http://doi.org/10.35784/iapgos.7300 received: 16.12.2025 | revised: 15.10.2025 | accepted: 18.11.2025 | available online: 20.12.2025

DEEP LEARNING-BASED PREDICTION OF STRUCTURAL PARAMETERS IN FDTD-SIMULATED PLASMONIC NANOSTRUCTURES

Shahed Jahidul Haque[1], Arman Mohammad Nakib[2]
[1]Nanjing University of Information Science & Technology, Department of Information and Communication Engineering, Nanjing, China,
[2]Nanjing University of Information Science & Technology, Department of Artificial Intelligence, Nanjing, China

Abstract. *The research creates a new approach to estimate essential dimensions of plasmonic nanoparticles that use the Finite-Difference Time-Domain (FDTD) simulation program. The research team uses EfficientNetB0 alongside ResNet50 and VGG16 deep learning models to obtain quick and exact simulations parameter predictions from simulation image data. The developed dataset consists of dielectric and magnetic field images that stem from FDTD simulated fields through representative materials MgF_2, Au, and glass. The preparation process for the dataset includes a systematic variation of 38 structural parameters for achieving sufficient coverage of potential configurations. VGG16 proved to be the most effective model from the testing group because it attained a training loss 0.1592, validation loss of 0.1607, and test loss 0.1625. The outstanding result shows deep learning techniques can be effectively used to boost nanophotonic device design speeds and optimization processes. The methodology developed in this work has the potential to reduce substantially the computational expenses together with simulation duration for nanostructure engineering processes.*

Keywords: FDTD plasmonic nanostructures, structural parameter & image-based prediction, deep learning

PROGNOZOWANIE PARAMETRÓW STRUKTURALNYCH W NANOSTRUKTURACH PLAZMONICZNYCH SYMULOWANYCH METODĄ FDTD Z WYKORZYSTANIEM UCZENIA GŁĘBOKIEGO

Streszczenie. *W niniejszym artykule opracowano nowe podejście do szacowania kluczowych wymiarów nanocząstek plazmonicznych z wykorzystaniem programu symulacyjnego opartego na metodzie różnic skończonych w dziedzinie czasu (FDTD). Zespół badawczy zastosował modele głębokiego uczenia, takie jak EfficientNetB0, ResNet50 oraz VGG16, w celu szybkiego i precyzyjnego przewidywania parametrów symulacji na podstawie obrazów uzyskanych z symulacji. Stworzony zbiór danych obejmuje obrazy pola dielektrycznego i magnetycznego uzyskane z symulacji FDTD dla reprezentatywnych materiałów: MgF_2, złota (Au) i szkła. Proces przygotowania danych uwzględniał systematyczną zmianę 38 parametrów strukturalnych w celu uzyskania odpowiedniego pokrycia możliwych konfiguracji. Spośród testowanych modeli, VGG16 okazał się najskuteczniejszy, osiągając błąd walidacyjny równy 0,1607. Uzyskane wyniki dowodzą, że techniki głębokiego uczenia mogą skutecznie przyspieszyć projektowanie oraz optymalizacji urządzeń nanofotonicznych. Opracowana metodologia ma potencjał znacznego ograniczenia kosztów obliczeniowych oraz czasu trwania symulacji w procesach inżynierii nanostruktur.*

Słowa kluczowe: plazmoniczne nanostruktury FDTD, prognozowanie parametrów strukturalnych na podstawie obrazu, uczenie głębokie

Introduction

Both nanophotonics and plasmonics have developed significantly, due to the unique ability of plasmonic structures working on optical light at the subwavelength scale. Due to the effect of light on the free electrons present on the surface of metallic nanostructures, these structures have opened many applications in high-resolution imaging, biological and chemical detection, data storage, and even the fabrication of miniaturized optical devices. An important aspect of their operation is the ability to modulate light on scales less than the wavelength which is desirable for accurate measurements of molecular or environmental changes necessary in health monitoring and environmental analysis.

The design and optimization of such nanophotonic devices involve complicated simulation schemes wherein the initial simulation was based on an FDTD approach. This numerical method is used to solve Maxwell's equations, modeling the behavior of electromagnetic waves traveling through challenging materials and interfering with complex nanostructures. However, as FDTD simulations consume a lot of computational resources especially when simulating structures with large geometries and a broad range of wavelengths, any new change in the parameters of a device e.g. its material composition, geometry or individual layer thickness requires a new simulation to be run. In turn, this make the design of optimized structures resource and time-intensive, not least when trying to optimize multiple variables.

To overcome these challenges, scholars have started investigating the application of machine learning (ML), and in particular deep learning (DL) as a framework for predictive modeling in scientific computing. The predictivity and capability of learning non-linear functions from large computational data make DL a candidate for bypassing repetitive FDTD simulations. In nanophotonic and materials science, DNN has recently been used to predict results by using structural parameters or material arrangements, saving much time as a design tool. For instance, deep learning has been used to predict electromagnetic responses or structural properties based on large sets of FDTD simulation data that take less time than full-scale simulation data.

The present work follows on from this line of research by utilizing DL to estimate the extended structural profile of a particular layered plasmonic nanostructure, made of MgF_2, gold, and glass. These materials were carefully selected based on their good optical and mechanical characteristics where MgF_2 is transparent in the infrared range, gold provides the possibility of plasmonic resonance, and glass ensures the stability of the overall device. For this purpose, a new data set has been produced using the FDTD simulation which exhibits electromagnetic field distributions for several structural conditions. As each material layer spans 500 nm and the variation of 38 structural parameters up to 2000 nm is accounted for, this dataset is a broad set of nominals that may be compared to highly time-consuming computations when simulated recurrently.

