Carleton University

Capstone Progress Report

An Artificial Neural Network For Simulating Silicon Photonic Devices

Partners:
Jonathan Levine
Jacob Ryall
Deng Mading
Yu Wan

Supervisors:
Winnie Ye
Leonard MacEachern

Author: Jonathan Levine

January 16th, 2022

1 Introduction

Artificial Neural Networks (ANNs), are an approach to machine learning in which a computer "learns" the relationship between a system's inputs and outputs through a network of nodes, weights and activation functions. A simplified diagram of a neural network is shown in Figure [1] below.

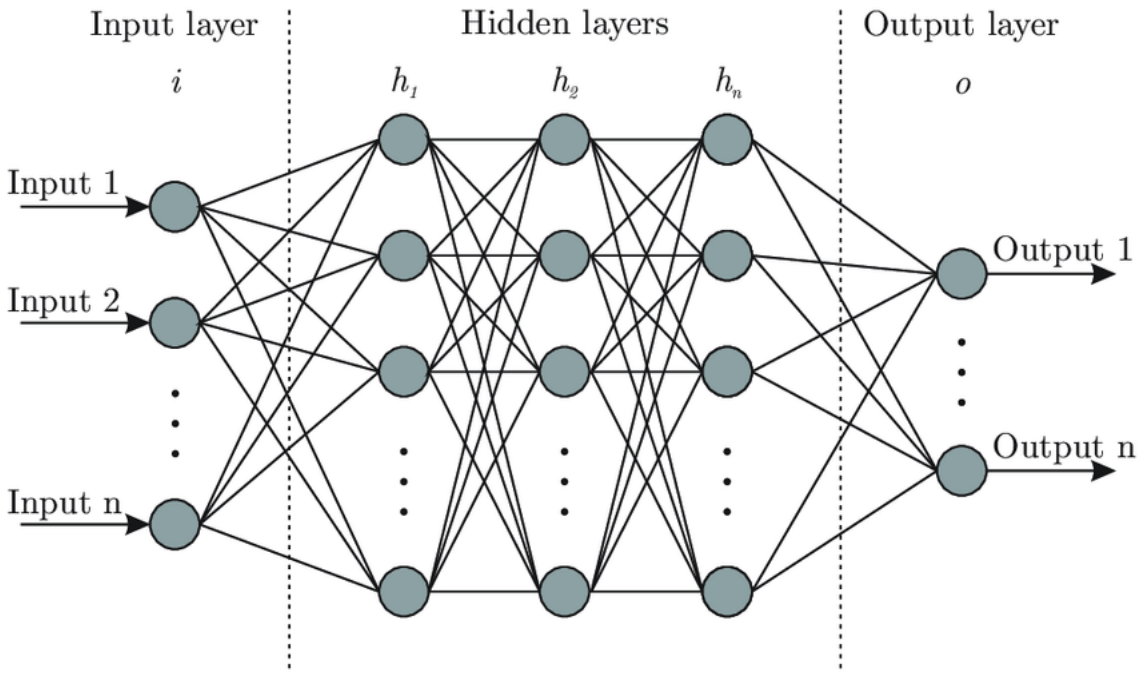

Figure 1: Neural Network Architecture [1]

Each of the nodes or *perceptrons* in the hidden layers, are connected to the next layer by weights and activation functions. These activation functions can be non-linear enabling the entire network to learn non-linear relationships between inputs.

At first the weights and biases throughout a network are assigned randomly, and thus the network does not predict the outputs accurately. Mathematical algorithms such as Optimizers are used during training to update the weights and biases in a neural network so that a loss function is minimized.

Because of a neural network's ability to learn non-linear relationships, they have recently

become very popular in Computer Aided Design (CAD) as they can offer quick and accurate simulation performance compared to traditional numerical techniques [2].

1.1 Background

The design of silicon-photonic devices requires finite-difference time-domain (FDTD) optical simulations. Software applications like Lumerical are used to do FDTD simulations.

FDTD simulations can be very time-consuming and require high performance hardware to run. Often it can take tens or hundreds of simulations to design a silicon-photonic device, further delaying the design process and the time to fabrication and testing [3]. Additionally these FDTD simulations requires a significant amount of power to run.

A well trained artificial neural-network can offer a powerful alternative to the standard numerical computation techniques by offering fast and accurate simulations. These ANN's can also reduce the amount of power needed to simulate silicon-photonic devices.

