University of Manitoba

Faculty of Science – Data Science

DATA 4010 – Data Science Capstone Project

Exploring Datasets for Machine-Learning-Enabled Microwave Imaging

Mai Nguyen (7924202)

Domain Expert: Dr. Ian Jeffrey, P.Eng.

(Price Faculty of Engineering - Department of Electrical and Computer Engineering)

2023-2024 Academic Year

Table of Content

I. Introduction:

Microwave imaging is one method to work on the inverse scattering problem where we want to determine either the physical or geometrical properties of scatters from the measured scattered field. This process is used in several fields, from healthcare to food production to security, to assist the task of estimating physical parameters from the observations of external or internal radiant energy in remote sensing, detecting human organs and biological systems in biomedical imaging and diagnosis [1]. Microwave allow interrogation and measurement of the target without destructive evaluation, meaning without damaging the object or the inspected field, along with other advantages including being more cost-effective compared to MRI and safer than X-Ray due to its non-ionizing properties [2].

In a microwave imaging experiment setup, several transmitting and receiving antennas are positioned around the inspected fields. The backscattering signals are then received by the receivers which become the input to the image reconstruction algorithms. To overcome the challenge of having noisy and blurry reconstructed image, some deep learning techniques has been proposed by researchers [1], [3-6]. Convolutional Neural Network (CNN) is a type of deep learning model utilized to classify and detect target object due to its ability to extract features using convolutional layers and classify using connected layers.

This project aims to leverage the power of machine learning to take a closer look at the scatter field images to hopefully gain insights about the experiment design as well as constructing a U-Net CNN tailored to the given dataset.

II. Background and Goals:

1. Project Background and Motivation:

While there are existing research papers on developing a U-Net CNN to work on image segmentation like the one proposed as a reading material by the domain expert, from the engineering perspective, there are remaining questions when constructing the experiment such as “What kind of targets should constitute the data?”, “How well will the model generalize when introducing new data of different physical properties?”, “What frequency should be used?”, “What is the preferred number of receivers and transmitters?”, “How to tell if the reconstructed image is correct?”.

2. Project Goals:

Through this project with Dr. Jeffrey, some project goals were proposed as followed. First, we want to learn from the field image data alone something about the parameters of the dataset (i.e., the frequency, the number of transmitting and receiving antennas, type of object, etc.) that are needed to ensure a good imaging process. Second, we want to build a U-Net CNN to work on our data generated based on the Richmond Solver algorithm from one of the graduate students from the domain expert's lab. Some modifications were made to the algorithm to output images specific to this project like the frequency used in the generation process, the physical properties of the generated target and the target shape.

III. Data Description:

1. Experiment Design:

In the lab, to collect image data, a number of transmitting and receiving antennas are set up around the inspected field D that contains the object with relative permittivity $\varepsilon(r)$ ($r \in D$) which is the physical property of the material that made up the object [1]. Microwave of a certain frequency will go from the transmitters, through the inspected field then received by the receivers. Microwave imaging consists of two parts: real and imaginary. However, typical techniques only reconstruct the real part of the permittivity due to its high accuracy compared to the imaginary part [7]. An illustration (Fig 1) of an example set up for the experiment is presented below with the field D containing object of permittivity ε surrounded by three transmitters and three receivers.

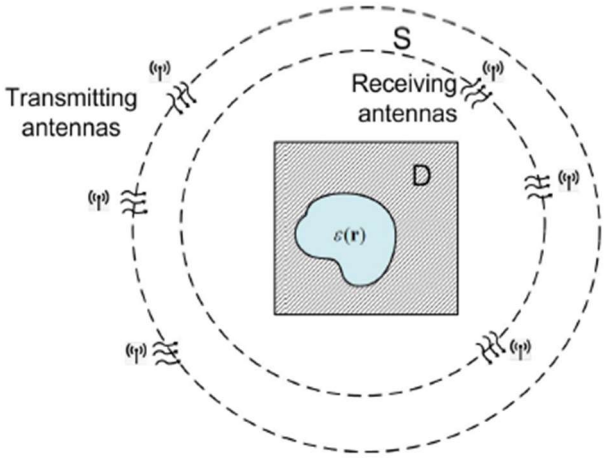

Fig 1: A typical set up for inverse scattering problem [1]

