Throughput Prediction in Dense IEEE 802.11 WLANs Using Graph Neural Networks

Rajasekar Mohan *, Aman Cyrano Dsouza, P. Punith, and J. Manikandan

Electronics & Communication Engineering Department, RR Campus, PES University, Bangalore, India;
Email: amanjosephite@gmail.com (A.C.D.), punithprakash2001@gmail.com (P.P.), manikandanj@pes.edu (J.M.)
*Correspondence: rajasekarmohan@pes.edu (R.J.)

***Abstract*—With the growing adaptation of Wi-Fi and the increased possibilities of complementing it with 5G, there is a need to exploit the fullest potential of the IEEE 802.11ac/ax and higher Wireless Local Area Network (WLAN) standards, especially in densely deployed scenarios. Machine learning techniques can be used to predict the performance of WLANs with the vast training data obtained through contemporary network simulators. They are quite useful to predict the throughput in crowded and dynamic deployments of WLANs where hand-crafted solutions may not be feasible. This paper presents a novel, data-driven approach that can contribute to improving the performance of next-generation WLANs. In particular, we employ a Graph Neural Network (GNN) model to predict the performance of Wi-Fi deployments by exploiting topology information and capturing complex wireless interactions. The network simulator, Komondor, is used to simulate different real-life scenarios for generating comprehensive datasets for training the model. Our approach addresses challenges related to energy efficiency, latency, and data rate in WLANs, and the regression model can be used to predict the throughput of a Basic Service Set (BSS) before it is deployed, allowing for better network design and optimization. The findings of this study demonstrated that GNNs can accurately forecast the throughput of BSSs in WLAN deployments in a given region with minimal information. Overall, our proposed approach can significantly influence the choice of topology for deployment, leading to optimal performance in crowded and dynamic WLAN scenarios.**

***Keywords*—IEEE 802.11ac/ax, Overlapping Basic Service Set (OBSS), Komondor, performance prediction, throughput, next-generation Wireless Local Area Networks (WLANs), Graph Neural Network (GNN), International Telecommunication Union (ITU) challenge**

I. INTRODUCTION

Wireless Local Area Networks (WLANs), popularly known as Wi-Fi, are one of the essential elements of the next generation of wireless communication technologies, of which IEEE 802.11, is the most commonly adopted standard [1, 2]. The demand for WLANs has increased in tandem with the rapid growth of mobile devices like laptops, tablets, and smartphones. Furthermore, WLAN systems can provide faster data rates in important areas (hotspots) and are more cost-effective as compared to other forms of broadband, which contributes to their appeal. A significant level of complexity is caused by the limited availability of frequency spectrum in fields like the industrial, medical, and scientific radio bands, as well as the rising throughput demands imposed by new applications with a voracious appetite for bandwidth and the variety of wireless network deployments currently in use [3, 4].

Although wireless communication networks have significantly improved our ability to communicate with one another, they also present a number of challenges that must be addressed and resolved. Such issues grow more significantly in dense environments, resulting in a considerable number of networks that overlap and have coexistence issues [5].

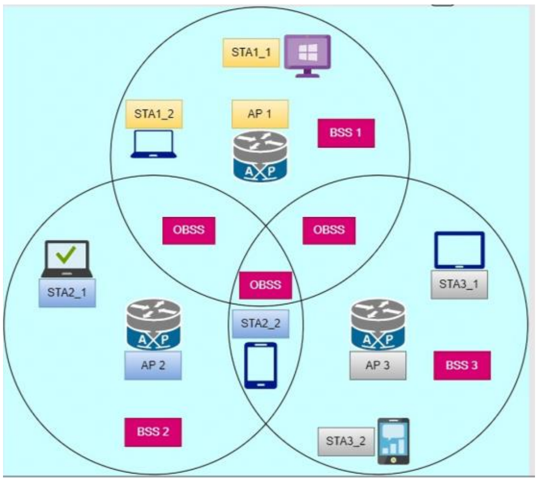


Figure 1. Overlapping BSS.

When two or more BSS can hear one another and are physically close enough to hear one another while using the same channels, they are said to be "overlapping". This is known as the Overlapping Basic Service Set (OBSS) problem (Fig. 1). Such situations result in interference issues and a drop in throughput (performance). [5] outlines the need for intelligent systems in place for OBSS operations where there may not be much coordination among the neighboring APs. This paper focuses on predicting and analyzing the throughput of WLANs [6]

Manuscript received March 6, 2023; revised April 23, 2023; accepted June 13, 2023; published November 10, 2023.

doi: 10.12720/jait.14.6.1198-1205

using Graph Neural Networks (GNNs) [7]. To showcase the effectiveness of our solution, we focus on a synthetic dataset with measurements from multiple dense IEEE 802.11ax WLAN deployments, generated using the Komondor simulator [7–10]. The primary goal of this paper is to accurately predict the throughput of WLANs based on various network parameters, such as the number of access points, the configuration of the network, and the number of users. One of the major works done in this research was data augmentation to increase the size of the dataset, reduce overfitting, and improve the prediction accuracy.

