1. **Introduction :**

Machine learning is transforming our world as we know it, into science fiction. Especially deep learning [1], surpassing human performance in a variety of fields, namely automatic speech recognition, image recognition, natural language processing, drug discovery and toxicology. However, this flexibility does not stop here because Artificial Neural Networks can compute any function at all. No matter what the function is, there is to be a ANN so that for every possible vector X, the value f(X) (or some close approximation) is output from the network [2]. This fact propelled researchers to widen the applications of ANNs even further, including the study of nanophotonic structures [3], optimization of photonic crystal nanocavities [4], and more recently, computing optical properties of a photonic crystal fiber [5]. Albeit deep learning seems promising, unfortunately it comes with its perks. Perhaps one of the challenges is that deep learning models benefit from a large amount of data to train, which is quite costly to acquire [5]. One of the solutions to overcome this issue is to artificially expand the data by the means of generative networks. Introduced by Goodfellow et al., Generative Adversarial Networks (GANs) [4], proved to be very impressive in tasks such as synthetic data generation [6], Synthetically Augment data for deep learning based image segmentation [7]. GANs power comes from the fact that they discover insights and structures within the data that allows them to make good approximations to the real dataset. In this paper we focus on determining one of the propagation features of a multi-channel Photonic Crystal Fiber (PCF) sensor based on Surface Plasmon Resonance (SPR), that is the confinement loss, by using a an Artificial Neural Network model and a special kind of generative networks (Wasserstein Generative Adverserial Netwrok) to expand our limited dataset and improve the accuracy of our ANN model. Our aim to eventually surpass the simulations performed using Full-Vectorial Finite Element Method (FV-FEM) to design, optimise and evaluate the sensor performance. A time comparaison between the performance of the ANN and simulation techniques has been made here [ref].

1. **The Dataset:**

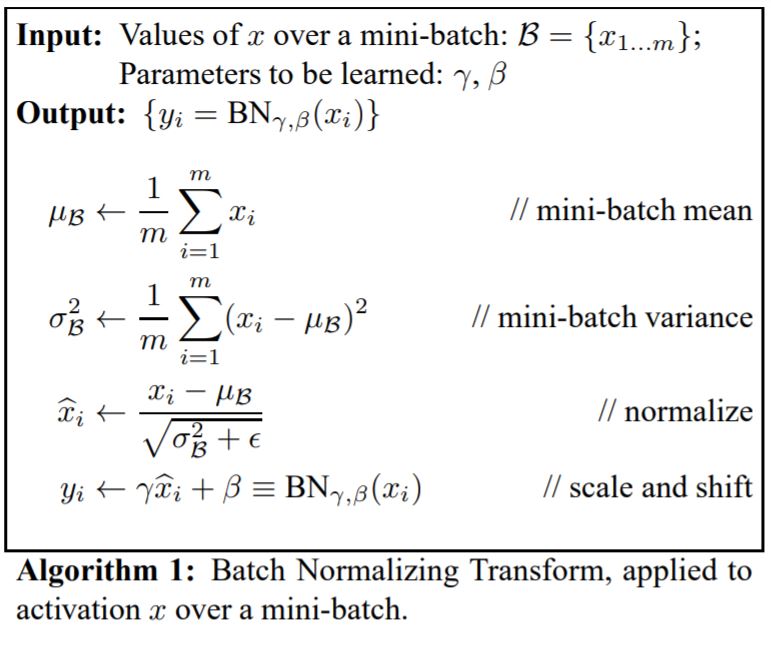
A labeled dataset of only 432 samples were collected in this work through simulations using FV-FEM. The length of the dataset made the task of building ANN model look quiet impossible, however the results were satisfying to some extent. The dataset consists of the wavelength λ, index of refraction n\_analyte, air-hole to air-hole distance Λ, and the air holes radii (d1, d2, d3), taken as our independent variables. The labels are the confinement loss of the photonic sensor. The set consists of 9 different configurations of the geometric properties (Λ, d1, d2, d3), for each configuration the confinement loss was calculated for 3 different analytes (Water (n=1.33), Ethanol (n=1.35) and several commercial hydrocarbon mixtures (n=1.36) [35]). 7 configurations were randomly selected for training both the WGAN and the ANN, 1 configuration was held out for validation, and the last configuration for testing. This data was preprocessed before fed in the networks. The indices of refraction {n\_analyte} are very close, which made it quiet difficult for the neural networks to differentiate between them, after many trials, the best choice was to take only the tens digit in 134, 135, 136, giving 4, 5, 6. Finally, we transform the confinement loss to the log scale.

