**Microstrip Line Fed Dielectric Resonator Antenna Optimization using ML Algorithms**

**Abstract:**

In this report, our team is attempting to implement various machine learning techniques into Antenna Design Optimization. As a reference antenna, a Cylindrical DRA (Di-electric Resonator Antenna) with appropriate design parameters. This Antenna Model is designed in the HFSS (High-Frequency Structure Simulator) environment, using which we generated the data-set and verified results.

Studying these ML algorithms, we are trying to prove the efficiency and reliability of these techniques over conventional optimization methods. Each method is analyzed adequately by first training the learning model with the generated data set. Training is followed by predictions made by each model for a given input, comparing the prediction with actual results helped in the accuracy analysis. Further, a detailed comparative analysis of these predicted values with the HFSS results was carried out to verify the accuracy.

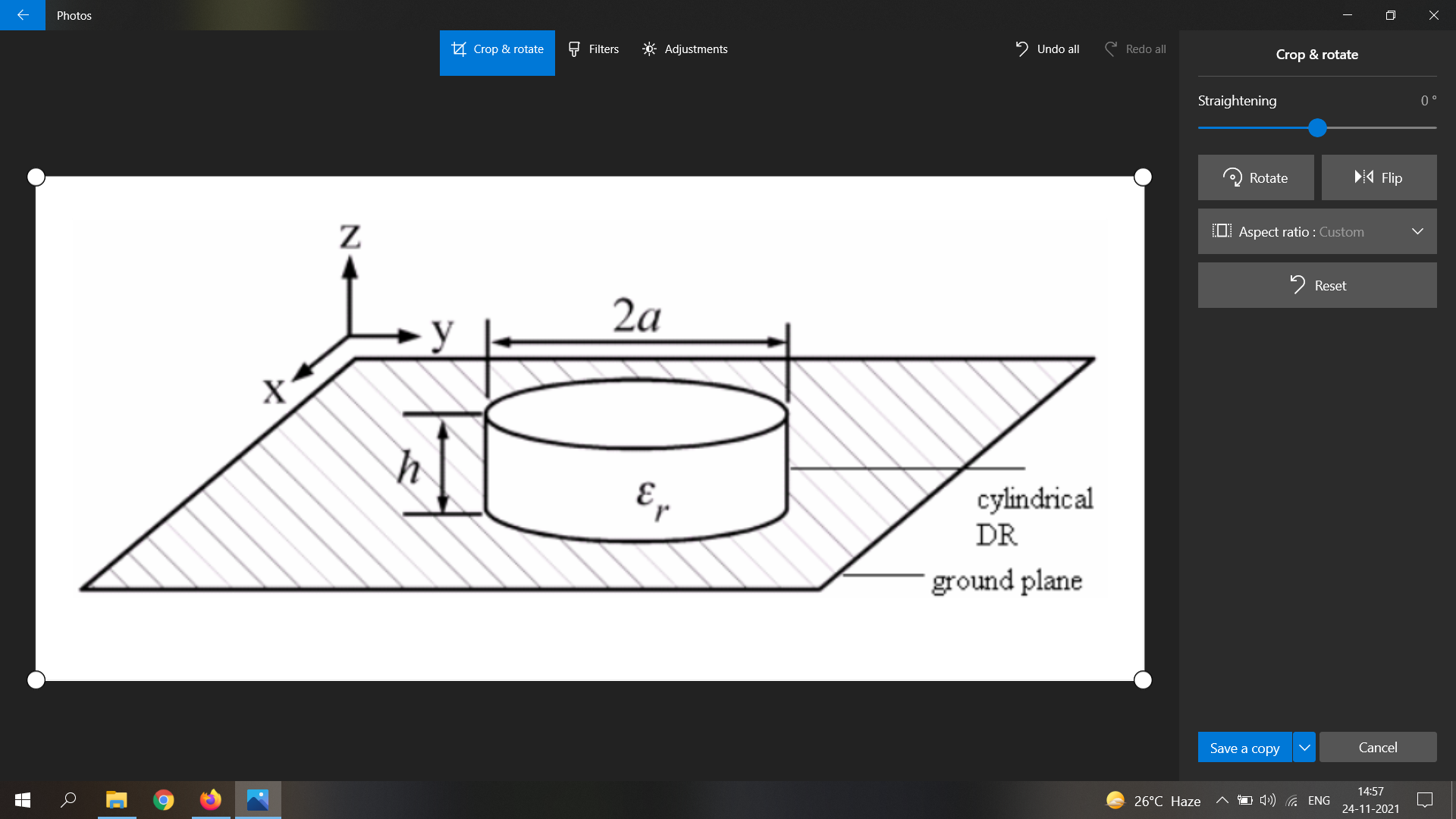
Alongside implementing conventional ANN, some work on some Knowledge-Based Neural Network techniques has also been done, followed by a comparative analysis.