1.2 Objective

The main objective of this project is to develop with PyTorch an artificial neural network to simulate a silicon-photonics device.

The design of the neural network for this project can be broken down into two main sections: **data acquisition** and **model training**. The data acquisition will be done by Deng Mading and Randi Wan and the neural network training and development will be done by Jacob Ryall and myself.

Members	**Individual Objectives**
Yu Wan & Deng Mading	Data Acquisition
Jacob Ryall & Jonathan Levine	ANN Development with PyTorch

Table 1: Individual Objective Breakdown

The data acquisition will be done with Lumerical. Lumerical simulations will be used

because device parameters can be varied and measurements can be done in an automated fashion. Lumerical has a MATLAB and Python API, so scripts can be written in those languages to automate simulations.

The neural network will be developed using the Python PyTorch library. With simulations of different device parameters, we can select what features impact a device the most and select those as the input into the neural-network.

The specific size and number of the hidden-layers of the network will be determined with trial and error. However a good starting point from the paper, **"An Open-Source Artificial Neural Network Model for Polarization-Insensitive Silicon-on-Insulator Subwavelength Grating Couplers,"** [3] set the dimension of the hidden layers to 100-50-50, plus 5 input neurons and 4 output neurons.

Initially each perceptron will be equipped with the Rectified Linear Unit (ReLu) Activation function; however this may be subject to change throughout the development process. The learning rate of the network will also be determined through trial and error.

2 Theory and Techniques

As outlined in the previous sections, the objective of this project is to develop a neural network that can accurately model a silicon-photonic device. The specific device type we are trying to model are subwavelength grating (SWG) couplers.

The emission spectra of SWG couplers are highly influenced by changes in their features. These features include: pitch, duty cycle, fill factor, fiber angle, wavelength and polarization of the coupled light. Figure [2] below illustrates the parameters of a grating coupler.

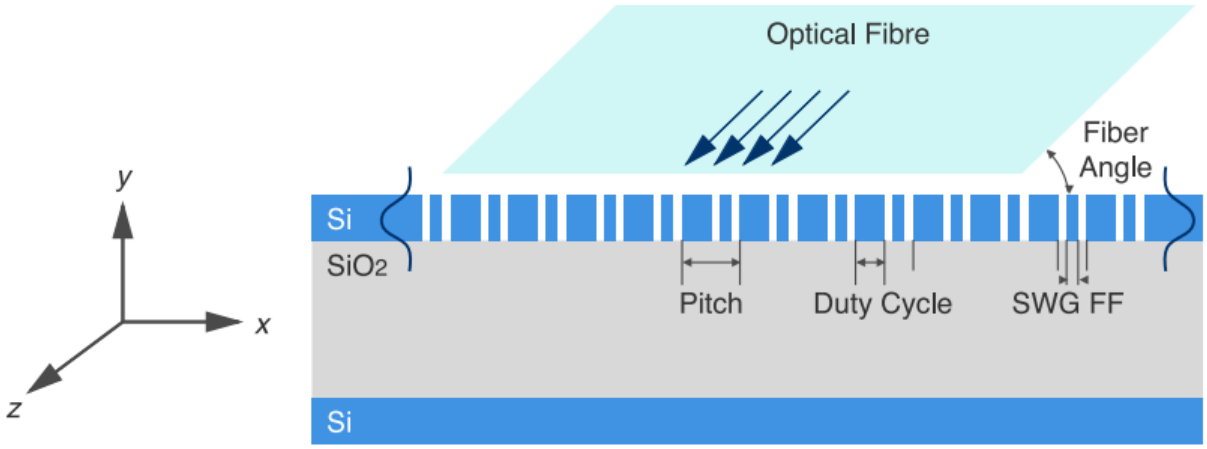

Figure 2: 2D Cross Section of an SWG Coupler

The use of neural networks to replace, or reduce the need for FDTD simulations is an emerging field and similar research has been done by a Graduate student at Carleton University, Dusan Gostimirovic, where he developed a neural network to model polarization-insensitive sub-wavelength grating couplers [3]. In his paper he was able to develop a neural network that performed to 93.2% accuracy of numerical simulations.

2.1 General Approach

With a large dataset that contains different values of pitch, duty cycle, fill factor, fiber angle, wavelength and polarization, and the resulting transmission, we could design a neural network to simulate an SWG coupler.