The purpose of this work was to build an effective predictor function using the DL that would allow calculation of these 38 structural parameters from the image data obtained with FDTD simulations directly. In the present work, different CNN models were used and these models comprised efficient NetB0, ResNet50, and VGG16, each of which has its advantages in the area of model complexity and feature extraction ability. It is therefore the intention of this study to train these models on the generated image data in as much as to derive an accurate yet computationally efficient mean for parameter prediction. The accomplishment of this goal is useful not only to resolve the issues related to high computational costs required for FDTD simulations but also to further develop the nanophotonic design domain by creating a more effective predictive model. By way of these improvements, the study established a significant platform for enhanced routes to design nanostructures at a faster and more affordable process that will benefit various applications that need precisely engineered plasmonic devices.

1. Literature review

Precedent literature research highlights colossal advancements within the application of deep learning (DL) and machine learning (ML) in increasing and accelerating the computationally challenging Finite-Difference Time-Domain (FDTD) simulations required in the design of plasmonic and nanophotonic structures. For example, Mahadi et al. (2024) analyzed the prospects of using Gated Recurrent Units (GRUs) to predict absorption spectra in plasmonic devices and found that while providing an accurate spectrum estimation, it is orders of magnitude more efficient than the standard fully discrete time-domain Finite-Difference Time-Domain (FDTD) approaches [9]. This approach is especially useful for applications where timely spectrum estimation is most crucial. He and Ye (2019) used ML to improve FDTD tasks by modeling electric field distributions in plasmonic nanoparticles to help SNIP inverse design more accurately and effectively, necessary for the development of tailor-made nanostructures.

Malkiel et al. (2018) proposed the use of an artificial DNN-based approach for characterizing nanostructures which essentially substituted conventional FDTD simulations with DL thus bringing down the computational burdens greatly. This innovation enabled more frequent changes of design for an important application in sensing and imaging [10]. Likewise, Adibnia et al. (2024) were concerned with optical behavior predictions using DL in plasmonic switches, demonstrating that DL can save electricity for conventional iterative FDTD simulations in terms of modeling switching behaviors [2]. In another novel application, Baxter et al. (2019) used DNNs in predicting plasmonic colors which is an efficient method of modeling FDTD output for display devices where accurate colors are needed [3].

Persson (2024) expanded this field further by employing CNNs to predict anisotropic nanostructure design and minimizing the simulation of specific geometries as well as enhancing FDTD results [14]. Masson et al. (2023) also employed DL in nanoplasmonics and explained how DL could successfully predict the material property to execute the role of enhancing the design efficiency in nanophotonic applications [12].

Likewise, in Du et al. (2021), DL was applied to estimate the geometric parameters of nanoparticles to similarly high accuracy, in contrast to the computationally complex FDTD requirement, which suggests the scalability of DL in nanoparticle design [4]. Verma (2023) implemented ANNs in FDTD to enhance photonic properties in metallic nanoparticles with greater computational parity by optimizing its parameters and computational paradigms [16].

This study also extends previous findings by Li et al. (2020) regarding DL's efficiency and accuracy of scattering behavior and real-time optical response predictions, respectively, making it a favorable model for any system in need of quick optical feedback [8]. In developing the core-shell nanoparticle property, Vahidzadeh & Shankar (2023) were able to invoke ML to predict properties to minimize overdependence on FDTD simulations for efficient design for complex core-shell structures [15].

For renewable energy applications, Manzhos et al. (2021) used DL to predict plasmonic behavior in solar cell structures by improving FDTD simulation for better light absorption in nanostructures [11]. Zhang et al. (2020) applied the evolutionary algorithms with ML to simulate the graphene metamaterials obtained accurate spectrum estimations and reinforced DL as the promising substitute of FDTD for the design of new advanced metamaterials [18]. Kazemzadeh (2022) employs the DL approach for optimizing the nanoplasmonic sensors for biomedical applications and it was shown how DL algorithms turned the fabricated design by FDTD for healthcare purposes [7].

Adibnia & Mansouri-Birjandi (2024) used DL for spectral prediction in the nonlinear plasmonic ring resonator switches and proved the efficiency of the DL approach for modeling the optical responses in such resonators [1]. This spectrum of research illustrates the gradual progression of adopting both DL and ML into nanophotonics and plasmonics, and where these techniques largely substitute or enhance FDTD simulations, as more efficient, accurate, and economical solutions to complex nanostructure developments for multifaceted applications in the fields of display technology, sensing, energy, and biomedical engineering. Other papers worked with Plasmonic nanoparticle simulations and inverse design using machine learning and also used Deep Learning to simulate the parameters [5, 6, 17].

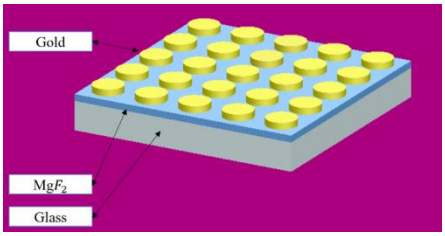
2. Methodology

2.1. Overview

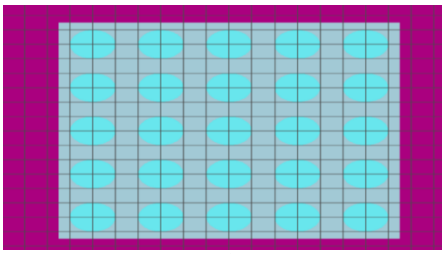
The methods section in this study explains how the authors designed and modeled plasmonic nanostructures and predicted their parameters. Based on the formulation, we created a set of images and their corresponding structural parameters and explicitly trained deep learning models using the Finite-Difference Time Domain (FDTD) method. This section discusses the FDTD simulation employed, the dataset generation, the deep learning model, and its training.

