Distributed Radio Resource Allocation Using Deep and Federated Learning in 6G Networks

Iacovos Ioannou*, Christophoros Christophorou*, Prabagarane Nagaradjane†, Vasos Vassiliou*
* Department of Computer Science, University of Cyprus and CYENS - Centre of Excellence, Cyprus
† Department of ECE, Sri Sivasubramaniya Nadar College of Engineering Chennai, India

***Abstract*—Efficient resource allocation in Device-to-Device (D2D) communication within 6G networks is crucial for enhancing overall network performance and efficiency. This paper presents a novel Deep Learning (DL) based approach for Radio Resource Allocation (RRA), leveraging Distributed Artificial Intelligence (DAI) using Belief-Desire-Intention eXtended (BDIx) agents, dynamic feedback allocation, and a Deep Feedback Neural Network (DFBNN). Additionally, Federated Learning (FL) is integrated to enable distributed training across BDIx agents, serving as D2D Relays (D2DR) or D2D Multihop Relays (D2DMHR), ensuring data privacy and reducing communication overhead. The proposed method is thoroughly evaluated against traditional graph-based and game-theoretic algorithms and Deep Feedforward Neural Networks (DFNN). Results demonstrate significant improvements in interference management, data rate, and execution time. By providing scalable, adaptive, and resilient resource allocation, this proposed method meets the stringent requirements of 6G applications, paving the way for more efficient and reliable network operations.**

***Index Terms*—6G networks, D2D communication, radio resource allocation, Deep Learning, DFBNN, Federated Learning, DAI**

I. Introduction

The evolution of wireless communication technology has been marked by rapid advancements and increasing demands for higher data rates, improved reliability, and lower latency. As we move towards the sixth generation (6G) of mobile networks, these requirements are further amplified by the emergence of new use cases such as the Internet of Things (IoT), autonomous vehicles, and massive Machine-type Communications (mMTC) [1], [2]. One of the key challenges in realizing these 6G use cases is efficient Radio Resource Allocation (RRA) [3], [4], particularly in Device-to-Device (D2D) communication. With the advent of 6G networks, efficient resource allocation for D2D communication is paramount. Traditional methods often fail to dynamically adapt to changing network conditions.

RRA in D2D communication involves distributing spectrum, power, and other resources to optimize throughput, latency, and interference. The main challenges include managing interference, ensuring scalability, and adapting to the dynamic wireless environment. D2D communication reuses cellular spectrum, causing potential interference between cellular and D2D users as well as among D2D users themselves [5], [4], [6]. Many devices in mMTC scenarios require scalable algorithms for real-time resource allocation. Additionally, varying wireless channel conditions and user mobility necessitate adaptive strategies [3]. Efficient RRA can significantly enhance network performance by maximizing throughput and minimizing latency [7], [6]. Optimizing transmission power and other resources can also lead to substantial energy savings, crucial for battery-operated IoT devices. Ensuring fair resource distribution maintains Quality of Service (QoS), which is vital for user satisfaction and Service Level Agreements (SLAs).

6G networks will support diverse use cases, such as IoT deployments, requiring efficient RRA to minimize interference and ensure connectivity. Autonomous vehicles need ultra-Reliable Low-Latency Communication (URLLC) for safety, necessitating dynamic resource allocation. Smart grids, environmental monitoring, and intelligent transportation systems in smart cities also require robust and scalable RRA solutions. Overlay spectrum sharing in D2D communications allows D2D devices to use dedicated spectrum, while underlay spectrum sharing allows simultaneous spectrum sharing with cellular users. Centralized RRA approaches are often infeasible for 6G networks due to high computational and communication overhead [1]. Thereby, distributed approaches using Artificial Intelligence (AI) and Machine Learning (ML) are becoming more popular [7]. DAI is crucial for managing the complexity and scale of 6G networks [8], [9].

This research builds on Belief-Desire-Intention eXtended (BDIx) agents, which autonomously manage radio allocations for D2D-Relays. BDIx agents operate based on beliefs (network state information), desires (objectives like maximizing throughput), and intentions (actions to achieve objectives) and can adapt to changes in network environment, such as user density variations or channel conditions. In distributed RRA, BDI and BDIx agents at various network nodes (e.g., base stations, user devices) make local resource allocation decisions. Benefits include scalability, adaptability, and resilience [7]. Distributed decision-making handles more devices without a proportional increase in computational load.

