v1: 2 August 2024

Research Article

Generative Artificial Intelligence Using Machine Learning on Wireless Ad Hoc Networks

Preprinted: 16 July 2024
Peer-approved: 2 August 2024

Qeios, Vol. 6 (2024)
ISSN: 2632-3834

Antonio Cortés Castillo[1]

1. Facultad de Informática, Electrónica y Comunicación, Departamento de Informática, Universidad de Panamá, Panama

In this article, we discuss the use of Generative Artificial Intelligence (GenAI) to improve the efficiency and performance of access to wireless points located in various spaces and specific places, which allows interaction with wireless mesh networks and enables the use of mobile devices to access all types of information in internal environments. Furthermore, we propose the use of generative neural networks, which are one of the pillars of GenAI, since they use a methodology from the perspective of Machine Learning that allows analysis of a large amount of data and detection of certain types of patterns that help in the better placement of access points for improved reception and connectivity. Images (heat maps), access point locations, positioning points, and bandwidth are analyzed, allowing new information to be created. On the other hand, to understand and model the general architecture of the wireless Ad-Hoc network, we use two processes that are part of neural networks, such as Multilayer Perceptron (MLP), and the Radial Basis Function (RBF), which is a function of predictors or independent variables or input variables that allows the prediction error in the output variables of the wireless network architecture to be reduced. Using these two processes does help reduce blind spots in those internal places where the wireless signal does not reach, resulting in a signal drop. Improving internal scenarios with wireless Ad-Hoc networks is what is required for better functioning and performance of the network infrastructure.

Corresponding author: Antonio Cortés Castillo, antonio.cortes@up.ac.pa

1. Introduction

The past few years have seen an evolution in wireless communication networks, which has allowed the number of nodes in wireless networks to grow exponentially. Concurrently, with the rise and assimilation of the Internet of Things (IoT) and the integration of new technologies such as Cloud Computing, Edge Computing, and Software Defined Networks [1], it has been possible to integrate a variety of services and applications into wireless networks, which means that Internet Service Providers (ISPs) have to improve their network infrastructures to provide greater bandwidth in the shortest possible time because of the huge real-time data usage.

Likewise, with this large number of access points or active nodes, in which you have a variety of mobile users moving in all directions in small and specific spaces, accessing all types of valuable and diversified information, security [2] when accessing these nodes is of very high relevance because collisions at the frequency level between these nodes can sometimes occur, which weakens the bandwidth and access to data, generating blind or dead spots where the wireless signal does not reach and therefore allows unwanted intruders to access the wireless network, generating some type of fraud or infecting the network with a worm, trojan, rumors...etc.

Given this situation that arises with access to data in wireless networks, we have given ourselves the task of using GAI. This is an area within artificial intelligence that allows, from existing data that is generated through a series of metrics in wireless network infrastructures, the generation of new original data and analysis of these data that help to improve performance and security in wireless networks.

In this research article, we set ourselves the following objectives:

- Analyze how wireless Ad-Hoc network infrastructures improve their performance, become more efficient, and produce better benefits for users using Generative Artificial Intelligence employing a Machine Learning approach, taking Generative Neural Networks as a reference.
- Collect those predictors to later parameterize them, such as signal levels, signal-to-noise ratio, signal-to-interference ratio, number of access points, noise levels, frequency band coverage, network coverage, PHY mode of the 802.11 protocols (a, b, g, n, ac, ax), which allow the neural network to be modeled from the processes called Multilayer Perceptron (MLP) and the Radial Basis Function (RBF).
- Create a model of the framework for the wireless Ad-Hoc network from Machine Learning, taking into consideration the independent or input variables, which will be represented through Generative Neural Networks visualized through MLP and RBF processes, respectively.

In this same direction, the significance of normalizing and metricizing the independent variables must be emphasized, since it helps to create a model with predictive and residual value from the entry of these covariates

 doi.org/10.32388/KON6WA.2

or number of units. At the same time, in Fig. 1, we can have several hidden layers, and for each of these layers, several units in these hidden layers help the performance of the model by activating various functions, for example, the hyperbolic tangent, among others.

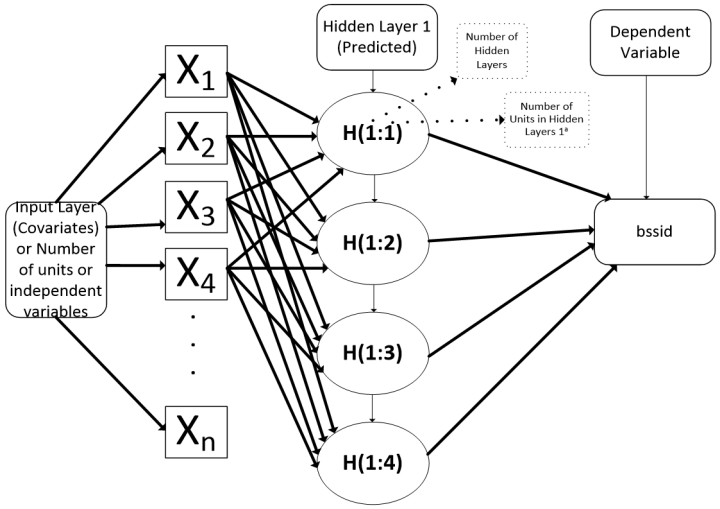

Fig. 1. Model with prediction value.

Indistinctly, regardless of the model that we use for the wireless network infrastructure, applying GAI and utilizing the two algorithms suggested in the neural network, MLP and RBF, respectively, should allow us, from the normalized independent variables, to select predictive values that enable the designed wireless network infrastructure to improve.

In turn, highly innovative generative machine artificial intelligence (GenAI) models, powered by Machine Learning and neural networks, can enable the digital transformation of organizations, thereby increasing productivity, efficiency, and problem-solving capabilities. Then, based on these predicted values, original content is generated to address complex challenges, allowing GenAI to revolutionize not just how organizations operate, but how innovation is structured.

However, if we take into consideration the increased use of Machine Learning and GenAI-level applications with large data volumes and computational complexity, this allows unprecedented demands on wireless infrastructures, requiring reliable, high-bandwidth, low-latency data, significantly higher structured cabling densities, and advanced cooling methods.

So far, wireless network infrastructure is preparing for artificial intelligence, and users need powerful and innovative network infrastructure solutions to help with the design, implementation, and scalability of back-end, front-end, and ambient network architecture storage for complex high-performance computing (HPC).

We also resort to the accelerator models of GenAI and Machine Learning, which consist of training (where a new aspect is learned) and inference (applying what is learned to new data). Typically, these neural and Machine Learning networks mimic the architecture and functions of the human brain to learn and generate new original knowledge by considering the analysis of patterns, nuances, and the general and variable characteristics of a massive and complex data set. Likewise, large language models (LLM), such as ChatGPT and Google Bard, are clear examples of these GenAI models, which are trained on large amounts of data to understand and generate plausible linguistic responses.

Ultimately, general-purpose CPUs that perform I/O and control operations sequentially cannot effectively extract large volumes of data in parallel from multiple sources and processes quickly enough, so Deep Accelerated Learning and GenAI use parallel processing-based graphics processing units (GPUs) to execute thousands of high-performance calculations.