The first step in the design of the neural network was generating a dataset for the input features and output labels. The process of generating a large dataset from Lumerical proved to be challenging because the simulation time for silicon-photonic devices takes days to complete. And so our approach was to figure out a way to automate this process.

3 Results and Discussion

To date as a group we have been able to automate the automate the process of making a dataset from Lumerical using a script that iterates through values of pitch, duty cycle, fill factor, fiber angle and wavelength to calculate the transmission value.

The script uses Lumerical's built-in editor and scripting language and can be found in Appendix [A]. Also, all the ANN development is done with the Python PyTorch library, and the python files can be found at: **https://github.com/JonathanALevine/Capstone**.

3.1 Design of The Network

We set-up the ANN model of the grating so that it has six input features: pitch, duty cycle, fill factor, fiber angle, wavelength and polarization, and one output label, transmission.

A starting point for the ANN architecture came from the paper [3] where there are three hidden layers set to 100-50-50 nodes, plus 5 input neurons and 4 output neurons. The input neurons are for the features: pitch, duty cycle, fill factor, fiber angle, wavelength and polarization and the output neurons are for the labels: max transmission left, max transmission right, center wavelength left, center wavelength right.

For our models we decided to include the wavelengths in the training set and set the output labels to solely the transmission value.

To date we have three neural net architectures trained. Network 1 is structured by 6 input neurons, three hidden layers with 100-100-100 neurons and 1 output neuron. Network 2 is structured by 6 input neurons, two hidden layers with 50-100 neurons and 1 output neuron. Finally, Network 3 is structured by fourth hidden layers with 50-100-200-100 neurons and 1 output neuron.

The layers in the networks are connected by the ReLu activation function and the optimizer used to train the networks was the Adam algorithm and the loss function is the mean-squared-error between the neural-net's prediction and the expected value. The results

of the training are found in the following section.

3.2 Results and Discussion

From Figure [3] we can see that the current architectures we have do not converge to a solution. We can see that even after 20,000 training epochs, the MSE on Network 1 is still above 0.6.