Distributed systems are resilient to failures, as a single agent's failure does not compromise the entire network. Therefore, distributed and autonomous DAI and BDIx agents are necessary for effective resource allocation in 6G networks [8], [9]. As 6G networks bring diverse and stringent requirements, efficient RRA is key to successful deployment. Distributed approaches, especially those using DAI and BDI agents, offer scalable, adaptive, and resilient solutions for massive D2D communication scenarios. Leveraging AI and ML ensures optimal resource utilization, enhancing network performance and user experience [1].

This paper presents a distributed RRA method using Federated Learning (FL) and Deep Neural Networks (DNN), like Deep Feedback Neural Network (DFBNN), to minimize average interference and maximize the sum rate while maintaining constant transmission power. This approach leverages FL to train models on decentralized data, ensuring data privacy and reducing communication overhead [3]. DNNs capture complex data patterns, enabling more accurate and efficient resource allocation decisions. Our proposed method have been thoroughly evaluated against traditional graph-based and game-theoretic algorithms and Deep Feedforward Neural Networks (DFNN), demonstrating significant improvements on scalability, adaptability, and overall network performance, making it suitable for 6G networks. The novelty lies in integrating advanced Deep Learning (DL) with DAI for RRA, utilizing BDIx agents and FL to manage resource allocation autonomously while preserving privacy. This research significantly advances RRA for 6G networks by combining BDIx agents and FL to optimize resource allocation. Specifically, the following contributions are made:

1) Introduction of a distributed RRA method using FL and DNNs.
2) Employment of DFBNN to capture complex data patterns, leading to more accurate and efficient resource allocation decisions.
3) Integration of FL for decentralized training across BDIx agents functioning as D2DR or D2DMHR, ensuring data privacy and reducing communication overhead.

The rest of the paper is structured as follows. Section II provides some background information on the RRA and the ML approaches used in this examination. The system description is elaborated in Section III. The methodology used, the model training details, the dataset creation algorithm for the ML training and testing, and the algorithm that the Base Station (BS) should execute to predict the best resource allocations are provided in Section IV. The efficiency of the investigated approaches is examined, evaluated, and compared in Section V in scenarios with varying device densities. Finally, Section VI includes concluding remarks and our future directions.

II. Background Work

1) Deep Feedback Neural Networks (DFBNNs): DFBNNs utilize feedback connections to predict optimal resource allocation strategies based on historical data and current state information. Unlike traditional feedforward networks, DFBNNs incorporate recurrent connections that allow information to be cycled back into the network with the use of i.e., LSTM, enhancing their ability to model temporal dependencies and complex patterns in data [10], [11].

The architecture consists of an input layer, multiple hidden layers with feedback loops, and an output layer. Each hidden layer processes inputs using a combination of linear transformations and non-linear activation functions, with feedback connections enabling the integration of past and present inputs. The network iteratively refines its predictions through these feedback loops, capturing dynamic changes in the input data [12]. The output layer generates the predicted output $\hat{y}$ using the transformation $\hat{y} = f(x, h; \theta)$, where h represents the hidden states and θ denotes the network parameters. The training process aims to minimize the loss function $L(\hat{y}, y)$, with y being the true output. Training employs backpropagation through time (BPTT), updating the weights θ via gradient descent according to $\theta \leftarrow \theta - \eta \nabla_\theta L(\hat{y}, y)$, where η is the learning rate [13].

2) Federated Learning (FL): FL, as described by [14], allows multiple devices to collaboratively learn a shared model while keeping their data localized, enhancing privacy and reducing central computation load. The global model at iteration t is formulated as $\theta_t = \frac{1}{K} \sum_{k=1}^{K} \theta_t^k$, where θ_t is the global model, and θ_t^k is the local model on device k. The objective is to minimize the global loss function aggregated from all devices. Each device updates its local model using its data: $\theta_t^k \leftarrow \theta_t - \eta \nabla_\theta L_k(\theta_t)$. These updated local models are then averaged to update the global model, ensuring that the model improves over time without centralizing the data from individual devices.

These control and transmission modes are essential for efficient D2D communication, with each mode offering specific advantages depending on the network structure and requirements. The DAI framework particularly leverages the DAI control type for its autonomous and parallel processing capabilities [9].

III. System Description

The network architecture, as shown in Fig. 1 consists of a BS that serves as the central unit responsible for coordinating the network's overall operations. The sub-network is composed of three types of devices: D2D-Relays (D2DR), D2D-Clients (D2DC), and D2D-Multihop Relay devices (D2DMHR). D2DR are devices that relay data between the D2DCs and potentially other relays. D2DCs are end-user devices requiring resource allocation for efficient communication. D2DMHR devices support multi-hop communication within the sub-network, facilitating extended reach and connectivity. All devices are equipped with BDIx agents to form the sub-network under the BS [9].