2.2. FDTD simulation setup

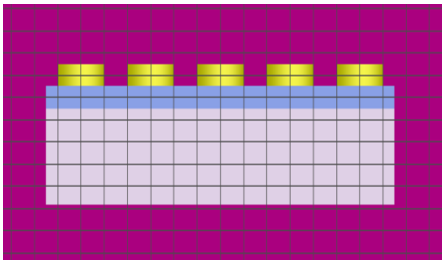
Fig. 1a illustrates the material structure of the whole design in the FDTD software. Figs. 1b, 1c, and 1d show the different dimensions of the structure, such as the XY view, YZ view, and XZ view.



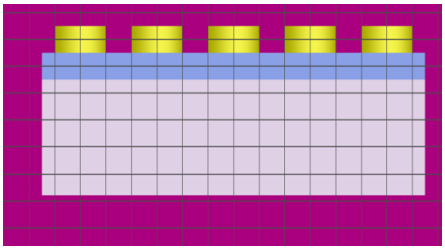
(a) material structure



(b) XY view of the structure



(c) YZ view of the structure



(d) XZ view of the structure

Fig. 1. Structure of the design (a–d)

The model for observing the optical properties of the layered plasmonic structure was simulated in the FDTD software using MgF_2, gold, and glass substrate. It needed to structure be designed to absorb infrared radiation, which is a key component of such uses as sensing. Ranges that were allowed to differ from one simulation to another include the thickness of layers, pattern density, and some dimensions of the material. Changes in any of these parameters produced a different image, that featured electromagnetic field distribution patterns.

2.3. Material properties

MgF_2 (Magnesium Fluoride): Selected for deposition as a dielectric layer because it is transparent in the infrared range.

Gold (Au): Applied for its plasmonic properties, it increases the field interaction at certain wavelengths.

Glass: Served as support as a base layer to the walls of the building.

2.4. Parameter configuration

In each FDTD simulation, 38 parameters, including layer thickness and feature spacing, were varied to produce 30 sets of images representing structural geometries.

2.5. Dataset creation

In all, 200 images were gathered for study with a host arrangement put in 10 groups according to structural and field differences. For each image, the segmentation produced 38 parameter values and these skills were logged in a CSV file for a supervised learning algorithm. The particular factors included Image Path, Circle_x_min, Circle_x_max, Circle_radius, Boundary_layer_min, Boundary_layer_mix, Source_x_min, Source_x_max, Source_y_min, Source_y_max, Monitor_x_min, Monitor_x_max, Monitor_y_min, Monitor_y_max, Mesh_x_min, Mesh_x_max, Mesh_y_min, Mesh_y_max, Gold_x_min, Gold_x_max, Gold_y_min, Gold_y_max, Gold_z_min, Gold_z_max, MgF2_x_min, MgF2_x_max, MgF2_y_min, MgF2_y_max, MgF2_z_min, MgF2_z_max, Source_z_min, Source_z_max, Monitor_z_min, Monitor_z_max, Mesh_z_min, Mesh_z_max, Monitor_point, Mesh_step, collectively offering information on the specific aspect of the structure to control the optical characteristics. Fig. 2 shows the dataset image samples where the categories are electric field (e), electric field at a specific region (e1), monitored magnetic field intensity (h1), energy flow or power density (p), integrated power through a surface (p1), transmitted power ratio (t), transmission through a particular structure (t1), combined electromagnetic field (e+h), frequency or wavelength-domain result (spectrum), time domain (time).

2.6. Deep learning models

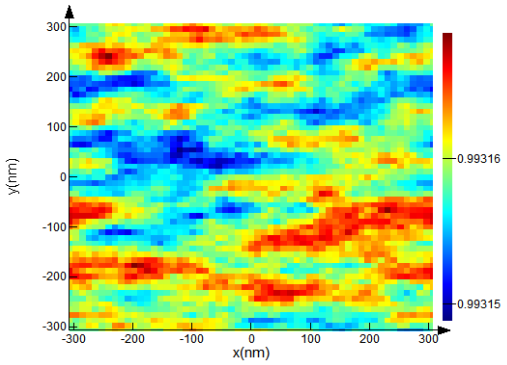
Later, three different pre-trained CNNs, namely EfficientNetB0, ResNet50, and VGG16 were employed for parameter prediction. There are many models and based on the objective of the study these models were chosen because they are capable of feature extraction and may have different levels of complexity d to test them for application in high dimensional regression.

EfficientNetB0: This efficient model applies compound scaling to adjust network depth, width, and resolutions.

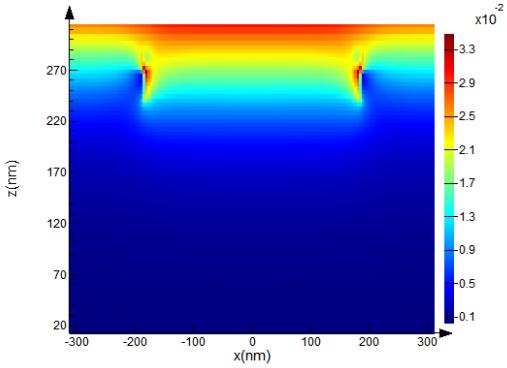
ResNet50: Resort to residual blocks that enable the network to be out of layers that do not contribute to enhancing feature extraction.

VGG16: A deeper neural model, all layers are of equal depth and are good at capturing spatial features.

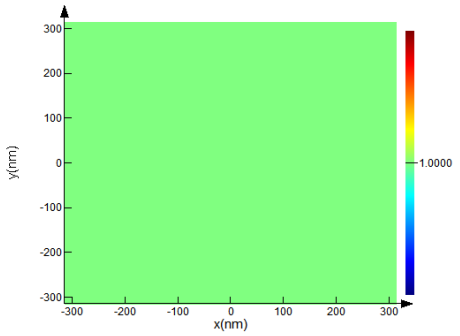
For each model, fully connected (dense) layers were included for regression at the output layer, which consisted of 38 nodes as the parameters.



