Nanophotonic Particle Simulation and Inverse Design Using Artificial Neural Networks

John Peurifoy[*] **Yichen Shen**[*†] **Li Jing**[*] **Yi Yang**[‡] **Fidel Cano-Renteria**[§]

Brendan Delacy[¶] **Max Tegmark**[*] **John D. Joannopoulos**[*] **Marin Soljačić**[*]

Abstract

We propose a method to use artificial neural networks to approximate light scattering by multilayer nanoparticles. We find the network needs to be trained on only a small sampling of the data in order to approximate the simulation to high precision. Once the neural network is trained, it can simulate such optical processes orders of magnitude faster than conventional simulations. Furthermore, the trained neural network can be used solve nanophotonic inverse design problems by using back-propogation - where the gradient is analytical, not numerical.

From molecular geometry search to nanophotonics designs [2, 5], inverse design problems and simulations play a large role in modern physics. Typically, these problems require optimization in high dimensional spaces. Particularly in photonics, where the forward calculations are well understood with Maxwell's equations [4], solving just one instance of an inverse design problem can often be a substantial research project. Currently, these problems are either meticulously solved by hand or solved numerically with the use of trial-and-error and extensive computing hours.

In this paper, we propose a novel method to further simulate light interaction with nanoscale structures and solve inverse design problems using Artificial Neural Networks (NNs) [6]. In this method, a NN is first trained to approximate a simulation; thus the NN is able to map the scattering function into a continuous, higher order space where the derivative can be found analytically. The "approximated" gradient of the figure of merit (FOM) with respect to input parameters is then obtained analytically with standard back-propagation [6]. The parameters are then optimized efficiently with the gradient descent method. Finally, we compare our performance with the standard gradient free optimization method and find our method is orders of magnitude faster and more effective than traditional methods.

1 NNs can learn and approximate Maxwell Interactions

We evaluate this method by considering the problem of light scattering from a multi-layer dielectric spherical nanoparticle — Fig. 1. Our goal is to use a NN to approximate this simulation. Specifically, we consider eight layers of alternating dielectric material (silica and TiO_2) between 30nm to 70nm thicknesses per layer. Thus the smallest particle we consider is 480nm in diameter, and the largest is 1,120nm. This problem can be solved analytically or numerically with the Maxwell equations, though for multiple layers, the solution becomes involved. The analytical solution is well known [4]. We used the simulation to generate 50,000 examples from these parameters with Monte-Carlo sampling.

[*]Department of Physics, Massachusetts Institute of Technology

[†]ycshen@mit.edu

[‡]Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology

[§]Department of Mathematics, Massachusetts Institute of Technology

[¶]U.S. Army Edgewood Chemical Biological Center, Aberdeen Proving Ground

Workshop on Deep Learning for Physical Sciences (DLPS 2017), NIPS 2017, Long Beach, CA, USA.

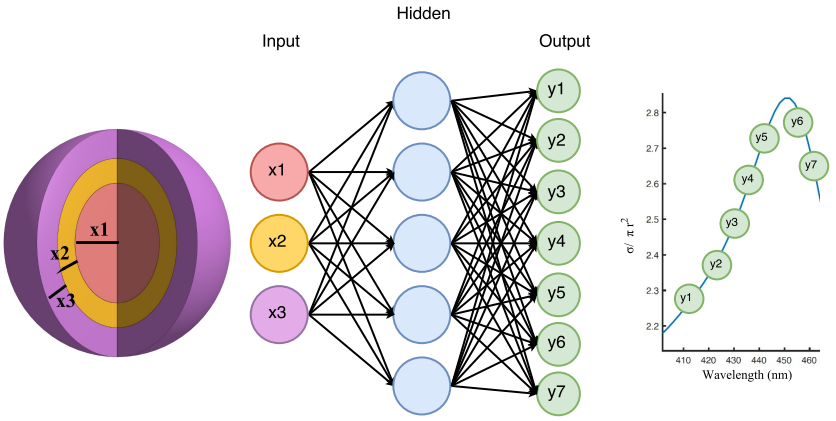

Figure 1: The NN architecture has as its inputs the thickness of each layer of the nanoparticle, and as its output the scattering cross section at different wavelengths of the scattering spectrum. Our actual NN has four hidden layers.