The remainder of the paper is structured as follows: Section II comprises the relevant works in this field. In Section III, we will present the methodology and experimental setup used in this study, such as the dataset generated by the Komondor network simulator and the workings of the GNN model, and explain how they can be used for throughput prediction. We will discuss the performance of the GNN model and present the findings from our trials in Section IV. Finally, in Section V, we will conclude the paper and discuss future work. Through this paper, we hope to demonstrate the potential of GNNs for predicting and analyzing the performance of WLANs. We believe that the results of this study can provide valuable insights into the behavior of these networks and can be used to improve their design and management [2, 4, 6].

II. Literature Review

In order to enhance the optimization of next-generation WLANs, the use of machine learning has emerged as a promising solution [2, 3]. Although its adoption in networks was still in its nascent stage, a great deal of work was required. In response to this, the International Telecommunication Union (ITU) made considerable efforts towards realizing and establishing the groundwork for this study [11]. In these circumstances, the ITU AI/ML in 5G Challenge [12] was launched, and enthusiastic academic scholars and researchers from around the world were called upon to come up with machine learning models to forecast the performance of next-generation WLAN deployments, amongst many others. To support the challenge, an open dataset from a network simulator was obtained and provided [9]. Increasingly, network simulators are used as ML sandboxes to enable future ML-aware communications. The use of network simulators in the field of WLANs has been growing in recent years, with the aim of improving their performance and efficiency. Network simulators can be used for training, testing, and validating machine learning models prior to deployment on live networks [13].

One such simulator is Komondor [8], which is designed to simulate IEEE 802.11ax features in densely deployed environments. The key attributes of the Komondor simulator where it has been shown to have validated performance include its ability to simulate transmissions in real-time, packet by packet, its high event processing rate, dependability, and beginner-friendly interface [8]. The estimation of the throughput of multiple active WLANs in a given location is a complex task that requires accounting for the interactions between these networks and the negative effects of collisions. The technique used by the Komondor simulator to perform this estimation is based on the distinction between “micro” and “macro” interactions, where micro interactions are caused by two or more nodes’ back-off timers expiring at the same time, and macro interactions refer to larger-scale network effects that are caused by factors such as interference and congestion [6]. While these techniques are promising individually, it will take a considerable amount of research to assess the actual performance benefits that result from the integration and application of multiple strategies [1]. With the advancement of technology and the increasing availability of computing power, deep learning methods such as reinforcement learning and supervised learning may help reduce training time and distribute learning tasks [3].

Several learning models have been proposed for Wi-Fi performance prediction so far. In Ref. [14–16], various machine learning techniques such as K-Nearest Neighbor (KNN), Feed Forward Neural Network (FNN), Random Forest (RF), Artificial Neural Network (ANN), and Gradient Boost (GB), among others, have been used to predict the throughput in Wi-Fi networks under varying network and environmental conditions [3]. Another such model is the Graph Neural Network (GNN), Soto *et al.* [7] took part in the challenge [12], which showcased its abilities. Soto *et al.* [7] suggest the use of the Graph Neural Network (GNN) as a novel neural network model that outperforms existing techniques and has a reasonable computing cost [17]. As may be seen, various approaches to modeling Wi-Fi performance have been put forth. The majority of them are mathematical models built on straightforward and simplifying assumptions, which reduces their complexity and leaves out important wireless interactions. GNNs utilize graph-based representations of data to analyze and understand complicated, non-Euclidean data and the relationships within it [7]. They are extremely useful as network topologies, their interactions, and data flow can be easily characterized in the form of a graph. This paper is an extension of the work carried out by Soto *et al.* [7].

Our contribution in this paper includes the following:

- A comprehensive and realistic dataset in meaningful deployment space with matching density of nodes, which can be used for further research and comparisons.
- Variation of features was introduced in the dataset in order to diversify the data so as to be representative of realistic scenarios, allowing the GNN model to adapt to these variations.

III. Methodology

This section provides a comprehensive description of the methods used to conduct the research and obtain the results. The processes and techniques employed to achieve the goals of our study are outlined. This information will enable other researchers to replicate our results and validate our findings. The methodology adopted for this study is based on prior research in the field, and with the intention of forecasting the numerical value of throughput for comparable or any arbitrary deployment, regression techniques were deemed more appropriate than a classification model [14]. The use of GNN as the machine learning technique was selected to ensure that the results accurately demonstrate its potential in predicting throughput in WLANs and stimulate further research in this area [7]. The process depicted in Fig. 2 encompasses a series of steps that are implemented in the following manner:

- Dataset generation: The dataset for this research is generated using the Komondor network simulator, which simulates the behavior of WLANs. The simulator was configured to simulate various scenarios, such as different network topologies, traffic loads, and channel conditions.
- Data pre-processing: The dataset generated is pre-processed to prepare it for the machine learning model. This involves cleaning the data, removing any redundant or irrelevant data, and normalizing the data to ensure that it is suitable for the model.
- Training the GNN model: We utilized the ML model from [7] and trained it on the preprocessed dataset. When the model's performance was not satisfactory, hyper parameters were tuned to optimize the model's performance.
- Model testing and evaluation: The model's performance is evaluated using various performance metrics and tested on unseen data to evaluate its generalization performance using the testing set. The results of the model's performance are analyzed to understand the factors that affect the throughput of a WLAN and to identify any potential areas for improvement.