After the Introduction section, this report consists of part 2; thereafter, related works; section 3, related to Materials and Methods; section 4, also proposed and related to the Experimentation phase; in section 5, Discussion, the results obtained from the data experimentation are analyzed, and conclusions and suggestions for further research are included in section 6.

2. Related Work

In [3], having signaled from several cells implies having frequency interference between different channels, which causes calls made through one of these channels to be distorted. Therefore, the authors propose to eliminate this

noise in the channel and use a multilayer perceptron network trained with Wilcoxon learning, which will help stabilize the inputs and outputs produced in the system through the wireless network system.

On the other hand, in [4], the authors use the multi-layer perceptron network applied to Wireless Sensory Networks to analyze security in this type of network. This is used to analyze the CSMA protocol located at the physical level, specifically at the MAC layer, to analyze the various denial of service attacks launched by unethical hackers. Likewise, in [5], the authors propose using a model based on multilayer perceptron-type networks to be trained from a set of data in such a way that it helps them optimize network congestion and use of the bandwidth. The network type they use is wireless mesh backbone networks.

Therefore, in [6], the authors use artificial neural networks by making use of the Multilayer Perceptron for the detection and classification of a diversity of attacks on wireless Ad-Hoc networks. In turn, in [7], the authors propose that there are malicious nodes that can destroy data in Mobile Ad-Hoc networks, and that to predict how much damage these viruses, malware, fake news, rumors, exploits can cause, the networks use neural networks, specifically the Multilayer Perceptron, to identify which of these perceptrons are the most harmful and can cause the most damage to the network. In this same direction, in [8], possible predictions are outlined using the Multilayer Perceptron to improve mobility in wireless networks, since the processes of connection and reconnection to the various access points, whether internal or external, of a certain space decrease network performance.

Similarly, in [9], the authors use multilayer distributed perceptron (MLPC) to detect various types of distributed denial of service (DDoS) attacks in modern vehicular communication systems. They use Apache Spark-level simulation processes to create the MLPC and Amazon Web Services (AWS) to train and determine the attack time at the distributed neural network level. However, in [10], the authors use a Multilayer Perceptron neural network with 4 hidden layers with 20 hidden units to detect dead links between nodes in complex networks.

3. Materials and Methods

We outline our methodology in this section for Generative Artificial Intelligence using Machine Learning in wireless Ad-Hoc networks. Selecting and classifying metrics or parameters are essentially two main tasks. In Fig. 2, the data collection task, training phase, and validation phase make up the order in which we propose to implement our system.

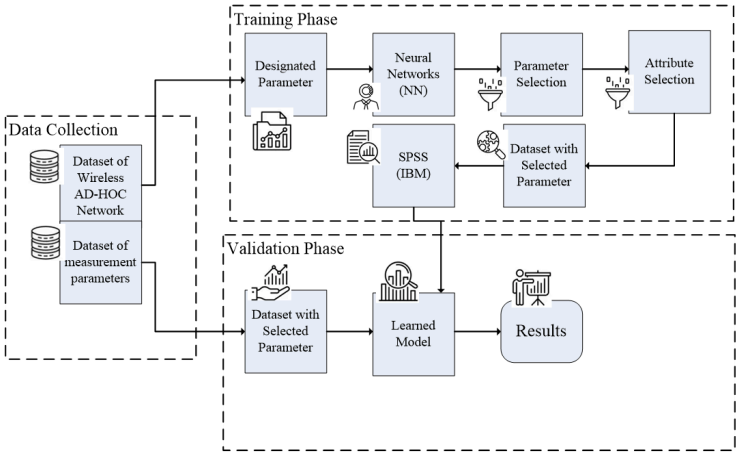

Fig. 2. Our proposed architecture

To validate our proposed architecture, we use experimental data, and this data is tested through simulation, in which active and offline access points are considered by creating data repositories of Ad-Hoc wireless networks that store a series of metrics. The set of processed metrics is introduced into the feature selection method. The metric selection method, contained in the learning of neural networks (NN), results in a list of important metrics that are classified according to the level of importance. To validate this shortened list of metrics, we use Machine Learning, represented using Multilayer Perceptron (MLP) neural networks and the Radial Basis Function (RBF).

3.1. Experimental data

The Ad-Hoc wireless network data set not only contains discrete type values but also numerical type values and character strings, which allow generating scalar and nominal type measurements. At the same time, the processing of experimental data must be carried out in advance. Therefore, two main stages are presented for the processing of experimental data, where first we must carefully select the metrics to transform them into attributes and subsequently assign them a numerical value, and second, the normalization stage of these

experimental data. It is worth mentioning that some variables, such as security, are considered as a string of characters, while others, such as BSSID, down_speed, up_load_speed, wireless_transmit_rate, among others, are considered to have numeric values. Once all the attribute values are converted to integer values, each of the attributes is linearly normalized between zero and one. Equation (1) displays the normalization formula.

$$Ni = \frac{xi - \min(x)}{\max(x) - \min(x)\prime} \qquad (1)$$

Where:

- N_i = normalization value.
- X_i = corresponds to the attribute's value.
- min(x) = minimal attribute value.
- max(x) = maximum attribute value.

3.2. Setting metrics

Metric learning, which includes metric extraction and selection [11], involves the capacity to simulate data flow patterns between various access points using unprocessed data from additional measures, known as "metric learning." To illustrate the relationship between detection performance and data traffic model quality, it is crucial to provide metrics to facilitate learning [12].

Metric selection and extraction are distinct processes. When new, non-redundant metrics are extracted from an area containing existing, raw metrics, the process is known as metric extraction [13]. In most cases, there are differences between the newly generated metrics and the metrics that have not yet been processed. However, choosing many metrics from the unprocessed metrics space is what metric selection entails. Consequently, without any modification, the generated metrics are only chosen from the original values.

In the same way, fewer new metrics produced from the original measurements are the goals of both metric extraction and selection. The metric selection in this article is done directly using neural networks (NN), while the metric extraction is done implicitly using the statistical analysis tool SPSS 25. We use NN to improve the bandwidth and load balance that is generated from the access points, in such a way that the areas, zones, or dead spots where there is no wireless signal reception can be reduced. With NN, we can select some metrics based on the heuristic weights of NN learning, which are crucial for learning from the neural network. We trained our NN using thirteen to fourteen variables, with two hidden layers and hyperbolic tangent and sigmoid functions. It is important to note that the more hidden layers there are, and for each of these layers, the number of units, covariates, or independent variables increases, the slower the model to be executed becomes when executed, and even the results to be obtained take longer than the usual estimate. The neural network model is depicted in Figure 3, which allows assessing the variation between the current value of a metric and its expected value, with each corresponding middle layer being represented by bias 1 and bias 2, respectively.