Next, we trained the NN using these examples. We used a fully connected network, with four layers and 250 neurons per layer, giving us 239,500 total parameters. The input was the thickness of each layer of material (with the material held fixed), and the output was the spectrum sampled at points between 400 to 800 nanometers. The network size was increased as the number of layers of material increased, with the maximum size being four hidden layers with 300 neurons each.

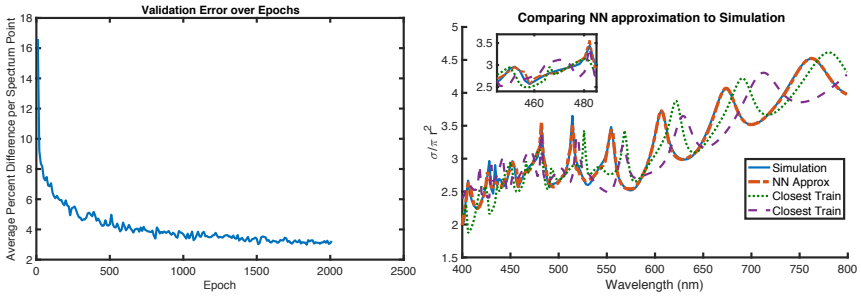

Figure 2: Left - Training loss for the eight layer case. Right - Comparison of NN approximation to the real spectrum, with the closest training examples shown here. The training examples are the most similiar larger and smaller particle respectively.

We trained the network using a batch size of 100, for around 16,000 epochs on most trials. The cost function we use is the mean-square-error between each spectrum point and output neuron. The training error is graphed in Fig. 2. The first application was to test the forward computation of the network to see how well it approximates the spectra it was not trained on — for an example see Fig. 2. Impressively, the network matches the sharp peaks and high Q features with much accuracy, even though the model was only trained with 50,000 examples — which is equivalent to sampling each layer thickness between 30-70 nanometers only four times.

To study if the network learned anything about the system and can produce features it was not trained on, we also graphed the closest examples in the training set it was trained on. The results from Fig. 2 visually demonstrate that the network is not simply interpolating, or averaging together the closest training spectra. This suggests that the NN is not simply fitting to the data, but instead learning some pattern about the input and output data such that it can solve problems it had not encountered, and to some extent generalize the physics of the system.

2 NNs solve Nanophontonic Inverse Design

With the weights fixed, we set the input as a trainable variable and used back-propogation to train the inputs of the NN. In simple terms, we run the NN 'backwards'.

We test this inverse design on the same problem as above - an eight layer nanoparticle made of alternating layers of TiO_2 and silica. We choose an arbitrary spectrum, and have the network learn what inputs would generate a similar spectrum. We can see an example optimization in Fig. 3. In order to ensure that we have a physically realizable spectra, the desired spectrum comes from a random valid nanoparticle configuration.

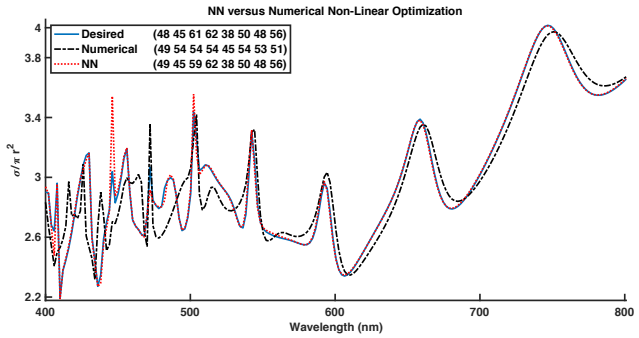

Figure 3: Inverse design for an eight layer nanoparticle. The legend gives the dimensions of the particle, and the blue is the desired spectrum. The NN is seen to solve the inverse design much more accurately.