For this project specifically, we used synthetic data generated based on the Richmond Solver algorithm. This algorithm creates image data without the need to physically set up the transmitting and receiving antennas along with choosing the object to put into the inspected field. This helped speeding up the data collection task and allowed having control over how the data is produced. For instance, we can quickly change the frequency or the number of transmitters or receivers or the shape, size, location, and other physical properties of the inspected object. The data was generated with 24 transmitters and 24 receivers which created 24x24 field images. The target image will have different domain size depends on the type of image data.

2. Data Generation and Description:

Utilizing the Richmond Solver algorithm with some appropriate changes, two types of image data were generated. For both types, the algorithm output the real and imaginary field image and the corresponding image of the original target. As mentioned above, only real field images along with their corresponding target image were used in this project.

The first type was generated using the EMNIST database. EMNIST database is a large database of handwritten character [8] that is commonly used for training various image processing systems. The image format is of 28x28 pixel and have the data structure that matches the MNIST dataset. The target has fixed location and size and was generated using a fixed frequency of 1e9 for this image type. There are in total of 630 images that were split into training, validation and testing images in the later step. (Fig 2) shows sample type one field and target images. In this case, the target image is a 1-by-1 mapping of EMNIST image with 28x28 shape.

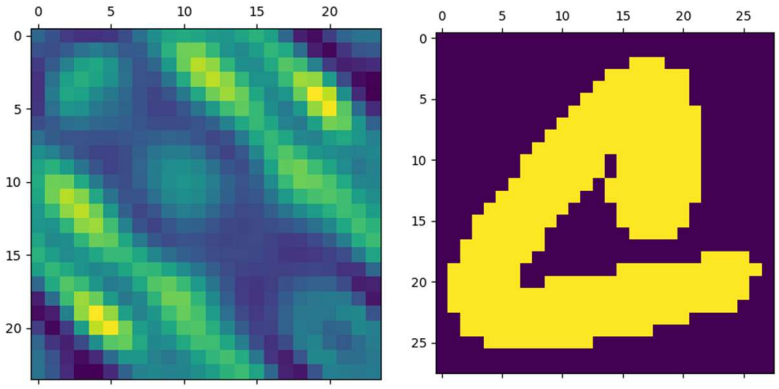

Fig 2: Sample field (left) and target (right) image from EMNIST

Another version of the EMNIST image data was also created for further testing. In this version, an extra value randomly drawn from the Gaussian distribution (Equation 1) with mean (μ) = 2.7 and standard deviation (σ) = 0.25 then Richmond Solver algorithm will loop over the list of coordinates in the target and apply that permittivity value proportional to how far it is from the peak. (Fig 3) shows sample image data with added permittivity which can be distinguished by the added green shade in the target image.

