DLSU Research Congress 2022
De La Salle University, Manila, Philippines
July 6 to 8, 2022

Simulation Neural Network Trained by Two-Dimensional Finite Difference Time Domain Dataset for Determining Frequency Responses of Metagratings

Francis Dela Cruz[1] and Romeric Pobre[1,*]
[1] *OPTICS Research Group, De La Salle University-Manila*
**Corresponding Author: romeric.pobre@dlsu.edu.ph*

Abstract: In recent years, Convolutional Neural Network (CNN) has been used for various machine learning tasks such as Image Recognition, Computer Vision, and Natural Language Processing. Recent developments on its application on designing electromagnetic devices such as metamaterials lead to fast and precise computation of seemingly slow and difficult to obtain frequency responses through numerical methods such as Finite Difference Time Domain (FDTD) and Finite Element Analysis (FEM). This study aims to develop a CNN, to be called as Simulation Neural Network (SNN), that can produce frequency responses of metagratings of different geometric configuration faster than FDTD can do. The CNN architecture is composed of 10 convolutional blocks, including the input block, and 4 fully dense layer, plus another dense layer for the sigmoid function, producing the discretized frequency responses. The SNN is written in Tensorflow 2 programming framework and trained using the frequency response data from the FDTD simulation of 2450 samples of metagratings, with loss set to mean squared error, metric set to accuracy, and optimizer set to Adam optimization algorithm. For the transmission dataset, the SNN training reported a loss of 1.3751e-4 and an accuracy of 82.63%, while the in the validation set, a mean square error of 2.496e-04 at 83.20% accuracy. Also, the SNN can produce predictions within seconds for bulk testing, in contrast to FDTD, giving off results within minutes. Plotting the FDTD results together with the predicted plots from SNN shows close predictions from the numerical method dataset. This makes the SNN as a powerful complement for numerical methods in analyzing electromagnetic structures.

Key Words: Neural Network; Metamaterials; Finite Difference Time Domain

1. INTRODUCTION

Neural Networks has received widespread recognition for various machine learning tasks, such as Image Recognition, Computer Vision, and Natural Language Processing. Neural Networks are generally considered algorithms that tries to mimic how brain neurons work, forming an interconnected network of nodes where learning happens by adjusting weights and biases using backpropagation methods. Various type of Neural Networks has been implemented for different applications, such as Artificial Neural Networks for prediction and classification tasks, Convolutional Neural Networks for image processing and computer vision, and Recurrent Neural Networks for Natural Language Processing and Translation. In recent years, Neural Networks have been used for

developing and designing electromagnetic devices for nanophotonics (Mall, 2020), wireless power transfer (Bui et al., 2020), and phase manipulation (Li et al, 2021). Most of the time, known numerical methods such as Finite Difference Time Domain (FDTD), Finite Difference Frequency Domain (FDFD), and Finite Element Analysis (FEM) are being used for solving electromagnetic wave-matter interaction and spectral responses. However, these methods are computationally slow and resource demanding, especially at more complicated models. To address these difficulties, there are various studies presenting the use of NN-enabled simulations, however, various methodologies only employed neural network architecture with only fully dense layers, which is limited when it comes to capturing structural complexities. One way to accommodate the ever-increasing geometrical and material demands is to use convolutional layers and blocks that can process layers of images. In the study of Mall et al., they showed that a convolutional neural network can be used for mapping 2D images of nanophotonic structures and their polarization conversion efficiencies, in a fraction of time compared to traditional full wave solvers such as FDTD. However, more work is needed when it comes to determining spectral responses of electromagnetic devices, such as metagratings using CNNs, while reducing the amount of time for simulation. The purpose of this work is to provide a neural network architecture that can simulate the frequency responses of metagrating elements comparable to numerical solvers in a short span of time. Using a training set from FDTD, the Simulation Neural Network (SNN) learns from the connections of the structural features and the corresponding frequency responses. The use of SNN significantly reduced the time needed for solving spectral responses of metagratings, with minimal deviation from the numerical solution.