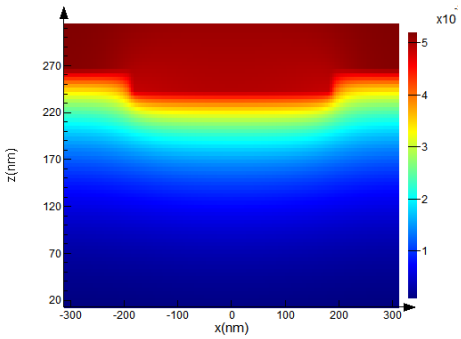
a) Electric field, e



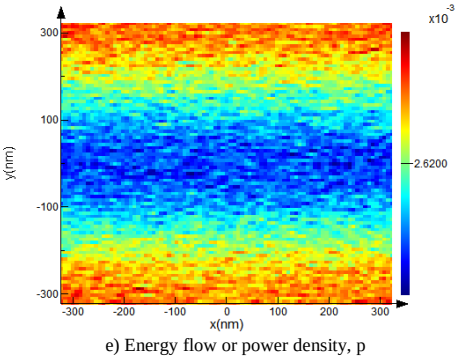
b) Electric field at a specific region e1



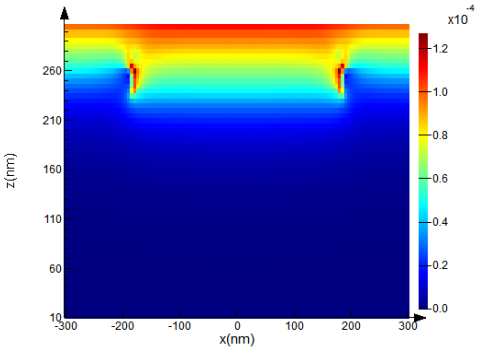
c) Combined electromagnetic field, e+h



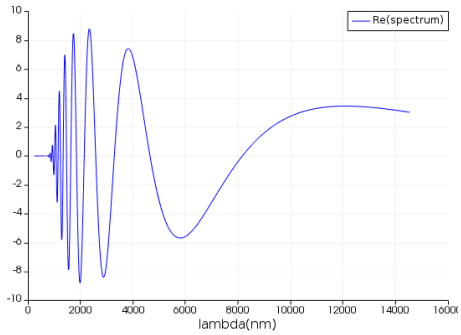
d) Monitored magnetic field intensity, h1



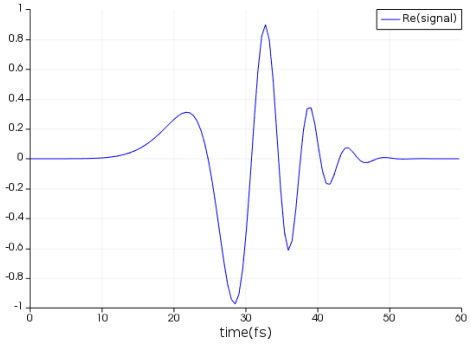
e) Energy flow or power density, p



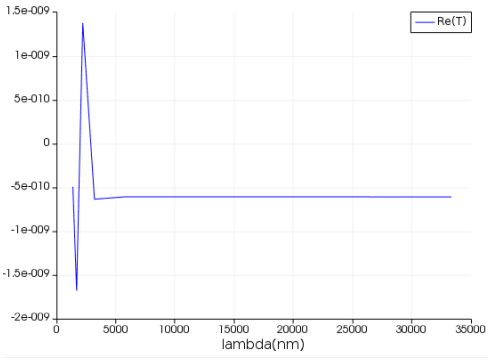
f) Integrated power through a surface, p1



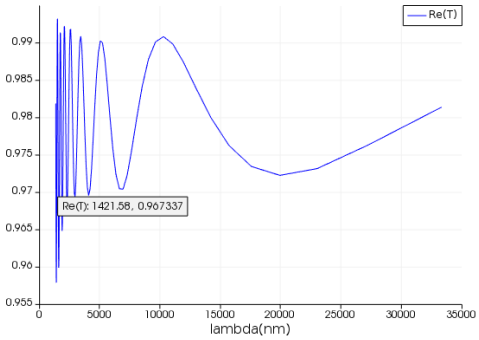
g) Frequency or wavelength-domain result, spectrum



h) Time domain



i) Transmission through a particular structure, t1



j) Transmitted power ratio, t

Fig. 2. Dataset image samples (a-j)

2.7. Model architecture: VGG16

The VGG16 is a CNN with simple and deep architecture because it is made up of many convolutions followed by fully connected layers. Being a model developed for classification tasks at first, VGG16 demonstrates great results in image processing because of its depth and uniformity. For this research, we modified the VGG16 architecture so that it can operate in a regression model since the aim was to make predictions of 38 structural parameters that are continuous, from FDTD simulation images. The initial complete connection layer for categorization was substituted by regression layers, allowing it to be used for parameter estimation.

Convolutional layers

They used twelve convolutional layers to extract features from images in VGG16 ranging from the basic level, and edges to higher level texture and shapes. The convolution operation in each layer is defined by:

$$Z_{i,j,k} = \sum_{m,n} X_{i+m,\ j+n} \cdot W_{m,n,k} + b_k \quad (1)$$

where:

$Z_{i,j,k}$ is the activation map output for the k-th filter at spatial position (i,j),

$X_{i+m,\ j+n}$ represents the input from the previous layer,

$W_{m,n,k}$ denotes the weights of the convolution filter, b_k is the bias term for the filter.