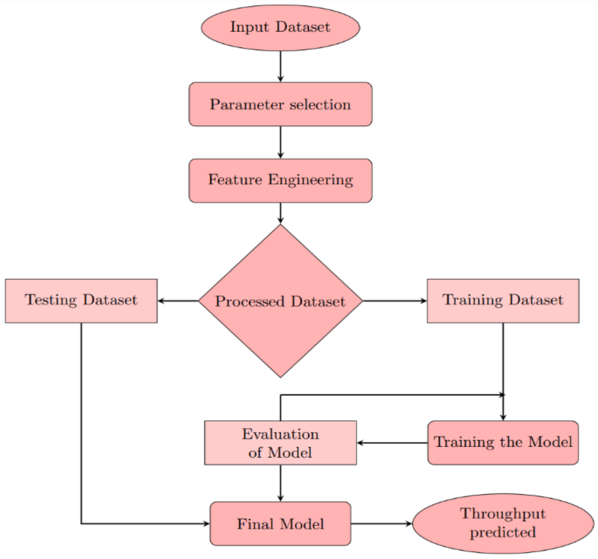


Figure 2. ML model creation workflow.

A. Dataset

The dataset for the model was generated using a network simulator called Komondor. Komondor is a tool that allows a user to simulate different wireless networks, including WLANs, for machine learning research and is capable of simulating various physical layer and Medium Access Control (MAC) layer protocols, as well as different network topologies [8]. It uses a set of algorithms to generate random graphs with specific properties, such as the number of nodes and edges, the distribution of node degrees, and the presence of specific structural patterns [10]. It generates data by simulating transmissions in real time, packet by packet, with the goal of correctly representing WLAN operations in a virtual environment. The virtual environment allows for the creation of different scenarios and interactions, which can make the dataset more representative of real-world data. This can lead to more robust and accurate models.

The simulator can be used to evaluate the performance of wireless networks, test new wireless protocols, and design and optimize wireless systems [13]. Additionally, the data can be visualized using tools such as heat maps and topology diagrams to gain a better understanding of the behavior of wireless devices and networks in the simulated environment. One advantage of using Komondor software is that it allows for the generation of large and diverse datasets, which can be used to train and test machine learning models. It also performs better than most commercial simulators in terms of high event processing rates, dependability, and beginner friendliness [8]. Overall, the use of Komondor software can help to improve the reproducibility and comparability of machine learning research on graph data.

In 2020, the ITU's AI/ML in 5G Challenge [12] presented an open dataset consisting of WLAN measurements obtained from Komondor for the invited participants to develop machine learning models that could predict the performance of future WLAN deployments. The dataset [9], as shown in Table I, consisted of six scenarios, each with 100 deployments, totaling 600 deployments in all. Despite the size of the dataset, it lacked diversity, was limited in its scope, and did not accurately reflect real-world deployments. Previous research suggested data augmentation [18] as a solution to the overfitting problem in models. The proposal was that generating a larger dataset with varied parameters could increase diversity and provide a more representative sample. Consequently, the creation of a more comprehensive dataset was deemed critical for advancing research in this field.

TABLE I. DEPLOYMENT CHARACTERISTICS OF THE ORIGINAL DATASET

Scenario ID	No. of APs	No. of STAs	Central Frequency (GHz)	Channel Bonding Model	Map Size (m^2)
sce 1a	12	[10–20]	5	4	80×60
sce 1b					70×50
sce 1c					60×40
sce 2a	8	[5–10]			60×40
sce 2b					50×30
sce 2c					40×20

Adapted from [7, 9].

Hence, in our work, we increased the size of the dataset fourfold, i.e., to 2400 deployments spread across 24 widely varied realistic scenarios, each of which represents an

OBSS environment made up of specified access points and stations associated with them. Each scenario is unique with respect to the parameters, including the number of APs, central frequency, and channel bonding schemes. The deployments themselves are unique, as across the deployments the number of STAs per AP is varied, as is the position of the APs and STAs too. Komondor takes input features such as node code, node type, (x, y, z) positions of all the APs and STAs inthe deployment, central frequency, channel configurations, channel bonding model, and other features pertaining to the simulations in the virtual network. It then constructs this OBSS, simulates it, and gives us output files containing Throughput, Airtime, Received Signal Strength Indicator (RSSI), Interference, and Signal to Interference plus Noise Ratio (SINR) for each of the APs and STAs [10]. The summary of the characteristics of the resulting dataset is depicted in Table II.

TABLE II. PROPOSED OBSS DEPLOYMENT STRUCTURE

Scenario ID	No. of APs*	Central Frequency (GHz)	Channel Bonding Model**	OBSS Space (m^3)
sce 3a	12	5	4	60×50×10
sce 3b			5	
sce 3c			6	
sce 3d			6	
sce 4a		2.4	4	
sce 4b			5	
sce 4c			6	
sce 4d			6	
sce 5a	10	5	4	50×40×10
sce 5b			5	
sce 5c			6	
sce 5d			6	
sce 6a		2.4	4	
sce 6b			5	
sce 6c			6	
sce 6d			6	
sce 7a	8	5	4	40×30×10
sce 7b			5	
sce 7c			6	
sce 7d			6	
sce 8a		2.4	4	
sce 8b			5	
sce 8c			6	
sce 8d			6	

*The number of STAs per AP varies in the range [10–20]. **Refer Table V.

