Communication

Machine Learning Techniques for Optimizing Design of Double T-Shaped Monopole Antenna

Yashika Sharma, Hao Helen Zhang, and Hao Xin, *Fellow, IEEE*

***Abstract*— In this communication, we propose using modern machine learning (ML) techniques including least absolute shrinkage and selection operator (lasso), artificial neural networks (ANNs), and *k*-nearest neighbor (kNN) methods for antenna design optimization. The automated techniques are shown to provide an efficient, flexible, and reliable framework to identify optimal design parameters for a reference dual-band double T-shaped monopole antenna to achieve favorite performance in terms of its two bands, i.e., between 2.4 and 3.0 and 5.15 and 5.6 GHz. In this communication, we also present a thorough study and comparative analysis of the results predicted by these ML techniques, with the results obtained from high-frequency structure simulator (HFSS) to verify the accuracy of these techniques.**

***Index Terms*— Antenna optimization, least absolute shrinkage and selection operator (lasso) shrinkage, linear regression, machine learning (ML), optimization.**

I. Introduction

The upcoming era of Internet of Things (IoT) has enabled an immense growth in the demand of application-specific antennas, which are needed for almost all electronic devices. Hence, the requirement of a smart and efficient way of antenna designing has become inevitable. Current antenna design heavily relies on the designer's empirical experiences and EM simulations. Traditional methods are inherently inefficient and computationally intensive, making them impractical when there are a large number of antenna design parameters to be optimized such as for 3-D printed antennas [1]. To address challenges for designing complex 3-D structures, machine learning (ML) techniques may be highly beneficial. ML has been widely used as an indispensable data analysis and decision-making tool in a broad range of applications, ranging from hand-written digit recognition [2] to human genomics [3]. Researchers have also explored optimization of antenna structures by applying heuristic optimization techniques like genetic algorithms and particle swarm optimization [4]–[6], 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 a global maxima or minima is identified. On the other hand, ML refers to all techniques and optimization algorithms of analyzing the data and finding the hidden mathematical relation in data such that we can relate the input behavior to the output behavior and make future predictions or decisions using this relationship. The main advantage of using ML techniques is once we have the relational model, we can predict the output for any data point rather than aiming for global optimal and minima points only. This property is very beneficial when we want to use the same data set for multiple different goals. As described in [7], there is some early work on applying ML techniques for antenna analysis and synthesis [8]–[11]. In [8], the performance of support vector machines (SVMs) is investigated for designing of a rectangular patch antenna and a rectangular patch array, while in [9], methodology to use SVMs for linear and nonlinear beamforming and parameter design for arrays and electromagnetic applications is mentioned. Another ML technique, artificial neural network (ANN) has also been applied in this field [10], [11]. In [12], clustering method is used to find the optimum position for shorting posts in microstrip patch array design so as to achieve acceptable bandwidth, scan angle, and polarization. As observed from here, some studies for using ML techniques for antenna design optimization have been conducted but a detailed analysis and systematic comparison of various ML techniques for antennas have not been reported. The main contribution of this communication is to fill the gap by presenting new classes of ML-based methods for automated antenna design optimization, evaluating their performance in terms of prediction accuracy and robustness and making comparisons with EM simulations. Our discovery suggests that ML is a promising choice to provide automated, computational feasible, and practically effective approaches for antenna design. The ultimate goal of this communication is to further extend the proposed ideas to more complex design of antennas and develop scalable and efficient algorithms to tackle computational challenges, by handling a large number of design parameters.

In this communication, we propose using ML techniques for antenna design optimization, and particularly consider ANNs, least absolute shrinkage and selection operator (lasso), and *k*-nearest neighbor (kNN). The feasibility of these new approaches for antenna design is demonstrated by their applications to optimize a reference double T-shaped monopole antenna [13] as shown in Fig. 1. The initial work

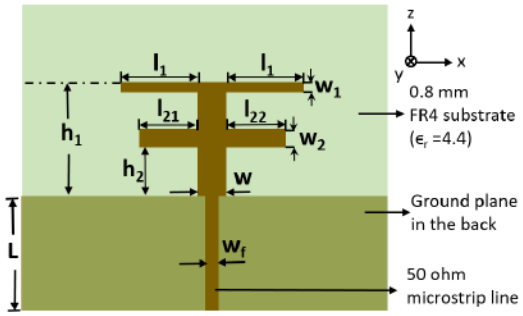

Fig. 1. Layout and design parameters of the referenced dual-band double-T monopole antenna (source: [13]).