We also compare our method to state of the art numerical nonlinear optimization methods. We tested several techniques, and found that interior-point methods [1] were most effective for this problem. We then compared these interior-point methods to our results from the NN, shown in Fig. 3. Visually, we can see that the NN is able to find a much closer minimum than the numerical nonlinear optimization method. This result is consistent across many different spectra, as well as for particles with different number of layers and materials.

Further results demonstrated the network was able to behave fine even in regions where ϵ has a strong dependence on ω, such as in J-Aggregates [3], where the spectra are very sharp and complex.

3 NNs can be used to optimize broadband and specific-wavelength scattering

For optimization, we want to be able to give the boundary conditions for a model (for instance how many layers, how thick of a particle, what materials it could be), and find the optimal particle to produce $\sigma(\lambda)$ as close as possible to the desired $\sigma_{desired}(\lambda)$. We consider two optimization problems: maximizing at a single wavelength, and maximizing a broad-spectrum.

To do this, we fix the weights of the NN, and create a cost function that will produce the desired results. We simply compute the average of the $\sigma(\lambda)$ inside of the range of interest, and compute the average of the points outside the range, then minimize this ratio.This cost function J is $J = \frac{\overline{\sigma_{in}}}{\overline{\sigma_{out}}}$.

Ideally, this optimization would be performed using metals and other materials with plasmonic resonances [3] in the desired spectrum range. These materials are well-suited for having sharp, narrow peaks, and as such can generate spectra that are highly efficient at scattering at precisely a single wavelength. Our optimization here uses solely dielectric materials. By using materials that do not have sharp plasmonic resonances, we force the NN to find a total geometry that still scatters at a single peak, despite the underlying materials being unable to. A figure showing the results of this for a narrow set of wavelengths close to 465 nanometers can be seen in Fig. 4.

Next, we consider the case of broadband scattering, where we want a flat spectrum across a wide array of wavelengths. In this case, we choose the same J as above - minimizing the ratio of values inside to outside. After training the network for a short number of iterations, we achieve a geometry that will broad-band scatter across the desired wavelengths. A figure of this can be seen in Fig. 4.

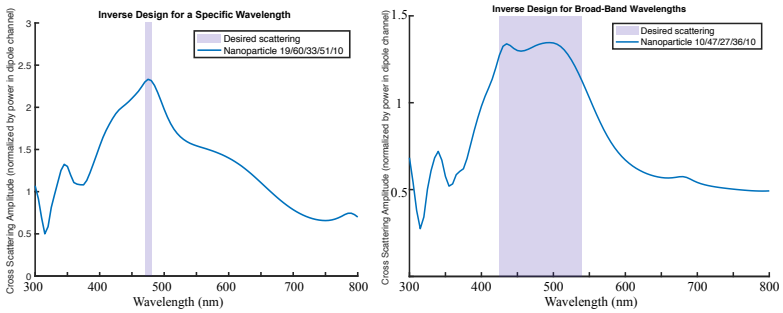

Figure 4: Spectra produced by using our approach as an optimization tool. Left - demonstrates scattering at a narrow range close to a single wavelength. Right - Demonstrates scattering across a broad-band of wavelengths. The legend specifies the thickness of each layer in nm, alternating TiO2 and silica layers.

4 Comparison of NNs with some conventional Inverse Design Algorithms

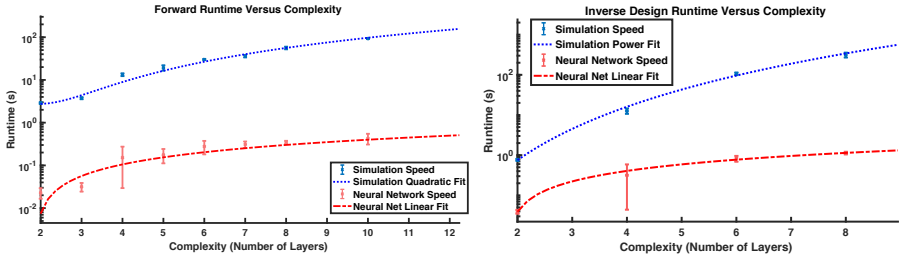

Figure 5: Left - Comparison of forward runtime versus complexity of the nanoparticle. The simulation becomes infeasible to run many times for large particles. The scale is log-log. Right - Comparison of inverse design runtime versus complexity of the nanoparticle. The runtime of the numerical optimization is seen to increase more quickly than that of the NN. The simulation is fit with a 4.5 degree power.