1. **The ANN and WGAN architectures:**

In this section we discuss the architecture chosen for both the ANN model and the WGAN. Before that, we would like to state that the choices we made for the hyper-parameters of both the ANN and the WGAN are rules of thumb and heuristics. Furthermore, both models were trained using the back-propagation algorithm [9].

1. **The ANN model:**

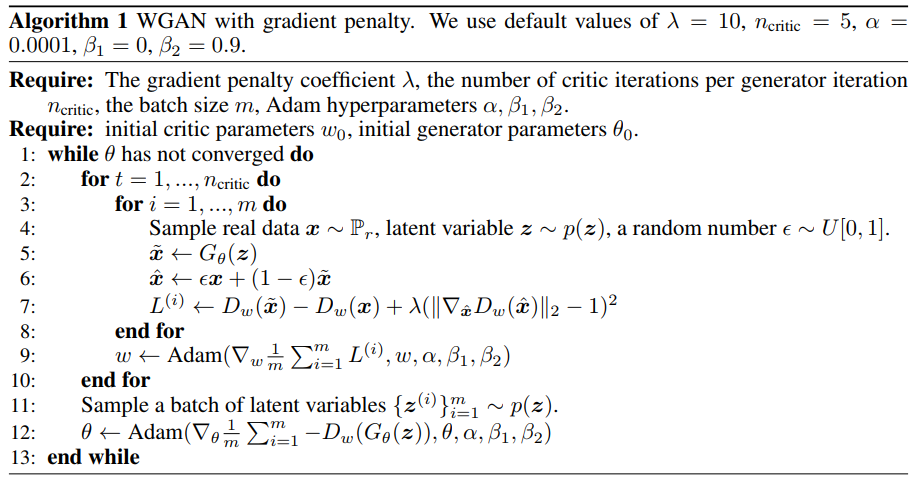
The ANN model is a fully-connected feed-forward MLP (Multi Layer Perceptron) consisting of an input layer, an output layer, and 5 hidden layers, 50 neurons for each hidden layer. Taking ReLU as the activation function, Adam as the optimizer and the mean suqared error as the loss/cost function. To reduce overfitting we use the Early stopping method [11], that is by saving checkpoints where the best validation mean squared error occurred as we iterate. To accelerate training and mitigate the problem of internal covariate shift, Batch Normalization algorithm was used [10]:



1. **The WGAN:**

In the original formulation of GANs, two neural networks, as shown in figure (3), contest with each other in a game (in the sense of game theory, often but not always in the form of a zero-sum game) [12]. The first of which called a discriminator gives an estimate of the probability that a given input is real or generated (fake). Whereas the second network is referred to as the generator, which outputs a data sample from a random noise vector called a latent variable usually given the symbol z supplied at its input. The error between the discriminator’s output and the actual labels (The real data samples vectors all labeled as 1) would then be measured by the means of a chosen metric. Introduced by Arjovsky et al [13] the Wasserstein distance metric (or the earth mover distance) proved to be very effective, instead of discriminating whether an input is real or generated, it provides a criticism of how far the generated data from the real data is, hence the discriminator network is referred to as the critic in the WGANs. Throughout this paper we will use yet the improved WGAN, the WGAN with Gradient penalty [reference], as it offers more stability to the network.

The algorithm for the WGAN-GP is as follows [reference]:



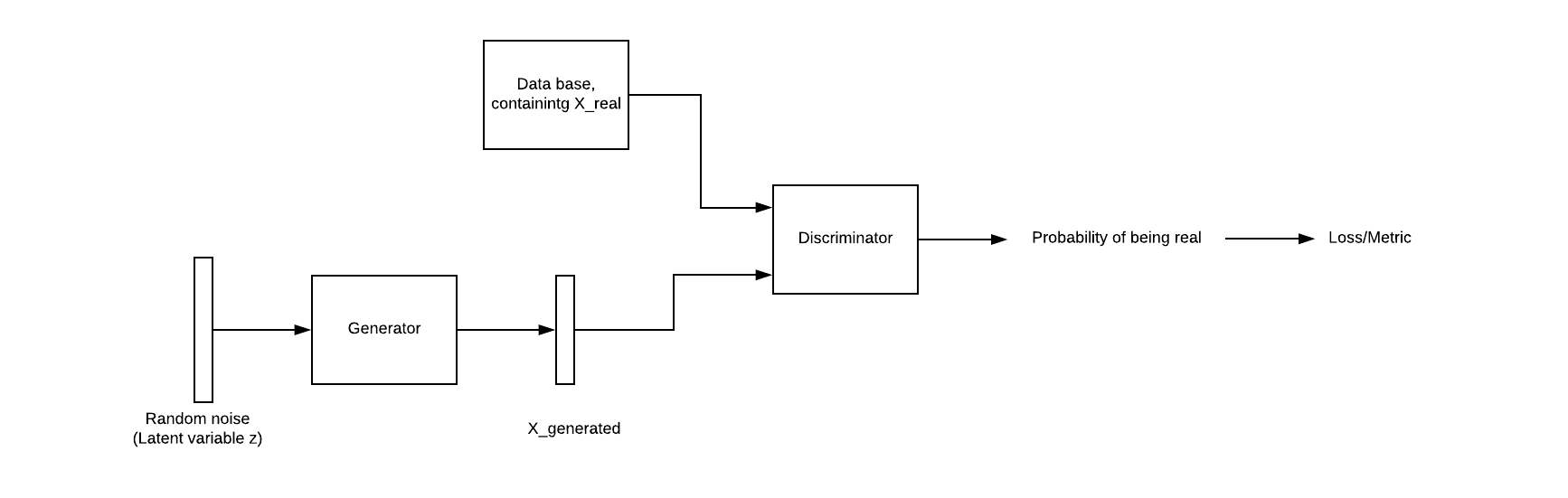


Figure (3)

Both the Critic and the generator networks are a fully-connected feed-forward MLP, having 4 hidden layers, with data\_batch\_size\*(2\*\*num\_of\_layer) neurons for each hidden layer for the critic wheras data\_batch\_size\* (2\*\*2) neurons for each hidden layer in the generator, therefore taking the form of an AutoEncoder [14]. The input vector for the Critic network consists of the 6 parameters discussed earlier for the ANN model, in addition the corresponding loss value, giving a total of 7 inputs. The generator is supplied with random noise vector having a dimension of 7, and outputs a vector that resembles that of the input to the critic network. LeakyReLU and ReLU were chosen as the activation fucntions for the Critic and Generator respectively and Adam as the optimizer for both networks. BatchNormalization technique was applied to the Generator, to stabilize the network.

1. **Experiments and numerical results:**
2. Training:

A.1 WGAN-GP:

We trained the WGAN-GP for 2000 epochs ( 1 epoch = 1 iteration over the whole training dataset) by monitoring the loss of the critic. We know that the training is over when this loss converges to 0 or fluctuates between positive and negative values in a very small interval. Figure (4), shows the curve of loss of the critic versus epochs. We also plotted the progress of the generator’s loss. The hyper-parameters chosen to be the same for both networks. They are demonstrated in the following table:

|  |  |  |  |
| --- | --- | --- | --- |
| Learning rate | Batch size | β1 | β2 |
| 2 e-04 | 12 | 0.5 | 0.9 |