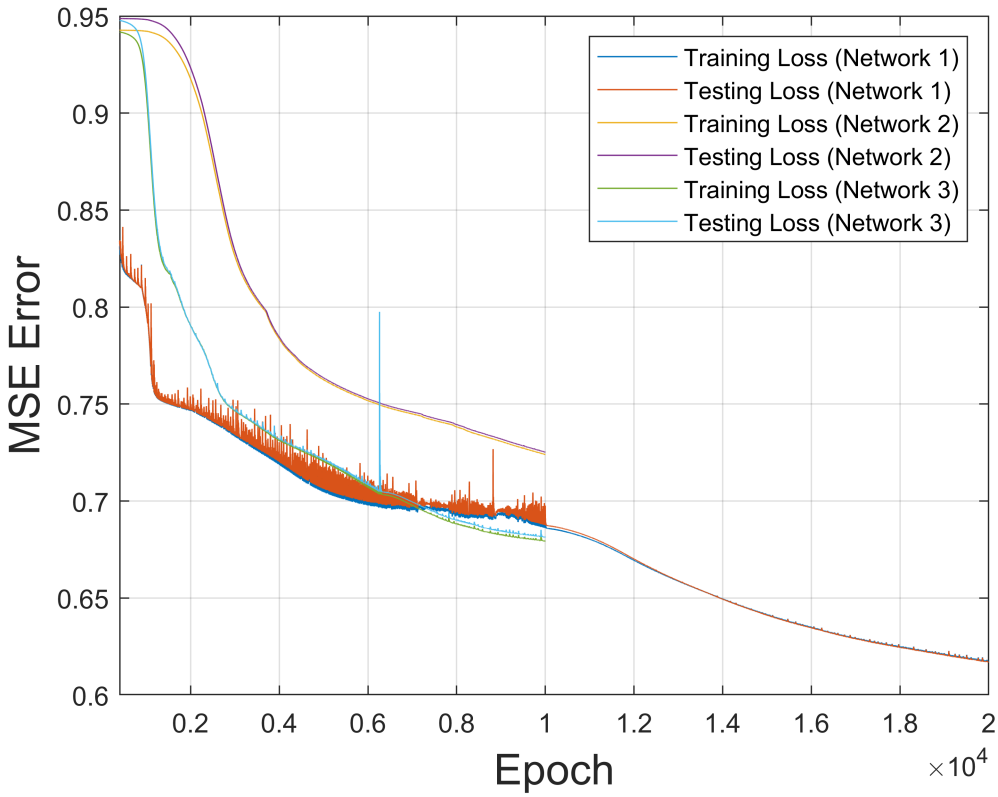

Figure 3: Mean-Squared-Error of The ANN's

We suspect that to model an SWG coupler accurately, larger network sizes are needed. We can see that Network 3 converges faster than the other two. Also we are starting to implement more sophisticated training methods. For example, we can see that the losses on Network 1 are relatively noisy from 1,000 to 10,000 epochs, but is more tame from 10,000 to 20,000 epochs. This indicates that perhaps the learning rate was set too high between 1,000

and 10,000 epochs. After 10,000 the learning rate in Network 1 was adjusted from 0.0001 to 0.00005.

Also, the amount of time required to these networks to converge is significant. They all required at-least 24 hours to train to this point, and even Network 3 needed 72 hours. These ANN's were trained on my home computer that has an i7 6700K 4.0 GHz CPU with 64 GB or RAM.

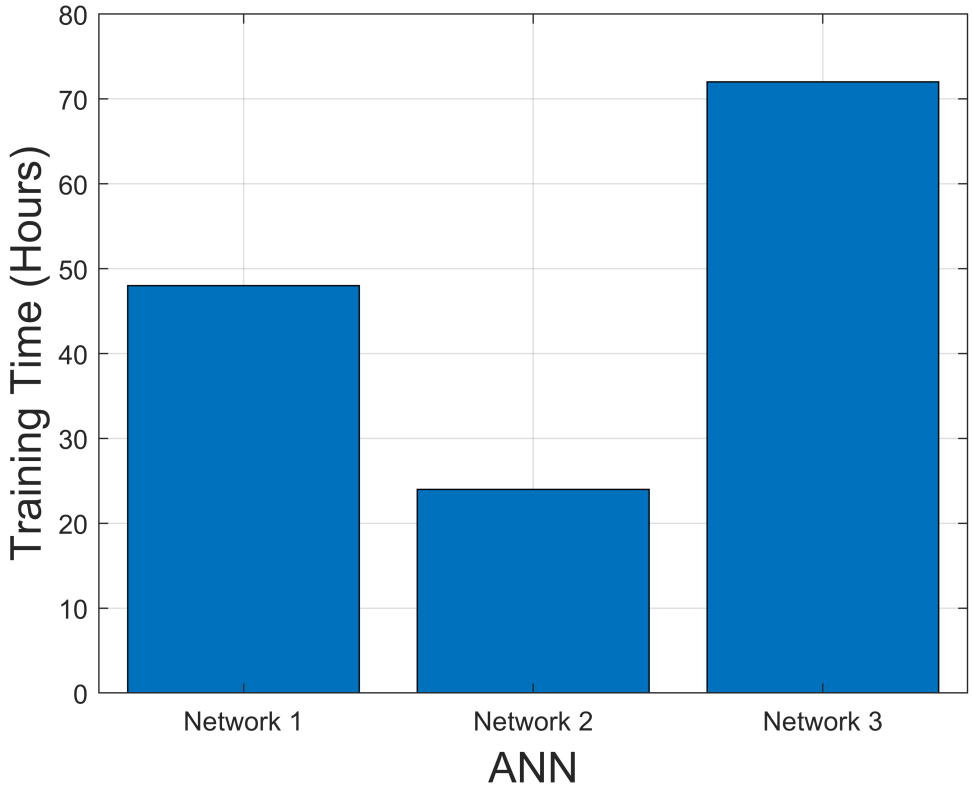

Figure 4: Training Time of The Different Network Architectures

4 Management and Time Table

SO far we have been sticking to our project timeline and the project has been moving on a good pace. For managing the project and ensuring that we stay on track, we will be having weekly meetings as a team every Friday from 1:30pm-2:30pm. Each member of the group is

part of Discord server where we report on our progress. Meetings throughout the week will also be scheduled from time to time to catch up.

The Friday afternoon meetings will be used to present each team members progress of the past week and assess what the next steps are. The overall project timeline can be found in Figure [5] below and the ANN development timeline can be found in Figure [6].