Manuscript received February 20, 2019; revised November 17, 2019; accepted December 3, 2019. Date of publication January 17, 2020; date of current version July 7, 2020. *(Corresponding author: Hao Xin.)*

Yashika Sharma is with the Department of Electrical and Computer Engineering, The University of Arizona, Tucson, AZ 85719 USA (e-mail: yashikasharma@email.arizona.edu).

Hao Helen Zhang is with the Department of Mathematics, The University of Arizona, Tucson, AZ 85721 USA (e-mail: hzhang@math.arizona.edu).

Hao Xin is with the Department of Electrical and Computer Engineering, The University of Arizona, Tucson, AZ 85719 USA, and also with the Department of Physics, The University of Arizona, Tucson, AZ 85721 USA (e-mail: hxin@ece.arizona.edu).

Color versions of one or more of the figures in this communication are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TAP.2020.2966051

related to this was reported in [14] where only the lasso technique was employed and targeted optimization function was weighted sum of fractional bandwidths of two bands of the reference antenna. In this communication, we use finer optimization function and explore more ML techniques to optimize the performance of the reference antenna. This helps in achieving better results as compared to [13] and [14].

Section II describes the main ideas of the suggested ML techniques as well as their computational algorithms and implementation. Their performance is evaluated in Section III and further compared with that of traditional EM simulation methods. Section IV contains the conclusion and the discussion of future work in this area.

II. Methodologies

To demonstrate the proposed idea, we use a reference double T-shaped monopole antenna [13] (as shown in Fig. 1) as the example. The performance of this antenna mainly depends on five design parameters l_{21}, l_{22}, w_1, w_2, and w. In the design process, we allow these five parameters to vary while keeping the other three parameters L, h_1, and h_2 at their fixed values, as mentioned in [13]. These five geometric parameters act as the explanatory variables or input variables for ML models. The data are collected in batches and the corresponding R-squared values of the fitted model trained by the accumulated data are monitored. As the sample size gets bigger, the R-squared value first substantially improves but then gradually levels off. In our numerical experiment, 450 samples give reasonably high and stable R-squared values (larger than 0.85 for each case). Hence, N is chosen as 450 sample points and data are collected by varying these five geometric parameters at different values using random sampling. Theoretically, increasing the training samples improves the prediction accuracy, as a larger number of sample points contain more information about the underlying data generation scheme and, therefore, lead to better estimation results. The same pattern is observed here when data are accumulated in batches. Therefore, we choose sufficient number of training samples in batches, till the R-squared value becomes stable. The sample points are simulated using ANSYS high-frequency structure simulator (HFSS) [15] and antenna reflection coefficients in the format of .s1p files are extracted from the simulations. Using each .s1p file, performance of this antenna for each sample point is assessed by extracting figure of merit (FOM), which is defined to obtain the maximal bandwidth in the two desired bands of the antenna. In this communication, FOM is defined as the sum of absolute values of reflection coefficient (S_{11}) in dB for frequency points in the range of 2.4–3.0 and 5.15–5.6 GHz. Basically, we add the absolute value of S_{11} at each of the frequency point falling in the band of interest in order to calculate the FOM for a given design and the same is represented mathematically as follows:

$$FOM = \sum_{\mathrm{f}=2.4}^{\mathrm{f}=3.0} |S_{11}(\mathrm{f})| + \sum_{\mathrm{f}=5.15}^{\mathrm{f}=5.6} |S_{11}(\mathrm{f})| \tag{1}$$

where f represents the frequency and $S_{11}(\mathrm{f})$ is the reflection coefficient value at that frequency. While collecting the sample points, these parameters take values within the following range of sample space, χ defined as: $l_{21} \in [6.3, 7.3]$, $l_{22} \in [6.3, 7.3]$, $w_1 \in [1, 3.5]$, $w_2 \in [1, 3.5]$, $w \in [1, 3.5]$, with each parameter taking a step size of 0.5 (all units are in millimeters). In the antenna design process, these five design parameters are input variables and FOM is the output or response variable. The training data are represented by $\{(X_i, Y_i), i = 1, 2, \ldots N\}$, where the input is $X = (l_{21}, l_{22}, w_{1,w_2}, w)^T \in \chi$ or its transformation. The output Y is the value of FOM. The goal is to learn a behavioral model based on the training set to best describe the relationship between the expected FOM ($\widehat{FOM}$) and the design parameters. We represent this behavioral model by