2. METHODOLOGY

2.1 Preparation of the Training Set

Metagratings can be modeled as a two-dimensional object, as the third dimension, usually the z-axis is assumed to be extended to infinity. An algorithm was implemented to create various shapes in a 64 x 64 binary grid using MATLAB. A total of 744 filled shapes were created, which the value of permittivity can be adjusted using the equation below:

$$ER = er1 + (er2 - er1) * ER \qquad \text{(Eq. 1)}$$

where:

ER = the 64 x 64 zero permittivity array
er1 = initial value of permittivity across the ER array
er2 = the desired value of permittivity.

The 64 x 64 binary array represents the permittivity distribution, which forms the geometrical features of the metagrating element. (Fig. 1). Common shapes such as circles, ellipses, and rectangles, with different rotational orientation were also used for sampling.

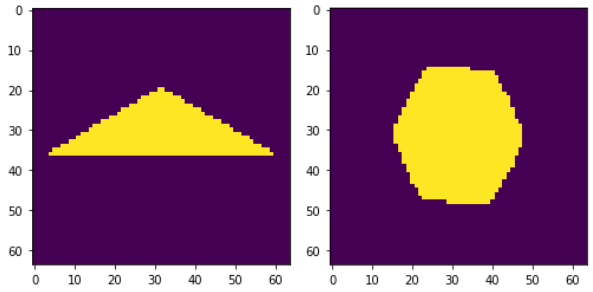


Fig. 1. A triangular (left) and hexagonal (right) metagrating elements.

Around 10%-20% of the total training sample is made for the test set. In this study, from 1340 samples of the training set, another 100 samples were added to serve as the test set for benchmarking the predictive capabilities of the SNN.

2.2 The FDTD Simulation

The 2D FDTD algorithm was implemented in MATLAB. The metagrating element structures represented by 64 x 64 array was saved using hdf5 data format. The HDF5 file was imported to MATLAB and then unpacked into MATLAB structure date containers. In the program dashboard, the FDTD simulation was set to run between 1 – 5 GHz with a 100 discretized number of frequency points. The physical dimension of the 64 x 64 grid is 1.5 cm x 1.5 cm, and then inserted in the simulation space with spacer regions scaled by 5% of the maximum wavelength. The relative permittivity used for the

element is 15 with electric conductivity set to 100 S/m. The frequency responses of the metagrating elements were solved for all the values of frequencies across 744 samples. The reflectance, transmittance, and absorptance data were all collected and stored in another HDF5 file container together with the 64 x 64 structure array to reduce storage size. The spectral responses plot can be seen in Fig 2.

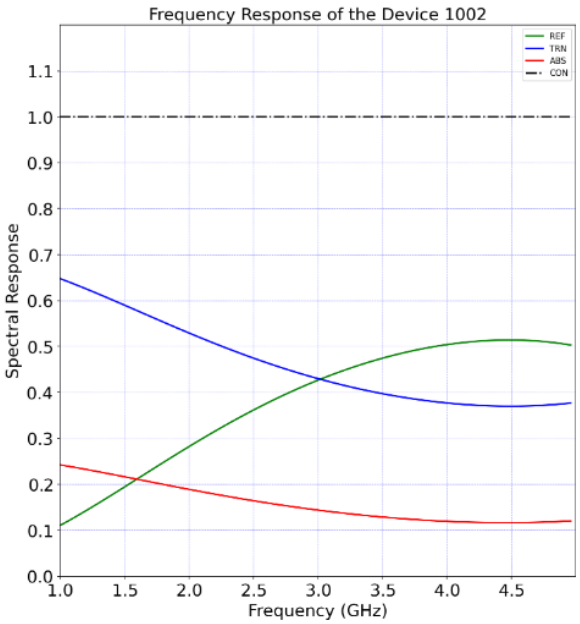


Fig. 2. Spectral responses of sample device 1002. Note that REF is the reflectance of the element, TRN is the transmittance, and ABS is the absorptance. CON represents the conservation, where REF + TRN + ABS = 1.