**Introduction:**

In this paper, we are presenting our work on Machine Learning for the application of Antenna Design. Detailed application analysis is done on Microstrip Line Fed DRA Antenna which finds versatile applications. Due to increased design complexities, designing antennas with highly specific and optimized performance has become difficult. Since a lot of variables have to be handled for designing, the use of EM solvers in a simulation environment is not very reliable and could prove highly inefficient. Machine Learning is a subject that has shown its usefulness and applications in different scientific fields, considering this we are putting in some efforts to prove the efficiency and usability of ML in the field of antenna design. Possible issues with each technique have also been discussed and compared with a few prior works. Several commonly used algorithms were explored and worked upon. ANN was also studied with proper implementation, in generalizing the outputs for over a wide range, ANNs are proved way better as compared to conventional ML techniques. The use of ANN however imposes certain issues, and a lot of work on tuning the model for obtaining the desired output is accompanied. The issues with ANN have been further taken into consideration by KBNN, which knocks most of the downsides of a conventional ANN model. Majorly two techniques of KBNN which are source-difference and prior-knowledge-input are discussed. The aim

**Cylindrical Dielectric Resonator Antenna:**

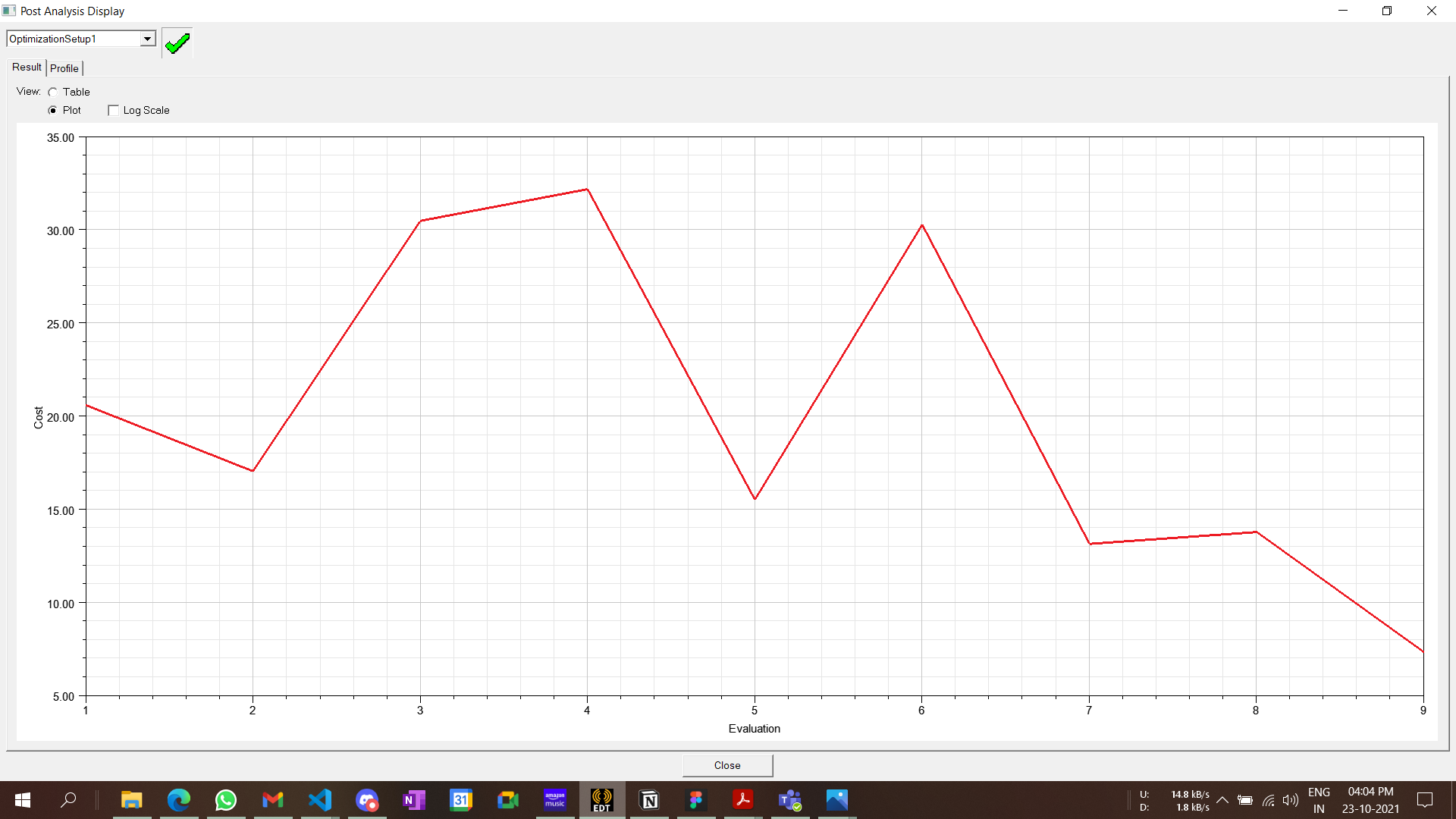
CDRA stands for Cylindrical Dielectric Resonator Antenna was, as evident from the figure, cylindrical attributes to its shape. It is utilized more often than not at frequencies greater than or equal to microwave frequencies ( > 1 GHz) and thus acts like a radio antenna. The Dielectric Resonator is made up of ceramic, which is the dielectric here, and is placed on a metal surface, ground plane. The choice of ceramic is due to the requirement of a material with a high-quality factor (20 < Q < 1000) and having low loss properties. Ideally, we require the dielectric constant between 10 < εr < 100 so that we can trade-off with other factors for various applications. DRA’s can perform the transformation of guided waves into unguided waves (RF signals) using radiating resonators. The size of DRA is inversely proportional to (εr)^½ and hence can be contained in a small space if the chosen material is of a high dielectric constant. Courtesy of this, DRA’s are now looked upon to be the possible solution to the requirement of smaller spacecraft equipment in the future. Also, the choice of low-loss material and the absence of any conducting material results in excellent radiation efficiency, thus making it optimal for applications operating at high frequencies. The resonant frequency of a CDRA can be roughly represented by, the resonant frequency is the frequency corresponding to the minima in the S11 sweep of the antenna.

