On the use of Generative adversarial neural networks for computing photonic crystal fiber optical properties

Aimen Zelaci, Ahmet Yaşlı, Cem Kalyoncu, and Hüseyin Ademgil

***Abstract*—Photonic crystal fibers (PCF) for specific applications are designed and optimized by both industry experts and researches. However, the potential number of combinations possible for a single application is very large. This issue combined by the speed of the commonly used Full Vectorial Finite Element Method (FV-FEM) simulators causes the task to take significant amount of time. As stated in the previous works, artificial neural networks (ANN) can predict the result of numerical simulations much faster. However, there are two issues with the methods proposed previously: the required number of samples for training and the generality of these methods. In this paper, we propose the use of generative adversarial networks (GAN) to augment the real dataset to train an ANN model. Experimental analysis suggest that the proposed combination not only accurately predicts the confinement loss even with limited amount of data but also GAN can be used to improve existing methods in the literature. Finally, it is shown that this system can predict the confinement loss over a range of analytes and wavelengths in a completely new geometric configuration and generic enough not to require any additional tuning when used in a new dataset.**

I. Introduction

Importance of PCF and SPR, written by AY or HA

Machine Learning (ML) techniques are becoming a common tool in many fields and surpassing human performance in many tasks, such as automatic speech recognition, image recognition, natural language processing, drug discovery and toxicology. Additionally, ANN can approximate any function proven by the Universality theorem [1]. This fact propelled researchers to widen the applications of ANN even further, including the study of nanophotonic structures [2], optimization of photonic crystal nanocavities [3], and more recently, computing optical properties of a photonic crystal fiber [4].

One of the most difficult challenges that deep learning models face is that they benefit from large amounts of data to train, which is often costly and/or difficult to acquire. One of the solutions to overcome this issue is to artificially expand the original training dataset by the means of generative networks. Introduced by Goodfellow et al., Generative Adversarial Networks (GAN) [5], proved to be successful in data generation [6]–[10].

In this paper, we focus on estimating confinement loss, one of the propagation features of multi-channel Photonic Crystal Fiber (PCF) sensors, using artificial neural networks. Specifically, we have based our system on Surface Plasmon Resonance (SPR). However, in the experiments section we have demonstrated that the designed system is generic enough to be applied to multiple PCF designs. The most important contribution of this research is the use of GAN phase, where the available data is expanded to be used in the training phase.

!! Literature survey !!

Use of ANN in PCF

SPR in PCF

This paper is organized as follows. Section II details the use of GAN to generate additional training samples for ANN as well as the proposed neural network architecture. Photonic crystal fiber design that is used for testing is described in details in section III. Detailed analysis of the experimental results are discussed in section IV. Finally, concluding remarks are made in Section V.

II. Proposed method

In this section, details of the proposed method is discussed. Training of the system contains two phases: a GAN phase and regression training phase. At the start of the training, a GAN is trained to generate additional data by using training samples. These generated samples are filtered to ensure they fall within the applicable range. Original training samples and generated samples are joined to train a fully-connected feed-forward multi-layer perceptron (MLP) neural network for the regression ask of estimating confinement loss. At the end of the training, the ANN will be used to decide the containment loss of a previously unseen set of input parameters. This architecture is illustrated in Figure 1. The details of the proposed GAN and ANN architecture is discussed in the following subsections.