$$\widehat{FOM} = h(l_{21}, l_{22}, w_1, w_2, w) + \epsilon \tag{2}$$

where ϵ is the error term and the function h is a flexible mapping, which can be linear or nonlinear, continuous or discontinuous function that is based on the main effects or including two-way interaction effects of the input parameters. ML methods are then used to search for the best $h \in H$, a class of candidate models, to describe the relationship between X and Y and make future predictions. To search for optimal design parameters based on the obtained ML model, a very fine grid over the entire space χ with a step size 0.1 mm (instead of 0.5 mm in the training set) is generated, consisting of a total of 2 126 696 design points. The FOM values at all the design points are then computed and the design parameter values that give the maximal FOM value are identified. The performance of this analysis is verified by comparing the predicted FOM at a set of test points with its actual value obtained from the HFSS simulation. In Sections II-A–II-C, the three ML techniques (namely lasso, ANN, and kNN) that have been used in this communication to obtain the behavioral model h are explained.

A. Lasso

The lasso technique is a sparse regression and predictor selection algorithm that estimates a linear model subject to some conditions. Basically, it searches the value of regression coefficients that can minimize the residual sum of squares subject to the condition that the absolute values of the regression coefficients are less than a constant [16]. Denote the training data by (m_k, n_k), $k = 1, 2, \ldots, N$, where $m_k = \{m_{k1}, \ldots, m_{kp}\}^T$ are p-dimensional predictor variables and n_k are associated responses. The linear estimation model predicts the response for a given input m_k, as $\widehat{n_k} = \alpha + \sum_p \beta_p m_{kp}$. In a simple linear regression problem, the values of $(\widehat{\alpha}, \widehat{\beta})$ are obtained by applying the least squares method, i.e., minimizing the difference between the actual response value n_k and the estimated value $(\widehat{n_k})$ as defined in the following:

$$(\widehat{\alpha}, \widehat{\beta}) = \operatorname{argmin} \left\{ \sum_{k=1}^{N} \left(n_k - \alpha - \sum_p \beta_p m_{kp} \right)^2 \right\}. \tag{3}$$

The standard regression analysis includes all the predictor variables in the model fitting, irrespective of the magnitudes of their effects on the output. If some predictor variables are not informative for prediction, these variables are regarded as unimportant and should be removed from the final model, in order to improve both prediction accuracy and model interpretability. This can be done by lasso, which calculates the estimate $(\widehat{\alpha}, \widehat{\beta})$ by minimizing the sum of residual squares subject to the constraint $\sum_p |\beta_p| \leq t$. Here, $t \geq 0$ is a tuning parameter, which controls the amount of shrinkage that is applied to the estimates.

In our analysis, we fit the linear model by using the function $lm()$ in R [17], which is a public-domain statistical computing and graphics software. To increase model flexibility and capture any nonlinear relationship between the predictors and the response, we have also considered and compared various forms of nonlinear transformation of the raw design parameters, including log, exponential, quadratic, cubic, etc., and finally decided that a quadratic transformation is the most appropriate. To implement the lasso procedure, we used the function $lasso()$ in R. For this lasso model, the five design parameters and their quadratic transformations are used as the predictor variables and the FOM is used as the response variable. The optimal value of the tuning parameter t is determined by the fivefold cross validation

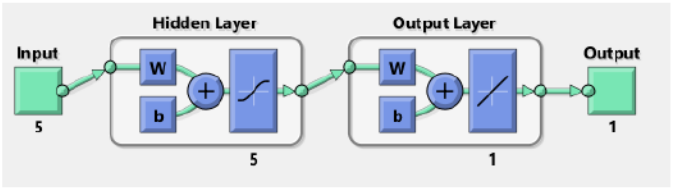

Fig. 2. ANN Architecture based on MLP.

method [18]. The final fitted model predicted by the lasso is as follows:

$$\begin{aligned}\widehat{FOM}_{lasso} &= 9.96l_{21}^2 + 16.43l_{22}^2 + 2.04w_1^2 + 0.79w_2^2 \\ &\quad - 6.48w^2 - 36.49l_{21}l_{22} + 0.93l_{21}w_1 + 2.12l_{21}w_2 \\ &\quad + 7.78l_{21}w + 2.59l_{22}w_1 + 0.57l_{22}w_2 + 6.44l_{22}w \\ &\quad + 0.06w_1w_2 - 3.46w_1w - 0.78w_2w + 92.35l_{21} \\ &\quad + 5.37l_{22} - 28.92w_1 - 7.42w_2 - 31.13w - 91.96. \end{aligned} \tag{4}$$