****

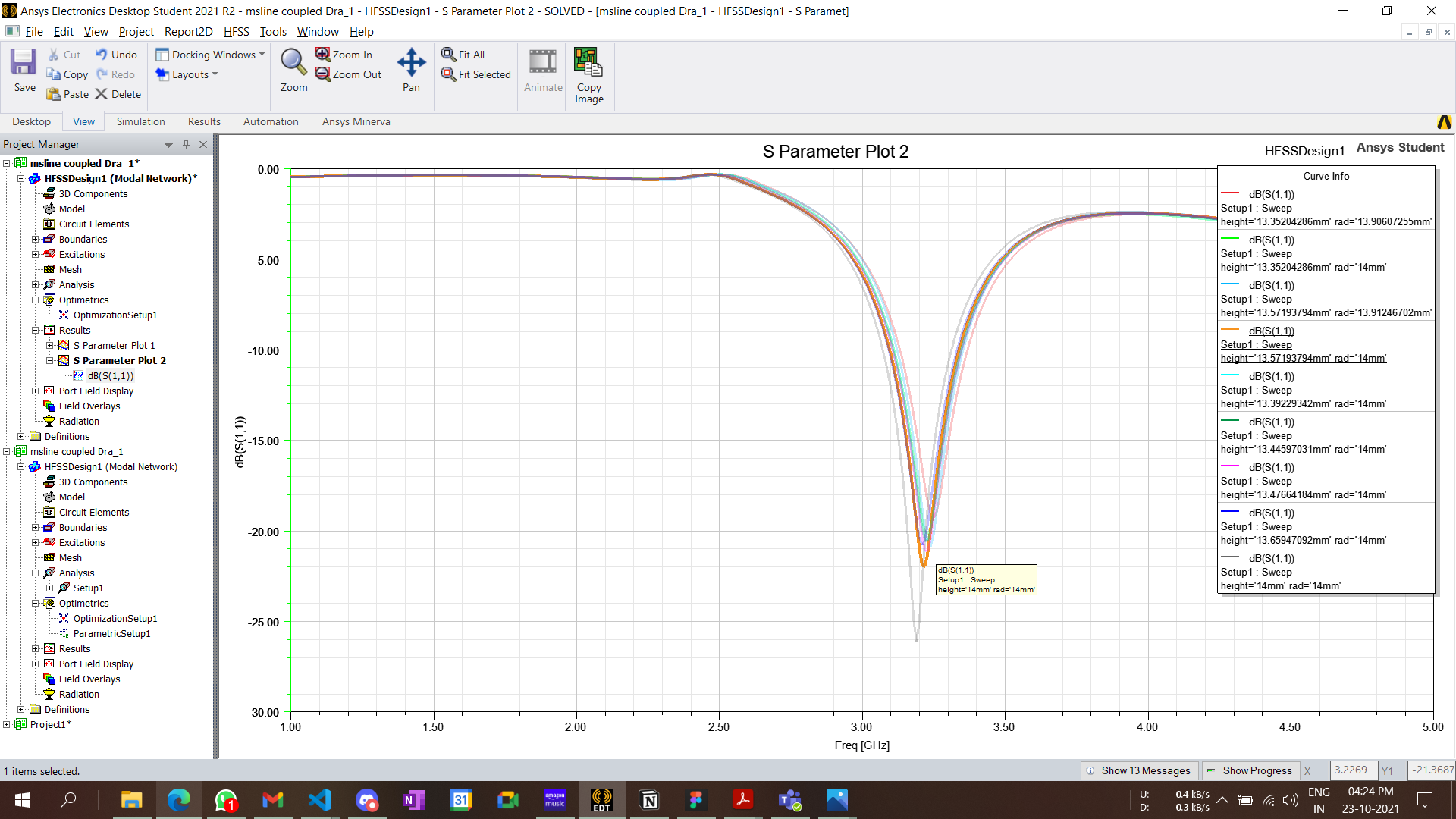
**Fig 1:** *Antenna configuration of cylindrical DRA*