The primary objective is to efficiently allocate Frequency Bands (FBs) to the D2DC devices using reusable bands (underlay) from the D2DR and D2D-Multihop Relay, ensuring minimal interference and optimized resource utilization. To achieve this, we employ FL to update the models of the D2DRs and D2DMHRs. This solution leverages DAI, where the model is embedded within the beliefs of the BDIx agents. Through this model, the relay devices can share FB information and, via FL, inform other devices of the shared FB. This collaborative approach ensures that updates and learning are distributed across the network, enhancing the efficiency and effectiveness of the resource allocation process.

IV. Methodology

A. Data Generation for the Dataset

To generate the dataset, an algorithm is employed under simulation. User Equipment (UEs) with BDIx agents are randomly positioned within a defined area under the BS. According to the Distributed Artificial Intelligence Solution

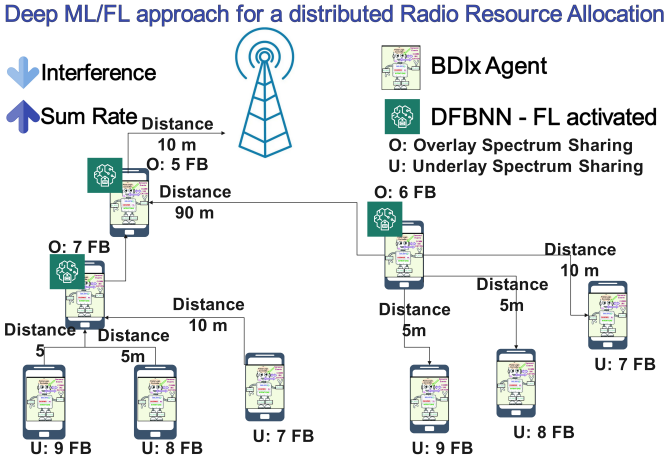


Fig. 1: Comparison of resource allocation methods.

(DAIS) plan, some BDIx agents are selected as D2DR or D2DMHR, serving as Cluster Heads (CHs) or clients when they enter the network. A key feature of this process is the dynamic FB allocation for clusters, with each CH receiving a unique FB. This dynamic allocation process, which takes into account interference levels, ensures minimal interference and efficient resource utilization.

The resulting tree structure network undergoes processing using the described algorithm to allocate radio FBs to each client (as shown in 1). The RRA algorithm is designed to allocate FBs to UEs within a 6G network using a hierarchical approach involving CHs. Initially, UEs equipped with BDIx agents are selected to be D2D-Clients (D2DCs to the CHs), D2DR, or D2DMHR, based on the DAIS plan executed upon network entry. For D2DR and D2DMHR, the algorithm assigns overlay FBs. The algorithm then computes the distances between each UE and its respective CHs, as well as the distances between each CH and the BS.

Following this, a list of available underlay FBs is initialized. That is, for each UE in the network, the algorithm checks for any unused underlay FBs. If available, these FBs are directly assigned to the UE. If not, the algorithm calculates the distance from the UE to its respective CH and assigns an FB based on the distance and CH's priority. Finally, the algorithm outputs the FB allocation for each UE. In summary, data generation and FB allocation in this context involve creating a dynamic and realistic network environment, selecting optimal CHs using the DAIS algorithm, and systematically allocating FBs based on calculated distances and CH priorities to ensure efficient resource utilization and minimal interference.

B. Features and Descriptions

Table I summarizes the features used to train the models.

C. DFBNN Model Architecture With FL Integration

The DFBNN model consists of multiple layers, with the architecture determined through hyperparameter tuning using the Optuna library [15]. The model includes dense and dropout layers, aimed at minimizing Mean Squared Error (MSE). The Neural Network (NN) model is implemented as a sophisticated architecture designed to leverage both

Algorithm 1 Radio Resource Allocation Algorithm

Require: List of UEs with BDIx agents, List of CHs (D2DRs/D2DMHRs), Distance information
Ensure: Allocation of FBs to each UE

To the selected CHs (D2DR and D2DMHR) that are selected based on the results of the tree creation (BDIx agents run DAIS algorithm on the entrance in the 6G network), assign overlay FBs
Compute distances between each UE towards its CH (D2DR/D2DMHR) and CH with BS
Create tree structure network with D2DR and D2DMHR as CHs
Initialize list of available underlay FBs
for each UE in the network **do**
    if there are available underlay FBs not assigned from the list of available underlay FBs **then**
        Assign the unused underlay FBs directly to UE
    else
        Calculate distance from UE to its respective CH
        Assign FB based on distance and CH priority
    end if
end for
Output the FB allocation for each UE

TABLE I: Features and descriptions for model training.