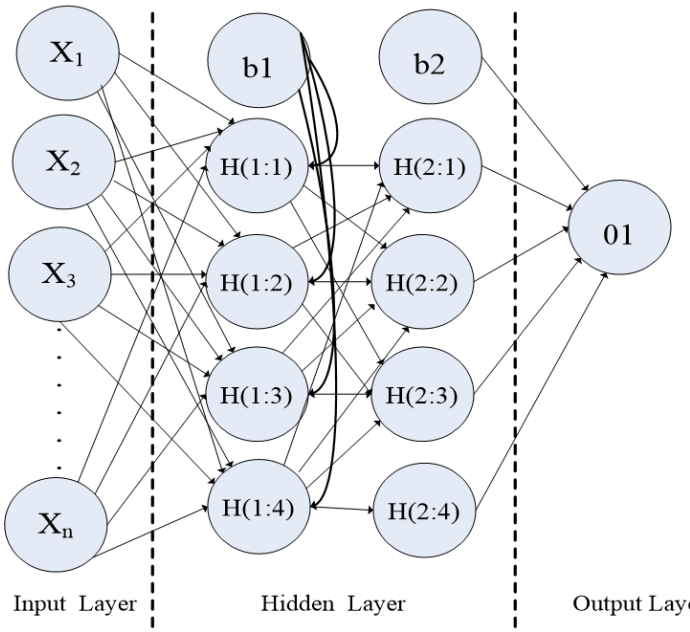

Fig. 3. Neural network (NN) model

doi.org/10.32388/KON6WA.2

In this same direction, the input layer in our proposed model will be determined by the metrics that are identified in closed scenarios at the level of wireless communications. Obviously, each of these metrics will be assigned a weight that allows the cluster of middle layers and the conglomerate number for each layer to be determined. The above will generate an output that allows you to validate and optimize the best metric for the wireless link. The weights of the start and end nodes, represented by α_{ij} as seen in Fig. 4, being extremely little or nonexistent, mean that the input metrics, corresponding to X_j, are meaningless for further propagation.

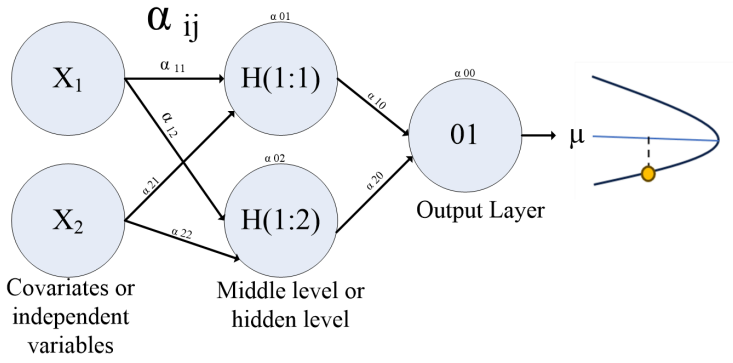

Fig. 4. The start and end nodes' weights represented (α_{ij})

Therefore, we only consider the weights in the middle layer; thus, one hidden layer is sufficient. For each unit, covariant, or independent variable, we define its importance value, as expressed in equation (2).

$$Vj = \sum_{i=1}^{\beta} |\alpha ij| \qquad (2)$$

In which:

- β = represents the quantity of metrics in the middle level.
- V_j = allows selecting the most relevant metrics, ordered according to the input metrics ordered descending. We select some metrics that have a Vj value larger than a threshold value (μ).

3.3. Metric classification

Once the metrics for our model were selected (see Table 1), we used the Multilayer Perceptron algorithm and the Radial Basic Functions. These two algorithms are part of Neural Networks, which in turn are part of supervised learning, allowing evaluation of a certain number of independent variables from which one or several hidden layers and one or several metrics are generated that allow validating the proposed model to manage better links at the wireless communications level in a closed environment. As a classifier, we chose SPSS 25 since it contains a neural network called Multilayer Perceptron (MLP) that allows the replacement of original metrics from a supervised method with a step for the hierarchical extraction of features. Likewise, the MLP neural network is like an NN, as shown in Fig. 5.

doi.org/10.32388/KON6WA.2

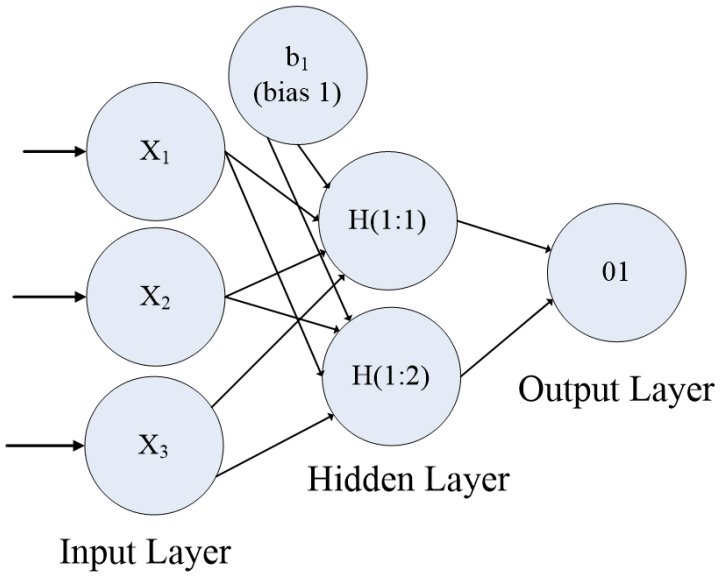

Fig. 5. MLP model

Compared to the NN model, the MLP neural network is characterized by several inputs, one or two levels of hidden layers, two or four levels of the quantity of metrics in the middle layer, and a single output. The nodes in the center, meanwhile, represent a set of novel metrics with high dimensions. This architecture allows data reuse after complex calculations have been executed. Likewise, the MLP neural network strives to learn effectively from a small amount of data to build deep networks by aggregating that data. However, at the middle layer, the results of each training session can fall hierarchically. By using several new measurements at various depth levels, this structure—known as MLP—can learn. The article's suggested MLP architecture is depicted in Fig. 6.

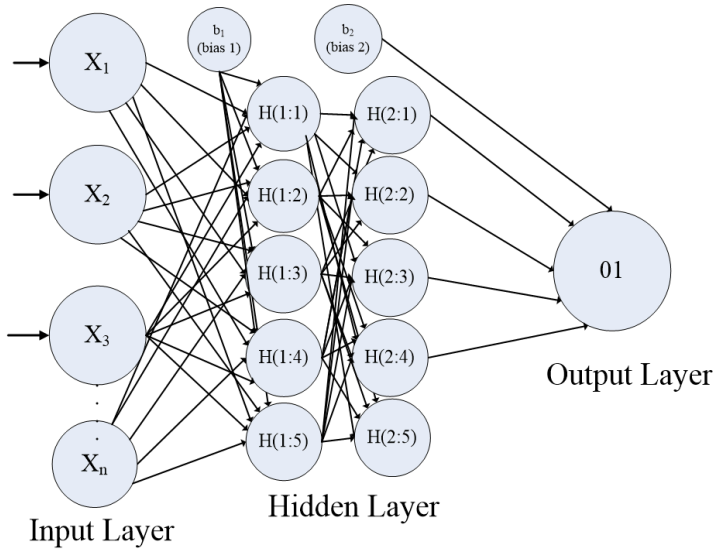

Fig. 6. Proposed MLP architecture

4. Experimentation

In this section, we show the different stages in detail that have been carried out at the experiment level and the results obtained from them. We first explain our tests and data set preparation, followed by the experimental results and analysis.

doi.org/10.32388/KON6WA.2

4.1. Experimental Setup

We constructed a data document using IBM SPSS version 25, with a total of 13 variables, to demonstrate that our proposed scheme can improve not only the bandwidth but also the data load balance between the interconnection of different access points in an Ad-Hoc wireless network when using Generative Artificial Intelligence. Table 1 shows the summary of the metrics used for testing.