We tested several techniques, and found that interior-point methods [1] were most suited for nanoparticle inverse design. To compare this numerical nonlinear optimization method to our NN, we use the same cost function for both, and code both the NN and simulation in Matlab.

We train a different NN on each number of particle layers from two to ten. The networks' size increased as we increased the number of layers. We tested the approximation speed, after-training, by averaging the runtime for 100 spectra. A plot of these results is shown in Fig. 5. Once fitting, it is evident that for complex problems, the simulation would struggle to run more than a few layers, while the NN would be able to handle more.

Next, we looked at the optimization runtime versus the problem complexity. To find the speed of this optimization, we chose a spectrum and set a threshold cost, and timed how long it took to find a spectrum below this cost or that converged to a local minimum. Results demonstrated that NN inverse design was able to handle more complex problems than the numerical inverse design — see 5.

5 Contributions

The results of this method suggest that it can be easily used and implemented, even for complex inverse design problems. The architecture used in the examples above — a fully connected layer — was chosen without much optimization, and still performs quite well. Our preliminary testing with other architectures (convolutions, dropouts, and residual networks) appeared to have further promise as well.

Perhaps the two most surprising results were how few examples it takes for the network to approximate the simulation, as well as how complex the approximation can really be. For instance, in the eight

layer case the NN only saw 50,000 examples over eight independent inputs. This means that on average it sampled only four times per layer thickness, and yet could reproduce the entire range of 30-70 nanometer layer thickness continuously. The approximation was even able to handle quite sharp features in the spectrum that it otherwise had not seen.

This method could be used in many other fields of computational physics; it would allow us to approximate physics simulations in fractions of the time. Furthermore, owing to the robustness of back-propogation, this method allows us to solve many inverse design problems without having to manually calculate the inverse equations. Instead, we simply have to write a simulation for the forward calculation, and then train the model on it to easily solve the inverse design.

Acknowledgments

This material is based upon work supported in part by the National Science Foundation under Grant No. CCF-1640012, as well as in part supported by the Semiconductor Research Corporation under Grant No. 2016-EP-2693-B. It is also supported in part by the U. S. Army Research Laboratory and the U. S. Army Research Office through the Institute for Soldier Nanotechnologies, under contract number W911NF-13-D-0001, as well as in part by the MRSEC Program of the National Science Foundation under award number DMR - 1419807.

References

[1] Michael J. Todd Arkadi S. Nemirovski. Interior-point methods for optimization. *Acta Numerica*, 2008.

[2] Peter Levay Barnabas Apagyi, Gabor Endredi. *Inverse and Algebraic Quantum Scattering Theory*. Springer-Verlag Berlin, 1996.

[3] C.W.Hsu Z.Zander S.Lacey R.Yagloski A.W.Fountain E.Valdes E.Anquillare M. Soljacic S.G.Johnson B.G.DeLacy, O.D.Miller and J.D.Joannopoulos. Coherent plasmon-exciton coupling in silver platelet-j-aggregate nanocomposites. *Nano Letters*, 15, 2015.

[4] David R. Huffman Craig F. Bohren. *Absorption and Scattering of Light by Small Particles*. Wiley, 1998.

[5] Alexander Y. Piggott, Jesse Lu, Konstantinos G. Lagoudakis, Jan Petykiewicz, Thomas M. Babinec, and Jelena Vuckovi. Inverse design and demonstration of a compact and broadband on-chip wavelength demultiplexer. *Nat Photonics*, 9(6):374–377, June 2015.

[6] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323:533 – 536, 1986.