Deep Learning Methods for Optimizing Terahertz Leaky Wave Antenna Design

J. Neronha, H. Guerboukha, and D. Mittleman

School of Engineering, Brown University, Providence, RI 02912

Introduction

The predictive power of neural networks and machine learning techniques more generally have been applied to a vast array problems ever since the neural network was proposed as a computational technique loosely modeling the brain in the mid-twentieth century. [1] This, of course, includes communications technology – neural networks have been used extensively in the field because of their unique ability to approximate accurate solutions to nonlinear problems that are commonplace in antenna design. Machine learning models are particularly useful in situations where an analytical solution cannot be obtained and/or numerical simulations are expensive [2, 3], for example when solving the "direct" problem of approximating an antenna's output [4] or modeling a metasurface [5], which could take a finite-element model hours or longer to solve, limiting real-time simulation and design. The "inverse" problem considers the opposite, for example reconstructing an image from scattered light [6].

We are particularly interested in the inverse problem of terahertz leaky-wave antenna (LWA) design because of its high applicability to the field of communications – given some arbitrarily desired far-field pattern, how can we quickly design, fabricate, and test an antenna that meets our needs? LWAs are simple metallic waveguides that have proven very effective in the terahertz range and are particularly interesting because they emit radiation at a frequency-dependent angle with a one-to-one relationship between frequency and angle, which is quite valuable given the narrow character of beams in the terahertz range. [7] They have been used in a number of diverse applications including link discovery [8], multiplexing and demultiplexing [9, 10], and for radar and object detection purposes [11]. Leaky-wave antennas also stand out for our purposes of rapid design and experimentation because of the ease of fabricating them using novel hot-stamping techniques [12].

This inverse problem has been explored in the terahertz range using deep neural networks in the context of designing structures to obtain an optimized geometry that produces the desired signal, particularly in relation to metasurface design [13, 14]. The inverse Leaky-wave antenna problem has also been explored, but using a genetic algorithm and outside of the terahertz range at much higher frequencies [15]. As a result, in this paper, we propose a model to predict the ideal LWA geometry that will generate desired far-field radiation in the terahertz range.

Methodology

We train a model to predict a slot geometry for a desired far-field signal. In particular, we consider a one-dimensional discretized slot split into thirty-six 0.5 micron sub-slots that can either be

transparent or metallic, similar to the approach taken in past explorations of metasurface design [15]. This design essentially creates a non-uniform periodic waveguide that has peaks which can be predicted by Floquet theory. In a leaky wave guide with plate separation h, the dispersion constant β for a given period Λ and Floquet mode p for a wave vector k_0 is given by:

$$\beta = \sqrt{k_0^2 - (\frac{\pi}{h})^2} + \frac{2\pi p}{\Lambda} \tag{1}$$

where k_0 is the wave number and h is the height of the slot. One primary objective design of antenna design is to generate peaks of specific magnitudes at specific locations. Thus, we wish to design a slot that will generate the desired amplitudes at the peaks from each possible Floquet mode. Our randomly generated slots form linear combinations of periodicities that greatly expands the potential peak profiles compared to a slot with a simple constant periodicity and makes the problem significantly more challenging to solve analytically, introducing the need for a model like ours. We wish to compute all possible phase constants for each possible periodicity across all Floquet modes; if they are valid (i.e. $1 < \frac{\beta}{k_0} = n_{eff} < 1$), we can generate the predicted scattering angle by $\theta = \cos^{-1}(n_{eff})$. In Figure 1 below, we compute all possible peak angles for an arbitrary discretized slot predicted by Floquet modes and overlay those peaks on a sample far-field signal generated numerically. We will extract these peaks from the simulation data and use them as the input training data along with the slot used to generate it in order to predict a slot design given some peak profile in the neural network model.