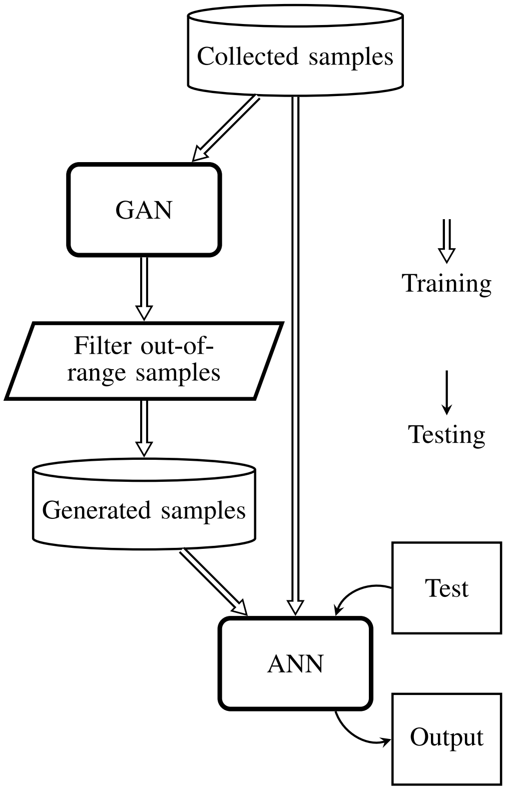


Figure 1: System architecture

A. Generative adversarial network design

Generative adversarial networks are introduced in [5]. We have employed GAN to augment the number of samples that are used in training. A GAN is composed of two neural networks: generator and discriminator. The aim of the generator is to transform fully randomized data into data that follows the distribution of the original dataset. The discriminator assesses the performance of the generator and provides feedback for training. Instead of directly training generator, it is trained through this feedback. This paradigm avoids over-fitting the data.

It is possible to use different metrics for training a GAN [5], [11], [12]; for this project we have selected Wasserstein distance metric called as WGAN. This variant is proposed by Arjovsky et al in [13]. In this system, discriminator is named as critic and it measures the distance between the generated and the real data. The reason behind choosing WGAN over other methods is to be able to determine a stopping criteria. In a regular GAN system, training is stopped when the generated data is deemed viable by an observer. Since GAN is often used in generation of image, video or audio, using a human in the loop is effective. However, in our problem, it is not viable for a human to judge the generated data. Automating this procedure to remove the human in-the-loop can lead to over or under-fitting, which in turn degrades the performance. WGAN uses an adaptive stopping criteria that does not have the issue mentioned above. Additionally, we have selected to incorporate Gradient penalty to improve WGAN as proposed in [14]. This improved WGAN system converges in a stable manner without having to fine tune hyper-parameters of the system. The flowchart of this system is given in Figure 2.

In this work, both the Critic and the generator are fully connected feed forward MLP models. The details of the networks are given in Table I. The output from this system should be input and output pairs that will be used to train ANN part of the system. In our SPR system, we have 7 input parameters and a single output parameter, making a total of 8 parameters that will be used in the GAN phase. In Section IV, we have experimented using a different PCF system with different inputs, resulting a different number of parameters for the system. The generator is supplied with the same number of parameters as its expected output, that is 8 for our dataset. These parameters are generated using Gaussian noise and is called latent variable.

Previous works [15], [16], demonstrated that it is possible for random data augmentation to weaken the performance of the model. Hence, it is important to sample the generated data in way to prevent performance degradation [17]. In the proposed system, we have included a filtering step for the generated data. This step uses a simple condition to discard the values that fall out of the desired range, for both the independent variables ($n_{analyte}, \lambda, \Lambda, d1, d2, d3, n_c$), and the confinement loss. Once the training phase is complete, the generator and the filter is used to augment the number of training samples that are available for ANN to train on.

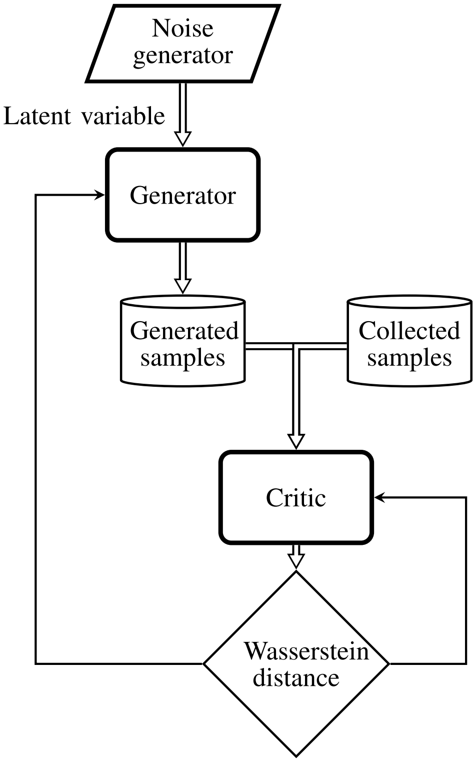


Figure 2: WGAN training

B. Artificial neural network design

The ANN model employed in this research is a fully-connected feed-forward Multi Layer Perceptron network consists of an input layer, an output layer, and 5 hidden layers. The ANN is designed and trained to estimate containment loss in a regression configuration. We have applied log_{10} to confinement loss in order to keep its numeric stability across a wide range [4]. Each hidden layer contains 50 neurons and uses Rectified Linear Unit (ReLU) activation function. Table I summarizes the details of this model.