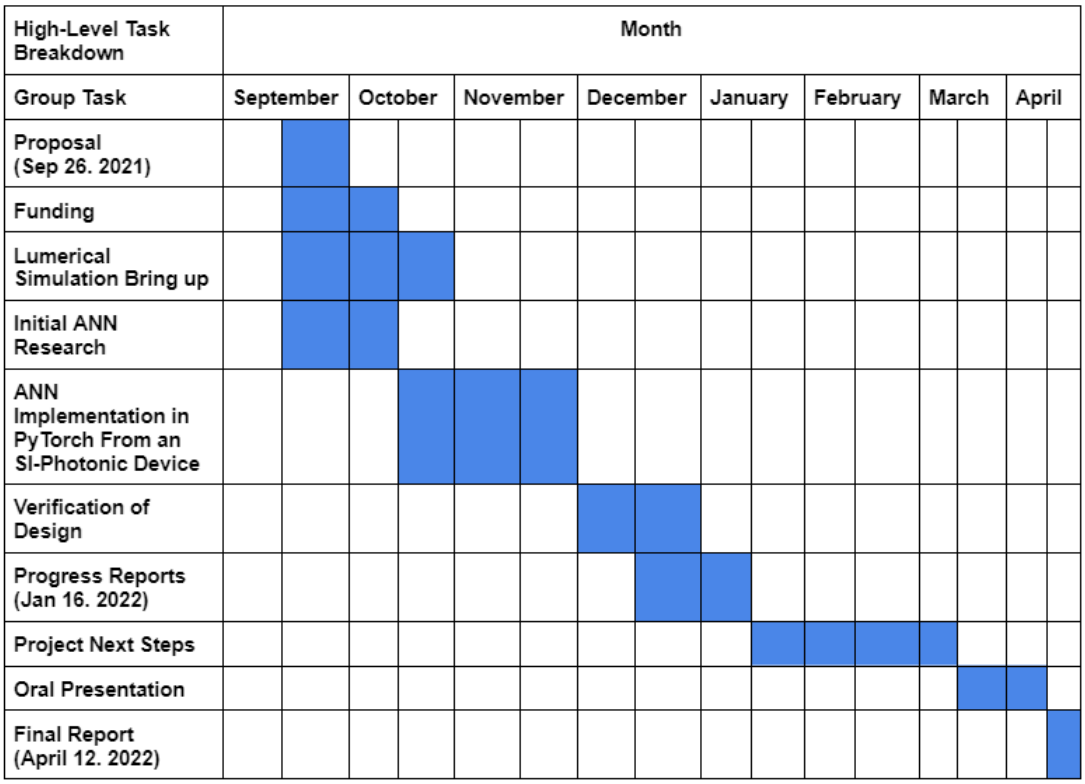

Figure 5: Overall Project Timeline For the Academic Year

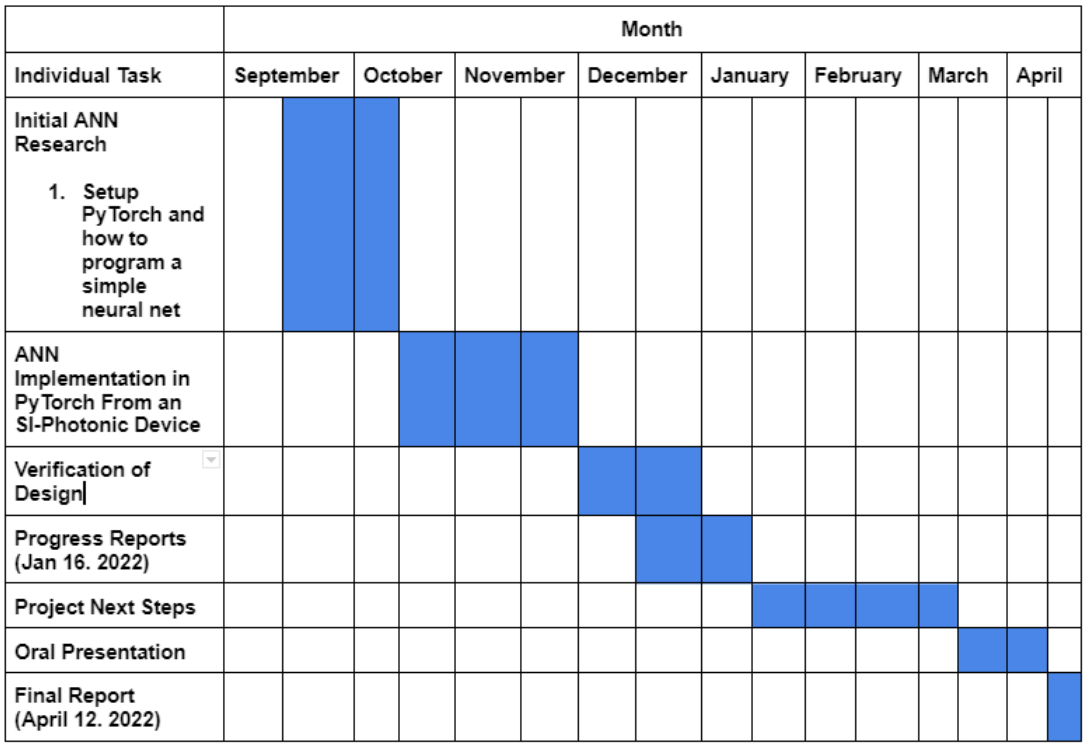

Figure 6: ANN Development Timeline For the Academic Year

5 Conclusion

In conclusion, since the proposal as a group we have gotten extremely familiar with Lumerical and how to script our own simulations. We have a method now of converting Lumerical simulations to usable datasets to train our neural networks. Additionally, we have learned to set up and design neural networks in PyTorch.

For the coming months we will be implementing more sophisticated training methods to improve the convergence of our ANN models and we will be testing them to see if they are capable of modelling different grating couplers.

A Lumerical Script

```
switchtolayout;

num = 5;

theta_vals = linspace(5, 20, num);
pitch_vals = linspace(0.5e-6, 1.5e-6, num);
duty_cycle_vals = linspace(0.4, 0.8, num);
fill_factor_values = linspace(0.2, 0.6, num);

current_time = now;

for(i=1:num){
for(j= 1:num){
for(k = 1:num){
for(z = 1:num){
switchtolayout;
select("fiber");
set("theta0", theta_vals(i));
select("gc");
set("pitch", pitch_vals(j));
set("duty cycle", duty_cycle_vals(k));
set("fill_little", fill_factor_values(z));
select("FDTD::ports");
set("source mode", "mode 1");
run;
```

```
a = getresult("FDTD::ports::port 3", "T");
switchtolayout;
select("FDTD::ports");
set("source mode", "mode 2");
run;
b = getresult("FDTD::ports::port 2", "T");