**Methodology:**

For the implementation purpose, firstly we have designed this CDRA on Ansys HFSS. Few design parameters which have been considered are (substrate), r (substrate radius), w1 (width of the feedline), w2 (width of the vertical feed line). These parameters are chosen as they significantly affect the S11 which is the FOM in most design requirements. Scattering parameters define the properties of microwave devices, S11 in this case refers to the amount of power that is reflected back when the antenna is fed with power, and its value should be as low as possible. However not only does S11 get to be the FOM in each requirement, but there could also be other constraints that could introduce tradeoffs. For our study, we will consider only S11 as the desired FOM. Working with a conventional EM simulator gets highly inefficient, as each design iteration requires high compute which makes the entire process time-consuming. Even though each operation needs proper manual supervision and expertise, automation in this entire process is a great requirement. To generate the dataset from HFSS we have implemented HFSS parametric. However, before the parametric analysis and generating the dataset for training and testing, a knowledge of the optimized values of each parameter is required. For this purpose, HFSS optimetrics [7] were used, here while configuration the desired output has to be added and the optimetrics generates the optimized values of the selected variable parameters using any selected algorithm. In our case, we applied the *Quasi-Newton (Gradient)* optimizer with 9 iterations, the required result was that the dBS11 <= -20dB. After these iterations, the parameter variables are set to near approximate optimized values.