Feature	Description
Latitude	The geographic latitude of the device's position.
Longitude	The geographic longitude of the device's position.
FB	The specific FB is allocated to the device by the algorithm.
Cluster Head Status	Indicates whether the device is a cluster head.

the strengths of traditional feedforward layers and advanced techniques like Long Short-Term Memory (LSTM) layers. This hybrid model is built using the TensorFlow library, which allows for efficient training and deployment of DL models.

The model begins with an input layer that takes in the data, with the input shape determined by the features of the dataset. This layer is crucial as it defines the structure and dimensionality of the input data that will be fed into the network. The first major layer in the model is an LSTM layer. LSTM is a type of Recurrent Neural Network (RNN) layer well-suited for sequential data. It can learn long-term dependencies, making it ideal for time-series data or any data with temporal aspects. The LSTM layer consists of several units, each with an internal memory cell that can retain information across many time steps. The specific number of units in the LSTM layer is defined as hyperparameters in our method, and the best parameter is evaluated during the evaluation of hyperparameters (with a range of 1 to 3). Following the LSTM layer, a Dropout layer is included. Dropout is a regularization technique used to prevent overfitting in NNs. It randomly sets a fraction of input units to zero during training, ensuring that the model generalizes well to new data. The dropout rate specified also as a hyperparameter (with a range of 0.2 to 0.5) determines the fraction of neurons that are dropped.

After the Dropout layer, the model incorporates several Dense layers. Dense layers, also known as fully connected layers, are a fundamental building block of NNs. Each neuron in a Dense layer receives input from all neurons in the previous layer, making these layers highly expressive and capable of learning complex patterns. The number of neurons in each Dense layer and their activation functions are also specified as hyperparameters (with a range of 32 to 256). Typically, the first Dense layers might use activation functions like Rectified Linear Unit (ReLU) to introduce non-linearity. In contrast, the final Dense layer uses a softmax activation function for classification tasks or a linear activation function for regression tasks.

The final layer of the model is the output layer. The configuration of this layer depends on the specific task at hand. The output layer typically uses a softmax activation function for classification tasks to produce probability distributions over the possible classes. A linear activation function might be used for regression tasks to output continuous values. The number of neurons in the output layer corresponds to the number of classes in a classification task or the number of predicted values in a regression task. The model is compiled with an optimizer, loss function, and metrics for evaluation. The optimizer, such as Adam or Stochastic Gradient Descent (SGD), is used to minimize the loss function during training. The loss function depends on the task: categorical cross-entropy for classification or mean squared error for regression. In the implementation, we include callbacks such as early stopping to halt training when the model's performance stops improving, which helps prevent overfitting. The final resulting model is shown in Fig. 2.

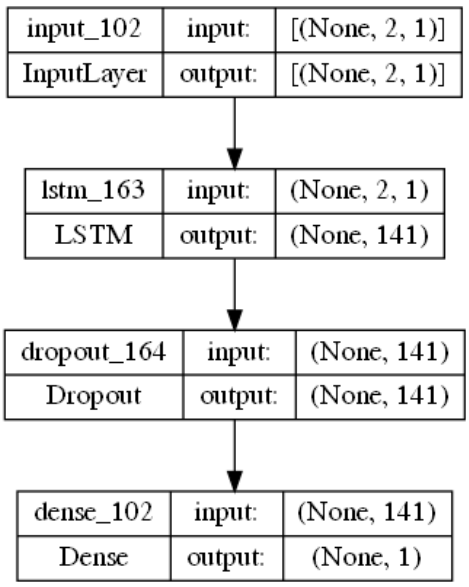


Fig. 2: DFBNN and DFBNN-FL Model.

In summary, the DNN model is a hybrid architecture that combines LSTM layers for handling sequential data with fully connected Dense layers for learning complex representations. Including dropout layers helps with regularization, and using appropriate activation functions ensures that the model can capture non-linear patterns in the data. This combination makes the model versatile and powerful for various ML tasks.

1) Federated Learning (FL) Integration: To enhance the privacy preservation and scalability of our method, we integrate FL, with the centralized model being hosted at the Base Station (BS). FL is a machine learning paradigm that enables decentralized training of models across multiple nodes, such as devices or research labs, without centralizing the data [14]. This approach is valuable in scenarios where data privacy and security are paramount, allowing institutions to collaboratively train a model while keeping their data on local servers. Each participating node, such as D2DR and D2DMHR devices, independently trains a local instance of the DFBNN model on its proprietary dataset. Instead of sharing raw data, these devices periodically send their model updates to the BS, ensuring that sensitive data remains within its source location, thereby preserving privacy and complying with regulations.