B. ANN Analysis

ANN is a computational model that is inspired by the function of biological neural networks. ANN consists of a group of artificial neurons, which process information over interconnection. There are many different ANN structures used in the literature. Multilayer perceptrons (MLPs) [19]–[21] which are successfully and commonly employed in engineering problems are preferred in this communication because of their ability to learn and model complex relationships. The MLP can be trained by many algorithms such as Levenberg–Marquardt (LM), back-propagation, and delta-bar-delta. In this communication, we train the MLPs using the LM algorithm [22], which has the abilities of fast learning and good convergence. For our problem setup, which has only five parameters in the input space and needs a regression model, the LM algorithm is one of the most efficient training algorithms for small- and medium-sized patterns. The MLP consists of three layers: an input layer, an output layer, and a hidden layer, as shown in Fig. 2. Neurons in the input layer distributes the input signal u_i to the neurons in the hidden layer. Each neuron j in the hidden layer adds up its input signals u_i after multiplying weights to each term depending on the respective connections w_{ij} from the input layer and computes its output v_j as a function g of the sum and a bias value, b_j added to it, that is

$$v_j = g\left(\sum_i w_{ij}u_i\right) + b_j \tag{5}$$

where $g(\cdot)$ can be a simple threshold function, a sigmoid, hyperbolic tangent, a radial basis function, a purelin function, etc. The output of neurons in the output layer is computed similarly. The simplest algorithm that can be used is a first-order error back-propagation (EBP) algorithm [23] which is one of the primitive training algorithms for neural networks. However, the EBP algorithm suffers because of its low training efficiency. This can be improved by using dynamic learning rates or by using the second order algorithms, such as Newton algorithm and LM algorithm [24] which will help in increasing the training speed of the EBP algorithm. The LM algorithm is a blend of vanilla gradient descent and Gauss–Newton iterations, which gives it better convergence speed over vanilla gradient descent algorithms. LM is also able to provide a solution for nonlinear least squares minimization. Since it combines the EBP algorithm and Newton algorithm, LM algorithm is considered to be most efficient algorithm for small- and median-sized patterns.

LM algorithm is implemented for training the neural network for this communication, using the neural network toolbox of MATLAB [25]. In our analysis, the data collected from .s1p file are divided into three parts: 70% of data are used for training and 15% each used for testing and validation, respectively. The input layer consists of five design parameters, the hidden layer of five hidden nodes, and the output layer of a single node for FOM.

C. kNN

The kNN [26] is an instance-based algorithm for supervised learning, which defines the similarity between sample points and makes predictions for new data points based on their similarity to the data that are already present in the training set. The similarity measure is typically expressed by a distance measure such as the Euclidean distance, cosine similarity, or the Manhattan distance. For any given new data point, the kNN algorithm first calculates its distance to all stored data points, which are used to determine its kNNs. Next, the outputs of these neighbors are gathered to produce a weighted average, which will then be assigned to the new data point. The weight of each neighbor is inversely proportional to its distance to the target data point. In simple words, the nearer neighbors contribute more to the average than the more distant one.

To implement kNN in this analysis, we first use tenfold cross validation to choose an appropriate value of k. It is observed that $k = 5$ gives the minimum cross validation error and therefore, five nearest neighbors are chosen for each data point for prediction. We use the function $knn.reg()$ in the FNN package of R [27] in order to obtain the kNN model for the reference antenna.

Hence, by using above-mentioned three ML techniques, we obtain three separate behavioral models to relate design parameters to FOM of the reference antenna. Next, these models are used to estimate performance of the antenna at various design points in terms FOM and also the accuracy of the estimations is tested by comparing the results with HFSS. More details regarding this have been mentioned in Section III.

III. Results and Discussion

Using the behavioral models obtained from the ANN model, lasso model, and kNN model, we identify the optimal design parameters which produce the highest FOM value. For this purpose, we predict the FOM values for all possible combinations of values of the five design parameters and locate the maximum FOM point. In particular, the five antenna design parameters are varied in the following range: $l_{21} \in [6.3, 7.3]$, $l_{22} \in [6.3, 7.3]$, $w_1 \in [1, 3.5]$, $w_2 \in [1, 3.5]$, $w \in [1, 3.5]$, with each parameter having a step size of 0.1 (all units are in millimeters).