The dataset is divided into two parts, with 80% allocated for training and 20% for testing purposes. The training set is then divided into two subsets, with 80% utilized for training and 20% reserved for validation, as shown in Table III.

TABLE III. SPLIT UP OF THE PROPOSED DATASET

Dataset	Values
Total No. of Deployments	2400
Training split	1536
Validation split	384
Testing split	480

The Komondor simulator provides numerous features at our disposal, but for training our model, we have only considered those that have an impact on the throughput of APs and STAs. Table IV shows all the simulation settings involved in Komondor [10].

TABLE IV. KOMONDOR SIMULATION SETTINGS

Parameters	Definition	Values
Tx pwr	Transmit power that a transmitter antenna produces at its output	20 dBm
CCA	Clear channel assessment value	−82 dBm
CE	Capture effect threshold	10 dB
Traffic load	Data traffic generation rate	Full buffer traffic
L	Length of the data payload in packets	12000 bits
CW	Contention window	16
N agg	Number of packets aggregated per transmission	64
CE model	Capture effect model	0
PIFS	PCF Interframe Space (PIFS)	0
Traffic model	Stochastic model of the traffic flows or data sources in a communication network	99

Adapted from [10].

B. Graph Neural Network

Graph Neural Networks (GNNs) are a type of neural network model designed to handle data that is structured as graphs [17]. They are used for a range of tasks, such as node classification, link prediction [19], and graph classification. GNNs excel at analyzing complex, non-Euclidean data [19] by utilizing graph-based representations of the information, making them ideal for analyzing networks such as WLANs. This is because they can capture the complex relationships present in the graph structure and the spatial information inherent in WLAN deployments [7]. The graph-based representations of the data allow for more accurate and efficient analysis compared to traditional neural networks, which are only equipped to handle vector or matrix inputs [17]. GNN is an optimizable transformation on all attributes of the graph (nodes, edges, global-context) that preserves graph symmetries (permutation invariances) [20]. It can learn these attributes and predict the output based on the input graph structure [7].

The dataset generated with Komondor first undergoes pre-processing, which involves replacing any anomalous values, such as Inf and NaN, with 0. We then generate a graph representation of the dataset that can be processed by the GNN model. The graph structure can be used to depict wireless networks: G = (V, E) [19, 20], where nodes such as APs and STAs are represented by V and wireless links are represented by E. This involves transforming each node and edge into a vector representation that includes features such as node attributes, edge attributes, and topological information [7] (Table V). The model then processes the data through a series of layers, each of which performs a set of operations on the input vectors. These operations typically include convolutional filters, pooling [19], and non-linear activation functions. The aim of these layers is to extract useful features from the input data and facilitate the propagation of information between nodes in the graph [20]. The flow of information between nodes occurs through the use of edges. After this process, the updated node representations are utilized to make predictions or classify data. This updating of node representations is accomplished through a sequence of layers (Fig. 3), each of which is trained using a supervised learning algorithm [21].

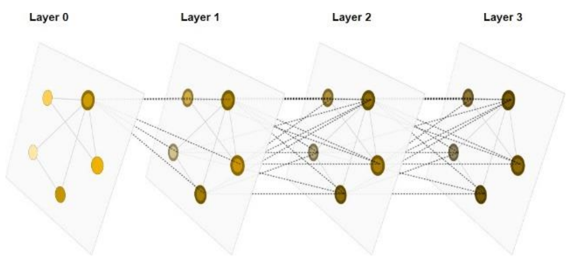


Figure 3. Graph neural network [20].

TABLE V. NODE AND EDGE FEATURES

Feature	Definition	Values
Node Features		
Node Type	Defines the given node	AP:0; STA:1
Node Code	Alphabetic code given to AP and STA	-
(x,y,z)	Coordinates of APs and STAs in 3-dimensions	-
CF	Band used for central frequency	2.4 / 5 GHz
CB Model	Channel bonding model (4: Log2 always-max (DCB): TX in the larger channel range allowed by the log2 mapping, 5: Log2 Always-max (DCB) with optimal MCS: picks the channel range + MCS providing max throughput, 6: Log2 probabilistic uniform: pickwith same probability any available channel range)	-
C p	The index of the primary channel for transmission	[1–7]
C min	The index of the lowest channel used for transmission	[1–7]
C max	The index of the highest channel used for transmission	[1–7]
SINR	Signal to Interference plus Noise Ratio	-
Airtime	Percentage of time each AP occupies each of the assigned channels	-
Edge Features		
Edge Type	Defines edge between APs and STAs	AP:1; STA:0
Distance	Euclidean distance between source and destination	-
RSSI	Received Signal Strength Indicator	-
Interference	Inter-AP interference sensed from every AP	-

Adapted from [18].