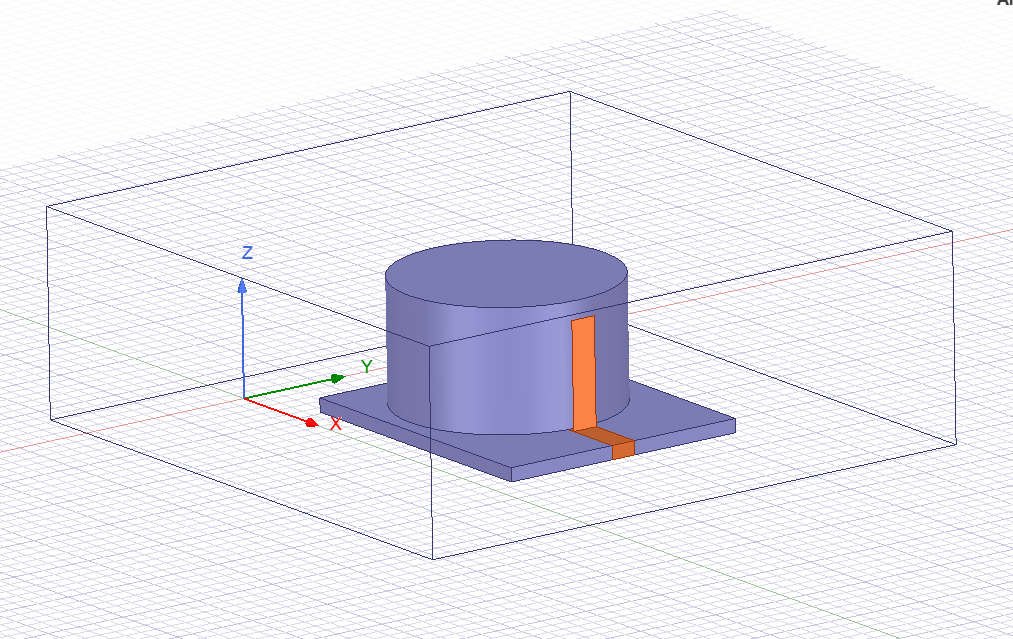


**Fig** *Cost representation for each iteration of Quasi-Newton (Gradient) optimizer*



**Fig** *S11 Sweeps for each optimization iteration*

Moving on to the parametrics, an HFSS parametric was added by adding in appropriate variable value range and step size. This can be added based on the HFSS optimization results, after the analysis the parametric sweeps can be generated and the dataset can be exported in a *.csv* file. The range and step size selected for *h,r, w1 and w2* are 13mm - 14mm: 0.2mm, 13mm-14.2mm: 0.4mm, 2mm - 3mm: 0.5mm, 2mm - 3mm: 0.5 mm. These ranges result in 216 parameter combinations, which means HFSS will have to perform 216 parameter operations. This process of generating the dataset with such minor iteration count itself shows the inefficiency of using such simulation softwares for parameter explorations or for complex antenna design optimization.



**Fig 2:** *Designed HFSS model of CDRA*

**Machine Learning**

A domain that recently has proved very useful with decision making and analytics in a wide variety of applications. In the telecommunications industry, machine learning has acquired a lot of traction due to automated self-service, predictive maintenance, enhanced customer service and satisfaction, network optimization, network monitoring, virtual support, and cost reduction. Machine learning has exploded in popularity over the last decade, affecting virtually every industry, including communications. Large computation capacity and economic viability were the primary drivers of this increase. In contrast to previous technologies, the magnitude of the data had an inverse relationship with efficiency [11]. In the communication industry, an antenna with circular polarization is required for uninterrupted wireless transmission. Providing gain to increase the radiated signal in a low-profile, cost-effective, multi-band configuration that maximizes signal level with little noise reception. Time savings, improved computational efficiency, lower operational costs, reduced simulation time, and reduced human effort were all advantages of ML-powered solutions in custom antenna design. Each day, antenna designers faced new obstacles due of the quick change in demand for multi-functional and small antennas. With techniques and optimization algorithms we can find the hidden mathematical relations in the input and output data considering which we can make future predictions. Other heuristic optimization techniques could also be used like genetic algorithms and particle swarm optimization but these algorithms search for the optimal solution by analyzing the output on individual data points and generating new and possibly better search directions until one global maxima or minima is identified. And once a model is trained the same data set gets beneficial for multiple output goals. Though some studies regarding ML implementation have been conducted as mentioned in [2], a comparative study could be required to get a clear picture. Our work tries to support ML as a promising choice to include in antenna design, and later on, this same idea can be extended to complicated designs even.

Broadly classifying there are two types of ML techniques - supervised learning and unsupervised learning. The major difference between the two techniques is the use of labeled data to train. Supervised techniques use labeled data and the model has the job of finding the mapping function. Whereas unsupervised learning uses unlabeled data and the model here itself has to infer patterns from the data set. Unsupervised learning models find the similarities and kind of cluster or group them together. It can be used to gather important insights from the dataset. However, since here we are trying to build a model which can predict the output for any data provided, we had to go forward with supervised learning models. Unsupervised learning is closer to Artificial Intelligence compared to supervised learning as it learns similarly as a child by observing its surroundings. But supervised learning is optimal for applications requiring an accurate response and therefore we can justify the use of it in our scenario. For our study, we are implementing ANN, KNN, Random Forest, DecisionTree, XGBoost, and KBNN.

**Training**

For Machine Learning models the generated training dataset has to be separated into training and test sets. This separation is helpful in testing the models for the test-set data for which it is never trained on, which helps in better evaluation of the prediction efficiency of each. For each model we are randomly picking up 30% of samples for in the test-set. Later the training data frames are utilized for the training of each model. For implementing KNN, Decision-Tree and Random-Forest *sklearn* python library is used, the value of K = 5 was chosen to minimize the cross-validation error. The training process of these models were quick and simple.