The results predicted by different ML techniques are compared with those obtained from the HFSS simulation tool and summarized in Table I. The first column of the top three rows in this table shows the design parameters' value that is obtained by each ML technique to give maximum FOM. The next three columns list the corresponding maximum FOM value predicted by each ML technique and the last column is the FOM obtained from HFSS if the design parameters mentioned in the first column are substituted in HFSS design. For comparison, the design parameters' value of the reference antenna as mentioned in [13] is also stated in the first column of the fourth row of Table I and in the next columns of the fourth row, the corresponding FOM value predicted by each technique and HFSS is shown. The fifth row in this table mentions the total computation time corresponding to each of the ML technique, i.e., the time it took to train the ML model

TABLE I

PREDICTED FOM, DESIGN PARAMETER VALUES, AND COMPUTATION TIME COMPARISON

Design parameters predicted by ↓ / FOM predicted by →	lasso	ANN	kNN	HFSS
lasso: l_{21}=7.3, l_{22}=6.3, w_1=1, w_2=3.5, w=3.5	298.56	297.68	286.75	298.06
ANN: l_{21}=7.3, l_{22}=6.3, w_1=1, w_2=3.5, w=3.5	298.56	297.68	286.75	298.06
kNN: l_{21}=7.3, l_{22}=6.3, w_1=1.2, w_2=3.3, w=3.1	286.35	280.47	288.67	280.53
Mentioned in [13]: l_{21}=7.3, l_{22}=7.3, w_1=1, w_2=3.5, w=3.5	288.17	288.68	273.69	289.67
Computation time: (Seconds)	789	1074	387	

TABLE II

SENSITIVITY ANALYSIS OF PREDICTED LASSO MODEL: PREDICTED FOM AND DESIGN PARAMETER VALUES WHEN COEFFICIENTS OF (4) ARE PERTURBED

Coefficient of variable↓	Initial Value	Perturbed Value	Maximum FOM value	
			For initial model	For perturbed model
$l_{21}{}^2$	9.96	9.80	FOM=298.56 at l_{21}=7.3,l_{22}=6.3, w_1=1, w_2=3.5, w=3.5	FOM=290.03 at l_{21}=7.3,l_{22}=6.3, w_1=1, w_2=3.5, w=3.5
$w_1{}^2$	2.04	2.60	FOM=298.56 at l_{21}=7.3,l_{22}=6.3, w_1=1, w_2=3.5, w=3.5	FOM=299.12 at l_{21}=7.3,l_{22}=6.3, w_1=1, w_2=3.5, w=3.5
w^2	-6.48	-6.00	FOM=298.56 at l_{21}=7.3,l_{22}=6.3, w_1=1, w_2=3.5, w=3.5	FOM=304.44 at l_{21}=7.3,l_{22}=6.3, w_1=1,w_2=3.5, w=3.5

and then search the corresponding best set of design parameters from that model.

In Sections III-A–III-C, the results of Table I for each ML techniques are discussed in more detail.

A. Lasso Results

The lasso model is trained with 20 parameters in the input space, consists of five original design parameters and their square, as well as their cross product terms. Fivefold cross validation [16] is used to determine the optimal value of the tuning parameter. Though lasso is known to be able to produce sparse solutions, for this case none of the coefficients becomes zero [as observed from (4)], which implies that all the predictors have nontrivial effects on the output performance. For the model predicted by lasso as represented in (4), the sensitivity analysis is performed by perturbing few coefficients and the results are mentioned in Table II. It can be observed from Table II that maximum value of FOM and the design parameter values for which this maximum occurs for the perturbed model remain almost the same to the values predicted from the actual model. This proves the robustness of the lasso model against slight random perturbations. The maximum FOM predicted by the lasso model is mentioned in first row of Table I, which can be achieved by choosing the design parameter values as: $l_{21} = 7.3$ mm, $l_{22} = 6.3$ mm, $w_1 = 1$ mm, $w_2 = 3.5$ mm, and $w = 3.5$ mm. The trained lasso model also predicts the FOM value at this design location equal to 298.56. To verify the results, an HFSS simulation for these values of design parameters is done. The corresponding value of FOM obtained from HFSS is 298.06.

B. ANN Results

The results obtained from neural networks are more precise than lasso. For neural network analysis, 70% of data were used for training and 15% each is used for testing and validation, respectively. The input layer consists of five nodes, each representing one design parameter, a hidden layer with five hidden nodes, and the output layer of one single node, representing the FOM. As observed from the second row of Table I, the maximum FOM is expected to occur at $l_{21} = 7.3$ mm, $l_{22} = 6.3$ mm, $w_1 = 1$ mm, $w_2 = 3.5$ mm, and $w = 3.5$ mm by the ANN model and the corresponding FOM value is predicted to be equal to 297.68. The FOM value for these design parameters is also checked through the HFSS simulation; the resulting value is 298.06. It can be observed from here that the results obtained from lasso and neural network are very close to each other.