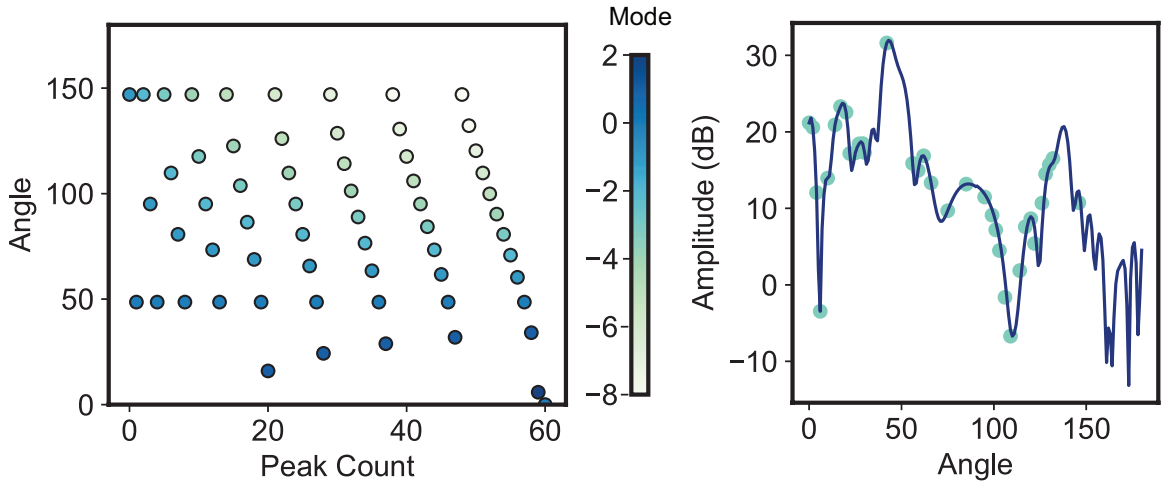
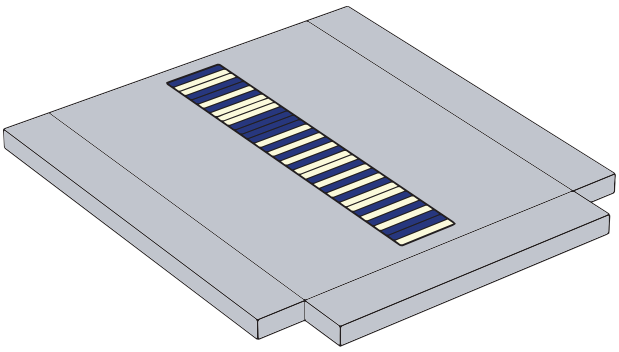


Figure 1: The left panel shows the possible Floquet modes from any linear combination of modes predicted by Equation 1; as expected, the positive modes occur in forward scattering (¡ 90deg) and the negative modes in back-scattering. The right panel shows these predicted Floquet modes overlaid on a sample peak profile generated in COMSOL.

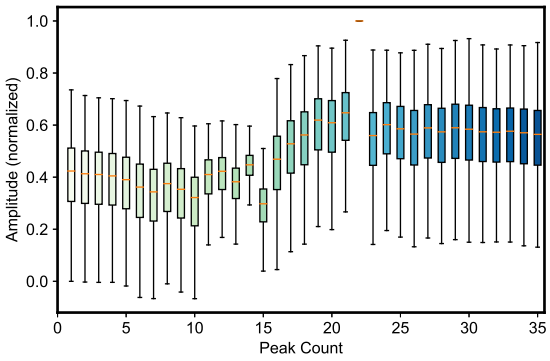
To generate our training data, we build a model of a leaky-wave antenna in COMSOL Multiphysics with a slot of length 18 mm divided into 36 sub-slots, each of which is 500 microns in length. 18 sub-slots (exactly half) of the slots are randomly chosen as to be transparent (i.e. a scattering boundary layer) whereas the balance remain metallic and thus do not leak radiation. We automate this random geometry generation and simulation using MPh [16], an open-source Python package that enables controlling COMSOL via its API. Basic combinatorics theory tells us that there are over 9 billion possible slot designs that can be generated, indicating the usefulness of deep neural networks in this prediction task given the sheer number of possible outputs; using only a few tens of

thousands sets of training data, we can predict an optimal slot design for an arbitrary peak profile. The simulation geometry is shown in Figure 2a below

Figure 2: Simulation geometry (left) and possible peak profiles (right)



(a) The leaky wave antenna geometry used to generate training data. Dark-colored sub-slots represent transparent areas while light colors indicate metallic boundaries; both are randomly generated so that the slots are half and half.



(b) A box plot showing the amplitudes at each peak; peak #23 is always the highest amplitude because it is the primary emission angle of all LWAs, periodic and not, at 200 GHz. A wide array of values are evident.