Implementation of ANN was done using *keras* library, which internally uses *tensor-flow* library. There is one input layer with the dimension of 5 as we have five input features for the model with a ‘relu’ activation function. 2 hidden layers are added with the same ‘*relu*’ activation function. The output layer of unit=1 provides the final result along with this the Adam optimizer and *mean\_squared\_error* was used as the loss function for the model. Adaptive Moment Estimation is a technique for optimising gradient descent algorithms. When working with huge problems with a lot of data or parameters, the method is quite efficient. It is efficient and takes minimal memory. It's essentially a hybrid of the 'gradient descent with momentum' and the 'RMSP' algorithms.

mt = aggregate of gradients at time t [current] (initially, mt = 0)

mt-1 = aggregate of gradients at time t-1 [previous]

∂L = derivative of Loss Function

∂Wt = derivative of weights at time t

β1 & β2 = decay rates of average of gradients in the above two methods.

The training is carried on for 100 epochs with a batch size of 10. Training of an ANN model is the time-taking process for machines with less compute potential. The training process was also analyzed to have a clear picture of how the model trains itself with the input training data set.

|  |  |
| --- | --- |
| **Fig 3:** *Loss vs Epoch graph for ANN* | **Fig 4:** *ANN Model validation loss vs Epoch graph* |

**Prediction**

Each of these models can now predict the values of S11 for a given antenna parameter in a particular frequency band. It’s better to test the model with a data-set with which it is not trained, to understand it’s precision better and analyze the prediction accuracy. If the predictions made are not up to the mark, then any of the following could be the case.

* The training data could be inappropriate or of poor quality.
* Model might require hyperparameter tuning or some cross-validation.
* Choice of the input features could be wrong.

There might be other reasons even, but for our use case, we tried working on the above-mentioned issues in case of the poor model prediction. Prediction metrics provides evaluative and comparative numbers which helps in opting out the best possible algorithm for a specific use case. Two most relied metrics are the r2-score and MSE (mean absolute error)

For instance, in case of ANN if the resultant metrics were not up to the mark the hyperparameters like training epochs, batch size, hidden layers, activation functions were tweaked a bit to have a better performance. On comparing the predicted outputs with the actual outputs, we obtained the r2 score, mean absolute error, mean squared error, max error. Table 1 contains these accuracy metrics for predictions made by each ML technique. To obtain these values we used the methods provided by “*sklearn.metrics”* . These metrics provide us a picture of the prediction effectiveness and helps in improving the training dataset or tuning the model.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | R2 Score | Mean Absolute Error | Mean Squared Error | Maximum Error |
| ANN | 0.94 | 0.42 | 1.02 | 35.34 |
| KNN | 0.98 | 0.11 | 0.31 | 30.51 |
| Decision Tree | 0.97 | 0.16 | 0.51 | 29.12 |
| Random Forest | 0.99 | 0.13 | 0.32 | 30.03 |
| XGBoost | 0.98 | 0.18 | 0.33 | 13.79 |

**Table 1** *Prediction metrics*

|  |  |
| --- | --- |
| **Fig 5** Predicted vs Actual values for *ANN* | **Fig 6** Predicted vs Actual values for *Decision Tree* |
| **Fig 7** Predicted vs Actual values for *KNN* | **Fig 8** Predicted vs Actual values for *Random Forest* |
| **Fig 9** Predicted vs Actual values for *XGBoost* |  |

Fig 5 – Fig 9 shows the accuracy plot of each model, since the X = Y line represents the ideal case, density of points near the line presents better accuracy result

In the accuracy analysis we took a fraction of the data-set (the test-set) to compare the

predictions, but now we will carry on the comparison of the HFSS results with the model predicted values. The data which we will use in this section will be randomly generated by uniform random distribution using the *NumPy* library in python. The range and step size values that we used for *h* and *r* in HFSS parametric will only be used here to generate a random data frame with 5 samples. The values of antenna parameters for a given frequency are set for the HFSS model and a simulation is carried out to get the value of S11 which can then be compared with the predicted values. For each data sample, an error is calculated to get an idea of the prediction accuracy considering the HFSS value as the actual value.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Sample | Height (mm) | Radius (mm) | Freq (GHz) | W1 (mm) | W2 (mm) | HFSS S11 |
| D1 | 14.10 | 14.55 | 2.42 | 2.25 | 2.6 | -0.37 |
| D2 | 13.04 | 13.30 | 1.84 | 2.3 | 2.7 | -0.43 |
| D3 | 13.19 | 13.74 | 1.95 | 2.7 | 2.45 | -0.44 |
| D4 | 14.42 | 13.12 | 2.41 | 2.4 | 2.8 | -0.51 |
| D5 | 13.86 | 14.57 | 2.56 | 2.8 | 2.1 | -0.77 |