At the BS, these model updates are aggregated using Federated Averaging (FedAvg), which combines the updates from all participating nodes to create a global model [14]. This global model is then used to update the local models on the CHs, ensuring that each CH benefits from the diverse data distributed across the network without compromising data privacy. This aggregation process ensures that the global model benefits from the diverse data distributed across the nodes, maintaining overall model accuracy and generalization capability while preserving data privacy.

The key parameters in the FL process include the number of clients, which is set to the number of CHs to ensure each cluster contributes to the training process. Communication rounds involve model updates being exchanged between clients and the central server (BS) for about 100 rounds, striking a balance between model performance and communication overhead. The aggregation method used is FedAvg, which integrates local models comprehensively by aggregating their updates. After the global model is updated at the BS, it is redistributed to the CHs, allowing them to update their local models accordingly. Data privacy techniques such as differential privacy, encryption, and Secure Multi-Party Computation (SMPC) are implemented to ensure data integrity and confidentiality during transmission. Regularization parameters, including dropout with a rate of 0.5 and L2 regularization with a coefficient of 0.01, are used to prevent overfitting.

Additionally, data augmentation strategies are employed to improve model robustness, with parameters such as rotation up to 15 degrees, flipping both horizontally and vertically, and scaling up to 10% zoom in or out is applied to the training data. Secure communication protocols such as public key cryptography and SSL on HTTPS are employed to ensure data integrity and confidentiality while transmitting model updates. This comprehensive approach maintains model performance and privacy, making it suitable for sensitive applications.

D. Training, Optimization and Validation

The model is trained on normalized synthetic data created from the BDIx agent's decisions to form the network tree under BS, with hyperparameters such as the number of layers,

units per layer, and dropout rates optimized using Optuna. The best model is selected based on validation loss. The training and hyperparameter optimization process involves several key steps.

Firstly, data normalization is performed using synthetic data to ensure features contribute equally to the training process, improving the neural network's performance and stability. This preprocessing step scales data to a standard range, typically between 0 and 1, or -1 and 1. Hyperparameter optimization is crucial for selecting the best set of parameters governing the learning process. Key hyperparameters optimized include the number of layers in the neural network, the number of units in each layer, and dropout rates. Dropout, a regularization technique, helps prevent overfitting by randomly setting a fraction of input units to zero during training, ensuring the model generalizes well to new data. Optuna, a hyperparameter optimization framework, is used to automate this process. The optimization involves defining an objective function to train the model and return the validation loss, sampling hyperparameters to propose new sets for each trial, training the model with the proposed hyperparameters, evaluating the model on a validation dataset, and iteratively adjusting the hyperparameters to minimize validation loss. This loop continues for a predefined number of trials or until a stopping criterion is met. Finally, the best model is selected based on the lowest validation loss observed during the trials, ensuring that the chosen hyperparameters lead to a model that generalizes effectively to new data.

The training and optimization of our DNN model involve a systematic approach to hyperparameter tuning using Optuna. By leveraging normalized synthetic data and focusing on minimizing validation loss, we ensure that the model achieves high performance and generalization capabilities. The use of Optuna allows us to efficiently explore the hyperparameter space and select the best model configuration for our specific application.

In our simulation, we utilize k-fold cross-validation with 10000 UEs with BDIx agents under the BS to ensure a robust and unbiased evaluation of the model's performance. Specifically, we partition the dataset into k subsets, training the model k times, each time using a different subset as the validation set and the remaining $k-1$ subsets as the training set. For the resulting testing phase, we adopt an 80-20 split, where 80% of the data is used for training and the remaining 20% is reserved for validation. This approach helps maximize the use of available data and provides a reliable assessment of the model's predictive capabilities.

V. Performance analysis

A. Evaluation Metrics and Simulation Parameters

Metrics for comparison include average interference, sum rate, total power consumption, and execution time. These metrics are evaluated across different device counts, ranging from 100 to 1000.

1) Average Interference (Watts): This metric measures the mean level of interference experienced by the devices in the network. It is calculated as the average interference power received by each device from all other devices: avg_interference $= \frac{1}{N}\sum_{i=1}^{N} I_i$, where I_i is the interference affecting the device i and N is the number of devices [16].

2) Execution Time (seconds): The time taken to compute the resource allocation for the given number of devices, including the computational time of the algorithm from input to output [17].

3) Sum Rate (bps): This metric measures the total data rate achieved by all devices in the network. It is the sum of the individual data rates of each device: sum_rate $= \sum_{i=1}^{N} R_i$, where R_i is the data rate of device i [17].

