (GEE) power allocation model. Eisen *et al.* [7] proposed a model-free primal-dual method to model the power resource allocation problem as a generic formulation. Then, they combined the conventional indirect optimization method with FNN to optimize the generic formulation. In addition to FNN, there are also attempts to use CNN to address the power management problem while gaining the merit of real-time performance. Lee *et al.* [8] employed the CNN architecture to optimize the power resources by feeding the channel information as a matrix. They claimed that CNN architecture can take advantage of the spatial features in the channel gain. Their results showed that the performance of CNN in power allocation tasks is similar to what of FNN. Chen *et al.* [9] used CNN to optimize power resources and spectrum reuse simultaneously. Specifically, they trained the CNN model using the sum of the mean squared error and the categorical cross-entropy loss functions for the spectrum reuse and transmit power values, respectively. Dai *et al.* [10] used CNN to optimize power by using several SoftMax

979-8-3503-7099-7/24/$31.00 ©2024 IEEE

blocks in the output layer. As a result, the output of their CNN model is the ratio of the transmission power of each user on the sub-carrier to the total power. Furthermore, Zhou *et al.* [3] proposed using FNN to optimize power under the constraint of the QoS requirement. Later, in [4], three different QoS-aware power optimization system models were proposed, and a based model was employed to optimize these models. Their numerical experiments verified the effectiveness of the three system models and the FNN-based optimization strategy.

Besides those works mentioned before, recently, lots of works have used deep learning to optimize power allocation. In Fatima et al,[11], The authors use Q-learning to allocate time slots allocation in M2M communication. A clustering algorithm is first used to overcome the congestion problem and then Q-learning is employed to assign conflict-free time slots for machines. Work Yi et al, [12] are using GAN and LSTM for resource allocation. GAN is used to generate the training dataset and LSTM is used to optimize the resource allocation in the M2M network. Sree et al, [13], they are using a Reinforcement learning-based Pointer Network to handle the power allocation.

All the aforementioned works are experimenting on small-scale simulation environments that utilize datasets of a limited size. In order to provide users with diverse QoS requirements in a close to realistic environment. In this work, we propose to use oversampling to level up the simulation environment of the work in [4], by exploring WGAN to oversample the dataset. The generated dataset is evaluated and demonstrates the expected effect in the power allocation optimization in the M2M network.

III. System Model and Problem Formulation

In this section, we initially outline the system model of a typical M2M network facing interference. Following that, we introduce multiple optimization models to address the power management issue.

A. System Model

To construct the base dataset for WGAN. We examine a standard M2M network situated in an area characterized by a disc with a radius of R_c. This M2M network consists of N pairs of *transmitters* (*T*x) and *receivers* (*R*x). Here, we follow the same number of transmitters and receivers provisioning [3, 4], which is one of the most popular provisions of M2M communications. We use $\mathcal{C}_T = \{1,2,\dots,N\}$ and $\mathcal{C}_R = \{1,2,\dots,N\}$ to denote the index sets of *T*x and *R*x, respectively.

The depicted generic network in Fig. 1 uses solid lines to indicate desired transmission links, while dotted lines showcase interfering links. For clarity, primary notations are summarized in Table I where the term "link ij" means the link from Tx i to Rx j, and "channel gain" alludes to the small-scale Rayleigh fading. The channel gains h_{kj} and h_{kk} follows the exponential distribution. The large-scale fading due to the path loss will be described with a power-law term. The shadowing effect is included in the large-scale fading with appropriately adjusted parameters.

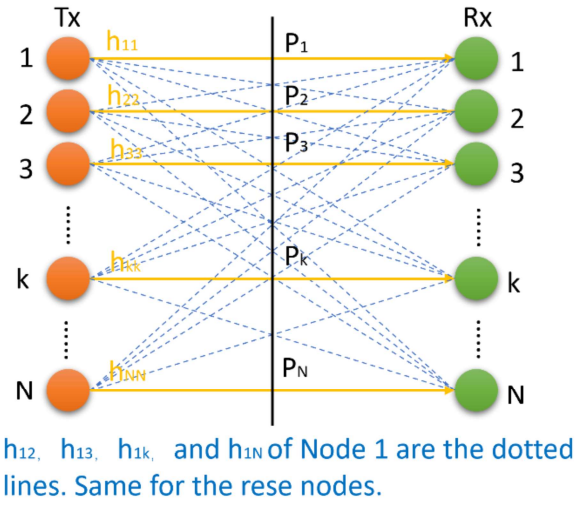

Fig. 1. Generic M2M network structure.