$$f(x) = \frac{1}{\delta\sqrt{2\pi}} e^{-\frac{1}{2}(\frac{x-\mu}{\delta})^2}$$

Equation 1: Property density function of Gaussian distribution

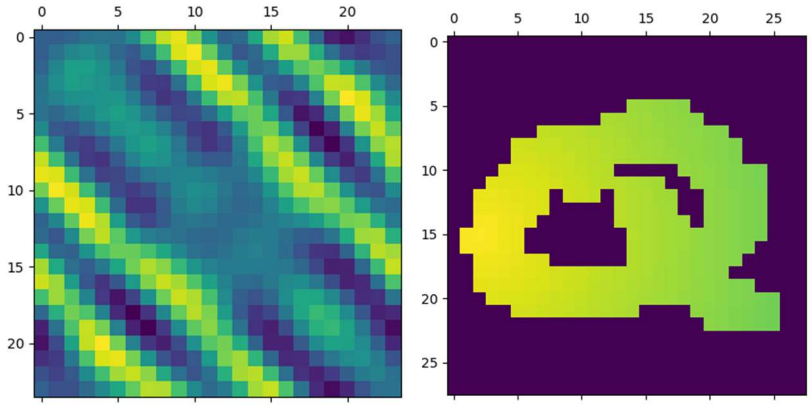

Fig 3: Sample field (left) and target (right) image with added permittivity from EMNIST

The second data type was generating from shape objects including circle, square and triangle. The target objects have fixed permittivity but different locations within the domain field, are roughly around the same size and were generated using frequencies in the range of 1e9, 2e9 and 3e9. (Fig 4) shows sample type two target images of circle, triangle and square with the corresponding field images. In this case, since we want to have the shape in different locations, the target image domain increases to 48x48. The circle radius is of 0.05 pixels, the side of a square is of 0.10 pixels, and the triangles are isosceles triangles with height of 0.10 pixels.