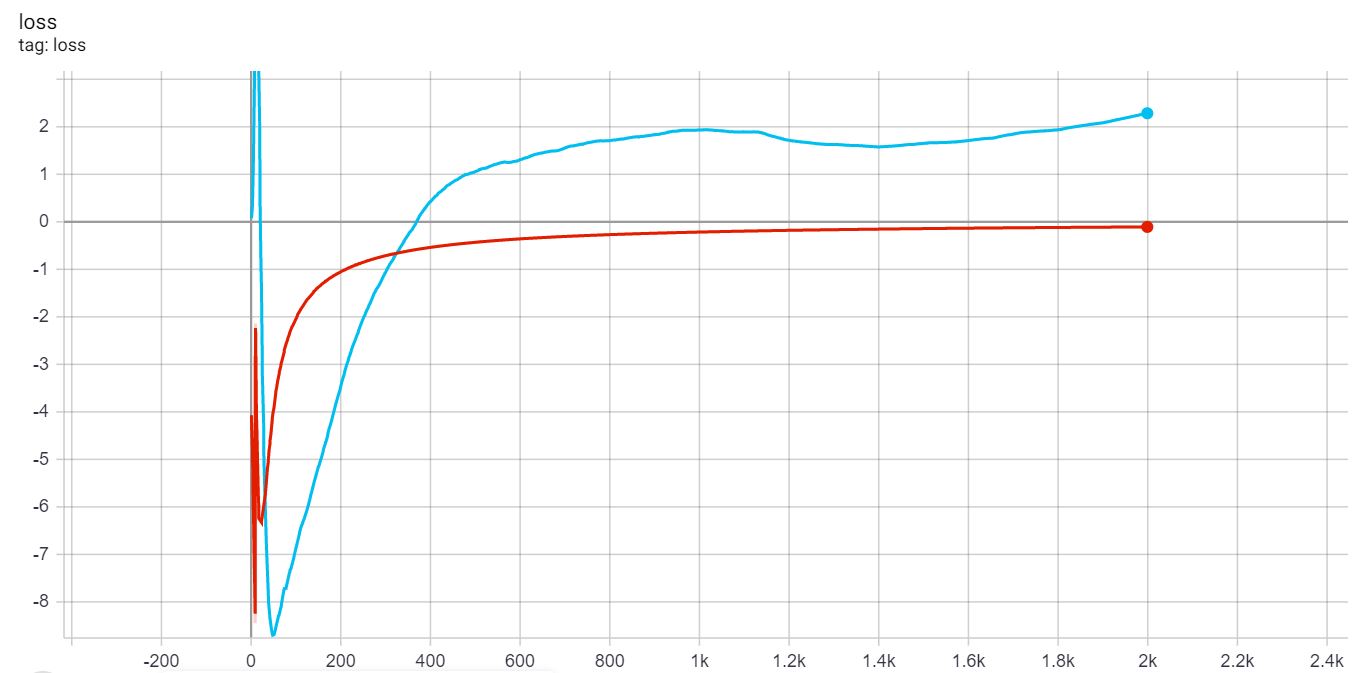


Figure (4)

A.2 ANN model:

We train the ANN model for more than 2000 epochs, starting from the orginal dataset. Then we augment the data by 1000 generated samples from our WGAN-GP. The MSE on the training sets ranges from 0.0030 to 0.0050. For all data sets the MSE on the training sets decreased in an acceptable manner, as shown in the following figure.