ReLU activation

Each convolutional layer's output undergoes a ReLU activation function, defined as:

$$f(x) = \max(0, x) \quad (2)$$

This introduces non-linearity, enabling the network to learn complex patterns and relationships in the data.

Pooling layers

Max pooling layers reduce spatial dimensionality by downsampling, which lowers the computational load and provides translation invariance. Pooling is defined as:

$$P_{i,j,k} = \max(X_{a,b,k}) \quad (3)$$

where $P_{i,j,k}$ is the pooled output for the k-th channel over the region (a,b).

Fully connected layers for regression

Specifically, the network features are extracted at the convolutional layers, but for regression, FC layers are used. Here we removed the classification layers and used a series of Dense layers that followed a last linear layer containing 38 neurons for each parameter.

Output layer

In this layer, the 38 nodes are linearly activated to predict continuous parameter values; therefore, this layer is a multi-output regression layer.

Loss function and optimization

For the regression task, the **Mean Squared Error (MSE)** was used as the loss function:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y_i})^2 \quad (4)$$

where y_i is the true parameter value and $\hat{y_i}$ is the predicted parameter value for each data sample n.

L2 Regularization

To reduce overfitting, L2 regularization was added, penalizing larger weights:

$$\text{Regularized Loss} = MSE + \lambda \sum_{j=1}^{p} w_j^2 \quad (5)$$

where λ controls regularization strength, and w_j are the weights.

2.8. Model architecture: EfficientNetB0

Codified by key layers, EfficientNetB0 also employed a compound scaling method for the width, depth, and resolution of inputs it used. This balance helped EfficientNetB0 deliver high performance with relatively fewer parameters, which makes it more computationally efficient, especially for datasets with fewer samples. In this work, EfficientNetB0 was fine-tuned for regression because the model was intended to predict 38 continuous features that describe plasmonic structure attributes FDTD.

EfficientNetB0 scales the width, depth, and resolution of the network using a compound coefficient, represented mathematically as:

$$\text{Width} = \alpha\cdot\phi,\ \text{Depth} = \beta\cdot\phi,\ \text{Resolution} = \gamma\cdot\phi \quad (6)$$

where: α, β, γ are constants determining scaling ratios for each dimension, ϕ is the scaling coefficient.

EfficientNetB0 uses $\alpha = 1.2$, $\beta = 1.1$, and $\gamma = 1.15$ to balance performance and efficiency, making it ideal for high-dimensional regression tasks.

Convolutional layers with swish activation

EfficientNetB0 employs Swish activation, defined as:

$$f(x) = x\cdot\sigma(x) \quad (7)$$

where $\sigma(x)$ is the sigmoid function. Swish helps smooth gradient flows, which is especially useful for deep architectures.

Squeeze-and-Excitation (SE) block

Each block in EfficientNetB0 contains an SE module that dynamically scales feature maps by their channel-wise importance. The SE block can be defined as:

$$s = \sigma(W2\cdot\text{ReLU}(W1\cdot z)) \quad (8)$$

where z is the global average-pooled feature map, W_1 and W_2 are trainable weight matrices, and s represents the scaled feature map.

Fully connected (dense) layers for regression

A new classification layer was proposed and thrown into fully connected layers which terminate in a linear layer of 38 nodes, each representing one parameter. The regression layers were incorporated with dropout regularization to reduce cases of overfitting.

Loss function and optimization

EfficientNetB0 was trained to minimize Mean Squared Error (MSE) with L2 regularization:

$$\text{Loss} = \text{MSE}+\lambda\sum_{=1}^{p} w_j^{\,2} \quad (9)$$

The Adam optimizer was used as the first choice with decay when the validation loss did not change for a few iterations.

2.9. Model architecture: ResNet50

ResNet50 is one of the widely used ResNet models, a convolution neural network based on a residual learning formula to solve the vanishing gradient problem in deeper learning architectures. ResNet50 incorporates residual blocks to help increase the depth and improve training for deeper networks by skipping layers that do not support the direction, a benefit that increases the model's learning capacity for intricate data sets. In this work, ResNet50 was used for regression on 38 continuous structural parameters obtained from FDTD-simulated images.

The ResNet50 architecture uses Shortcut connections that enable gradients to flow directly through the network and skip the convolutional layers, stabilizing training deep networks. Each residual block in ResNet50 can be expressed as:

$$y = F(x,\{W_i\})+x \quad (10)$$

where: x is the input feature map, $F(x,\{W_i\})$ represents the residual mapping learned by stacked layers with weights $\{W_i\}$,and y is the output feature map after the shortcut connection.

The residual connection also precisely maps to identity, so modifications can be learned instead of features from scratch. This makes training deep networks possible, which also enhances accuracy.

Each residual block in ResNet50 includes two or three convolutional layers followed by batch normalization and ReLU activation, defined as follows:

Convolution operation

$$Z_{i,j,k} = \sum_{m,n} X_{i+m,\ j+n} \cdot W_{m,n,k} + b_k \quad (11)$$

where $Z_{i,j,k}$ is the output of the convolution, $X_{i+m,\ j+n}$ is the input, and $W_{m,n,k}$, and b_k represent weights and bias.

Batch normalization

Batch normalization standardizes activations within each batch, improving convergence and reducing internal covariate shift. For activations x, it is defined as:

$$\hat{x} = \frac{x-\mu}{\sqrt{\sigma^2+\epsilon}} \quad (12)$$

where μ and σ^2 are batch statistics and ϵ is a small constant for numerical stability.