ID	Metric name	Type	Width	Decimal	Label	Columns	Align	Measure	Role
1	down_load	Numeric	3	0	Download_speed	14	Right	Nominal	Input
2	upload_speed	Numeric	3	0	upload_speed	12	Right	Nominal	Input
3	wireless_transmit_rate	Numeric	3	0	wireless_transmit_rate	16	Right	Nominal	Input
4	signal_level	Numeric	2	0	signal_level	10	Right	Nominal	Input
5	signal_to_noise_radio	Numeric	2	0	signal_to_noise_radio	17	Right	Nominal	Input
6	signal_to_interference_radio	Numeric	2	0	signal_to_interference_radio	20	Right	Nominal	Input
7	noise_level	Numeric	2	0	noise_level	10	Right	Nominal	Input
8	İssues_with_SNR	Numeric	2	0	İssues_with_SNR	14	Right	Nominal	Input
9	low_signal_level	Numeric	2	0	low_signal_level	13	Right	Nominal	Input
10	high_level_of_noise	Numeric	2	0	high_level_of_noise	14	Right	Nominal	Input
11	overlapping_channels_SIR	Numeric	2	0	overlapping_channels_SIR	19	Right	Nominal	Input
12	low_download_rate	Numeric	2	0	low_download_rate	15	Right	Nominal	Input
13	low_upload_rate	Numeric	2	0	low_upload_rate	13	Right	Nominal	Input

Table 1. Summary of metrics used in testing

Two experimental tests are described below:

1. Experimental test No. 1

In this first test, 13 variables with active access points and a double hidden layer neural network containing 4 hidden units are used.

There is the following network information, which is seen in the following Table 2.

Input Layer	Hidden Layer(s)			Output Layer	
Covariates	**Number of Hidden Layers**	**Number of Units in Hidden Layer 1[a]**	**Number of Units in Hidden Layer 2[a]**	**Dependent Variables**	**Number of Units**
13	2	4	4	1	1

Table 2. Network Information for Experimental test No. 1

It is important to highlight that at the level of the input layer, the rescaling method for standardized covariates or min-max normalization is presented, and the one we use is the standardized one. At the hidden layer level, we have the activation function, which is used as information of the network, and we use the hyperbolic tangent. On the other hand, at the output layer level, we have the scale-dependent rescaling method, and in this case, we use adjusted normalization. Similarly, at the level of the activation function, we use the hyperbolic tangent, and as the error function, the sum of the squares is used.

In Fig.7, you can see how the neural network is constructed from the Multilayer Perceptron (MLP).

doi.org/10.32388/KON6WA.2

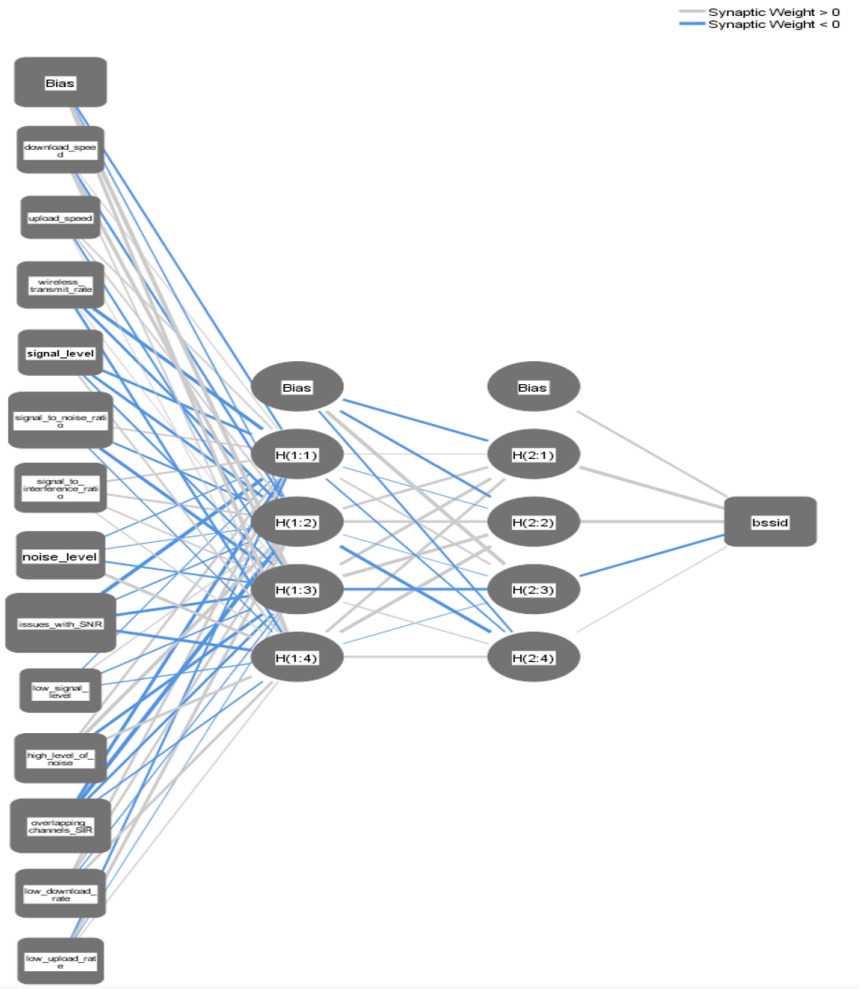

Fig. 7. Neural network based on MLP for 2 hidden layers and 4 units for the 1st and 2nd hidden layers for Experimental test No. 1

In Table 3, the estimation of the metrics is observed at the neural network level.

doi.org/10.32388/KON6WA.2

Parameter Estimates

Predictor		Predicted: Hidden Layer 1 H(1:1)	H(1:2)	H(1:3)	H(1:4)	Hidden Layer 2 H(2:1)	H(2:2)	H(2:3)	H(2:4)	Output Layer bssid
Input Layer	(Bias)	-.269	-.125	.709	.495					
	download_speed	.043	-.295	.562	.131					
	upload_speed	.136	.145	-.231	.015					
	wireless_transmit_rate	-.675	-.360	.010	-.300					
	signal_level	-.488	-.455	-.250	.220					
	signal_to_noise_ratio	.264	-.299	-.578	-.093					
	signal_to_interference_ratio	.315	.333	.227	.092					
	noise_level	-.129	-.070	-.286	.517					
	issues_with_SNR	-.775	-.152	-.578	-.620					
	low_signal_level	-.077	.083	-.173	-.117					
	high_level_of_noise	.300	.663	-.499	.471					
	overlapping_channels_SIR	-.498	-.768	-.346	-.170					
	low_download_rate	.347	.296	-.074	.355					
	low_upload_rate	-.280	.401	-.025	.071					
Hidden Layer 1	(Bias)					-.392	-.364	.710	-.245	
	H(1:1)					.046	-.045	.247	-.169	
	H(1:2)					.427	.490	-.016	-.611	
	H(1:3)					.508	.750	-.614	.127	
	H(1:4)					.378	.706	-.039	.355	
Hidden Layer 2	(Bias)									.330
	H(2:1)									.772
	H(2:2)									.718
	H(2:3)									-.391
	H(2:4)									.069

Table 3. Metric estimation for Experimental test No. 1

In Table No. 3, we can observe the estimated values for the issues_with_SNR metric in hidden layer one and the number of elements for each of the layers.