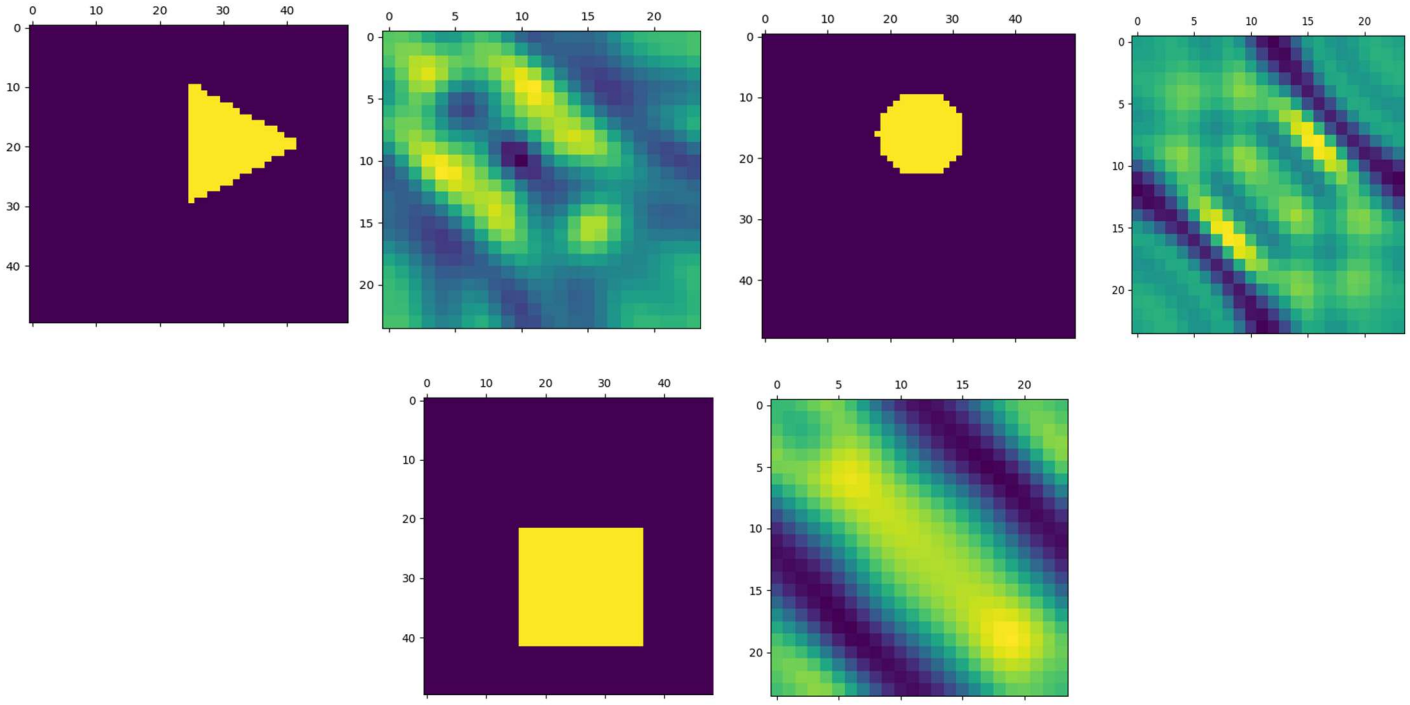

Fig 4: Sample field (left) and target (right) image of triangle, circle and square

Overall, EMNIST data generation only took approximately 10 minutes for all 630 pairs of field and target images. On the contrary, shape images took around an hour to generate 100 pairs of field and target images.

IV. Model Development and Evaluation:

1. Clustering Algorithm:

a) Background and Motivation:

The first goal of this project is to see if there is any interesting insight arises from just looking at the field image by itself without knowing anything about the target or how the experiment is set up. After some discussion with the domain expert, we chose to take a closer look at shape images generated by different frequencies as mentioned above in the data generation and description section. Our hypothesis is that by using image data clustering, the field images of the same frequency and/or same shape will appear together in the same cluster.

b) Algorithm:

Data clustering seeks to divide data points into groups so that the points within each group are more similar to one another than they are to the ones outside of them. K-means clustering is a

partition-based technique designed to minimize a specified clustering measure by repeatedly reassigning data points to different clusters until an optimal clustering is achieved [9]. When working with image data, k-means can be utilized as part of a deep clustering network [9]. For this project specifically, we want to test with using k-means algorithm alone. Minibatch k-means was used since it does not use all the datapoint for cluster distance calculation but uses a subset of the training data instead which reduces the clustering time as well as convergence time [10].

Based on the hypothesis, the parameter n_clusters of minibatch k-means was set to 3, which represents three different frequencies, and then to 2, which represents two different target shapes. The training data was initially consisting of images generated from circle and square targets only. After some preliminary training and testing, triangle target was introduced into the dataset. To hopefully simplify the clustering result for easier interpretation, only image data of two different target shapes were used at once for training, for example, data with circles and triangles or data with circles and squares.

c) Training Setup:

The initial shape dataset consists of 700 images of squares and circles with ratio 50:50. They all have fixed sizes but the location is randomly selected inside the domain field. The images of each target shape contain equal number of images generated using three different frequencies 1e9, 2e9 and 3e9.

The second shape dataset consists of the same number and ratio of circle and triangle images which replaces squares due to the potential effect of geometric similarity between square and circle. This will be discussed later in the result section. It was also generated using the above-mentioned frequencies.

For both dataset, first, the model was trained with the parameter n_clusters equals to 3 to hopefully get images generated by different frequencies to end up in the same cluster. Then, the parameter was set to 2 to observe whether each cluster consists of only one target shape.

d) Results and Evaluation:

We observed the clustering results and then evaluated using the Silhouette Coefficient (Equation 2) which measures the similarity of an object compared to its own cluster and to other clusters, with value ranging from -1 to 1. Silhouette coefficient close to 1 means good cluster separation, close to 0 means overlapping clusters, negative means wrong cluster assignment. We also looked at the Elbow plot to see if there is any other way to set the number of clusters.

$$s_i = \frac{b_i - a_i}{\max\{a_i, b_i\}} \text{ if } |C| > 1$$

$$s_i = 0 \text{ if } |C| = 1$$

Equation 2: Silhouette Coefficient formula for data point i in cluster C

(a_i: mean intra-cluster distance, b_i: mean nearest-cluster distance)

i. Square and Circle Dataset:

When setting n_clusters to 3, we observe the following (result in Table 1):

- The frequency of 3e-9 only appears in cluster 1. This might suggest that field images generated by frequency equals 3e-9 looks different from those generated by the other two frequencies.
- The other two clusters have quite similar results. This might suggest that the field images with frequencies 1e-9 and 2e-9 are similar.