2.3 The Simulation Neural Network

The current study implemented a Convolutional Neural Network in Tensorflow 2 programming framework and run using Google Colaboratory to take advantage of the cloud computing capabilities. To match the FDTD parameters, the same parameters was initialized in the dashboard of the Simulation Neural Network. The training set was imported also in the python script through the HDF5 read command, and then all the data were separated and arranged in a way that each metagrating element is the training feature and the frequency responses to be training labels. The training

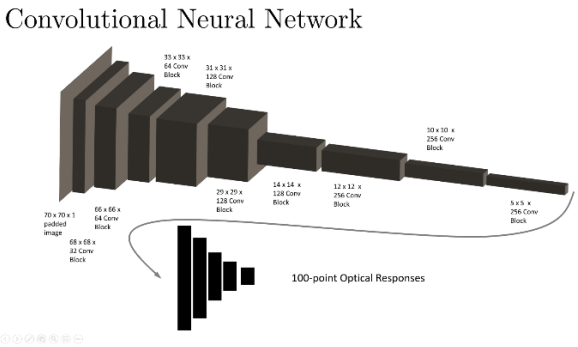


Fig. 3. Visual representation of the Simulation Neural Network Architecture. Better resolution at the end of the document.

labels were divided into three, one for reflectance, one for transmittance, and one for absorptance. Separate programs were prepared for each spectral response but only the transmission simulation was discussed. The CNN architecture can be seen in Figure 3, summarized in Table 1. The simulation was set to work under Adam Optimizer, with loss set to 'mean squared error' and metric set to 'accuracy'. The model was trained with batch size of 64 for 500 epochs.

Table 1. Simulation Neural Network Architecture Summary

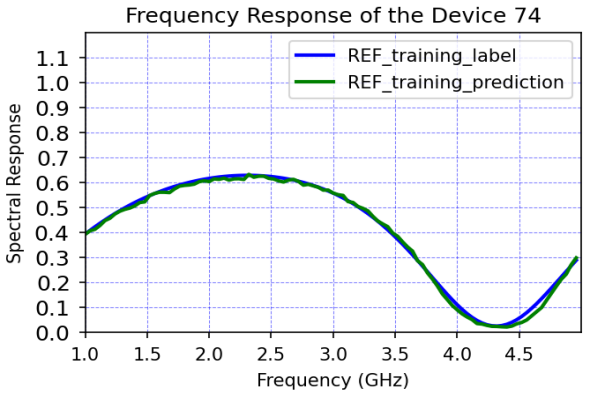
Layer	Output Shape	Available
zero_padding2d (ZeroPadding2)	(None, 70, 70, 1)	0
conv2d (Conv2D)	(None, 68, 68, 32)	320
batch_normalization (BatchNorm)	(None, 68, 68, 32)	128
re_lu (ReLU)	(None, 68, 68, 32)	0
conv2d_1 (Conv2D)	(None, 66, 66, 64)	18496
batch_normalization_1 (BatchNorm)	(None, 66, 66, 64)	256
re_lu_1 (ReLU)	(None, 66, 66, 64)	0
max_pooling2d (MaxPooling2D)	(None, 33, 33, 64)	0
dropout (Dropout)	(None, 33, 33, 64)	0
conv2d_2 (Conv2D)	(None, 31, 31, 128)	73856
batch_normalization_2 (BatchNorm)	(None, 31, 31, 128)	512