In our model, we have used mini-batch training with 48 samples as the batch size. In contrast to full-batch training, mini-batch training allows more frequent gradient calculations, improving the stability and reliability of the training [20], [21]. The downside of using mini-batches are added computational complexity during

Table I: Details of ANN models

Parameter	Generator	Critic	Regressor
Hidden layers	5	5	5
Neurons in hidden layers	128	$2^{layer\_index} \times 32$	50
Batch size	32	32	48
Activation function	ReLU	Leaky ReLU	ReLU
Optimizer	Adam [18]	Adam	Adam
Input normalization	None	None	None
Layer normalization	-	Batch [19]	Batch
Loss function	$-Critic(generated)$	$Critic(gen) - Critic(col) + Penalty_{gradient}$	MSE

training. However, it has no adverse effect on the testing phase.

We have adopted Adam [18] as the optimizer and mean squared error (MSE) as the loss/cost function. In addition, we use Batch Normalization algorithm [19] to accelerate training, reduce the effects of initial randomized state and mitigate the problem of internal covariate shift. In the proposed system, the ANN is trained using the training samples from the original dataset and the augmented samples generated by the GAN phase.

III. Photonic crystal fiber design

PCF design details, written by AY. Following is only for reference

A labelled dataset of 432 samples was collected in this work, through simulations using FV-FEM. This dataset consists of the wavelength λ, index of refraction $n_{analyte}$, air-hole to air-hole distance Λ, and the air holes radii per ring $d1$, $d2$ and $d3$, and real part of the refractive index (n_c) taken as our independent variables. The labels are the confinement loss of the PCF. The set consists of nine different configurations of the geometric properties (Λ, d1, d2, d3), for each configuration the confinement loss was calculated for three different analytes (Water (n=1.33), Ethanol (n=1.35) and several commercial hydrocarbon mixtures (n=1.36)).

We will fix the pattern and geometrical shape of the cladding air holes, and vary the wavelength λ, index of refraction $n_{analyte}$, air-hole to air-hole distance Λ, and the air holes radii per ring $d1$, $d2$ and $d3$. Then the results of this study can be carried to different types of SPR-based PCF sensors, using other suitable deep learning approaches such as Recurrent and Convolutional neural networks.

IV. Experiments

A. Experimental setup

In order to prove the effectiveness of the proposed setup, we have performed multiple experiments over two datasets. We have built the first dataset that is used in the experiments. This is an SPR PCF configuration ... and collected using FV-FEM. Details of this dataset is explained in Section III and it is referred as SPR dataset. Having nine configurations, we have performed 9-fold testing on this dataset. Each fold tests a single configuration using the other configurations as training data. Slicing the data this way allowed us to guide the network to predict the loss on a configuration that has not used in training. Please note that we have applied log_{10} to confinement loss to scale it.

We have used mean square error (MSE) in the comparisons. All algorithms are implemented using TensorFlow with Keras library in Python and experiments are per-

Table II: Datasets that are used in experiments

	SPR	PCF
Samples	432	1117
Configurations	9	-
Parameters	7	5
Output	log_{loss}	log_{loss}
Testing	48 (1 configuration)	10%
Testing method	9-fold	10-fold

formed on laptop with Intel i7-5600 CPU and 8GB of RAM. TensorFlow library is set to use only CPU for the experiments.

Additionally, we have used the dataset that has been used in [4]. This dataset contains over 1000 samples. Inputs to this data is .... and outputs are .... We have only used log of confinement loss as the output. This dataset is referred as PCF. Details of the datasets that are used in the experiments are given in Table II. As a final note, the range and domain of these datasets are different; therefore, all numeric comparisons should be drawn within the same dataset.