In Table No. 4, we can see the level of importance of the independent variables that is derived from the estimation of the metrics.

Independent Variables	Importance	Normalized Importance
download_speed	.059	36.1%
upload_speed	.023	14.0%
wireless_transmit_rate	.059	35.9%
signal_level	.047	28.9%
signal_to_noise_ratio	.137	83.6%
signal_to_interference_ratio	.082	50.1%
noise_level	.066	40.3%
issues_with_SNR	.164	100.0%
low_signal_level	.034	21.0%
high_level_of_noise	.082	49.9%
overlapping_channels_SIR	.120	73.2%
low_download_rate	.072	44.1%
low_upload_rate	.054	32.9%

Table 4. Level of importance of independent variables for Experimental test No. 1

We can observe, in Table 4, that the independent variable that prevails 100% is the one called issues_with_SNR with an importance level of .164. It is followed by signal_to_noise_ratio in relevance level with 83.6% and an importance level of .137, and in third place, the overlapping_channels_SIR metric with 73.2%, whose relevance level is .120. With these values, we can deduce that the relationship between the signal-to-noise ratio and the overlap between the various channels covered by the wireless spectrum considerably affects the performance at the level of bandwidth and the load balance between the various access points that may be connected at a certain time.

doi.org/10.32388/KON6WA.2

2. Experimental Test No. 2

In this second test, 13 variables with active access points and a double hidden layer neural network containing 8 hidden units are used.

There is the following network information, which is seen in the following Table 5.

Input Layer	Hidden Layer(s)			Output Layer	
Covariates	Number of Hidden Layers	Number of Units in Hidden Layer 1[a]	Number of Units in Hidden Layer 2[a]	Dependent Variables	Number of Units
13	2	8	8	1	1

Table 5. Network Information for Experimental Test No. 2

In this new network information, what changes with respect to experimental test No. 1 is the amount of the number of units in the first and second hidden layers, since we go from 4 to 8 units in the number of hidden layers, of which there are 2.

In Fig. 8, you can see how the neural network is constructed from the Multilayer Perceptron (MLP).

doi.org/10.32388/KON6WA.2

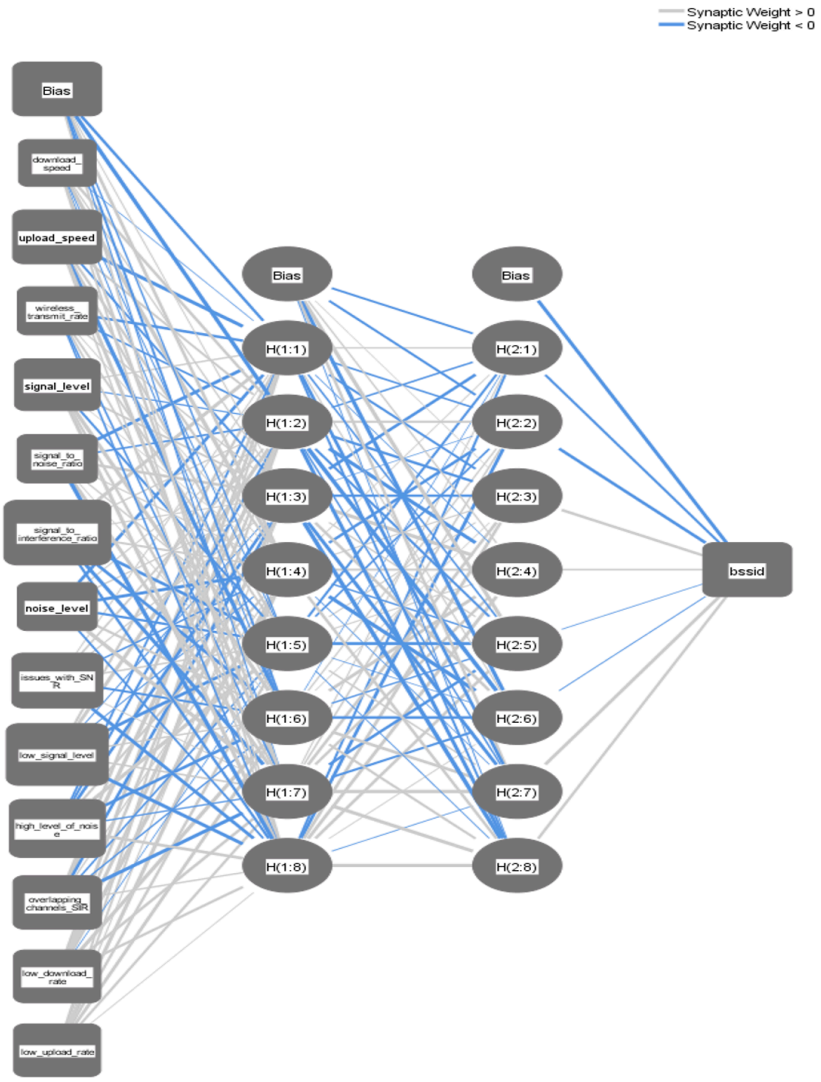

Fig. 8. Neural network based on MLP for 2 hidden layers and 8 units for the 1st and 2nd hidden layers for Experimental Test No. 2

To the extent that the number of hidden layers increases and with it the number of units for each of these layers, as we can see in Fig. 8, the level of complexity of the Multilayer Perceptron becomes increasingly complex, which helps obtain a more accurate and efficient predictive value.

In Table 6, the estimation of the metrics is observed at the neural network level.