4) Total Power Consumption (Watt): This metric measures the total power consumption of all devices in the network during communication. It is the sum of the power used by each device: total_power $= \sum_{i=1}^{N} P_i$, where P_i is the power consumption of device i [17].

The simulation parameters used in this research include 10,000 training data points, the system is limited to 100 FBs, a frequency of 2.4 GHz, and a bandwidth of 20 MHz. Table II shows the simulation parameters used in this research.

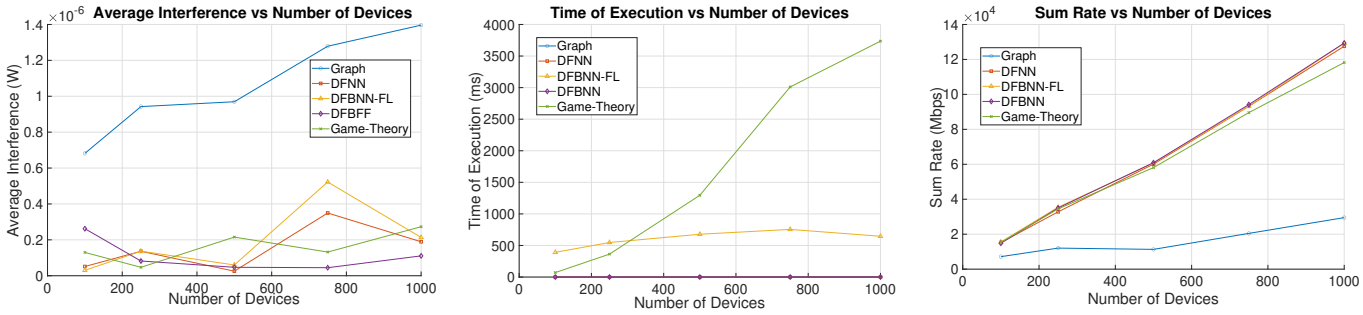
TABLE II: Simulation Parameters

Parameter	Value (Units)
Training Data	`10000 samples`
Number of Frequency Bands	`100 frequency bands`
Frequency	`2.4 GHz`
Bandwidth	`20 MHz`

B. Evaluation of RRA approaches with DFBNN & DFBNN-FL

The proposed DFBNN-based approach and the FL case are compared against traditional graph-based [18], game-theoretic methods [19], and DFNN [20] approaches. The DFBNN shows significant improvements in reducing interference and power consumption while maintaining high data rates and lower execution times. Fig. 3 illustrates the performance comparison of various methods as the number of devices increases. The methods compared are Graph, DFNN, DFBNN, DFBNN-FL, and Game-Theory.

Fig. 3a illustrates the relationship between average interference and the number of devices for various methods, including Graph, DFNN, DFBNN, DFBNN-FL, and Game-Theory. Interference is a critical parameter in wireless networks, affecting communication system performance and reliability. The Graph method shows a steady increase in interference as the number of devices grows, starting at about 0.6×10^{-6} W for 100 devices and reaching 1.4×10^{-6} W at 1000 devices, indicating inefficiency in managing interference with increasing network density. Conversely, methods like DFNN and DFBNN maintain minimal interference levels, below 0.4×10^{-6} W, making them more suitable for densely populated networks. DFBNN-FL and Game-Theory also perform well, albeit with slightly higher interference levels. These results underscore the effectiveness of advanced algorithms like DFNN and DFBNN in controlling interference, highlighting the need for appropriate interference management strategies to ensure stable and high-quality communication in device-dense environments.



(a) Avg. Interference vs Number of Devices (b) Time of Execution vs Number of Devices (c) Sum Rate vs Number of Devices

Fig. 3: Performance comparison of different methods based on the number of devices.

Fig. 3b compares the computational efficiency of the methods. Execution time is critical for real-time applications and high-performance computing. The Game-Theory method shows the highest execution time, starting at around 500 milliseconds for 100 devices and rising sharply to 4000 milliseconds for 1000 devices, making it impractical for real-time applications as network size increases. The Graph method also sees a notable increase in execution time, from about 500 to 3500 milliseconds. In contrast, DFNN, DFBNN, and DFBNN-FL methods demonstrate significantly lower execution times, with DFBNN remaining consistently low across all device counts. These findings highlight the importance of computational efficiency in dynamic, time-sensitive environments and emphasize the need to balance execution time with other performance metrics like interference and sum rate to optimize overall wireless network performance, especially in large-device scenarios.