re_lu_2 (ReLU)	(None, 31, 31, 128)	0
conv2d_3 (Conv2D)	(None, 29, 29, 128)	147584
batch_normalization_3 (BatchNorm)	(None, 29, 29, 128)	512
re_lu_3 (ReLU)	(None, 29, 29, 128)	0
max_pooling2d_1 (MaxPooling2	(None, 14, 14, 128)	0
dropout_1 (Dropout)	(None, 14, 14, 128)	0
conv2d_4 (Conv2D)	(None, 12, 12, 256)	295168
batch_normalization_4 (BatchNorm)	(None, 12, 12, 256)	1024
re_lu_4 (ReLU)	(None, 12, 12, 256)	0
conv2d_5 (Conv2D)	(None, 10, 10, 256)	590080
batch_normalization_5 (BatchNorm)	(None, 10, 10, 256)	1024
re_lu_5 (ReLU)	(None, 10, 10, 256)	0
max_pooling2d_2 (MaxPooling2	(None, 5, 5, 256)	0
dropout_2 (Dropout)	(None, 5, 5, 256)	0
flatten (Flatten)	(None, 6400)	0
dense (Dense)	(None, 4096)	26218496
re_lu_6 (ReLU)	(None, 4096)	0
dense_1 (Dense)	(None, 2048)	8390656
re_lu_7 (ReLU)	(None, 2048)	0
dropout_3 (Dropout)	(None, 2048)	0
dense_2 (Dense)	(None, 1024)	2098176
re_lu_8 (ReLU)	(None, 1024)	0
dropout_4 (Dropout)	(None, 1024)	0
dense_3 (Dense)	(None, 512)	524800
re_lu_9 (ReLU)	(None, 512)	0
dense_4 (Dense)	(None, 100)	51300

Total params:	38,412,388
Trainable params:	38,410,660
Non-trainable params:	1,728

3. RESULTS AND DISCUSSION

The algorithm was tested using the 1340 training samples and 100 test samples. After 500 epochs of training, the SNN reached a loss of 1.3751e-4 and an accuracy of 82.63%, while the in the validation set, a mean square error of 2.496e-4 at 83.20% accuracy. Both the training and validation frequency responses were plotted against the frequency responses solved by numerical method such as 2D-FDTD, as you can see in Figure 3 for the training set and Figure 4 for the test set.



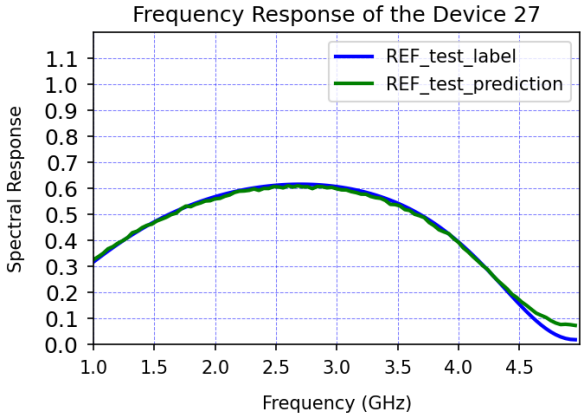
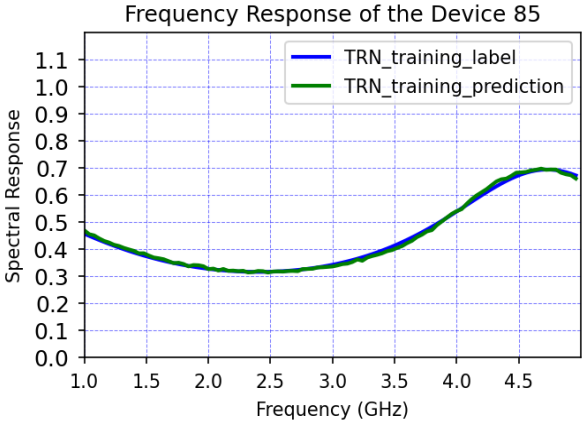
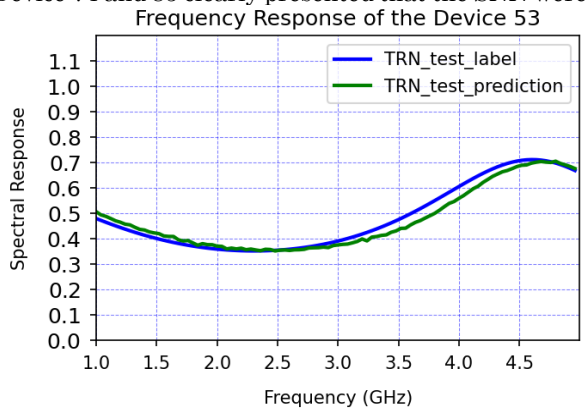


Fig. 3. Comparison of spectral responses between FDTD (training set) and SNN predicted results.