The operations of GNNs can be broken down into two main steps: message passing and updating. In the message passing stage [19], each node in the graph communicates a "feature vector" to its neighboring nodes, which is a combination of the node's original feature vector and the feature vectors of its neighbors. The purpose of this step is to gather information from the node's surroundings. The model starts with an initial representation for each node and then updates the representation of subsequent nodes incrementally based on the representations of its neighboring nodes [19–21]. This repetition continues for multiple iterations until a final representation of each node is achieved, which is then utilized for the intended task. In the updating stage, the feature vectors of all nodes are updated based on the messages they received (Fig. 4). This process is repeated until the feature vectors converge [20]. To obtain the output of a GNN, a readout function is applied to the final feature vectors of the nodes.

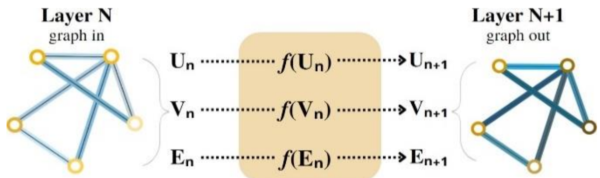


Figure 4. A single layer of a GNN model [20].

The GNN model operates on the MetaLayer architecture (Fig. 5) [22]. To accurately update the features of the input graphs, the model utilizes a node model and an edge model. The input graph is first fed into the input layer of the model. From there, the data is then processed by the edge model, which performs the necessary updates on the edges, such as changes in position coordinates and interference between nodes from one layer to the next. The edge model is configured with a Multilayer Perceptron [20] that applies a linear transformation to the incoming data, followed by a ReLU activation layer.

The edge model is designed to gather all the edge features along the connected edges. This is achieved by utilizing two dense layers. The edge model then passes on the aggregated data to the node model, which is composed of two MLPs (Multilayer Perceptrons). Each MLP in the node model has two dense layers, which act as linear layers that transform input features into output features by utilizing weight matrices and biases. The node model has two blocks. The first block, the aggregate function block, calculates the average of the embeddings for all neighboring nodes and combines the edge features into the node features. The second block, known as the update function, takes the aggregated values from the aggregate function and updates the state of the nodes within the graph layer by layer. In our model, the mean is used as the aggregation method.

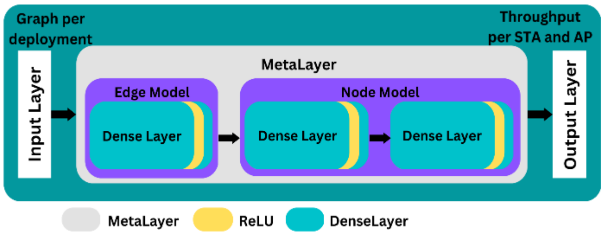


Figure 5. Metalayer architecture [7].

Overall, GNN is a remarkable tool that can be leveraged to analyze and comprehend datasets created by the Komondor software. With the utilization of node attributes and connectivity structures as its features, GNNs have the ability to identify crucial patterns and connections within the data. This makes them highly efficient at resolving a vast array of tasks that are based on graphs. GNNs offer an optimal solution for processing graph-structured data, and their capability to grasp complex relationships is what makes them an ideal choice for examining networks such as WLANs.

IV. Result and Discussion

With the datasets generated using Komondor for the varied and realistic scenarios, the GNN model was trained with an 80:20 ratio between train-test datasets. The comprehensive and exhaustive dataset of 2400 deployments with the critical parameters varying was used to train the GNN model and enable it to capture the complex inter-relationships between the stations and the access points. The results obtained with respect to the performance metrics to evaluate the model are tabulated for different combinations of testing in Tables VI and VII. The GNN model, having been trained with different deployment space sizes and parameters, has been able to generalize well. The effective throughput of the OBSS (i.e., the throughputs of the APs) predicted with reference to the expected values (obtained through Komondor) falls within the band of accuracy between 72 and 81% over 480 of a variety of test cases. It is imperative to observe here that the GNN model proposed by Soto *et al.* [7, 18] had the best RMSE score of 8.73 Mbps as compared to the proposed work's best RMSE score of 4.45 Mbps. With the help of the enhanced datasets and the selection of the critical parameters, the model was able to mimic the behavior of the OBSS deployment of the network elements far better than others. The improved results give renewed confidence to pursue further tweaking of the model's configuration parameters to achieve better results for much more diverse deployments while factoring in other related concerns [3].

Following the training of the GNN model, we administer the test data in the form of a graph consisting of nodes and edges along with their attributes. The model then predicts the target values for each node, which in our case is the throughput for each AP and STA. The results of the throughput prediction using GNN on the dataset generated by Komondor were analyzed using the following evaluation metrics to assess the performance of the model.

- RMSE: Root Mean Square Error measures the average difference between values predicted by the model and the actual values.

$$RMSE = \sqrt{\frac{1}{n}\{\sum_{i=1}^{n}(y-y')^2\}} \quad (1)$$

y—observations, y'—predicted values and i—No. of observations

- MAE: Mean Absolute Error measures the average of the absolute error values, i.e., the difference between predicted and the actual values.