doi.org/10.32388/KON6WA.2

Parameter Estimates

Predictor		Predicted																
		Hidden Layer 1								Hidden Layer 2								Output Layer
		H(1:1)	H(1:2)	H(1:3)	H(1:4)	H(1:5)	H(1:6)	H(1:7)	H(1:8)	H(2:1)	H(2:2)	H(2:3)	H(2:4)	H(2:5)	H(2:6)	H(2:7)	H(2:8)	bssid
Input Layer	(Bias)	-.348	-.591	.072	.380	-.140	-.249	-.103	.184									
	download_speed	-.013	.346	.118	.541	-.075	-.086	.309	.194									
	upload_speed	-.563	.247	-.245	.275	.258	-.534	.454	.142									
	wireless_transmit_rate	-.400	-.232	.136	.026	.238	.182	-.164	-.179									
	signal_level	.152	-.126	.246	.014	-.257	.543	.283	.504									
	signal_to_noise_ratio	-.515	-.235	.452	.116	.141	.052	.463	-.622									
	signal_to_interference_ratio	.508	.124	.099	.257	-.466	-.431	-.527	-.324									
	noise_level	-.429	.091	.135	-.526	-.338	.331	.633	-.226									
	issues_with_SNR	.076	.183	.111	-.171	.175	-.288	.062	-.478									
	low_signal_level	.271	.488	-.070	-.082	-.487	.171	.214	-.538									
	high_level_of_noise	-.224	-.265	-.393	.270	.564	-.441	-.142	.444									
	overlapping_channels_SIR	.358	.541	-.374	.124	-.169	-.146	-.562	.162									
	low_download_rate	.417	.337	.440	.674	-.021	.536	.379	.347									
	low_upload_rate	.246	.234	.098	.506	.330	.437	.300	.043									
Hidden Layer 1	(Bias)									-.266	-.295	.004	.132	.528	-.517	.335	-.064	
	H(1:1)									.152	-.196	-.273	-.025	-.168	.094	-.416	-.240	
	H(1:2)									-.187	.308	-.382	-.597	-.271	-.684	-.151	-.481	
	H(1:3)									-.430	-.007	-.372	.631	.137	.383	-.477	-.681	
	H(1:4)									.050	-.386	.113	.333	-.170	-.629	-.023	.344	
	H(1:5)									-.258	-.186	.082	.097	-.459	-.021	.153	-.064	
	H(1:6)									-.008	.118	.018	.480	-.153	-.422	.459	.292	
	H(1:7)									.082	.163	.369	.336	-.384	-.296	.621	.564	
	H(1:8)									-.257	-.430	.403	.439	.393	.061	-.061	.620	
Hidden Layer 2	(Bias)																	-.485
	H(2:1)																	-.276
	H(2:2)																	-.432
	H(2:3)																	.397
	H(2:4)																	.224
	H(2:5)																	-.047
	H(2:6)																	-.065
	H(2:7)																	.452
	H(2:8)																	.337

Table 6. Metric estimation for Experimental test No. 2

In Table No. 6, we can observe the estimated values for the signal_to_interference_ratio metric in hidden layer one and the number of elements for each of the layers.

In Table No. 7, we can see the level of importance of the independent variables that is derived from the estimation of the metrics.

Independent Variables	Importance	Normalized Importance
download_speed	.028	17.6%
upload_speed	.079	49.6%
wireless_transmit_rate	.037	23.4%
signal_level	.064	40.1%
signal_to_noise_ratio	.039	24.8%
signal_to_interference_ratio	.159	100.0%
noise_level	.038	23.6%
issues_with_SNR	.088	55.1%
low_signal_level	.138	86.8%
high_level_of_noise	.110	68.9%
overlapping_channels_SIR	.077	48.2%
low_download_rate	.072	45.0%
low_upload_rate	.071	44.2%

Table 7. Importance level of the independent variables for Experimental test No. 2

In Table 7, we can observe that the level of importance, which has a normalized importance of 100.0%, falls on the independent variable signal_to_interference_ratio, which effectively shows us that the overlap that may occur in the waves emitted by the access points can affect performance in terms of bandwidth and the management of large volumes of data in the wireless Ad-Hoc network. Another metric that indicates low performance in this type of network is related to the metric called low_signal_level, which has an importance level of .138 and a normalized importance of 86.8%, since this is due precisely to the collision at the level of waves that the access points may present.

4.2. Dataset

The grouping of the independent variables rests in the data repository that we have called SSID-GWi-Fi-Hot Access Points.sav, which has been built and designed under the standards of the SPSS statistical tool version 25 of

doi.org/10.32388/KON6WA.2

IBM. Each of the covariates is assigned an identifier with its respective name, data type, field width, decimals, and a label with which each of these independent variables is identified, a value field, the width field of the column, the alignment of the field, type of measurement, and the role that the field will play in the measurement process, which is normally an output.

4.3. Performance Evaluation

In the case of experimental test No. 1, the following results are obtained, which we can see in the following Fig. 9, in which the independent variables are assessed based on the importance of normalization.

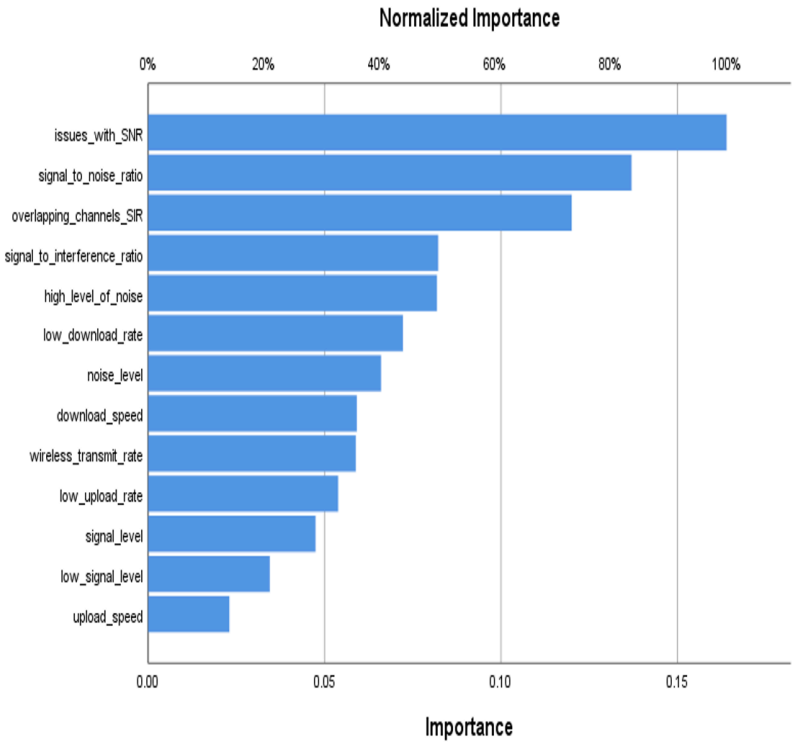

Fig. 9. Importance of Normalized for experimental test No. 1

In Fig. 9, we can see that from the 0.15 importance level, the metric or independent variable issues_with_SNR obtains 100% of the normalized importance, which means that the events that occur in the Ad-Hoc wireless network have an intrinsic relationship with the relationship present between the noise and the signal in the wireless communication medium, in our case, the overlap that may occur in the waves emanating from each of the access points.

On the other hand, in the case of experimental test No. 2, the following results are obtained, which we can observe in Fig. 10, and where the covariates are assessed based on their level of importance and the normalized importance.

doi.org/10.32388/KON6WA.2

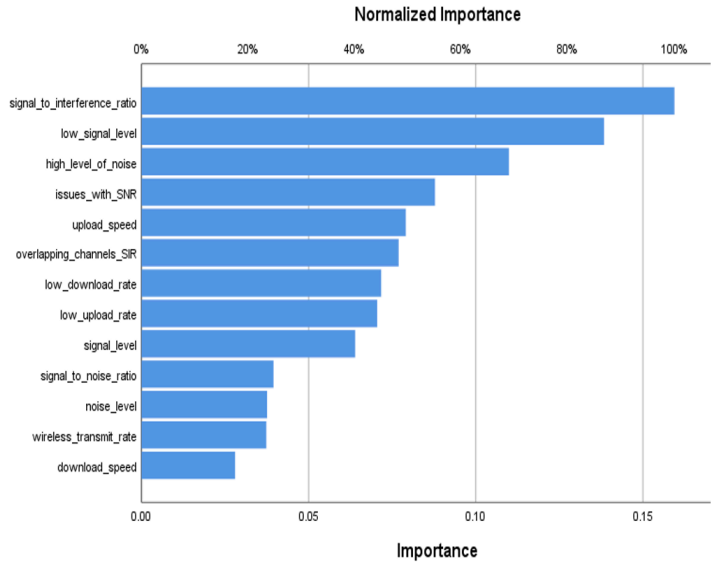

Fig. 10. Importance of Normalization for experimental test No. 2

Indeed, in Fig. 10, we can see how the metric signal_to_interference_ratio, with an importance level of 0.15, obtains a normalization importance of 100%, which shows that the average signal interference can decrease the performance of the signals in Ad-Hoc networks. Similarly, there are two other independent variables that are key in this type of Ad-Hoc wireless network scenario, such as low_signal_level and high_level_of_noise, with importance levels of .138 and .110, respectively, and which in turn represent a normalization importance of 86.8% and 68.9%, consecutively, which shows that these metrics also affect this type of network.