Cluster	**Frequency**	**Shape**	**Count**
0	1e-9	Circle	117
		Square	38
		Total	**155**
	2e-9	Circle	63
		Square	87
		Total	**150**
1	3e-9	Circle	80
		Square	90
		Total	**170**
2	1e-9	Circle	13
		Square	132
		Total	**145**
	2e-9	Circle	37
		Square	43
		Total	**80**

Table 1: Clusters for square and circle images when n_clusters = 3

When setting n_clusters to 2, we observe the following (result in Table 2):

- In cluster 1, even though the target shape counts do not have a significant difference (95 circles and 118 squares), if we take a closer look at the frequencies, we can see that 170 of them are 3e-9 and only 43 are 2e-9. Moreover, frequencies of 1e-9 and 3e-9 do appear in the same cluster. This agrees with our previous observation about frequency of 3e-9 generating considerable different field image of square target object.
- The other cluster has an approximately equal breakdown of shape and frequency. This might suggest that there exists no significant difference between the field images of circle and square. Geometrically, circle and square might be too similar for the algorithm to distinguish solely from their scatter fields.

Cluster	**Shape**	**Frequency**	**Count**
0	Circle	1e-9	130
		2e-9	85
		Total	**215**
	Square	1e-9	170
		2e-9	102
		Total	**272**
1	Circle	2e-9	15
		3e-9	80
		Total	**95**
	Square	2e-9	28
		3e-9	90
		Total	**118**

Table 2: Clusters for square and circle images when n_clusters = 2

The Silhouette coefficient were around 0.1429 and 0.1990 consecutively for n_clusters = 3 and n_clusters = 2 which indicates clusters overlapping. The Elbow plot (Fig 5) does not indicate a clear optimal choice for the number of clusters. Therefore, circle and square might not be the best target shape to investigate.

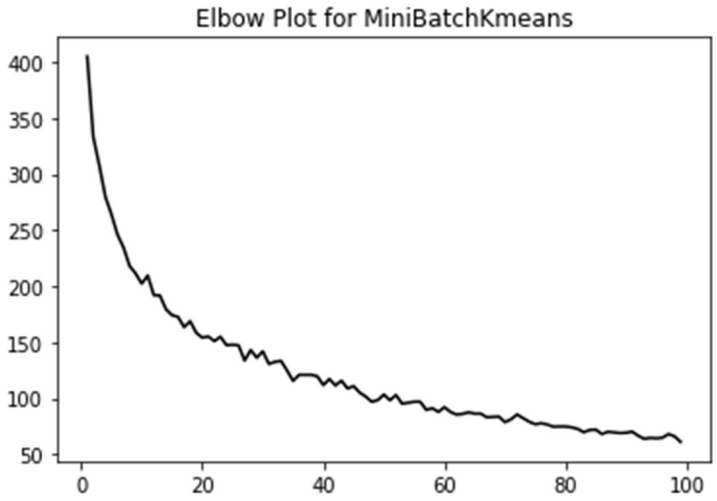

Fig 5: Elbow plot for circle and square dataset

ii. Triangle and Circle Dataset:

Afterwards, considering the similarity of circle and squares, we decided to train the model on triangle and circle images with three different frequencies instead.

When setting n_clusters to 3, we observe the following (result in Table 3):

- In cluster 0, field image of circles with frequency 2e-9 appears along with field images of both shapes with frequency 1e-9.
- In cluster 1, there are a more significant amount of frequency 3e-9 images.
- In cluster 2, there are only triangles with frequency 1e-9. This might indicate a significant difference in the field image of triangles generated by 1e-9 frequency.
- It can also be seen that frequencies 1e-9 and 3e-9 are not in the same cluster. This agrees with our finding from the square and circle dataset that these two frequencies might generate significantly different field images.