In the following subsections, we have performed experiments to show the performance of the base ANN system by its own comparing with the state-of-the-art method in the literature [4], improvement gained by employing GAN phase, and finally the computational advantages of using ANN over the simulator. 5000 training epochs are used for [4] as suggested in the original paper. All numeric comparison results are averaged over multiple executions as listed in Table II. Non-averaged data used in figures, such as configuration estimation, are selected from non-extreme cases for the compared methods.

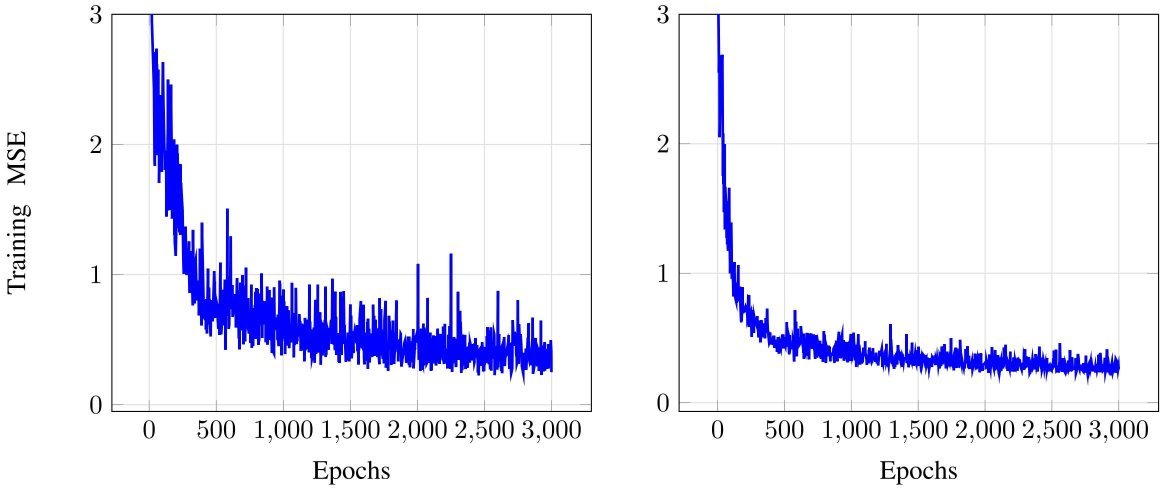
B. *Performance of ANN*

In this set of experiments, we have analyzed the performance of the proposed ANN design. The first experiment involves in training loss over SPR dataset. In this experiment, we have trained ANN model for up to 3000 epochs and reported the training loss in Figure 3. According to this experiment, training our ANN architecture more than 2500 epochs does not yield to any improvement. Therefore, we have set the training epoch limit to 2500 in all subsequent experiments. However, this number depends on the dataset and at times additional training can improve the result. Additionally, the benefit of augmented samples from the GAN phase is evident even at the training as they reduce the oscillations in the training loss.

We have performed experiements comparing the proposed ANN architecture with the state-of-the-art method proposed in [4]. Results of this experiment is demonstrated in Figure 4. In the experiment using SPR dataset, ANN models are predicting a new configuration and the number of samples available for training is lower. Therefore, they both have higher error rate compared to the PCF dataset. According to Figure 4c, $n_{analyte} = 1.35$ case, the ANN method proposed in [4] does not fit to the curve caused by the change in the wavelength. This can be caused by the fact that this method uses full-batch learning and therefore under-fits the model [21]. Regardless of this, both models have similar MSE due to the shift in the curve fit. This shift is caused because the network is never exposed to this particular geometric configuration.

C. *Performance boost of GAN*

The first set of experiments performed on GAN phase is the number of the augmented samples. In these experiments, we have augmented the proposed ANN system with varying number of samples. According to these experiments we have reached the minimum MSE at 1000 samples. Adding further samples slightly reduces the performance while increasing the training time. The results of this experiment is demonstrated in Figure



(a) With augmentation

(b) Without augmentation

Figure 3: Training MSE of the ANN model with (a) and without (b) GAN phase obtained from SPR dataset