5. Discussion

In the case of experimental test Nº 1, we used the following linear equation (3).

$$y = 0.59 + 0.81^{*}x \qquad (3)$$

This linear function allows us to generate a predicted value regarding the dependent variable, bssid, which contains a single unit that corresponds to the output layer, as we can see in Fig. 11.

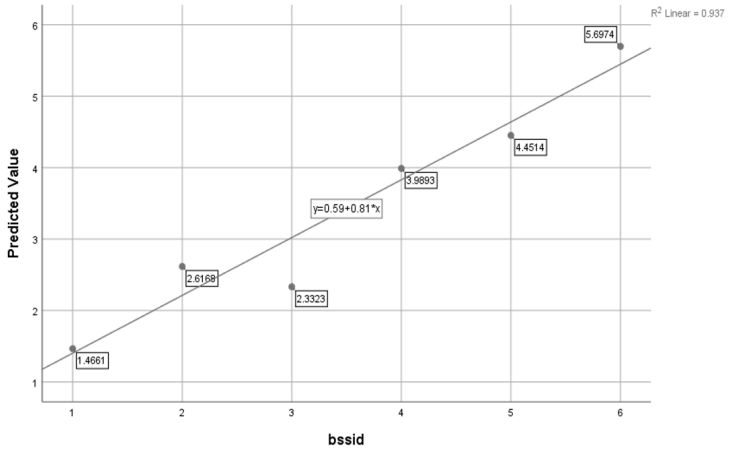

Fig. 11. Predictive value regarding the dependent variable bssid (Experimental Test Nº1)

In Fig. 11, we can observe that in the regression line, there are a total of 4 values that are above the line, which means that they are significant values that represent a regression model and that serve as a response to a set of

doi.org/10.32388/KON6WA.2

data, which, in our case, is identified by the independent variables. This allows us to quantify the relationship between one or more predictor variables and a response variable (bssid).

We also consider the residual values and their relation to the predictive values, as we can see in Fig. 12.

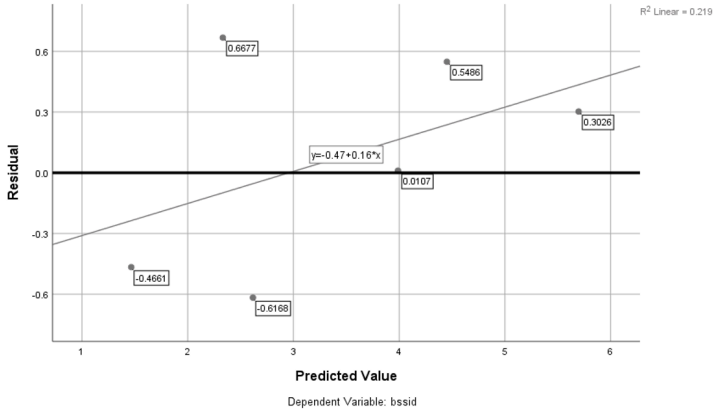

Fig. 12. Residual values with respect to the predictive values of a bssid response variable

In Fig. 12, there is the following linear equation (4).

$$y = 0.47 + 0.16^{*}x \qquad (4)$$

At the same time, this linear function allows us to observe the relationship between the independent variables and the dependent variables. In this case, in Fig. 12, we can see that there are only 2 values that are above the regression line, which means a lower correlation between the observed values and a predicted value in the regression analysis.

In this same direction, in experimental test No. 2, we have the following linear equation (5).

$$y = 2.25 + 0.35^{*}x \qquad (5)$$

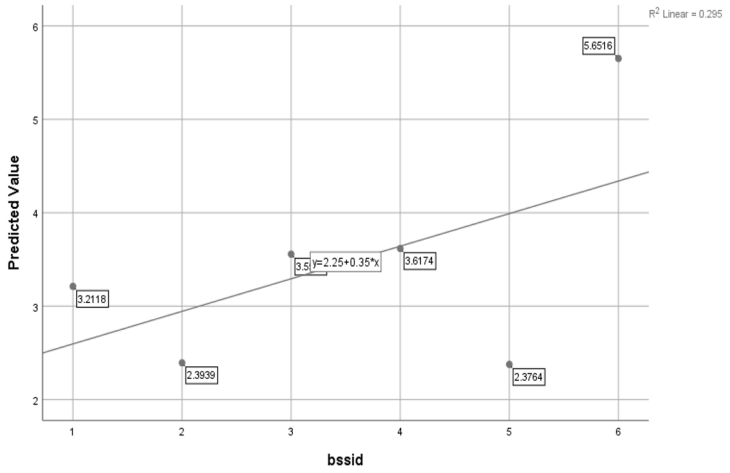

Fig. 13. Predictive value regarding the dependent variable bssid (Experimental Test Nº2)

In Fig. 13, we can see that there are only 3 optimal values above the regression line, which tells us about the level between the independent variables and the response variable. Therefore, the regression model indicates the level of granularity that may exist within your data set.

On the other hand, we also have the relationship between residual values and predictive values, as we see in Fig. 14.

doi.org/10.32388/KON6WA.2

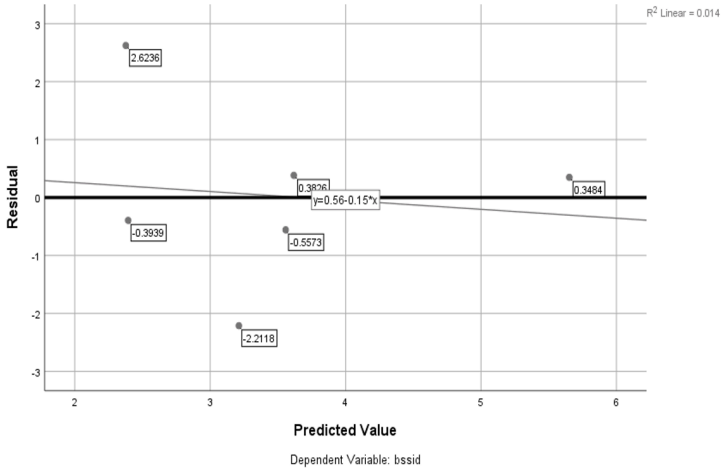

Fig. 14. Residual values with respect to the predictive values of a bssid response variable

In Fig. 14, we can see that the regression line is inclined in a decreasing manner, in which only 3 optimal values are observed. Thus, it seems that the relationship of one or more independent variables with respect to the dependent variable is low, or at least does not completely satisfy the response variable.