**Table 2** 5 Random data samples.

**Fig 10** *Graph representing the S11 comparison for each Dataset.*

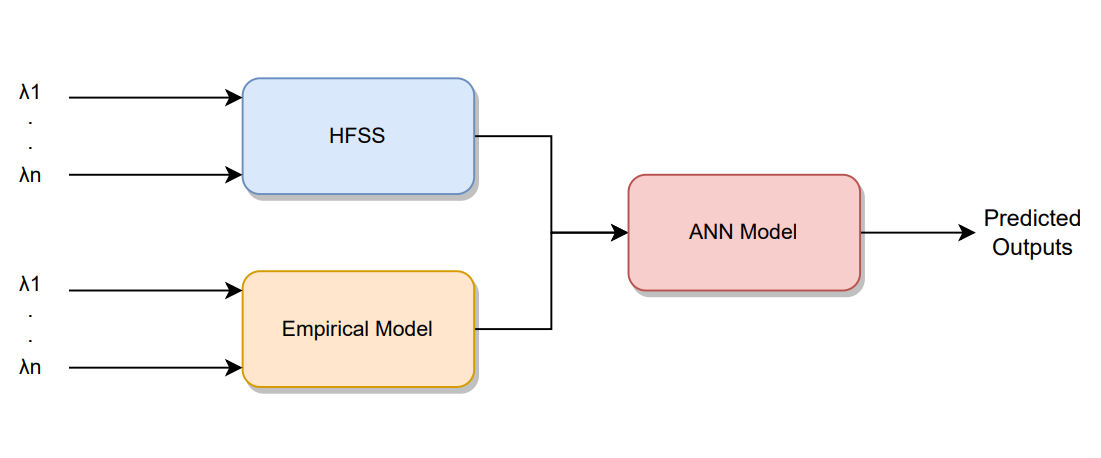
**KBNN**

Accuracy and prediction capability of ANN should be way more than other implemented ML algorithms, however in the accuracy analysis the results are not as expected. Primarily the low accuracy results of ANN are a reason for less training data-set on which the model was trained on. ANN can map complex data relations because of different implied layers, but the data required for training should have large samples, for less training samples other conventional models can easily outperform. Generating large training-samples through HFSS is not a feasible job, as each iteration requires heavy computation which makes the process highly inefficient. However, when the training data is difficult to obtain, design of experiment (DOE) techniques can also be implied [8], but the data requirement could still be too large. To tackle such issues some existing models which we have implemented could be used to represent additional prior knowledge. These models are already trained on limited data-set thus could just predict outputs for specific inputs, this can be leveraged to enhance the training of ANNs. Some work has been done on implementing KBNN on microwave components, which focused on reduction of training data requirement for ANNs [9]. This prior knowledge can be implemented using different methods, like *prior knowledge input* (PKI), *source difference* [10]*.* In the PKI method the ANN is trained to map the same input-output relation but along with the training data other inputs are also added. Additional data is generated using the empirical models, so for PKI the additional training data is generated using *KNN, Decision-Tree, Random-Forest and XGBoost* models. In the source difference method, the ANN is trained on the error (difference) between the HFSS output and the source-model output. The mapping of this error matrix is simpler thus requiring less training. Considering λ as a design parameter vector, for a particular vector the S(λ) is the S11 obtained from HFSS and similarly F(λ) obtained from the empirical model. Further finding out the difference between both results in the error matrix.

The ANN is now trained on the E(λ), the final required value of S11 can be obtained by summation of the ANN prediction with the empirical model’s output for a parameter vector.