Fig. 3c shows the sum rate, or total data transmission capacity, achieved by each method as the number of devices increases. The sum rate, measured in Mbps, is crucial for assessing network throughput and efficiency. All methods show an almost linear increase in sum rate with more devices, indicating good scalability. The Graph method has a lower sum rate compared to others, starting around 2000 Mbps and reaching 6×10^4 Mbps at 1000 devices, suggesting it may not fully utilize network capacity in larger networks. In contrast, DFNN, DFBNN, DFBNN-FL, and Game-Theory methods achieve around 14×10^4 Mbps at 1000 devices, demonstrating superior scalability and efficiency for high-density networks. These advanced algorithms are particularly effective for applications requiring high data rates, such as video streaming and large data transfers. This data underscores the necessity of sophisticated algorithms to optimize network performance as device density increases, allowing network designers and engineers to make informed decisions based on specific performance requirements and network conditions.

C. Evaluation of DFBNN vs DFBNN-FL

The results pertaining to DFBNN and DFBNN-FL encompass a comparison of their performance, data efficiency, and privacy benefits. The evaluation metrics include segmentation accuracy, data transfer efficiency, and privacy preservation [21]. Figure 4 provides a visual representation of the comparative performance of DFBNN and DFBNN-FL across these metrics.

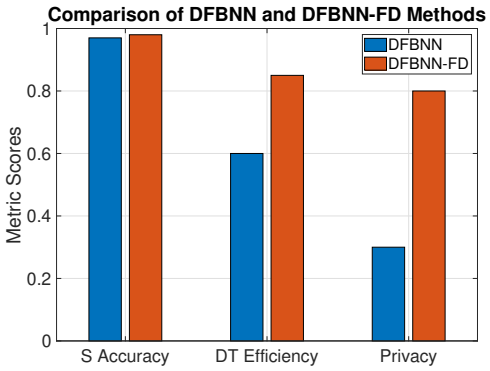


Fig. 4: Comparison of DFBNN and DFBNN-FL Methods: Segmentation Accuracy, Data Transfer Efficiency, and Privacy Preservation.

The results comparing DFBNN and DFBNN-FL were evaluated across three key metrics: segmentation accuracy, data transfer efficiency, and privacy preservation. The DFBNN-FL approach showed a slightly higher segmentation accuracy (0.98) compared to DFBNN (0.97), indicating its effectiveness in correctly segmenting target areas using decentralized data. Data transfer efficiency was significantly improved with DFBNN-FL scoring 0.85 versus 0.6 for DFBNN, due to transmitting model updates instead of raw data, leading to more efficient bandwidth use. Privacy preservation was also markedly better in DFBNN-FL, with a score of 0.8 compared to DFBNN's 0.3, ensuring that sensitive data remained localized and secure. Overall, DFBNN-FL offers high segmentation performance, enhanced data transfer efficiency, and superior privacy preservation, making it a highly advantageous alternative to the traditional DFBNN method.

VI. Conclusion and Future Work

The proposed DL-based method for RRA in 6G networks has significantly improved over traditional methods in managing interference, enhancing data rates, reducing power consumption, and optimizing execution time. The integration of DFBNNs and FL ensures scalable, adaptive, and resilient solutions for efficient resource distribution in dense and dynamic 6G environments. FL in particular, supports distribution and

can be effectively used by the BDIx agents acting as D2DR or D2DMHR. This decentralized model training approach preserves data privacy and reduces communication overhead, enhancing overall network performance.

Future work will extend the current method by incorporating real-world deployment scenarios to validate the robustness and practicality of our approach. We aim to explore the impact of various environmental factors on the model's performance and further refine the FL strategies to enhance data privacy and security. Additionally, the development of hybrid models integrating other advanced AI techniques could provide even more efficient resource allocation solutions. Collaborations with industry partners for field trials will be pursued to ensure our proposed methods' practical applicability and scalability in actual 6G network deployments. Further investigation into the specific roles of D2DR and D2DMHR in the FL process will also be a key focus to maximize their potential in distributed resource allocation.

VII. Acknowledgment

This work has received funding from the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 739578, the ADROIT6G project of the SNS-JU under Grant Agreement No. 101095363, and the Government of the Republic of Cyprus through the Deputy Ministry of Research, Innovation and Digital Policy.

References

[1] S. K. Pattnaik, S. R. Samal, S. Bandopadhaya, K. Swain, S. Choudhury, J. K. Das, A. Mihovska, and V. Poulkov, "Future Wireless Communication Technology towards 6G IoT: An Application-Based Analysis of IoT in Real-Time Location Monitoring of Employees Inside Underground Mines by Using BLE," *Sensors*, vol. 22, p. 3438, 2022.