In Table 8, we make a comparison between the linear equations and the regression analysis, considering the data obtained from experimental tests 1 and 2, respectively.

	Linear equation	**Regression analysis (R^2 Linear)**	**Percentage of regression analysis**
Experimental test N_0.1			
Predicted Value vs. bssid	y = 0.59 +0.81*x	0.937	93.7%
Residual vs. Predicted Value	y = 0.47 + 0.16 *x	0.219	21.9%
Experimental test N_0.2			
Predicted Value vs. bssid	y = 2.25 + 0.35*x	0.295	29.5%
Residual vs. Predicted Value	y = 0.56 – 0.15*x	0.014	1.4%

Table 8. Linear equations vs. regression analysis between Experimental test N_0.1 and N_0.2, respectively

In Table 8, we can see that the percentage of the regression analysis obtained by experimental test No. 1 is 93.7%, which is equivalent to 0.937 in linear regression. This is followed by the 29.5% of experimental test No. 2, with a 0.295. Likewise, experimental test No. 1 presents a better relationship of its independent variables with respect to the response variable than the results presented in experimental test No. 2. Therefore, with 2 hidden layers and 4 elements for each layer, better results are obtained regarding the relationship of the covariates with the dependent variable than without using 4 hidden layers and 8 elements for each of these layers.

6. Conclusion

Generative Artificial Intelligence (GenAI), within the context of Machine Learning and through its neural networks such as the Multilayer Perceptron (MLP) and the Radial Basis Function (RBF), allows us to carry out a series of analyses based on a group of data, in our case known as independent variables and response variables. The contribution of this work lies in selecting those variables or metrics that are captured through a simulator suitable for scenarios that contain Ad-Hoc wireless access points in combination with Mesh networks and converting them into measurable variables that allow improvement of this type of scenario starting from this set of independent variables.

As mentioned previously, 2 experimental scenarios have been used, in which tests were carried out with MLP, using 2 hidden layers and between 4 to 8 numbers of elements for each of these hidden layers, respectively. The

doi.org/10.32388/KON6WA.2

results show that to the extent that we increase the number of hidden layers and the number of elements for each of these hidden layers, the percentage of the Regression Analysis (linear R2) tends to decrease, which indicates that this percentage of the relationship between these independent variables and the response variable does not obtain the best optimal solution but simply a predictive or approximate value that allows providing a solution to the problem presented by the Ad-Hoc wireless network scenarios, such as bandwidth, blind or dead spots, and the data load balance that must be managed at a given time.

As we can see, there are a series of metrics or predictors that affect these scenarios of Ad-Hoc wireless networks, among which it is worth mentioning the average interference in the signal, the average noise in the signal, the channel overlap, the low signal level, a high noise level in the communication medium, among others.

As future work, it would be necessary to experiment with more levels of layers and numbers of elements per layer; for example, 4 hidden layers and 16 elements for each hidden layer based on the metrics or variables that are identified in Information-Centric Networks [14]. Although for this type of experiment, more computational equipment will be required to allow the results obtained to be satisfactorily managed.

Statements and Declarations

Data Availability

The dataset used in this study (SSID-GWi-Fi-Hot Access Points.sav) is publicly available at https://github.com/acortescastillo/SSID-GWi-Fi-Hot-Access-Points.sav-.git. Detailed variable definitions (name, type, field width, labels, measurement roles, etc.) are provided within the repository.

References

1. ^*Cheng Y, Yin B, Zhang S (2021). "Machine Learning for Wireless Networking: The Next Frontier." IEEE Wireless Communications. **28**(6):176–183. doi:10.1109/MWC.001.2100005.*
2. ^*Shen W, Wang W (2018). "Node Identification in Wireless Network Based on Convolution-al Neural Network." 2018 14th International Conference on Computational Intelligence and Security (CIS). 238–241. doi:10.1109/CIS2018.2018.00059.*
3. ^*Guha DR, Patra SK (2010). "Cochannel Interference Minimization Using Wilcoxon Mul-tilayer Perceptron Neural Network." 2010 International Conference on Recent Trends in Information, Telecommunication and Computing. 145–149. doi:10.1109/ITC.2010.50.*
4. ^*Rao PS, Varma KVSRP, Satapati RA, Vamsidhar E (2010). "Multilayer perceptron based secure media access control protocol for wireless sensor networks." 2010 IEEE International Conference on Computational Intelligence and Computing Research. 1–5. doi:10.1109/ICCIC.2010.5705769.*
5. ^*Eyobu OS, Edwinah K (2023). "A Machine Learning-Based Routing Approach for Wire-less Mesh Backbone Networks." IEEE Access. **11**:49509–49518. doi:10.1109/ACCESS.2023.3277431.*
6. ^*Canêdo DRC, Romariz ARSR (2019). "Intrusion Detection System in Ad Hoc Net-works with Artificial Neural Networks and Algorithm K-Means." IEEE Latin America Transactions. **17**(07):1109–1115. doi:10.1109/TLA.2019.8931198.*
7. ^*P BD, Prasad NA, Dhanraj, M TN (2023). "Adaptive Voting Mechanism with Artificial Butterfly Algorithm based Feature Selection for IDS in MANET." 2023 IEEE International Conference on Integrated Circuits and Communication Systems (ICICACS). 1–7. doi:10.1109/ICICACS57338.2023.10099861.*
8. ^*Gutiérrez D, Toral S (2019). "Deep Neuronal Based Classifiers for Wireless Multi-hop Net-work Mobility Models." 2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA). 602–607. doi:10.1109/ICMLA.2019.00111.*
9. ^*Jaton N, Gyawali S, Qian Y (2023). "Distributed Neural Network-Based DDoS Detection in Vehicular Communication Systems." 2023 16th International Conference on Signal Processing and Communication System (ICSPCS). 1–9. doi:10.1109/ICSPCS58109.2023.10261135.*
10. ^*Madapuzi Srinivasan S, Truong-Huu T, Gurusamy M (2018). "TE-Based Machine Learning Techniques for Link Fault Localization in Complex Networks." 2018 IEEE 6th International Conference on Future Internet of Things and Cloud (FiCloud). 25–32. doi:10.1109/FiCloud.2018.00012.*
11. ^*Wang Z (2015). "The Application of Machine Learning on Traffic Identification." BlackHat, USA.*
12. ^*Palmieri F, Fiore U, Castiglione A (2013). "A distributed approach to network anomaly detection based on independent component analysis." Concurrency Computation Practice and Experience. **26**(6):1113–1129. Wiley.*
13. ^*Aminanto ME, Kim K (2017). "Detecting Impersonation Attack in WiFi Networks Using Deep Learning Approach." In: Choi D, Guilley S (eds) Information Security Applications. WISA 2016. Lecture Notes in Computer Science. vol 10144. Springer, Cham. 136–147. doi:10.1007/978-3-319-56549-1_12.*
14. ^*Castillo AC (2023). "An Overview of Integration of the Virtualization of Network Functions in the Context of Information Centric Networks." 2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME). 1–7. doi:10.1109/ICECCME57830.2023.10253441.*

Declarations

Funding: No specific funding was received for this work.
Potential competing interests: No potential competing interests to declare.