In the present model, a half-duplex is assumed, i.e., a node cannot receive signals while simultaneously transmitting signals. For example, at a particular moment, the link from *T*x 1 to *R*x 3 is different than the link from *T*x 3 to *R*x 1. So, in general, $b_{jk} = b_{kj}$, $b \in \{h, r, \alpha\}$ is not necessary. Also, $r_{kk} \neq 0$, since it represents the distance from *T*x k to *R*x k. σ_k^2 characterizes the *additive white Gaussian noise* (AWGN). With these elaborations, the *signal-to-interference-plus-noise ratio* (SINR) for the receiver k is expressed as:

$$u_k \triangleq \frac{h_{kk}\left(P_k / r_{kk}^{\alpha_{kk}}\right)}{\sigma_k^2 + \sum_{j=1, j\neq k}^{N} h_{jk}\left(P_j / r_{jk}^{\alpha_{jk}}\right)} \tag{1}$$

where usually $1.6 < \alpha_{jk} < 9$. Note that in (1) h_{jk}, h_{kk}, r_{jk}, and r_{kk} are *random variables* (RVs), while the variables P_j and P_k are the entities to be optimized (referred to as the *decision variables* in optimization literature).

TABLE I. List of Main Notations in (1)

Term	Explanation
h_{jk}	Channel gain of the interference link from Txj to Txk
h_{kk}	Channel gain of the desirable link paired with *T*xk and *R*xk
N	Number of *T*x-*R*x pairs
r_{jk}	The length of link from Txj to Txk (meter)
u_k	SINR of Rxk (dB)
$u_{k,\min}$	QoS threshold of SINR (dB)
P_k	Transmitter power of *T*xk (dBm)
α_{jk}	Path-loss exponent of link from Txj to Txk
σ_k^2	Noise power of Rxk

B. Problem Formulation

In the present work, the *weighted sum-rate* (WSR) is used to characterize the system model since it clearly describes the overall throughput of the system. The basic WSR problem was investigated in [1,6]. Here we adopt WSR as the objective

function. Moreover, we introduce the QoS constraints as an enhancement. The augmented WSR problem is formulated as follows:

$$maximize \ \sum_{k=1}^{N} w_k \log_2(1+u_k) \quad (2)$$

$$s.t. \quad u_{k,min} \le u_k \quad (3a)$$

$$0 \le P_k \le P_{k,max} \quad (3b)$$

$$k = 1,2,\dots,N$$

where u_k is defined in (1). w_k denotes the bandwidth. $P_{k,max}$ refers to the allowed maximum power of k.

The optimization model, in Eq. (2) and (3), can be equally transferred to the following optimization model.

$$\min_{P_k, k=1,\dots,N} \sum_{k=1}^{N} w_k [log_2(1+\mu_k)]^{-1} \quad (4)$$

$$s.t. \quad u_{k,min} \le u_k \quad (3a)$$

$$0 \le P_k \le P_{k,max} \quad (3b)$$

$$k = 1,2,\dots,N$$

Detailed explanation can be found in [6].

Next, we utilize WGAN to oversample the dataset from system and optimization problem we mentioned here.

IV. WGAN FOR OVERSAMPLING

A. GAN

GAN is a class of machine learning frameworks proposed in 2014. GANs are used primarily in unsupervised machine learning and are known for their capability to generate data that are similar to, but not exactly the same as, data they were trained on. A GAN is composed of two machine-learning models: a generator and a discriminator. The generator network tries to produce data. Starting with some random noise, it refines its output over time to create data that resembles a certain dataset. The ultimate goal of the generator is to create a dataset that is very close to the real data, at least from the perspective of data distribution. The discriminator network tries to distinguish between genuine data from the dataset and fake data produced by the generator. The ultimate goal is that the output of the discriminator is not able to distinguish the real and fake datasets. The training process is called adversarial training. Ideally, we input both the real data from the task and fake data from the generator into the discriminator, and the final accuracy of the discriminator is 50% which means that it's incapable of classifying the data as real or fake. GAN has been applied to a variety of domains: (1) Image generation. (2) Style transfer. (3) Data augmentation (oversampling). (4) Generating new art and music. (5) Create realistic video game environments for more immersive gaming experiences.

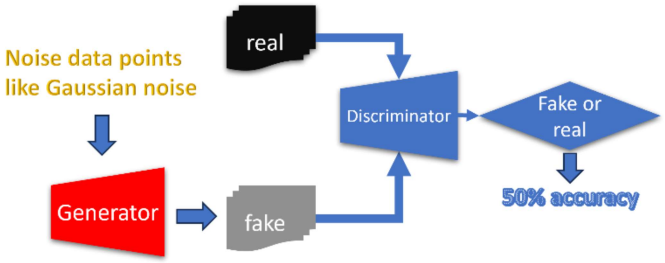

Fig. 2. GAN architecture.