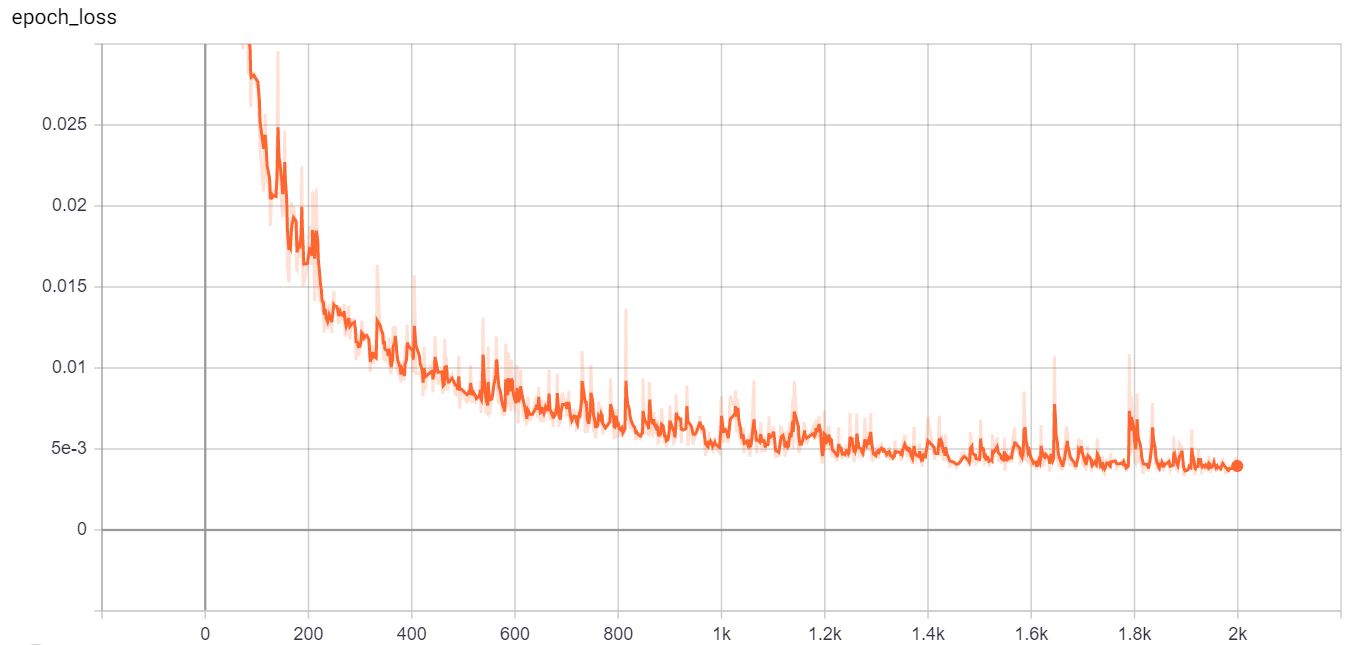


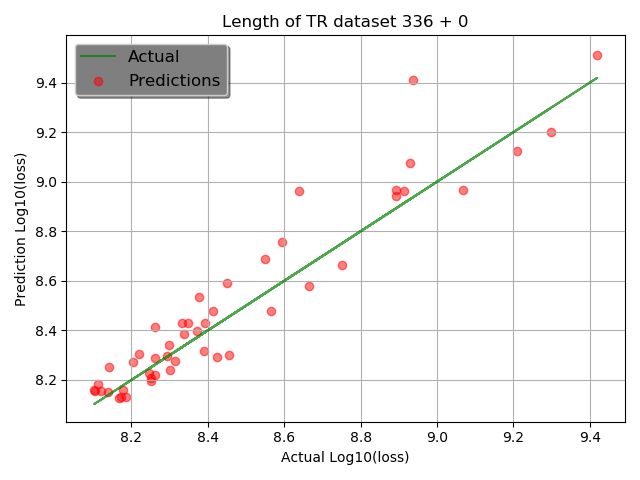
Figure (5) Smoothed curve of training loss versus epochs, length of training data 3000.

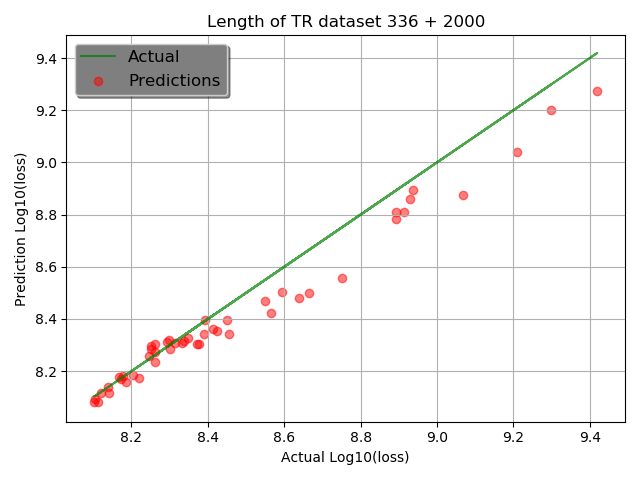
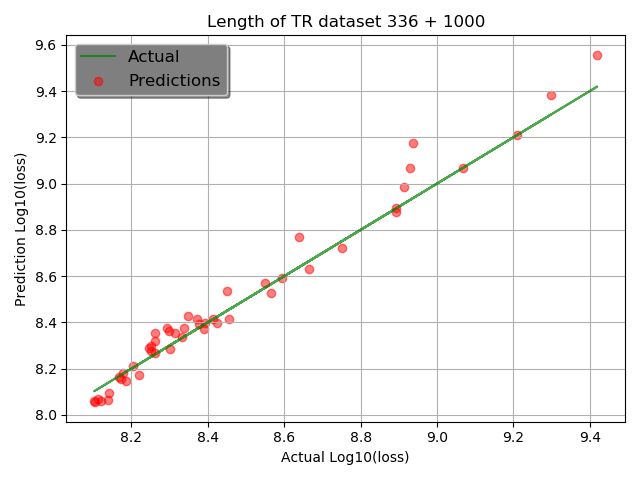
1. **Numerical results:**