# Write to file
string = num2str(theta_vals(i)) + ", ";
string = string + num2str(pitch_vals(j)) + ", ";
string = string + num2str(duty_cycle_vals(k)) + ", ";
string = string + num2str(fill_factor_values(z)) + "\n";
for(number = 1:size(a.T, 1)){
        string = string +
        "(" + num2str(a.lambda(number, 1)) +
        ", " +  num2str(a.T(number,1)) + "), ";
}
string = string + "\n";
for(number = 1:size(b.T, 1)){
        string = string +
        "(" + num2str(b.lambda(number, 1)) +
        ", " + num2str(b.T(number, 1)) + "), ";
}
string = string + "\n";
filename = "Results_" + num2str(current_time) + ".txt";
write(filename, string, "append");
}
}
```

```
}
}
```

References

[1] (2021, Sept. 25). "Designing Your Neural Networks" [Online]. Available: https://www.google.com/url?sa=i&url=https%3A%2F%2Ftowardsdatascience.com%2Fdesigning-your-neural-networks-a5e4617027ed&psig=AOvVaw3eQb6L4YoFnAPSMdc1Ix6m&ust=1632703391203000&source=images&cd=vfe&ved=0CAsQjRxqFwoTCMiYnpu0m_MCFQAAAAAdAAAAABAN

[2] (2021, Sept. 25). "Efficient Modeling of Complex Analog Integrated Circuits Using Neural Networks" [Online]. Available: https://www.researchgate.net/publication/305410960_Efficient_Modeling_of_Complex_Analog_Integrated_Circuits_Using_Neural_Networks

[3] D.Gostimirovic and W.N.Ye, "An Open-Source Artificial Neural Network Model for Polarization-Insensitive Silicon-on-Insulator Subwavelength Grating Couplers," ***I****EEE JOURNAL OF SELECTED TOPICS IN QUANTUM ELECTRONICS, VOL, 25, NO.3, MAY/JUNE 2019.*