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y-y'| \quad (2)$$

- MAPE: Mean Absolute Percentage Error gives a measure of the relative error between predicted values and actual values expressed in terms of a percentage.

$$MAPE = \frac{100\%}{n}\sum_{i=1}^{n}\frac{|y-y'|}{y} \quad (3)$$

- Prediction Accuracy: It tells us how close the predicted value is to the simulated value.

$$Accuracy = 100 - MAPE \quad (4)$$

From Table II, it is observed that there is a decreasing trend in the number of APs as the space constraints increase, while the network density shows an increasing trend. Analysis of Figs. 6 and 7 reveals that the prediction performance of the proposed model improves as the network scenarios become denser. Specifically, Fig. 6 shows a general decreasing trend in the average Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) scores, while Fig. 7 shows a decreasing trend in the average Mean Absolute Percentage Error (MAPE) scores and an increasing trend in the predicted accuracy with increasing network density. The minimum and maximum values of MAE, RMSE, MAPE, and prediction accuracy are indicated in Tables VI and VII, respectively. These trends indicate that the model can accurately predict network performance in scenarios with higher network densities and more limited deployment space constraints. It can be concluded that the proposed model is effective for predicting network performance in dense network scenarios.

TABLE VI. Test Results I

Scenario	Avg. MAE [Mbps] (min–max)	Avg. RMSE [Mbps] (min–max)
Sce 3	5.55 (2.09–15.24)	7.85 (3.25–20.21)
Sce 4	3.96 (1.63–10.09)	6.13 (2.69–15.45)
Sce 5	3.86 (1.70–9.82)	5.45 (2.65–13.24)
Sce 6	3.07 (1.44–7.19)	4.45 (2.40–9.69)
Sce 7	3.62 (1.19–10.45)	4.96 (1.99–13.22)
Sce 8	3.20 (1.24–8.49)	4.62 (1.94–11.41)

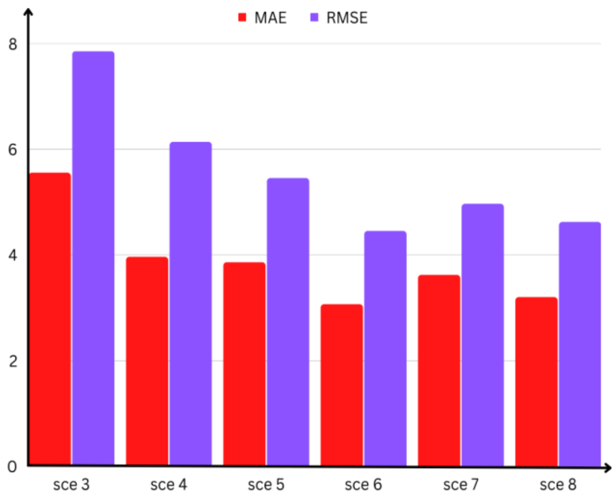


Figure 6. MAE and RMSE across different test scenarios.

TABLE VII. TEST RESULTS II

Scenario	Avg. MAPE (%) (min–max)	Avg. Predicted Accuracy (%) (min–max)
Sce 3	26.85 (22.62–39.48)	73.15 (60.52–79.77)
Sce 4	27.74 (18.84–43.96)	72.26 (56.04–81.16)
Sce 5	25.62 (19.60–38.14)	74.38 (61.86–80.40)
Sce 6	23.73 (14.47–41.49)	76.27 (58.51–85.53)
Sce 7	19.72 (12.55–36.12	80.28 (63.88–87.45)
Sce 8	19.80 (12.65–38.02)	80.20 (61.97–87.35)

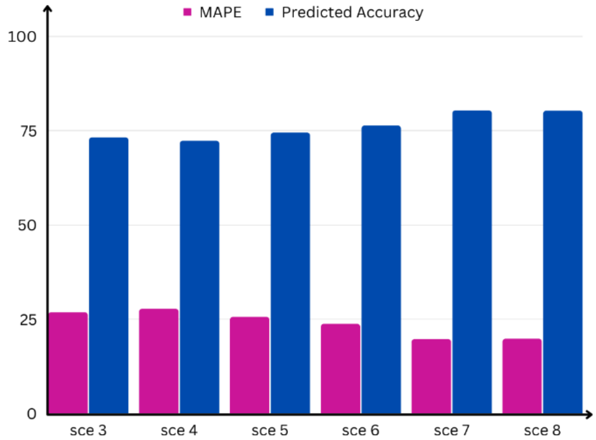
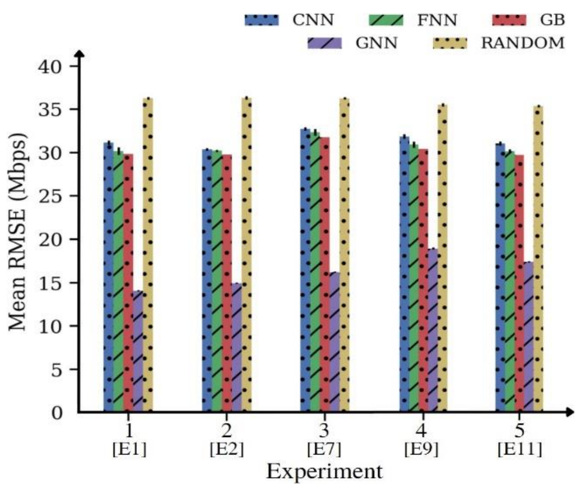


Fig. 7. MAPE and Predicted Accuracy across different test scenarios.