25,000 instances of COMSOL simulation data is used as the data source for our deep neural network, with a test-train split of 80-20. Figure 2b above shows the possible values for each peak, giving an approximate range over which we can design peak profiles for this particular geometry. The network, which is built with TensorFlow, has six dense layers, each of which have 2000 neurons and use a LeakyReLU activation function with $\alpha = 0.1$. The model uses the Adam optimizer and has a learning rate of 0.0001 and predicts the output of each individual sub-slot with a sigmoid activation function indicating the probability that a given sub-slot should be transparent. The model architecture is visualized in Figure 3.

The tricky part of the inverse problem, as we consider here, is finding an appropriate loss function – the primary objective is to generate a slot that minimizes the difference in signal compared to what is desired, but in order to use this idea as our loss function, we would need to compute the

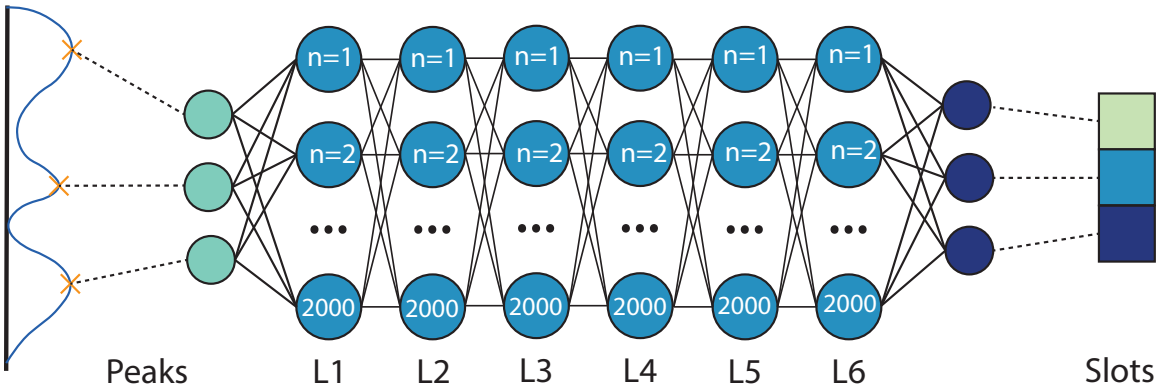


Figure 3: Neural network architecture schematic; peaks are used as the input data and passed through six hidden layers to predict the ideal slot design that will yield the closest possible signal.

far-field signal for each slot we generate as we train the model, which would be extraordinary (and unfeasibly) computationally expensive. Thus, instead, we turn to the idea of making the generated slot look as similar to the given training slot for a given peak profile as possible. Given that this is essentially a binary classification problem for each sub-slot, we use three weighted binary cross-entropy (BCE) loss functions, which are summed together to form our overall loss function with the following weightings:

1. BCE between the true and predicted slot (2x)
2. BCE between the first derivative of the true and predicted slot using a Prewitt filter (1x)
3. BCE between the second derivative of the true and predicted slot using a Laplacian filter (1x)

Results and Discussion

Binary Predictions

For some given peak geometry, the model returns each individual sub-slot's probability of being transparent (i.e. a value close to 0 indicates the sub-slot should be metal, whereas a value close to 1 indicates a transparent sub-slot) using a sigmoid activation function. In Figure 4 below, for two sample peak profiles, the true slots (left), predicted slot (center) and rounded ML slot (right, i.e. making the 16 slots with the highest probabilities transparent and vice versa). Both of these predicted slots share many similarities with the true slot but there are also some differences – the testing accuracy was about 68% averaged over all of the training samples.