C. kNN Results

As mentioned earlier, for the kNN analysis here, we use k = 5 and Euclidean distance-based weighted average to estimate FOM for any design parameter set using its training data. As shown in the third row of Table I, the maximum FOM predicted by kNN is equal to 288.67, which is given by the design parameters $l_{21} = 7.3$ mm, $l_{22} = 6.3$ mm, $w_1 = 1.2$ mm, $w_2 = 3.3$ mm, and $w = 3.1$ mm; for this particular set of values, HFSS gives FOM = 280.53. This implies a percentage error of 2.90%.

Fig. 3(a)–(e) shows the plots to compare performance of the ML techniques. The values of FOM predicted by ML techniques and obtained from HFSS are plotted with respect to each of the five design parameters. For each plot, only one design parameter is varied along the horizontal axis, while the other four design parameters are kept as constant mentioned on the top of each graph. For the HFSS results, each design parameter is varied with a step size of 0.5 mm because simulation for each design parameter set is quite time-consuming. For the three ML technique-predicted FOM, each design parameter is varied with a step size of 0.1 mm because once the ML models relating the input and output parameters have been established, FOM can be predicted much faster compared to doing HFSS simulations. The vertical axis depicts the FOM values. The red curve represents FOM from lasso prediction, black curve from ANN prediction, pink curve from kNN prediction, and blue curve for FOM values calculated from HFSS simulations. From Fig. 3, it is observed that the values from lasso and neural networks are quite close to the actual FOM value obtained from HFSS, since the red and black curves closely trace the blue curve in all the plots. This proves the accuracy of lasso prediction and ANN prediction. On the other hand, the kNN prediction (pink curve in Fig. 3) is little deviated

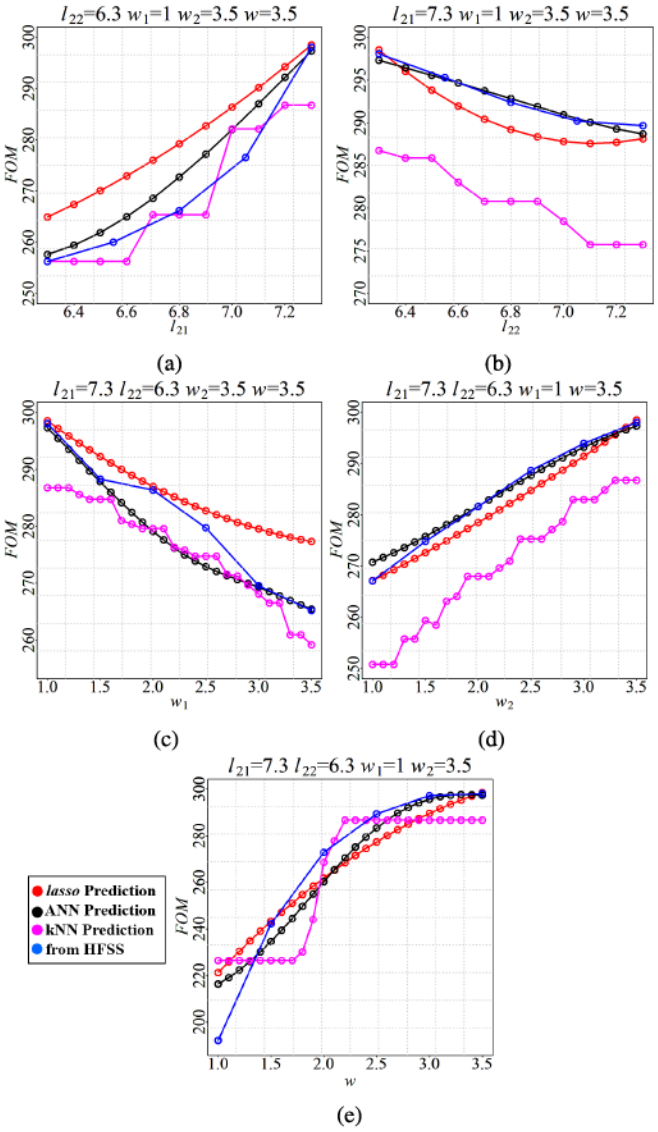

Fig. 3. Predicted and simulated FOM values with respect to change in (a) l_{21}, (b) l_{22}, (c) w_1, (d) w_2, and (e) w while keeping other four constants to the values shown on top of each plot and legend mentioned on bottom-left.

from the actual values obtained from HFSS and the other two ML techniques. Still, the kNN model is able to predict dependence of FOM on each design parameter like the other models. One possible reason for less precision in the kNN results can be due to its simple model structure, which is sensitive to irrelevant or redundant features because all features contribute while making prediction [26]. Also, the distribution of data and type of distance used also affect the prediction values in the case of kNN. Therefore, these are factors that may be responsible for the deviation of kNN prediction compared to the other two techniques.