Training efficiency of KBNN supports high dimensional parametric sweeps and design space exploration, however despite the positives there are several drawbacks even. Since empirical models are representing additional knowledge which is employed in the training of ANN, these models must be accurate enough. For several microwave design applications finding the empirical model beforehand could be a difficult task e.g., hybrid copper-graphene interconnects [10]. Fig 2 and Fig 3 represent the accuracy plot of PKI and Source Difference KBNN models. Table 1 contains the prediction metrics, and provides a comparison of the accuracy performance. Both the KBNN techniques provided better accuracy results as compared to ANN as evident from its MSE and R2 Score. The accuracy results of both the SD and PKI techniques are in close similarity.

|  |  |
| --- | --- |
| **Fig 11** *Error matrix is generated on which the ANN is trained.* | **Fig 12** *Predicting the FOM for particular input parameters.* |



**Fig 13** *Prior Knowledge Input KBNN*

|  |  |  |
| --- | --- | --- |
|  | MSE | R2 |
| ANN | 1.02 | 0.94 |
| SD KBNN | 0.51 | 0.97 |
| PKI KBNN | 0.73 | 0.96 |

**Table 3** *Comparison of prediction metrics.*

|  |  |
| --- | --- |
| **Fig 14** PKI accuracy Chart | **Fig 15** Source Difference accuracy chart |

**Conclusion**

In this study we went through different Machine Learning techniques which can be leveraged into antenna design optimization. A brief description of each technique was presented followed by their use cases and application. The reference CDRA antenna was then explained with its design characteristics. Later with the HFSS design we tried to optimize the parameters for a specific goal with optimetrics and then with parametrics we generated the training and test data set.

Above mentioned 5 ML techniques were used to firstly train the model followed by tuning each model by analyzing the prediction metrics. Finally for few data samples the predicted values were compared with the HFSS calculated values. This brought us to infer that the prediction performance of all these techniques were quite similar except ANN, despite being from DL domain its model didn’t show satisfactory outputs due the small-sized data set and less optimized hyperparameter tuning.

The limitations of the ANN model were addressed by KBNN. We discussed two major techniques of implementation which were *source-difference* and *prior-knowledge-input.* With fewer training samples better model accuracy were achieved as evident from the prediction metrics, even the training time of these models were less comparatively.

**Git Repository** = [OmSingh5092/final-year-project: Microstrip Line Fed Dielectric Resonator Antenna Design using Machine Learning (github.com)](https://github.com/OmSingh5092/final-year-project)

**References**

[1] H. Xin and M. Liang, “3-D-printed microwave and THz devices using polymer jetting techniques,” Proc. IEEE, vol. 105, no. 4, pp. 737–755, Apr. 2017.

[2] A. Massa, G. Oliveri, M. Salucci, N. Anselmi, and P. Rocca, “Learning By- examples techniques as applied to electromagnetics,” J. Electromagn. Waves Appl., vol. 32, no. 4, pp. 516–541, Mar. 2018.

[3] M. Hagan and M. Menhaj, “Training feedforward networks with the Marquardt algorithm,” IEEE Trans. Neural Netw., vol. 5, no. 6, pp. 989–993, 1994.

[4] S. B. Imandoust and M. Bolandraftar, “Application of k-nearest neighbor (kNN) approach for predicting economic events: Theoretical background,” Int. J. Eng. Res. Appl., vol. 3, no. 5, pp. 605–610, Sep./Oct. 2013.

[5] Tianqi Chen and Carlos Guestrin, “XGBoost: A Scalable Tree Boosting System” University of Washington

[6] ANSYS Electromagnetics Suite 15.0, ANSYS, Canonsburg, PA, USA, 2013.

[7] Siyang Sun, Yinghua Lv, Jinling Zhang, Zhidong Zhao, “Optimization based on genetic algorithm and HFSS and its application to the semiautomatic design of antenna”, Article May 2010

[8] P. Watson and K. C. Gupta, EM-ANN models for via interconnects in microstrip circuits, IEEE MTT-S Int Microwave Symp Dig, 1996, pp. 1819]1822.

[9] P. M. Watson, K. C. Gupta, and R. L. Mahajan, Development of knowledge based artificial neural network models for microwave components, IEEE MTT-S Int Microwave Symp Dig, 1998, pp. 9]12.

[10] Rahul Kumar, S. S. Likith Narayan, Somesh Kumar , Sourajeet Roy , Senior Member, IEEE, Brajesh Kumar Kaushik, Senior Member, IEEE, Ramachandra Achar , Fellow, IEEE, and Rohit Sharma , Senior Member, IEEE, “Knowledge-Based Neural Networks for Fast Design Space Exploration of Hybrid Copper-Graphene On-Chip Interconnect Networks”, Article

[11] Kanhaiya Sharma, Ganga Prasad Pandey, Efficient modelling of compact microstrip antenna using machine learning, Regular Paper.