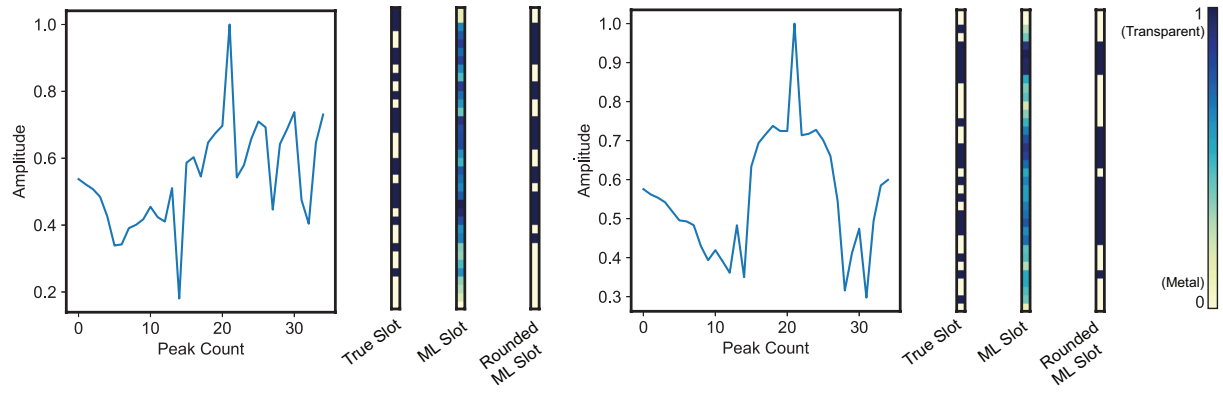


Figure 4: Model predictions for two different peak profiles. The left-most slot design is the randomly generated slot used to generate this profile, the center plot shows the sigmoid probabilities of a transparent sub-slot, and the right-most plot shows the top 16 sub-slots from the probabilities.

However, what determines the usefulness of our model is not the similarity of the slot but the similarity of the signal it generates to the objective. In order to consider the accuracy and usefulness of our model, we extract the predicted slot geometries, choose the top 16 slots based on the probabilities, and pass 500 of these new generated slots into our COMSOL model in order to compare the signal from the ML-generated slot to the originally desired peak profile. We then compare the peak profiles to the objective signal and calculate using the mean square error (MSE), as shown in the top left panel of Figure 5 below; the shape of the distribution matches that of similar work [5]. In the other panels of Figure 2, we plot the objective peak profile in black against the peak profile of an ML-generated slot, where the color corresponds to the four colored buckets

depicted in the histogram. Overall, the agreement is quite excellent, particularly for the first two buckets (green and light blue), which compose about 60% of testing samples with trends and relative amplitudes of the peaks agreeing very well. Even in the lower-tiered buckets, like in the bottom right panel of Figure 2, many of the peak profile trends are well matched – however, there are some areas where the model does not perform as well, resulting in a higher MSE. Overall, though, the agreement is excellent and shows that the model can be used to design an acceptable slot for many potentially desired peak profiles.

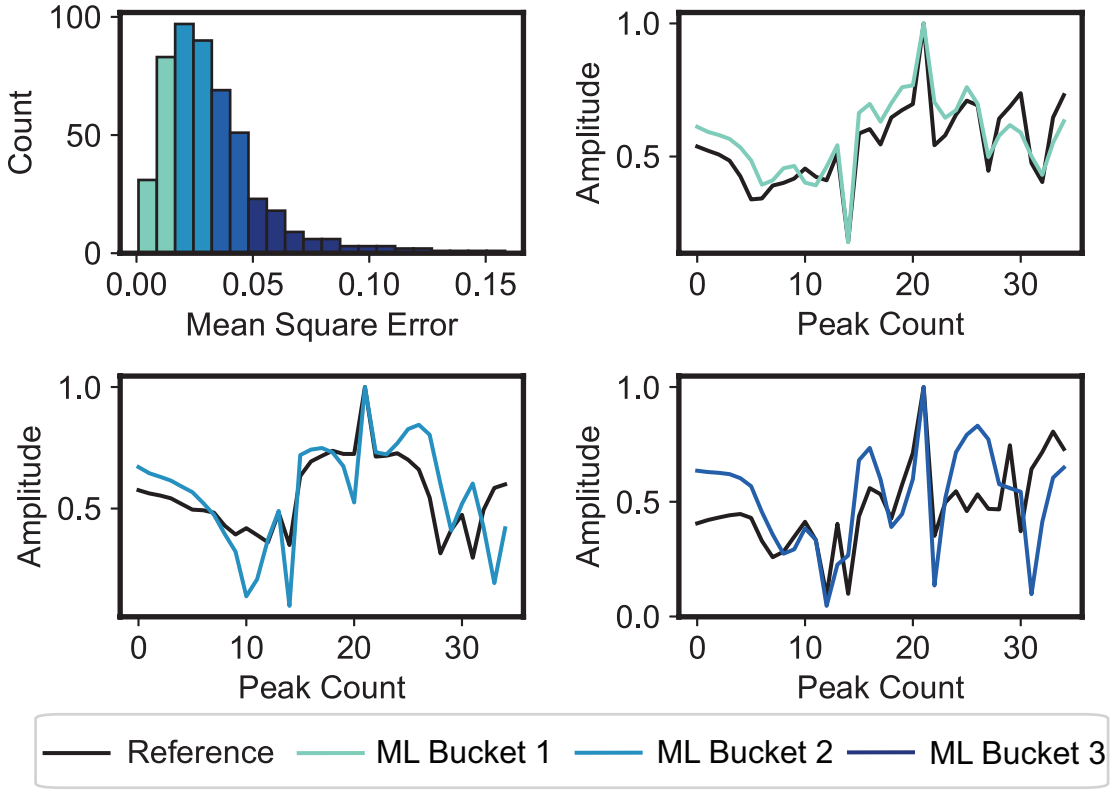


Figure 5: Results from COMSOL simulations of ML-generated slots. The top left panel shows an MSE histogram while the others compare the ML-slot generated peak profile to the objective.