B. WGAN

WGAN is an improved design over the standard GAN that was introduced to address the training instabilities and convergence issues found in the training of traditional GANs. It includes the critic and generator. WGAN introduces a new distance-measuring approach to measure the distance between the distribution of the data generated by the generator and the real data distribution. Instead of the Jensen-Shannon divergence that the original GANs use, WGAN employs the Wasserstein distance (also known as the Earth Mover's distance). This distance measuring approach provides smoother gradients, which in turn can lead to a more stable training process. The exact formula for Earth Mover's distance is below. x and y stand for the real and the generative samples respectively. $\prod(\mathbb{P}_r, \mathbb{P}_s)$ is the set of all joint distribution of $\delta(x,y)$ where the marginal distributions of x and y are $\mathbb{P}_r$ and $\mathbb{P}_s$.

$$W(\mathbb{P}_r, \mathbb{P}_s) = \inf_{\delta \in \prod(\mathbb{P}_r, \mathbb{P}_s)} \mathbb{E}_{(x,y)\sim\delta} \|x-y\| \quad (5)$$

In WGAN, the output is a score indicating the "realness" or "fakeness" of a sample instead of the probability from the discriminator of the original GAN. The loss function of WGAN is derived from the Wasserstein distance. One advantage of WGANs is that the training loss correlates better with the visual quality of generated samples. In traditional GANs, a low discriminator loss doesn't necessarily mean high-quality generated samples. However, with WGANs, when the loss decreases, it tends to indicate an improvement in the generator's good performance to generate realistic fake samples.

The classic design of WGAN is followed in this work. The criticism contains 3 layers. Two dense layers with 128 and 64 neurons respectively, and one output layer only contains 1 neuron for scoring. The generator has 4 layers with an output layer of 210 neurons.

C. WGAN for oversampling

Using WGAN for oversampling involves generating generative samples and comparing them with the original dataset to evaluate the performance. The overall model construction has no big difference from the traditional training and testing process of a machine learning model. The typical process includes preparation of the dataset, constructing the WGAN model, and training of WGAN. However, the only difference is that after finishing the training of WGAN, we utilize the WGAN to generate the generative dataset. The fake

dataset is combined with the original real dataset for the evaluation process. In our design, we are using a Forward Neural Network to evaluate the performance of the generated dataset.

In this paper, we use the dataset in [4] as the original dataset. Put simply, we utilized the widely recognized nonlinear programming problem (NLP) solver, Fmincon, available in the Matlab toolbox, to produce the original dataset. The Fmincon solver was motivated by two main reasons: firstly, it establishes comparable solution benchmarks for subsequent machine learning model evaluations; and secondly, being iterative in nature, Fmincon embodies the characteristics typical of such methods. For a deeper understanding, one can consult references [3, 4]. The overall architecture of our work is shown in Fig. 3.

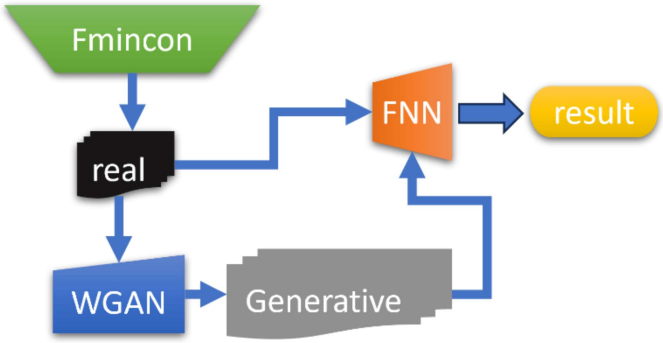

Fig. 3. The architecture of the proposed work.

.

V. Experimental Results

In this section, we verify the usefulness of the dataset generated by the WGAN model (we call it a generative dataset) through extensive numerical experiments. To test the validation of the generative dataset, we use the Gaussian Mixture Model to fit the real dataset and the generative dataset. We compare the GMM model parameters of the two datasets. To compare the effectiveness of the generative dataset with the real dataset, we trained an FNN to measure the performance on a test set. All the numerical experiments below are implemented in *Python* and *TensorFlow* on a computer with an Intel Core i7 CPU and 16 GB RAM.