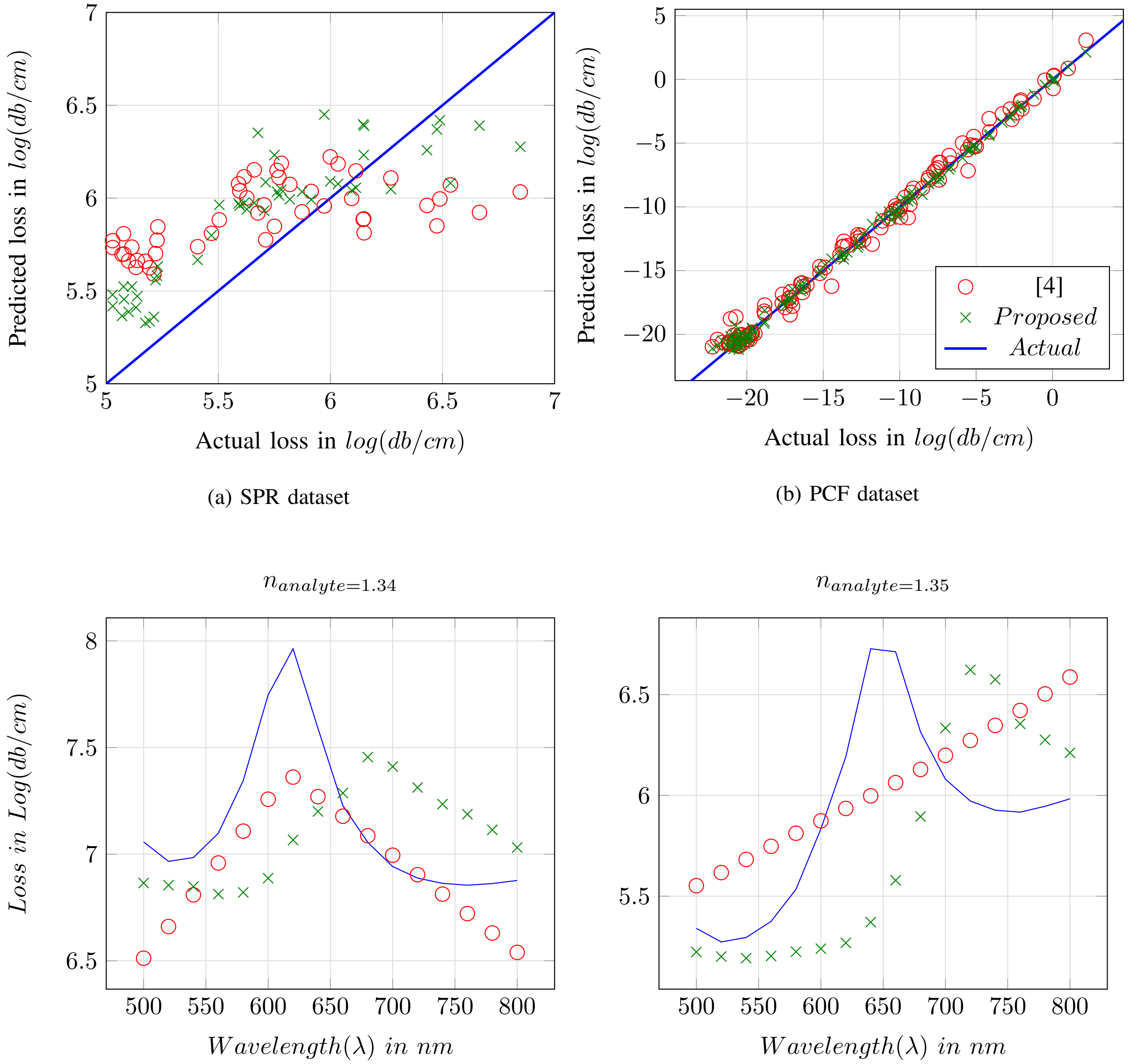
5. The number of augmented samples may depend on the dataset that is used. However, in our experiments 1000 augmented samples produce good results in both datasets.

We have performed experiments to showcase the improvement gained by augmenting training samples using GAN phase. In Figure 6, the results of these experiments has shown. An additional note to take away from this experiment is that the dataset with the limited number of samples has the highest improvement, which is demonstrated in a clearer fashion in Table III. This difference is due the fact that the number of training samples in this dataset is insufficient to train the regressor network causing reduced performance.