The maximum FOM predicted by both lasso and ANN occurs for the same set of design parameter values (l_{21} = 7.3 mm, l_{22} = 6.3 mm, w_1 = 1 mm, w_2 = 3.5 mm, and w = 3.5 mm). These values are substituted in the HFSS model and the simulated results are compared with the results for design parameters set mentioned in [13] in order to compare the bandwidth performance. Fig. 4 shows the S_{11} plot for these two cases. It can be observed that for design parameters predicted by lasso and ANN (red curve), larger bandwidth in the two bands is achieved compared to that for the design parameters

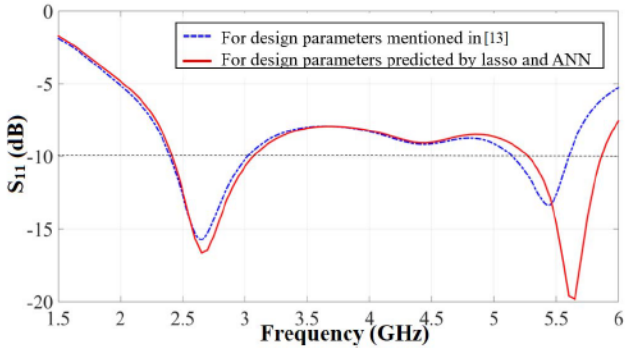

Fig. 4. Simulated S_{11} versus frequency response for the design parameters values mentioned in [13] (blue curve) and for design parameters values predicted by lasso and ANN (red curve) in order to attain maximum FOM.

mentioned in [13] (blue curve). Hence, in this communication by using lasso and ANN modeling, we not only save time during optimization but are also able to achieve better performance for the reference antenna structure.

As compared to the optimization done using EM solvers, ML optimization is quite fast. This is clearly evident from the fifth row of Table I, as the computation time of each algorithm is just a few hundreds of seconds which is significantly less than the time taken by EM solver to simulate even one design point. If the same optimization has to be done for these five design parameters by using an EM solver alone, it would take a few hours. In addition, we can change the optimization goal later after data collection by using ML optimization, while the same is not true for a standard optimization by an EM solver alone. Also, optimization using an EM solver requires designer's empirical knowledge to choose the best solution, while ML optimization automatically chooses the optimal design. The main advantage of the ML techniques is its superior ability to solve large-scaled optimization problems. The successful results for this communication prove ML techniques to be a new and powerful tool to tackle difficult design problems such as the initial design. These techniques can further be used to solve even more complex problems. For example, one can allow a flexible choice of the antenna shape, structure, and material, by formulating a large optimization problem and then use the ML to search within a high-dimensional input space for optimal design. To achieve this, we will need to collect more training data.

IV. Conclusion

In this communication, three ML techniques namely lasso, ANN, and kNNs are used to automatically identify the optimal values of the design parameters for a reference antenna where it can provide the best performance in terms of bandwidth of two bands. The brief description about these techniques is first presented in this communication and then how these techniques are applied to a reference double T-shaped monopole antenna is explained. With the help of these ML techniques, performance of the reference antenna is analyzed for 2 126 696 design points within a few seconds by learning from the training data set of 450 data points only. Compared to kNN, both ANN and lasso give more accurate predictions in our study. In summary, these new methods are more efficient than traditional method of EM simulation optimization for achieving optimal antenna design.

The results obtained from this research imply that ML techniques have the power to revolutionize EM simulation technology. Due to computational power limits of EM tools, it is challenging

and time consuming to optimize complex antenna designs like 3-D antenna structures involving a large number of design parameters. This problem can be addressed by incorporating ML techniques in simulation tools. The ultimate goal of this communication is to further generalize ML methods for complex design structures such as structures manufactured by 3-D printing technology. For example, 3-D printing enables designs with large numbers of degree of freedom and, therefore, optimizing all parameters through EM simulations is both tedious and computationally intensive. Our preliminary results have shown that ML techniques may be able to enable versatile and potentially automated design of antennas which will be beneficial for a number of applications including IoT.

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] T. Hastie and P. Simard, "Metrics and models for handwritten character recognition," *Stat. Sci.*, vol. 13, no. 1, pp. 54–65, 1998.

[3] T. S. Furey, N. Cristianini, N. Duffy, D. W. Bednarski, M. Schummer, and D. Haussler, "Support vector machine classification and validation of cancer tissue samples using microarray expression data," *Bioinformatics*, vol. 16, no. 10, pp. 906–914, Oct. 2000.