Device 74 and 85 clearly presented that the SNN were



trained to learn the connection between the metagrating element and its reflectance-transmittance.

Fig. 4. Comparison of spectral responses between FDTD (test set) and SNN predicted results. Device 27 and 53 were not originally part of the training set. This clearly shows that the SNN can solve the frequency response of the metagrating element just like how FDTD does.

For the time needed to produce results, Table 2 and 3 reported the simulation time difference between two methods for solving a single element. There is a striking gap between the solving time of FDTD and SNN.

Table 2. Simulation Time for getting the frequency response of a Single Metagrating Element

Methods	Batch Size	Simulation Time
FDTD	1	12-31 seconds
SNN	1	6.367e-4 seconds
Difference		11.9994-30.9994 seconds

Table 3. Simulation Time for getting the frequency response of a 100 Metagrating Element all at once.

Methods	Batch Size	Simulation Time
FDTD	1329	75600 seconds
SNN	1329	1.56 seconds
Difference		39598.44 seconds

4. CONCLUSIONS

Based on the results of the SNN, the frequency responses of the simulated elements are in almost the same as the responses solved using the FDTD method. Not only that the SNN algorithm learned from the training set, but it also able to predict the frequency responses of samples it has not seen before, which all belongs to the test set. Also, the simulation time difference between two solving methods is significantly huge, as SNN can solve responses of metagratings in less than 2 seconds for 1329 elements, in contrast to FDTD that the same 1329 samples can take almost 11 hours. This makes the SNN a vital tool for fast prototyping and solver of electromagnetic device elements. Albeit the need to use the FDTD as a source of training set, a validated SNN can further synthesized more training samples, which can lead to much faster and robust prediction of frequency responses.

5. ACKNOWLEDGMENTS

I would like to acknowledge the Department of Science and Technology – Philippine Council for Industry, Energy, and Emerging Technology

Research and Development for the funding the research project entitled "Development of Smart Polymer Metamaterial for Wearable Biosensors". Also, I'm grateful to Dr. Rene Batac for valuable insights about machine learning, to Dr. Wilfred Espulgar for important insights about how LSPR works that helped me gain new ideas about FDTD implementation, and to Dr. Raymond Rumpf of University of Texas, El Paso, for his golden lecture notes and incredible teaching materials such as video lectures and MATLAB webinars that made my FDTD and other numerical solvers possible.

6. REFERENCES (use APA style for citations)

Mall, A., Patil, Abhijeet P., Sethi A., & Kumar A. (2020). A cyclical deep learning based framework for simultaneous inverse and forward design of nanophotonic metasurfaces[Electronic version]. Nature Scientific reports. Retrieved February 20, 2021, from https://www.nature.com/articles/s41598-020-76400-y

Bui H.N., Kim J.S., & Lee J.W. (2020). Design of Tunable Metasurface Using Deep Neural Networks for Field Localized Wireless Power Transfer [Electronic version]. IEEE Access . Retrieved February 20, 2021, from https://ieeexplore.ieee.org/abstract/document/9239375

Li J., Li X., Wu Q., Wang L., & Gao L. (2021). Neural network enable metasurface design for phase manipulation [Electronic version]. Optics Express, Vol 29, No. 2. Retrieved February 20, 2021, from https://opg.optica.org/oe/fulltext.cfm?uri=oe-29-2-2521&id=446592

Answers to Reviewer's Questions:
2. Characterization of Neural Networks usually involves three steps. One is for checking the results using the training set, one for the validation set, and one for the test set. Training set characterization will show how it fare against the material it learned from. The validation set comes from the original distribution where the training set was come from, but these are samples that were set aside. Knowing how the machine will predict the validation set responses will give insights on the machine's performance against data coming from the same source or distribution. For the test set, the data comes from neither of the two distributions, showing that a good response on this dataset will give better insights on the effectivity of the machine predictions. It is a common practice within the machine learning community that the test set is set to be at least 10-20% of the training set.
3. I have not tried other algorithms. I used convolutional neural networks due to its capability to use 2D-3D objects to identify features due to the convolutional blocks that is commonly used in image recognition.

Convolutional Neural Network