Following the wireless network setting in [3, 4], we use N = 10 pairs of transceivers, radius = 1000 meters, P_{max} = 21dBm, AWGN power = -143.97dBm, and QoS thresholds = 15 dB. With these network parameters and Fmincon, we generate 3,000 successful optimization results as training samples. The 3,000 training samples are used to train the WGAN model. The WGAN model is used to generate 2000 fake samples. Then those two datasets are used to train an FNN to verify the effectiveness of the generative dataset in allocating power resources. To test the performance of FNN, a variety range of 8 QoS thresholds (5dB, 10dB, 15dB, 20dB, 25dB, 30dB, 35dB, and 40dB) are generated to form a test set, and each QoS threshold value has 5,000 samples. The test set contains some QoS thresholds that are not included in the training set to verify the robustness of the trained FNN model. Experimental performance is measured according to the optimal power WSR value in eq. (2), i.e., the sum-rate value, the QoS satisfaction rate, and computational costs.

A. The number of training samples of the WGAN model

To verify the effectiveness of the WGAN model in generating new data, we tested the performance of WGAN under various sizes of training samples, i.e., [1000, 2000, 3000, 4000, 5000, 10000]. Under each size of training samples, we train the WGAN model, then we use the trained model to generate 1000 fake generative data samples to do the following analysis.

First, we use GMM to fit both the training samples and the generative samples. Then the mean and covariance values are compared to measure if the WGAN is able to generate fake samples that follow a similar distribution of the network. The result is shown in the following Table II.

TABLE II. The Difference Between Real and Generative Datasets

#. Training Samples	Channel Gain Diff.		Deployment Diff.	
	Mean	Covariance	Mean	Covariance
1000	0.0287	-0.0287	0.0157	-0.0157
2000	0.0190	-0.0190	0.0266	-0.0266
3000	-0.0086	0.0086	0.0117	-0.0117
4000	-0.0147	0.0147	-0.0087	0.0087
5000	-0.0004	0.0004	0.0102	-0.0102
10000	-0.0055	0.0055	0.0002	-0.0002

As can be observed from Table II, the WGAN can generate network data that follows a similar distribution as the real data. In addition, with the increase of the training samples of WGAN, the differences between mean and covariance are decreased for both the channel gain and network deployment.

In this work, we feed the channel gain, network deployment information, and the related optimal power to WGAN. That is the trained WGAN model not only needs to learn to generate the network configuration data but also needs to generate the power values to maximize the sum-rate and meet the minimum QoS constraint. Therefore, to test the QoS satisfaction performance of the WGAN, the QoS satisfaction rate [6] is calculated based on eq. (3a) for all test samples with a 15 minimum QoS value.

The following Table III shows the sum rate and average training time of the WGAN model.

TABLE III. The Difference Between Real and Generative Datasets

#. Training samples	1000	2000	3000	4000	5000	10000
QoS Satisfaction Rate (100%)	67	97	100	100	100	100
Training time (s)	3.99	11.56	18.74	26.11	30.69	74.27

As illustrated in Table III, the QoS satisfaction rate and the training time increase as the number of training samples increases. If the training samples are above 3,000, the WGAN model is able to achieve about a 100% QoS satisfaction rate.

This observation provides us with a way to pick the appropriate size of the training dataset.

In summary, after comparing the distribution and QoS satisfaction rate of the data generated by WGAN trained with different sizes of training sets, we can conclude that WGAN is able to find the hidden pattern between the network configuration and its related power value. In general, if the WGAN is trained with a dataset that has more than 3,000 samples, it will get a satisfactory performance in generating new data.

B. Sum-Rate Performance Comparison

To verify the effectiveness of the generated data in power allocation-related tasks, we use the real training samples and the generative samples to train the FNN model to predict the power values. The test set contains 1000 samples and is collected using Fmincon under a 15 QoS constraint value. We use WGAN to generate datasets with different numbers of data samples, i.e., [1000, 2000, 3000, 4000, 5000]. Next, we use the generated dataset with the real dataset to train different FNN models. The performance results of those models are tested using the same test set. Note that to conduct a fair comparison, all FNN models follow the same setting as indicated in [5], and are trained with the same parameters. The sum rate and the QoS satisfaction rate of different FNN models on the same test set are summarized in Fig. 4.