[2] N. H. Mahmood, S. Böcker, I. Moerman, O. A. López, A. Munari, K. Mikhaylov, F. Clazzer, H. Bartz, O.-S. Park, E. Mercier, *et al.*, "Machine type communications: key drivers and enablers towards the 6g era," *EURASIP Journal on Wireless Communications and Networking*, vol. 2021, no. 1, p. 134, 2021.

[3] X. Li, T. Pan, M. A. Alim, and M. T. Thai, "Resource Allocation in Highly Dynamic Device-to-Device Communication: An Adaptive Set Multi-Cover Approach," *IEEE Transactions on Wireless Communications*, vol. 18, no. 10, pp. 4607–4619, 2019.

[4] K. H. Alzoubi, M. Bin Roslee, and M. A. A. Elgamati, "Interference management of d2d communication in 5g cellular network," in *2019 Symposium on Future Telecommunication Technologies (SOFTT)*, vol. 1, pp. 1–7, 2019.

[5] D. Hong and S. Kim, "Interference management in D2D communication," *Smart Device to Smart Device Communication*, vol. 9783319049632, pp. 89–111, 2014.

[6] M. S. M. Gismalla, A. I. Azmi, M. R. B. Salim, M. F. L. Abdullah, F. Iqbal, W. A. Mabrouk, M. B. Othman, A. Y. Ashyap, and A. S. M. Supa'at, "Survey on Device to Device (D2D) Communication for 5GB/6G Networks: Concept, Applications, Challenges, and Future Directions," *IEEE Access*, vol. 10, no. Vlc, pp. 30792–30821, 2022.

[7] M. Giordani, M. Polese, M. Mezzavilla, S. Rangan, and M. Zorzi, "Toward 6g networks: Use cases and technologies," *IEEE Communications Magazine*, vol. 58, no. 3, pp. 55–61, 2020.

[8] ADROIT6G, "Dai-driven open and programmable architecture for 6g networks," *arXiv preprint arXiv:2403.05277*, 2022.

[9] I. Ioannou, C. Christophorou, V. Vassiliou, and A. Pitsillides, "A novel distributed ai framework with ml for d2d communication in 5g/6g networks," *Computer Networks*, vol. 211, p. 108987, 2022.

[10] A. Graves, "Generating sequences with recurrent neural networks," *arXiv preprint arXiv:1308.0850*, 2013.

[11] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," *Physica D: Nonlinear Phenomena*, vol. 404, p. 132306, 2020.

[12] T. Mikolov, M. Karafiát, L. Burget, J. Cernockỳ, and S. Khudanpur, "Recurrent neural network based language model," 2010.

[13] W. Zaremba, I. Sutskever, and O. Vinyals, "Recurrent neural network regularization," *arXiv preprint arXiv:1409.2329*, 2014.

[14] H. B. McMahan, E. Moore, D. Ramage, and S. Hampson, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.

[15] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 2623–2631, 2019.

[16] T. Lou, H. Tan, Y. Wang, and F. C. Lau, "Minimizing average interference through topology control," in *Algorithms for Sensor Systems: 7th International Symposium on Algorithms for Sensor Systems, Wireless Ad Hoc Networks and Autonomous Mobile Entities, ALGOSENSORS 2011, Saarbrücken, Germany, September 8-9, 2011, Revised Selected Papers 7*, pp. 115–129, Springer, 2012.

[17] I. Ioannou, C. Christophorou, V. Vassiliou, and A. Pitsillides, "A distributed ai/ml framework for d2d transmission mode selection in 5g and beyond," *Computer Networks*, vol. 210, p. 108964, 2022.

[18] T. D. Hoang, L. B. Le, and T. Le-Ngoc, "Resource allocation for d2d communication underlaid cellular networks using graph-based approach," *IEEE Transactions on Wireless Communications*, vol. 15, no. 10, pp. 7099–7113, 2016.

[19] Z. Luan, H. Qu, and J. Zhao, "Using game theory for radio resource management of RRC layer in LTE-A," in *International Conference on Advanced Communication Technology, ICACT*, pp. 41–44, 2012.

[20] Z. A. and D. R. M. . D. M., "Wireless networks design in the era of deep learning: Model-based, ai-based, or both?," *IEEE Transactions on Communications*, vol. 67, no. 10, pp. 7331–7376, 2019.

[21] D. Chai, L. Wang, L. Yang, J. Zhang, K. Chen, and Q. Yang, "A survey for federated learning evaluations: Goals and measures," *IEEE Transactions on Knowledge and Data Engineering*, 2024.