In Figure 6c, prediction of new configuration with and without GAN phase is compared. In the case with $n_{analyte} = 1.33$, the network without GAN phase has failed to predict the curve correctly. Even though GAN boosted network has predicted the curve lower than it is, the peak is in the correct wavelength, which could help PCF designers to pin-point highest loss.

We have also applied GAN phase to the network architecture proposed in [4]. In this experiment, GAN phase reduced the performance slightly. However, as we have discussed earlier, full batch training strategy is the cause of this due to increased number of samples. In fact, when we reduce the batch size down to 900 samples, GAN phase starts to improve the score of this architecture.

The final set of experiments shows the stability that GAN phase adds. In Table III, we have listed minimum, maximum, average MSE along with the standard deviation between folds in the testing. It is clear that the GAN phase not only improves the average case but significantly improves the worst case while reducing the deviation, which is important in a live setting. Even in a dataset with enough examples to train the network (PCF), using GAN phase reduces the worst case error by 32%. This added stability can also be observed in Figure 3, where the training loss has less fluctuations when the data augmentation is employed.



(a) SPR dataset

(b) PCF dataset

(c) Configuration estimation on SPR dataset

Figure 4: Testing results without GAN augmentation phase

Table III: Detailed performance analysis

Method	SPR			PCF		
Dataset	Proposed		[4]	Proposed		[4]
GAN Phase	Yes	No	No	Yes	No	No
Average	**0.314**	0.894	0.987	**0.134**	0.156	0.167
Best	**0.006**	0.042	0.047	0.095	**0.093**	0.106
Worst	**1.102**	6.792	5.955	**0.181**	0.265	0.416
Std. dev.	**0.437**	2.213	1.940	**0.031**	0.046	0.090

D. *Computational performance*

In this section we have determined both testing and training time of the proposed system in comparison to the simulation approach. In Table IV, the result of these experiments are published. Training time is total training time while testing is per sample. These tests are performed over SPR dataset. Please note that the training time is considered offline time; i.e., it is the time that

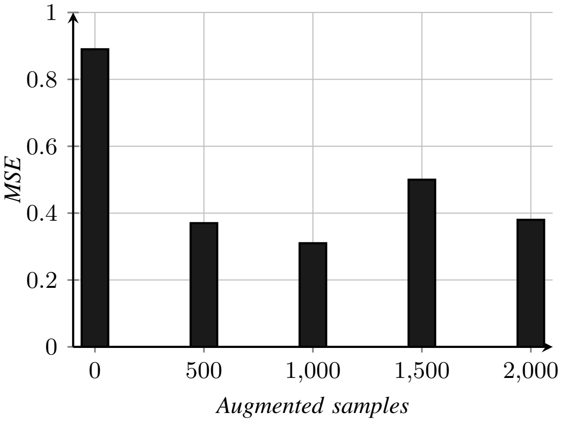


Figure 5: Effect of increasing augmented samples on MSE.

will only be used once, before the system is deployed. Once trained, the ANN can work alone without the need for GAN or training again. With this information, it is evident that the ANN based methods are far faster than the simulation approach by several orders of magnitude. Even when training time is included, total time spent to calculate the confinement loss will be faster on a GAN boosted ANN compared to an FV-FEM simulator after only 5 samples.

The training time difference between ANN systems is caused by the increased depth in the proposed architecture as well as mini-batch training strategy. Training time is dominated by GAN phase; however, this phase does not add any overhead to the testing time. The testing time difference between two networks are caused by the depth of the neural network (5 hidden layers instead of 3).

We have also experimented with TensorFlow using GPU via an additional system that contains NVidia GTX 1660Ti GPU. However, due to the relatively small size of the network, using GPU slightly reduced the computational performance instead of increasing it.