ReLU activation

The ReLU activation introduces non-linearity:

$$f(x) = \max(0, x) \quad (13)$$

Fully connected layers for regression

For regression tasks, we removed the classification layer of ResNet50 and added a series of fully connected (dense) layers that output 38 linear units to estimate each of the structural parameters we desired.

Loss function and optimization

We employed Mean Squared Error (MSE) as the loss function with L2 regularization to mitigate overfitting:

$$\text{Loss} = \text{MSE}+\lambda\sum_{j=1}^{p} w_j^{\,2} \quad (14)$$

where λ is the regularization parameter. As the optimizer's choice, Adam was applied using the learning rate decay strategy, in which the learning rate was to be reduced in case of validation loss stagnation for a definite number of iterations.

3. Experiments and results

The dataset is separated into two parts, 80% of which is the training set to use data augmentation to train the model and 20% is the validation set to keep track of performance and adjust it throughout the training. The dataset split does not include a single test set isolated to test the model, but rather attempts to test the model by uploading external images to find out which way the model works with completely unseen data. In this manner, model development is taken care of by training and validation, and external image uploads act as the testing stage.

3.1. Training and evaluation of VGG16 model

Data preparation

Dataset: The dataset consists of 200 images of FDTD-simulated plasmonic structures, divided into 10 categories,and 38 parameters per image overall.

Preprocessing: Images were also altered by resizing them to 224 × 224 pixels and then normalized.

Training configuration

Optimizer: Although the optimizer's default state was set to the Adam algorithm with a learning rate of 10^{-5}, learning rate decay was implemented if the validation loss was not reduced.

Batch Size: 16.

Epochs: 70 (depending on which value the validation loss does not continue to decrease early stopping is performed).

Table 1. Losses of VGG16 model

Metric	Value
Training Loss	0.1592
Validation Loss	0.1607
Test Loss	0.1625

The test MSE of 0.1625 indicates that VGG16 effectively generalized to new data, accurately predicting the 38 parameters with low error. Fig. 3 illustrates the VGG16 model's training, validation loss and Table 1 shows the values.

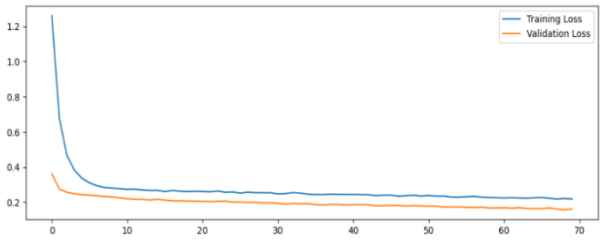


Fig. 3. Training and validation loss graphs of the VGG16 model

Discussion on VGG16 model performance

VGG16 proved to be more effective than other architectures because it could capture fine-grained spatial features of FDTD images. Due to its feature-extraction ability provided by deep convolutional layers, together with the regularization adopted, it was able to learn the parameter-image mapping in this high-dimensional space.

3.2. Training and evaluation of EfficientNetB0

Data preparation and model configuration

Dataset: These 200 FDTD-generated images were normalized and resized to improve the visualization of features upon superimposition over the original RGB images.
Batch Size: 16
Epochs: 70, with early stopping.
Optimizer: Adam with 10^{-5} learning rate, ReduceLROnPlateau was used here to reduce the learning rate when necessary.

Table 2. Losses of EfficientNetB0 Model

Metric	Value
Training Loss	0.2895
Validation Loss	0.291
Test Loss	0.2888

Results

EfficientNetB0 showed a good ability to predict with a slightly higher test loss than VGG16 suggesting they possibly overfit even though it has very few parameters. Fig. 4 shows the training, validation loss of the EfficientNetB0 Model and table 2 shows the values.

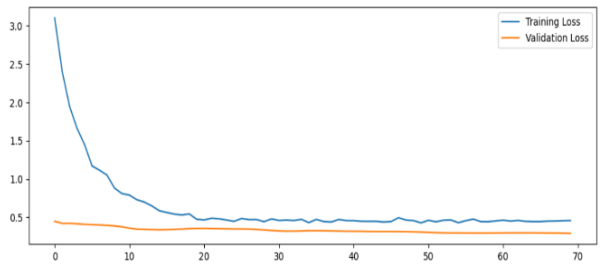


Fig. 4. Training and validation loss graphs of the EfficientNetB0 model

Analysis

The compound scaling of EfficientNetB0 makes it computationally efficient, though it lacks the complexity that might be essential to capture multiscalar spatial patterns in FDTD data.

3.3. Training and evaluation of ResNet50

Data preparation and model configuration

Dataset**:** 200 FDTD-simulated images processed for model input.
Batch Size**:** 16
Epochs**:** 70, with early stopping and learning rate decay.
Optimizer**:** Adam with an initial learning rate of 10^{-5}.

Table 3. Losses of ResNet50 model

Metric	Value
Training Loss	0.3258
Validation Loss	0.326
Test Loss	0.3239

Results

Although ResNet50 has provided reasonable performance, loss values were a bit higher than VGG16, which indicates that ResNet architecture may need more regulatory measures or may require better generalization of data augmentations. Fig. 5 shows the training, validation loss of the ResNet50 Model and table 3 shows the values.

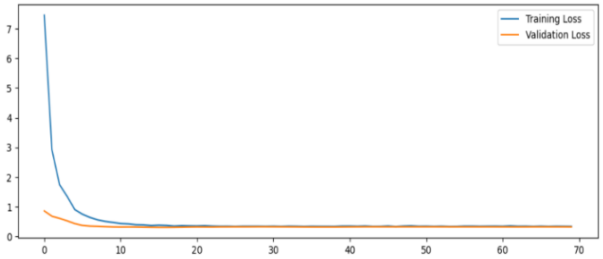


Fig. 5. Training and Validation Loss Graphs of the ResNet50 Model

3.4. Combined analysis and discussion

The connectivity pattern in ResNet50 provided an opportunity to learn deeper hierarchies and feature mapping in the images provided by FDTD. However, even though it has achieved a higher loss at the end of epochs, it shows that ResNet50's depth can result in the overfitting of a comparatively small dataset. The next improvements could focus on different forms of training, or apply ensemble methods to enhance future generalizations.