[4] S. Koziel and S. Ogurtsov, "Multi-objective design of antennas using variable-fidelity simulations and surrogate models," *IEEE Trans. Antennas Propag.*, vol. 61, no. 12, pp. 5931–5939, Dec. 2013.

[5] D. Z. Zhu, P. L. Werner, and D. H. Werner, "Design and optimization of 3D frequency selective surfaces based on a multi-objective lazy ant colony optimization algorithm," *IEEE Trans. Antennas Propag.*, vol. 65, no. 12, pp. 7137–7149, Dec. 2017.

[6] Y. Rahmat-Samii, J. M. Kovitz, and H. Rajagopalan, "Nature-inspired optimization techniques in communication antenna designs," *Proc. IEEE*, vol. 100, no. 7, pp. 2132–2144, Jul. 2012.

[7] 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.

[8] Z. Zheng, X. Chen, and K. Huang, "Application of support vector machines to the antenna design," *Int. J. RF Microw. Comput.-Aided Eng.*, vol. 21, no. 1, pp. 85–90, Jan. 2011.

[9] M. Martinez-Ramon and C. Christodoulou, "Support vector machines for antenna array processing and electromagnetics," *Synth. Lectures Comput. Electromagn.*, vol. 1, no. 1, pp. 1–120, Jan. 2006.

[10] T. Sallam, A. B. Abdel-Rahman, M. Alghoniemy, Z. Kawasaki, and T. Ushio, "A neural-network-based beamformer for phased array weather radar," *IEEE Trans. Geosci. Remote Sens.*, vol. 54, no. 9, pp. 5095–5104, Sep. 2016.

[11] K.-C. Lee and T.-N. Lin, "Application of neural networks to analyses of nonlinearly loaded antenna arrays including mutual coupling effects," *IEEE Trans. Antennas Propag.*, vol. 53, no. 3, pp. 1126–1132, Mar. 2005.

[12] G. Dadashzadeh, M. Kargar, Y. Torabi, and B. Rahmati, "Broad-band and wide scan phased array element design using data mining," *Appl. Comput. Electromagn. Soc. J.*, vol. 31, no. 3, pp. 244–251, 2016.

[13] Y.-L. Kuo and K.-L. Wong, "Printed double-T monopole antenna for 2.4/5.2 GHz dual-band WLAN operations," *IEEE Trans. Antennas Propag.*, vol. 51, no. 9, pp. 2187–2192, Sep. 2003.

[14] Y. Sharma, J. Wu, H. Xin, and H. H. Zhang, "Sparse linear regression for optimizing design parameters of double T-shaped monopole antennas," in *Proc. IEEE Int. Symp. Antennas Propag. USNC/URSI Nat. Radio Sci. Meeting*, Jul. 2017, pp. 347–348.

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

[16] R. Tibshirani, "Regression shrinkage and selection via the lasso: A retrospective," *J. Roy. Stat. Society, B (Stat. Methodol.)*, vol. 73, no. 3, pp. 273–282, Jun. 2011.

[17] *R: A Language and Environment for Statistical Computing*, R Found. Stat. Comput., Vienna, Austria, 2013.

[18] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference and Prediction*, 2nd ed. New York, NY, USA: Springer, 2009.

[19] I. Develi, "Application of multilayer perceptron networks to laser diode nonlinearity determination for radio-over-fibre mobile communications," *Microw. Opt. Technol. Lett.*, vol. 42, no. 5, pp. 425–427, Sep. 2004.

[20] J. B. Bradley, "Neural networks: A comprehensive foundation," *Inf. Process. Manage.*, vol. 31, no. 5, p. 786, Sep. 1995.

[21] J.-S. Jang, "Self-learning fuzzy controllers based on temporal backpropagation," *IEEE Trans. Neural Netw.*, vol. 3, no. 5, pp. 714–723, 1992.

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

[23] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, Oct. 1986.

[24] J.-M. Wu, "Multilayer potts perceptrons with Levenberg–Marquardt learning," *IEEE Trans. Neural Netw.*, vol. 19, no. 12, pp. 2032–2043, Dec. 2008.

[25] H. Demuth and M. Beale, "Neural network toolbox for use with MATLAB—User's guide version 4.0," MathWorks, Natick, MA, USA, Tech. Rep., 2004.

[26] 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.

[27] A. Beygelzimer *et al.*, "Fast nearest neighbor search algorithms and applications, R package, version 1.1.2.1," R Found. Stat. Comput., Vienna, Austria, Tech. Rep., 2018.