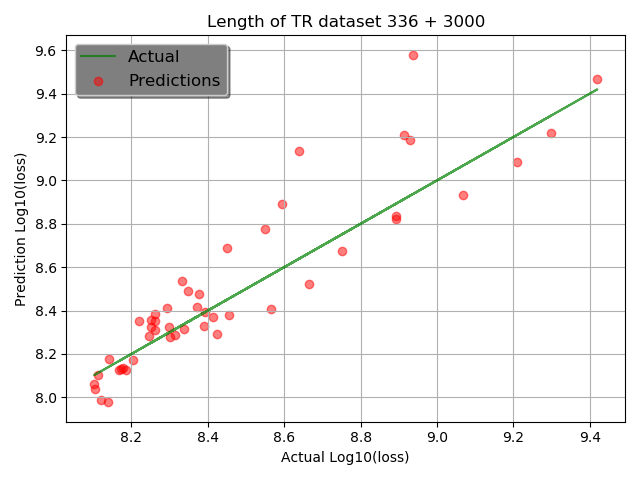
The following table summarizes the experiments:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Length of dataset | Learning rate | Batch size | β1 | β2 |
| 336 | 1e-04 | 8 | 0.9 | 0.999 |
| 1000 + 336 | 1e-04 | 8 | 0.9 | 0.999 |
| 2000 + 336 | 2e-04 | 16 | 0.9 | 0.999 |
| 3000 + 336 | 2.5e-04 | 20 | 0.9 | 0.999 |

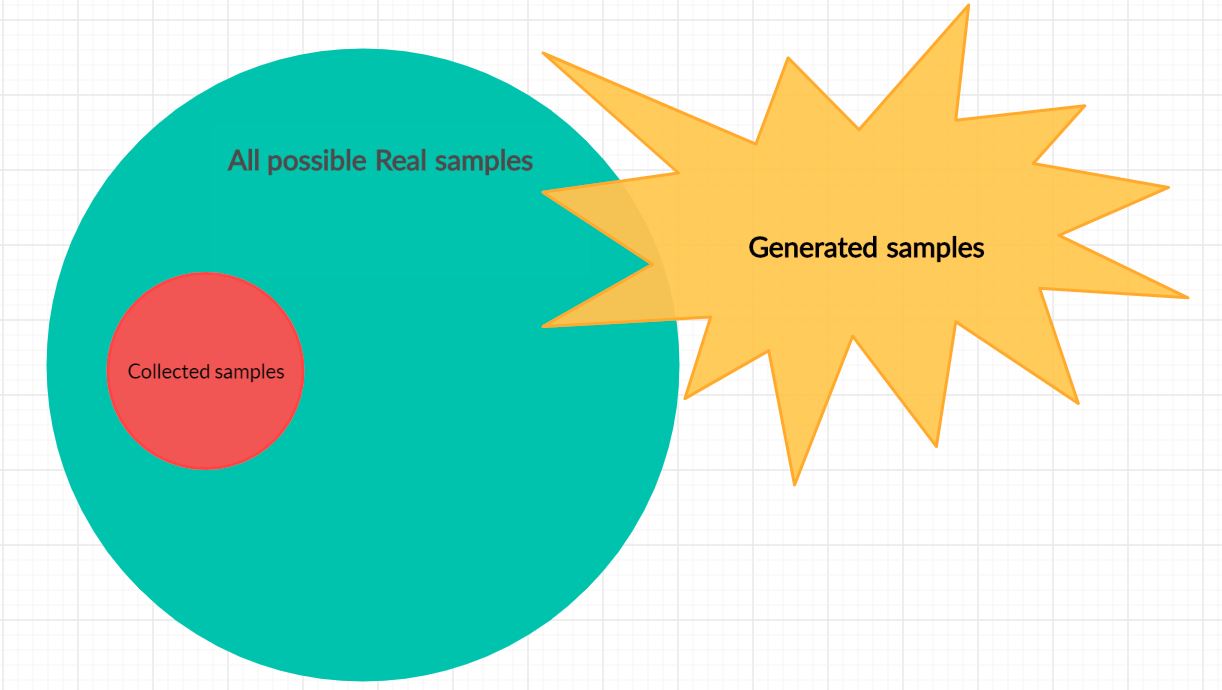
The following is the plots of predictions of the test set versus the actual values.







From the previous results we note that as we increase the data above 1000 samples, the MSE increases. This may seem paradoxical to what we mentioned earlier, that deep learning models benefit from large datasets to train. However, the reason why this is happening is due to the fact that the generated samples did not entirely contain the real samples as we hoped it would, and of course this is due to the fact that we don’t have enough data in this experiment to train the WGAN-GP in the first place! Therefore, the generated samples start to expand in the wrong direction, as demonstrated by the following Venn diagram:



Therefore, our best augmentation is by 1000. Next we plot the predictions of this model versus the wavelength.