Evaluation metrics

Table 4 shows the comparison of three models.

Table 4. Comparison of all Models

Model	Validation Loss	Test Loss
EfficientNetB0	0.291	0.2888
ResNet50	0.326	0.3239
VGG16	0.1607	0.1625

Analysis

VGG16 performed better than other kinds of networks because of its potential to address complex spatial information handling. Although EfficientNetB0 is an efficient network, fluctuations in the model parameters could not be captured appropriately, or this might be because of the restricted model size.

In order to further illustrate the performance of the models we were used to test sample image which was not visible and was compared with the predicted values against the true values. The Table 5 summarizes the predictions of VGG16, ResNet50 and EfficientNetB0. VGG16 also gave the nearest predictions to the real values in the majority of parameters when compared to the other two models. This proves that VGG16 behaves more predictively on unseen data in our experiments. On the whole, the findings indicate the feasibility of practicality of the proposed models, with VGG16 being the best.

Table 5. Test and predicted values of the test image

Parameter	True Value	VGG16 Predicted	ResNet50 Predicted	EfficientNetB0 Predicted
Circle_x_min	185	181.92337	176.20316	215.00941
Circle_x_max	245	241.0471	214.78952	240.07141
Circle_radius	79	107.27265	78.712776	86.88177
Boundary_layer_min	38	12.94438	9.175221	-4.9068174
Boundary_layer_max	64	64.11309	63.69646	64.03258
Source_x_min	-570	-660.99316	-689.36707	-654.25397
Source_x_max	570	592.575	558.9657	555.6847
Source_y_min	-570	-636.73254	-692.3318	-692.5504
Source_y_max	570	617.10034	562.10944	572.1373
Monitor_x_min	-280	-320.48346	-350.44702	-309.69818
Monitor_x_max	280	285.40384	262.076	294.51648
Monitor_y_min	-280	-325.1408	-350.7416	-336.7725
Monitor_y_max	280	279.25845	290.50894	265.79102
Mesh_x_min	-280	-330.40387	-345.17172	-332.93652
Mesh_x_max	280	297.67297	284.0118	288.7856
Mesh_y_min	-280	-322.83112	-350.85315	-374.89114
Mesh_y_max	280	276.65918	286.72977	277.90216
Gold_x_min	-280	-326.2986	-351.46222	-355.43503
Gold_x_max	280	273.81323	280.50262	275.33655
Gold_y_min	-280	-338.2107	-340.96518	-350.45605
Gold_y_max	280	287.5291	277.2876	292.74664
Gold_z_min	26.5	2.7478118	3.7242126	-8.412284
Gold_z_max	239.5	214.8449	217.60916	222.16992
MgF2_x_min	-280	-332.02725	-340.0608	-343.87402
MgF2_x_max	280	298.601	285.16437	310.82095
MgF2_y_min	-280	-328.6493	-334.82016	-309.2372
MgF2_y_max	280	288.63135	277.42896	283.77515
MgF2_z_min	177.5	182.1871	157.8646	221.78505
MgF2_z_max	266.5	112.438385	62.300987	32.23016
Source_z_min	1210	1222.7931	1207.1602	1229.1868
Source_z_max	1210	1216.8757	1188.4434	1192.3567
Monitor_z_min	1410	1428.2998	1402.2832	1417.1187
Monitor_z_max	1410	1423.0444	1394.084	1391.2072
Mesh_z_min	141	138.73506	135.62877	113.84221
Mesh_z_max	271	263.58746	250.25621	249.49942
Monitor_point	98	62.475952	55.606926	65.464424
Mesh_step	17	4.9715495	1.6483359	4.6041627

3.5. Reason to use loss function instead of accuracy

In these code implementations, the models (EfficientNetB0, ResNet50, and VGG16) are assessed based on the Mean Squared Error (MSE) loss since these models are developed for regression tasks not classification. Here's why:

Nature of prediction

These models estimate structural quantities – thickness, and distances between layers in nanostructures – that are numeric, not nominal values. In regression tasks, getting an impression of how close actual values are to the predicted values (using MSE or similar measures) is more informative than accuracy, which applies to classification.

Loss function for regression

Regression (loss functions such as MSE) measures the precision of the models in terms of the error in the continuous value predictions, and the loss functions penalize the error through a mathematical function. For instance, if the model estimates a structural parameter to be somewhat off, then MSE will measure the extent of this error with some precision, but accuracy would not be perceptive of this fineness.

Minimizing error, not maximizing correct predictions

The models are trained to minimize error instead of aiming for the highest possible number of correct classifications as this is not applicable where there are continuous outcomes.

By doing so, these models can solely center on the MSE to reduce the error in predicting the continuous variables and estimate how reasonably the models approximate the true parameter.

3.6. Novelty of the work

In this paper, a strategy that incorporates FDTD simulation data and deep learning architectures is to estimate a diverse range of structural parameters in plasmonic nanostructures. In contrast to the previous studies which dealt with the prediction of at most 6 optical properties and/or certain structural dimensions, the present study enriches the model predictive functionality by training EfficientNetB0, ResNet50, and VGG16 models on a dataset of the FDTD-derived images. The novelty of this work is based on the low prediction error of 38 parameters simultaneously and proving that VGG16 performs unexpectedly well with negligible validation loss, thus allowing the least computational time as compared to the conventional iterative FDTD technique.