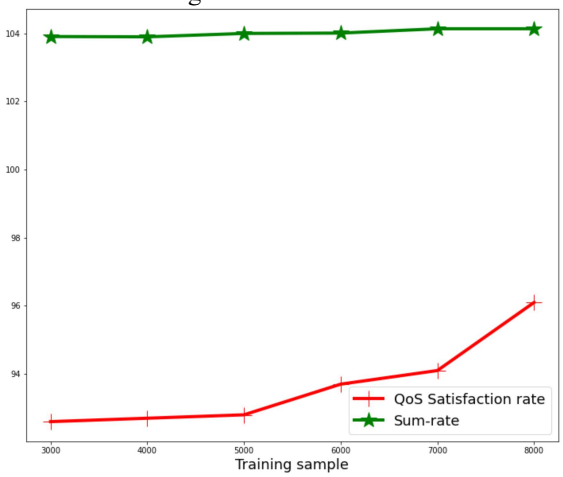

Fig. 4. Comparison the sum-rate and QoS Satisfaction rate over different training samples.

As illustrated in Fig. 4, an increase in the number of generative data samples corresponds to a noticeable improvement in the predictive performance of the FNN model. This outcome aligns with our initial motivation for employing the WGAN to generate data effectively. Notably, when a small number of generative samples, such as 1,000 or 2,000, are added to the real data samples, the influence of the generative data may not be readily discernible. However, in the course of this experiment, it became evident that surpassing the quantity of real data with an equivalent or greater number of generative samples significantly enhances the predictive power of the FNN model.

VI. CONCLUSION

In this paper, we use WGAN to produce generative samples for the FNN based optimization of the power management problem with several different objective functions or constraints. To set a fair comparison platform, we first compared the difference in data distribution between the real samples and the generative samples. Then extensive numerical comparison experiments are conducted to measure the QoS satisfaction rate of the generative samples. The results demonstrate that WGAN can create decent generative samples, while only requiring about 18s of additional generation time. In principle, the WGAN's superior performance lies in the ability to learn the distribution of the channel gains and path losses with a smooth gradient objective function. It is also noted that the generative samples can achieve a very high QoS satisfaction rate.

REFERENCES

[1] H. Sun, X. Chen, Q. Shi, M. Hong, X. Fu and N. D. Sidiropoulos, "Learning to Optimize: Training Deep Neural Networks for Wireless Resource Management," *IEEE Transactions on Signal Processing*, vol. 66, no. 20, pp. 5438-5453, 2018.

[2] A. Zappone, M. Debbah, and Z. Altman, "Online energy-efficient power control in wireless networks by deep neural networks," In *2018 IEEE 19th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2018, pp. 1-5.

[3] J. Zhou, X. Liu, Y. Tao, and S. Yu, "QoS-aware power management with deep learning," In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2019, pp. 289-294.

[4] J. Zhou, X. Liu, and C. Huang, "Machine Learning for Power Allocation of a D2D Network," In *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1-6.

[5] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359-366, 1989.

[6] Q. Shi, M. Razaviyayn, Z. Luo, and C. He, "An iteratively weighted MMSE approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Transactions on Signal Processing*, vol. 59, no. 9, pp. 4331-4340, 2011.

[7] M. Eisen, C. Zhang, L. F. Chamon, D. D. Lee, and A. Ribeiro, "Learning optimal resource allocations in wireless systems," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2775-2790, 2019.

[8] W. Lee, M. Kim, and D. Cho, "Deep power control: Transmit power control scheme based on convolutional neural network," *IEEE Communications Letters*, vol. 22, no. 6, pp. 1276-1279, 2018.

[9] M. Chen, J. Chen, X. Chen, S. Zhang, and S. Xu, "A Deep Learning Based Resource Allocation Scheme in Vehicular Communication Systems." In *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1-6.

[10] M. Dai, Q. Huang, Z. Lu, B. Chen, H. Wang, and X. Qin, "Power Allocation for Multiple Transmitter-Receiver Pairs Under Frequency-Selective Fading Based on Convolutional Neural Network," *IEEE Access*, vol. 8, pp. 31018-31025, 2020.

[11] Hussain F, Anpalagan A, Khwaja AS, Naeem M. Resource allocation and congestion control in clustered M2M communication using Q‐learning. Transactions on Emerging Telecommunications Technologies. 2017 Apr;28(4):e3039.

[12] Xu, Yi-Han, Xin Liu, Wen Zhou, and Gang Yu. "Generative adversarial LSTM networks learning for resource allocation in UAV-served M2M communications." IEEE Wireless Communications Letters 10, no. 7, 2021, pp. 1601-1605.

Das SK, Rahman MS, Mohjazi L, Imran MA, Rabie KM. Reinforcement learning-based resource allocation for M2M communications over cellular networks. In 2022 IEEE Wireless Communications and Networking Conference (WCNC) 2022 Apr 10, pp. 1473-1478.