Fig. 8 depicts the average or mean RMSE values by Soto *et al*. [7] for various other important techniques, viz., CNN, FNN, GB, GNN, and Random Forest, for data generated using Komondor. The best five experiments (E1, E2, E7, E9, and E11), i.e., an extract of the 16 experiments conducted, are used here for comparison. It is observed that the proposed approach in this paper for building the datasets and choosing the configuration parameters far outweighs the previous approaches, as the best mean RMSE score is brought down to as low as 4.45 Mbps.



*E1, E2, E7, E9, E11 refer to the study [7]

Figure 8. Mean RMSE across experiments comparing different ML models [7].

Besides, about 17 different configurations used by the authors in their earlier work [23], as shown in Fig. 9, indicate that the performance of the current work exhibits consistent and superior RMSE values, thereby substantiating the improvement of the results for such dense deployments of WLANs.

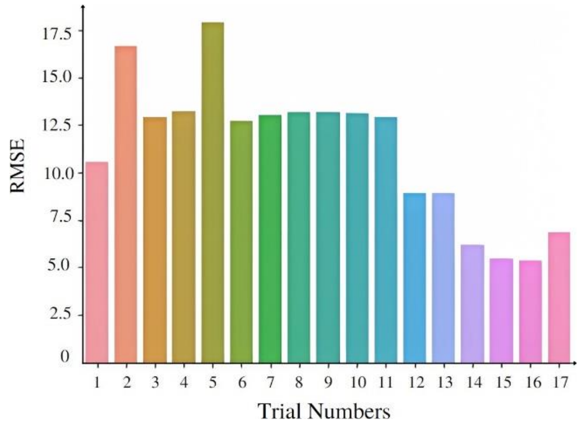


Figure 9. RMSE comparison of different trials [23].

V. CONCLUSION

WLANs are now densely deployed in a variety of real-world settings, like homes, offices, airports, and universities, and, as they continue to grow exponentially, there will be a demand to optimize their throughput before they are deployed in a setting. By employing ML techniques, a network designer can potentially enhance the network by dynamically adjusting transmit power, channel selection, resource allocation, and modulation schemes to optimize and improve the performance of networks in these scenarios. In this context, network simulators like Komondor can be of great utility and contribute to evaluating possible situations in a real network. Using simulators, we can test, validate, and anticipate possible use cases and configurations (even ML optimizations) before they occur in the network, thus providing confidence. However, the cost of simulations may become a big issue since, typically, the more reliable a simulator is, the more expensive it is to run. In this paper, we investigate the utilization of machine learning in the compact and efficient characterization of wireless networks. In particular, we have used an ML model that can abstract all the complexity behind a simulator and replace it with a single model that can be more efficient than a simulator when executed. The final goal is to apply ML techniques to wireless communication networks, and if we can find ML models with high accuracy for reproducing simulated data, then it is very likely that we can do the same in real networks. We can use this knowledge to make the necessary configuration changes in the event of a drop in performance.

In future work, incorporating more sophisticated Graph Neural Network (GNN) architectures, such as Graph Attention Networks (GATs) or Graph Convolutional Networks (GCNs) [21], which have been found to perform better in graph-based tasks, could be one way to enhance the model employed in the study. We would also recommend taking a look at Graph-Sage for possible improvements [19, 21]. Another possible improvement would be to introduce the model to real-world data [16], which will enable it to make better predictions and learn more about real-world circumstances. Finally, we can work on scaling the model to handle larger datasets and more complex wireless networks, which would enable the model to be used in a wider range of real-world applications.

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Rajasekar conceptualized the problem statement and designed the experimentation process. Aman conducted extensive literature reviews, meticulously maintained version control of the data, refined datasets for experiments, and undertook the documentation of the project's developments. Punith generated the dataset, adapted the machine learning model, and carried out the training process. J. Manikandan was instrumental in orienting the team's effort, giving deep insights, and changing the course of work when needed most. All authors participated in reviewing the manuscript.

Acknowledgment

We are grateful to Francesc Wilhemi of Nokia Bell Labs, Germany, for helping us understand the Komondor software and supporting us at different stages. We also thank Mr. Rahul Gangarapu and Mr. Sahishnu Suresh from PES University for their support in dataset generation during the initial stages of the research.