4. Conclusion

In this study, a deep learning model, which applies VGG16 model, was used to learn 38 structural parameters of FDTD-simulated images with a high accuracy (training loss 0.1592, validation loss 0.1607, and test loss 0.1625). The findings validate the strength of VGG16 in revealing the spatial complexity of electromagnetic field distributions, which proves its use in fine-grained structural tasking of biosensing, imaging, and storage of high-dense data. The proposed approach significantly lowers the computational cost as compared with the traditional design iteration based on FDTD, and it also speeds up the design optimization. Moreover, this paper demonstrates the opportunity of applying deep learning to nanophotonic design processes and indicates the possibility of expanding the framework to other computationally expensive models, including finite-element and finite-volume models. On the whole, the results will act as a foundation to hasten the simulated design in nanophotonics and other scientific and engineering-based fields.

5. Future work

Future work should investigate additional means of enriching the datasets, to include materials with different compositions, and extend the analyzed parameters from the current 38. Moreover, if the given model architectures are not the best or if one wants to improve the predictive accuracy, there is space to do so. Studying other deep learning models, such as transformers and other progressive CNN structures, can also be used to conclude handling larger datasets or complex distribution of fields. The authors also recommend extending the use of this predictive framework to other computational domains of finite-element or finite-volume kind to extend its applicability to the variety of electromagnetic simulation platforms.

References

[1] Adibnia E. et al.: A deep learning method for empirical spectral prediction and inverse design of all-optical nonlinear plasmonic ring resonator switches. Scientific Reports 14(1), 2024, 5787.

[2] Adibnia E., Ghadrdan M., Mansouri-Birjandi M. A.: Nanophotonic structure inverse design for switching application using deep learning. Scientific Reports 14(1), 2024, 21094.

[3] Baxter J. et al.: Plasmonic colours predicted by deep learning. Scientific reports 9(1), 2019, 8074.

[4] Du Q., Zhang Q., Liu G.: Deep learning: an efficient method for plasmonic design of geometric nanoparticles. Nanotechnology 32(50), 2021, 505607.

[5] He J. et al.: Plasmonic nanoparticle simulations and inverse design using machine learning. Nanoscale 11(37), 2019, 17444–17459.

[6] Jahan T. et al.: Deep learning-driven forward and inverse design of nanophotonic nanohole arrays: streamlining design for tailored optical functionalities and enhancing accessibility. Nanoscale 16(35), 2024, 16641–16651.

[7] Kazemzadeh M.: Deep Learning and Optimised Nanoplasmonic Sensors for Label-free Biomedical Applications (Doctoral dissertation, ResearchSpace@ Auckland). 2022.

[8] Li Y. et al.: Predicting scattering from complex nano-structures via deep learning. IEEE Access 8, 2020, 139983–139993.

[9] Mahadi M. K. et al.: Gated recurrent unit (GRU)-based deep learning method for spectrum estimation and inverse modeling in plasmonic devices. Applied Physics A 130(11), 2024, 784.

[10] Malkiel I. et al.: Plasmonic nanostructure design and characterization via deep learning. Light: Science & Applications 7(1), 2018, 60.

[11] Manzhos S. et al.: Modeling of plasmonic properties of nanostructures for next generation solar cells and beyond. Advances in Physics: X 6(1), 2021, 1908848.

[12] Masson J. F., Biggins J.S., Ringe E.: Machine learning for nanoplasmonics. Nature Nanotechnology 18(2), 2023, 111–123.

[13] Nakib A. M. et al.: Advanced Simulation Datasets for Deep Learning-Based Photonic and Electromagnetic Research using FDTD Methods. International Journal of Engineering and Advanced Technology Studies 12(4), 2024, 1–16.

[14] Persson P.: Inverse Design of Anisotropic Nanostructures using modern Deep Learning methods (Master's Thesis in Engineering Physics, Umeå University). 2024.

[15] Vahidzadeh E., Shankar K.: Insights into the Machine Learning Predictions of the Optical Response of Plasmon@ Semiconductor Core-Shell Nanocylinders. Photochem 3(1), 2023, 155–170.

[16] Verma S.: Evaluation of Photonic Characteristics of Plasmonic Integrated Metallic Nanoparticles with the help of Artificial Neural Network Parameterisation (Doctoral dissertation, University of London). 2023.

[17] Xu X., Aggarwal D., Shankar K.: Instantaneous property prediction and inverse design of plasmonic nanostructures using machine learning: current applications and future directions. Nanomaterials 12(4), 2022, 633.

[18] Zhang T. et al.: Machine learning and evolutionary algorithm studies of graphene metamaterials for optimized plasmon-induced transparency. Optics Express 28(13), 2020, 18899–18916.

M.Sc. Shahed Jahidul Haque
e-mail: haque019701@gmail.com

The present position of Shahed Jahidul Haque is at the School of Information and Communication Engineering within Nanjing University of Information Science & Technology in Nanjing Jiangsu China. The researchers from School of Information and Communication Engineering at Nanjing University of Information Science & Technology investigate modern technologies in their field under their research scope of information and communication engineering. Haque simultaneously works on current research projects while expanding his knowledge base within these subject areas.

https://orcid.org/0009-0004-7572-5337

M.Sc. Arman Mohammad Nakib
e-mail: armannakib35@gmail.com

Arman Mohammad Nakib has been in teaching profession since 2015. He is a Ph.D. candidate and completed his master's in Artificial Intelligence and bachelor in Electrical and Electronic Engineering. His research focus on OCT Imaging, Medical Image Processing, Machine & Deep Learning, Computer Vision, Robotics, Multi-Sensor Data Processing, Internet of Things, Automation.

https://orcid.org/0009-0006-4986-8806