Table IV: Execution time of algorithms

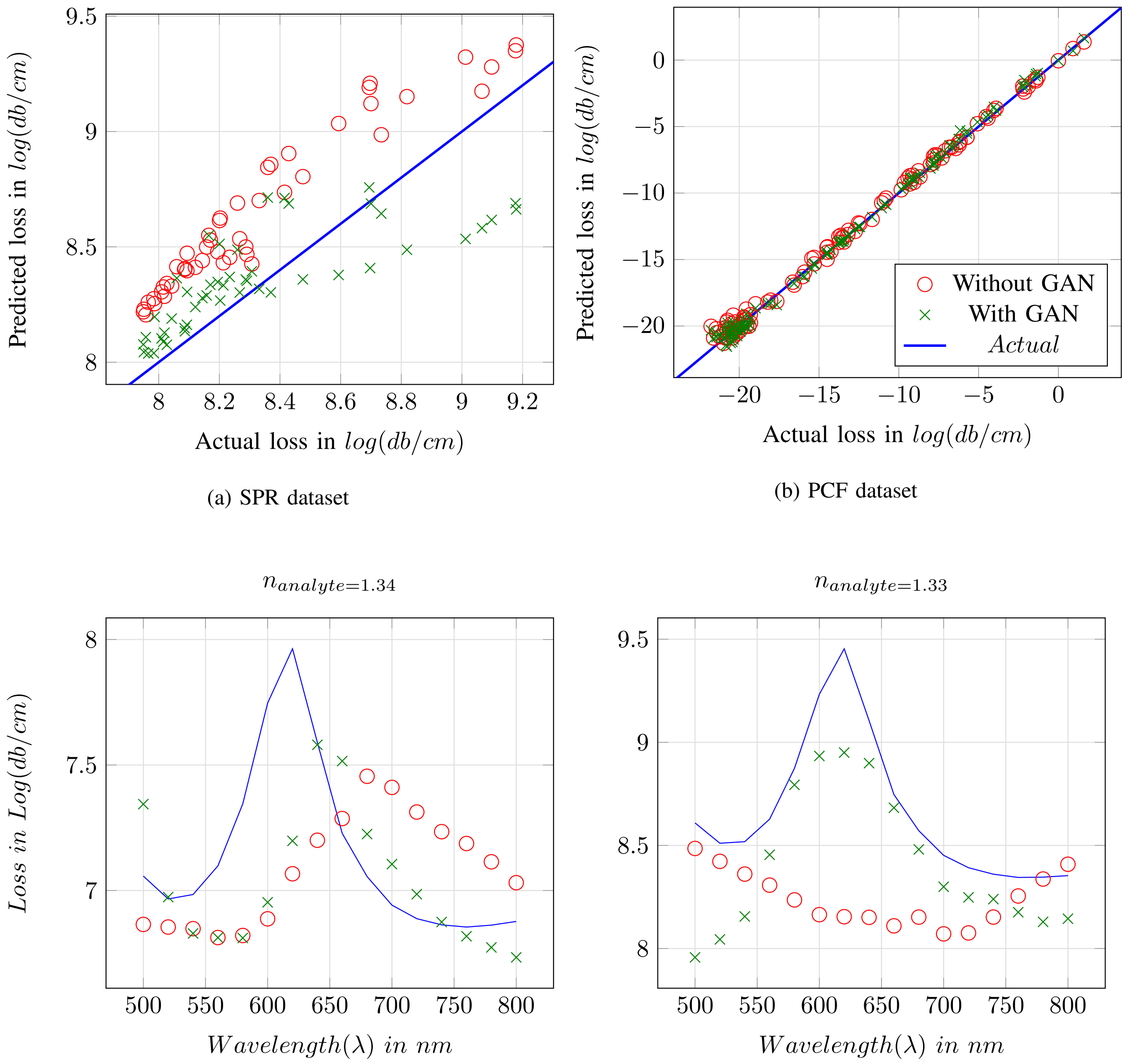
Method	Training (s)	Testing (ms)
Proposed ANN	42	3.6
Proposed ANN with GAN	449	3.6
GAN phase only	370	-
[4]	20	2.7
FV-FEM	-	106958

V. Conclusion

In this paper, we have demonstrated that the use of generative adversarial networks can benefit neural networks that are designed to compute photonic crystal fiber optical properties. Additionally, we have also proposed and demonstrated a neural network that is both more generic and more accurate to estimate the confinement loss compared to the state-of-the-art network. The GAN phase also reduces the required samples to fully train the ANN, reducing the work necessary to employ the proposed system. The proposed combination is shown to be general enough that even though it is designed for the SPR dataset built for this research; in the experiments, it surpassed the accuracy of the method proposed in [4] on the dataset this method is optimized on.