Conclusion

References

[1] W. S. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *The bulletin of mathematical biophysics*, vol. 5, no. 4, pp. 115–133, 1943.

[2] Y. Kim, "Application of machine learning to antenna design and radar signal processing: A review," in *2018 International Symposium on Antennas and Propagation (ISAP)*, pp. 1–2, 2018.

[3] A. Massa, D. Marcantonio, X. Chen, M. Li, and M. Salucci, "Dnns as applied to electromagnetics, antennas, and propagation—a review," *IEEE Antennas and Wireless Propagation Letters*, vol. 18, no. 11, pp. 2225–2229, 2019.

[4] H. M. Yao and L. J. Jiang, "Machine learning based neural network solving methods for the fdtd method," in *2018 IEEE International Symposium on Antennas and Propagation USNC/URSI National Radio Science Meeting*, pp. 2321–2322, 2018.

[5] C. C. Nadell, B. Huang, J. M. Malof, and W. J. Padilla, "Deep learning for accelerated all-dielectric metasurface design," *Opt. Express*, vol. 27, pp. 27523–27535, Sep 2019.

[6] Y. Sun, Z. Xia, and U. S. Kamilov, "Efficient and accurate inversion of multiple scattering with deep learning," *Opt. Express*, vol. 26, pp. 14678–14688, May 2018.

[7] H. Guerboukha, R. Shrestha, J. Neronha, O. Ryan, M. Hornbuckle, Z. Fang, and D. M. Mittleman, "Efficient leaky-wave antennas at terahertz frequencies generating highly directional beams," *Applied Physics Letters*, vol. 117, no. 26, p. 261103, 2020.

[8] Y. Ghasempour, R. Shrestha, A. Charous, E. Knightly, and D. M. Mittleman, "Single-shot link discovery for terahertz wireless networks," *Nature Communications*, vol. 11, no. 1, p. 2017, 2020.

[9] N. J. Karl, R. W. McKinney, Y. Monnai, R. Mendis, and D. M. Mittleman, "Frequency-division multiplexing in the terahertz range using a leaky-wave antenna," *Nature Photonics*, vol. 9, no. 11, pp. 717–720, 2015.

[10] J. Ma, N. J. Karl, S. Bretin, G. Ducournau, and D. M. Mittleman, "Frequency-division multiplexer and demultiplexer for terahertz wireless links," *Nature Communications*, vol. 8, no. 1, p. 729, 2017.

[11] Y. Amarasinghe, R. Mendis, and D. M. Mittleman, "Real-time object tracking using a leaky thz waveguide," *Opt. Express*, vol. 28, pp. 17997–18005, Jun 2020.

[12] H. Guerboukha, Y. Amarasinghe, R. Shrestha, A. Pizzuto, and D. M. Mittleman, "High-volume rapid prototyping technique for terahertz metallic metasurfaces," *Opt. Express*, vol. 29, pp. 13806–13814, Apr 2021.

[13] Y. Deng, S. Ren, K. Fan, J. M. Malof, and W. J. Padilla, "Neural-adjoint method for the inverse design of all-dielectric metasurfaces," *Opt. Express*, vol. 29, pp. 7526–7534, Mar 2021.

[14] E. Dejband, J.-W. Li, P.-C. Peng, and T.-H. Tan, "Prediction of thz absorption and inverse design of graphene-based metasurface structure using deep learning," in *2021 30th Wireless and Optical Communications Conference (WOCC)*, pp. 21–23, 2021.

[15] S. Jafar-Zanjani, S. Inampudi, and H. Mosallaei, "Adaptive genetic algorithm for optical metasurfaces design," *Scientific Reports*, vol. 8, no. 1, p. 11040, 2018.

[16] J. Hennig, M. Elfner, and J. Feder, "Mph-py/mph: Mph 1.1.5," Feb. 2022.