References

[1] B. Bellalta, "IEEE 802.11ax: High-efficiency WLANS," *IEEE Wireless Communications*, vol. 23, no. 1, pp. 38–46, February 2016. doi: 10.1109/MWC.2016.7422404

[2] F. Wilhelmi, S. Barrachina-Munoz, B. Bellalta, C. Cano, A. Jonsson, and V. Ram, "A flexible machine-learning-aware architecture for future WLANs," *IEEE Communications Magazine*, vol. 58, no. 3, pp. 25–31, March 2020. doi: 10.1109/MCOM.001.1900637

[3] S. Szott *et al.*, "Wi-Fi meets ML: A survey on improving IEEE 802.11 performance with machine learning," *IEEE Communications Surveys Tutorials*, vol. 24, no. 3, pp. 1843–1893, 2022. doi: 10.1109/COMST.2022.3179242

[4] M.-S. Kim, T. Ropitault, S. Lee and N. Golmie, "A throughput study for channel bonding in IEEE 802.11ac networks," *IEEE Communications Letters*, vol. 21, no. 12, pp. 2682–2685, Dec. 2017. doi: 10.1109/LCOMM.2017.2747511

[5] Z. Zhong, P. Kulkarni, F. Cao, Z. Fan, and S. Armour, "Issues and challenges in dense WiFi networks," in *Proc. 2015 International Wireless Communications and Mobile Computing Conference (IWCMC)*, Dubrovnik, Croatia, 2015, pp. 947–951. doi: 10.1109/IWCMC.2015.7289210

[6] B. Bellalta, "Throughput analysis in high density WLANs," *IEEE Communications Letters*, vol. 21, no. 3, pp. 592–595, March 2017. doi: 10.1109/LCOMM.2016.2632109

[7] P. Soto, M. Camelo, K. Mets, F. Wilhelmi, D. Goez, L. A. Fletscher, N. Gaviria, P. Hellinckx, J. F. Botero, and S. Latre, "ATARI: A graph convolutional neural network approach for performance prediction in next-generation WLANs," *Sensors*, vol. 21, 4321, 2021.

[8] S. Barrachina-Munoz, F. Wilhelmi, I. Selinis, and B. Bellalta, "Komondor: A wireless network simulator for next-generation high-density WLANs," *Wireless Days (WD)*, pp. 1–8, 2019. doi: 10.1109/WD.2019.8734225

[9] F. Wilhelmi. (2020). [ITU-T AI Challenge] Input/Output of project "Improving the capacity of IEEE 802.11 WLANs through machine learning". [Online]. Available: https://zenodo.org/record/4106127

[10] S. Barrachina-Munoz and F. Wilhelmi. (2018). Komondor user's guide (manual). [Online]. Available: https://usermanual.wiki/Document/komondoruserguide.34621546 4/pdf

[11] ITU-T. (2019). Recommendation Y.3172. Architectural framework for machine learning in future networks including IMT-2020. [Online]. Available: https://www.itu.int/rec/T-REC-Y.3172/en

[12] ITU. (2020). ITU AI/ML in 5G challenge. [Online]. Available: https://www.itu.int/en/ITU- T/AI/challenge/2020/Pages/default.as px

[13] F. Wilhelmi, M. Carrascosa, C. Cano, A. Jonsson, V. Ram, and B. Bellalta, "Usage of network simulators in machine-learning-assisted 5G/6G networks," *IEEE Wireless Communications*, vol. 28, no. 1, pp. 160–166, February 2021. doi: 10.1109/MWC.001.2000206

[14] R. Mohan, K. V. Ramnan, and J. Manikandan, "Machine learning approaches for predicting throughput of very high and extreme high throughput WLANs in dense deployments," in *Proc. 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, 2021, pp. 1–6. doi: 10.1109/ICCCNT51525.2021.9580

[15] F. Wilhemi, R. Mohan, and K. V. Ramnan *et al.*, "Machine learning for performance prediction of channel bonding in next-generation IEEE 802.11 WLANs," *ITU Journal on Future and Evolving Technologies*, vol. 2, issue 4, 2021.

[16] P. Marques and C. E. Sansores, "Machine learning models for the prediction of Wi-Fi links performance using a CityLab testbed," in *Proc. 8th International Workshop on ADVANCEs in ICT Infrastructures and Services*, pp. 1–8, 2020.

[17] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, Jan. 2009. doi: 10.1109/TNN.2008.2005605

[18] P. Soto, D. Goez, N. Gaviria, and M. Camelo. (2020). ITU-ML5G-PS-013: Improving the capacity of IEEE 802.11 WLANs through machine learning. GitHub. [Online]. Available: https://github.com/ITU-AI-ML-in-5G-Challenge/ITU-ML5G-PS-013-ATARI/blob/master

[19] J. Zhou *et al.*, "Graph neural networks: A review of methods and applications," *ScienceDirect*, Apr. 08, 2021. https://doi.org/10.1016/j.aiopen.2021.01.001

[20] B. Sanchez-Lengeling, E. Reif, A. Pearce, and A. B. Wiltschko. (2021). A gentle introduction to graph neural networks. *Distill*. [Online]. Available: https://distill.pub/2021/gnn-intro

[21] S. Karagiannakos. (2021). Best graph neural network architectures: GCN, GAT, MPNN and more. *AI Summer*. [Online]. Available: https://theaisummer.com/gnn-architectures/

[22] PyTorch.torch.nn module. What is torch.nn really? PyTorch Tutorials 1.13.1+cu117 documentation. [Online]. Available: https://pytorch.org/tutorials.html

[23] R. Mohan, K. V. Ramnan, and J. Manikandan, "Predicting the throughput of next generation IEEE 802.11 WLANs in dense deployments," *Procedia Computer Science*, vol. 203, pp. 24–31, 2022. doi: 10.1016/j.procs.2022.07.006