Another important aspect in this paper is the experiments performed on a dataset that is designed to test the ability of artificial neural networks to estimate the properties of a new configuration. For this purpose, we have built a database based on Surface Plasmon Resonance with varying geometric configurations. During the testing, a new configuration that is not used for the training is used. The proposed ANN, augmented with GAN phase is shown to estimate the new configuration to an acceptable error rate.

The most important benefit of using machine learning methods instead of simulation is the speed factor. In the experiments it is shown that the machine learning methods work few orders of magnitude faster than



(a) SPR dataset

(b) PCF dataset

(c) Configuration estimation on SPR dataset

Figure 6: GAN Phase experimental results

the simulation method. For different types of sensors, Convolutional and Recurrent neural networks can be of great advantage to discover the optimal patterns and geometrical shapes of the cladding air holes. This potential will be explored in a future work.

The code for this work is available at: https://github.com/Aimen-Zelaci/SPRPCF_ANN/.

References

[1] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251 – 257, 1991. [Online]. Available: http://www.sciencedirect.com/science/article/pii/089360809190009T

[2] Y. Kiarashinejad, M. Zandehshahvar, S. Abdollahramezani, O. Hemmatyar, R. Pourabolghasem, and A. Adibi, "Knowledge discovery in nanophotonics using geometric deep learning," *Advanced Intelligent Systems*, vol. 2, no. 2, p. 1900132, 2020.

[3] T. Asano and S. Noda, "Optimization of photonic crystal

nanocavities based on deep learning," *Optics express*, vol. 26, no. 25, pp. 32 704–32 717, 2018.

[4] S. Chugh, A. Gulistan, S. Ghosh, and B. Rahman, "Machine learning approach for computing optical properties of a photonic crystal fiber," *Optics Express*, vol. 27, no. 25, pp. 36 414–36 425, 2019.

[5] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in neural information processing systems*, 2014, pp. 2672–2680.

[6] T. Schlegl, P. Seeböck, S. M. Waldstein, U. Schmidt-Erfurth, and G. Langs, "Unsupervised anomaly detection with generative adversarial networks to guide marker discovery," in *International conference on information processing in medical imaging*. Springer, 2017, pp. 146–157.

[7] Z. Zheng, L. Zheng, and Y. Yang, "Unlabeled samples generated by gan improve the person re-identification baseline in vitro," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 3754–3762.

[8] M. Frid-Adar, E. Klang, M. Amitai, J. Goldberger, and H. Greenspan, "Synthetic data augmentation using gan for improved liver lesion classification," in *2018 IEEE 15th international symposium on biomedical imaging (ISBI 2018)*. IEEE, 2018, pp. 289–293.

[9] F. H. K. d. S. Tanaka and C. Aranha, "Data augmentation using gans," *arXiv preprint arXiv:1904.09135*, 2019.

[10] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," *arXiv preprint arXiv:1712.04621*, 2017.

[11] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 2794–2802.

[12] M. Lucic, K. Kurach, M. Michalski, S. Gelly, and O. Bousquet, "Are gans created equal? a large-scale study," in *Advances in neural information processing systems*, 2018, pp. 700–709.

[13] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein gan," *arXiv preprint arXiv:1701.07875*, 2017.

[14] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. C. Courville, "Improved training of wasserstein gans," in *Advances in neural information processing systems*, 2017, pp. 5767–5777.

[15] S. Ravuri and O. Vinyals, "Seeing is not necessarily believing: Limitations of biggans for data augmentation," 2019.

[16] K. Shmelkov, C. Schmid, and K. Alahari, "How good is my gan?" in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 213–229.

[17] B. Bhattarai, S. Baek, R. Bodur, and T.-K. Kim, "Sampling strategies for gan synthetic data," *arXiv preprint arXiv:1909.04689*, 2019.

[18] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[19] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," *arXiv preprint arXiv:1502.03167*, 2015.

[20] D. Masters and C. Luschi, "Revisiting small batch training for deep neural networks," *arXiv preprint arXiv:1804.07612*, 2018.

[21] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," *arXiv preprint arXiv:1609.04836*, 2016.