Cluster	**Frequency**	**Shape**	**Count**
0	2e-9	Circle	96
		Total	**96**
	1e-9	Circle	130
		Triangle	39
		Total	**169**
1	2e-9	Circle	4
		Triangle	40
		Total	**44**
	3e-9	Circle	80
		Triangle	110
		Total	**190**
2	1e-9	Triangle	51
		Total	**51**

Table 3: Clusters for triangle and circle images when n_clusters = 3

When setting n_clusters to 2, we observe the following (result in Table 4):

- In cluster 0, there are both circle and triangle images. While the circles have all three frequencies, we only see 2e-9 and 3e-9 for triangles.
- In cluster 1, there are only triangle images with frequency 1e-9. This agrees with our observation above.

Cluster	**Shape**	**Frequency**	**Count**
0	Circle	1e-9	130
		2e-9	100
		3e-9	80
		Total	**310**
	Triangle	2e-9	40
		3e-9	110
		Total	**150**
1	Triangle	1e-9	90
		Total	**90**

Table 4: Clusters for square and circle images when n_clusters = 2

The Silhouette coefficient in testing were 0.2042 and 0.347 consecutively for n_clusters = 3 and n_clusters = 2 which still indicates some overlapping. The Elbow plot (Fig 6) suggests changing the choice of number of clusters to possibly 4.

However, we did not proceed with changing the number of clusters since we needed to discuss and figure out the physics and design side of the experiment.

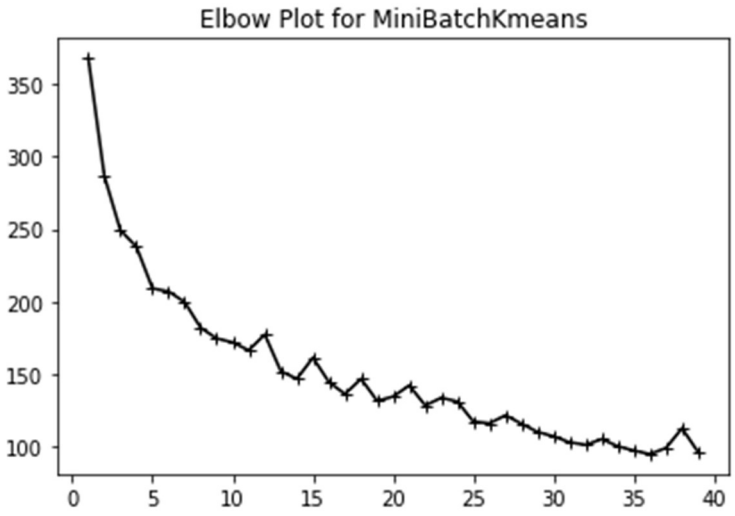

Fig 6: Elbow plot for circle and triangle dataset

2. U-Net Convolution Neural Network:

a) Background:

There are several proposed U-Net architectures [1], [3-6] for medical image segmentation which is the technique of dividing a digital image into several. Segmentation aims to simplify and/or change an image's representation into something more comprehensible and straightforward to examine. Finding boundaries (lines, curves, etc.) and objects in photos is usually accomplished using image segmentation. U-Net works well with the image segmentation task as a result of downsampling in the contracting path which significantly increases the effective receptive field of the network. This is crucial for the Inverse scattering problem since a wide field of view across the input image can greatly improve the prediction for each pixel of the output image [11].

b) Initial Exploration:

To get started with image classification, we tried building a simple binary classifier using Random Forest and then Support Vector Machine. Data for training was a simple dataset consists of triangle and circle shape images generated using 3 different frequencies. Both classifiers seemed to overfit with this small dataset of only 528 training images. The result, therefore, did not provide any useful information. We decided not to continue exploring this path at this point and moved on to U-Net CNN.

c) Network Architecture:

In this project, however, the U-Net CNN was built based on the architecture proposed in [1] with some modifications to work better with our data. Overall, the U-Net architect consists of

a contracting and expansive path which when put together resembles the letter “U”, hence the name U-Net. The contracting path on the left side of the U-Net architecture consists of repeating 3x3 convolution, batch normalization, and rectified linear unit (ReLU) (Equation 3), then followed by a 2x2 max pooling operation. The expansive path on the right side of the U-Net architecture is similar to the contracting path but with a 3x3 up-sampling convolution replacing max pooling. Skip connection is added from the input of the neural network to its output layer due to their shared similar important features. Moreover, this approach deals with the vanishing gradient problem during training [12]. Other method to address the vanishing gradients problems is batch normalization which fixes the means and variances of layer inputs, offers a faster training and reduces the dependence of gradients on the scale of the parameters or their initial values. [13]

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

Equation 3: Rectified Linear Unit (ReLU) activation function

EMNIST image data was used for training and testing U-Net CNN. The input image is of shape 24x24x1, i.e: 24 rows, 24 columns and 1 channel. However, the desired output is of shape 28x28x1, i.e: 28 rows, 28 columns and 1 channel. Therefore, to deal with the mismatch shape of the input and output image data, two U-Net architecture were developed for this project with similar contracting paths but with small modifications to the preprocessing part and the result of the expansive paths to achieve the desired image shape.

For the first architecture (Fig 7), zero-padding is added to the input image to create 28x28x1 image. The contracting and expansive path are as described above. To get the output segmentation map, a 1x1 convolution is added to the end.

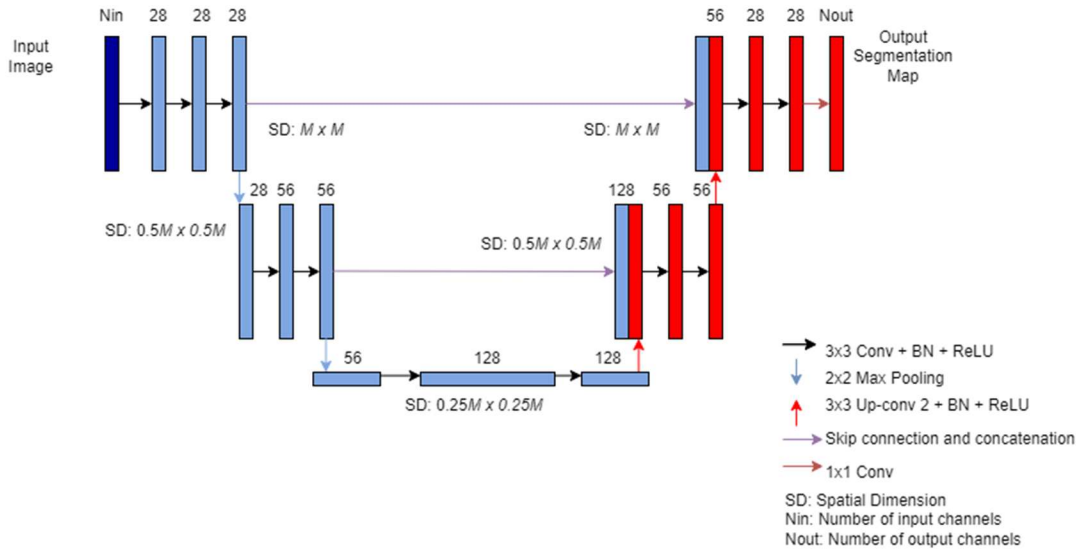

Fig 7: First proposed U-Net architechture

For the second architecture (Fig 8), the contracting and expansive paths are similar to those of the first architecture. However, instead of adding zero-padding to the input image, a flatten layer is added to the output of the last ReLU, followed by a dense layer, then finally a reshape layer and a 1x1 convolution to get the desired output shape. Sigmoid activation function was used in this last layer since we want to output.

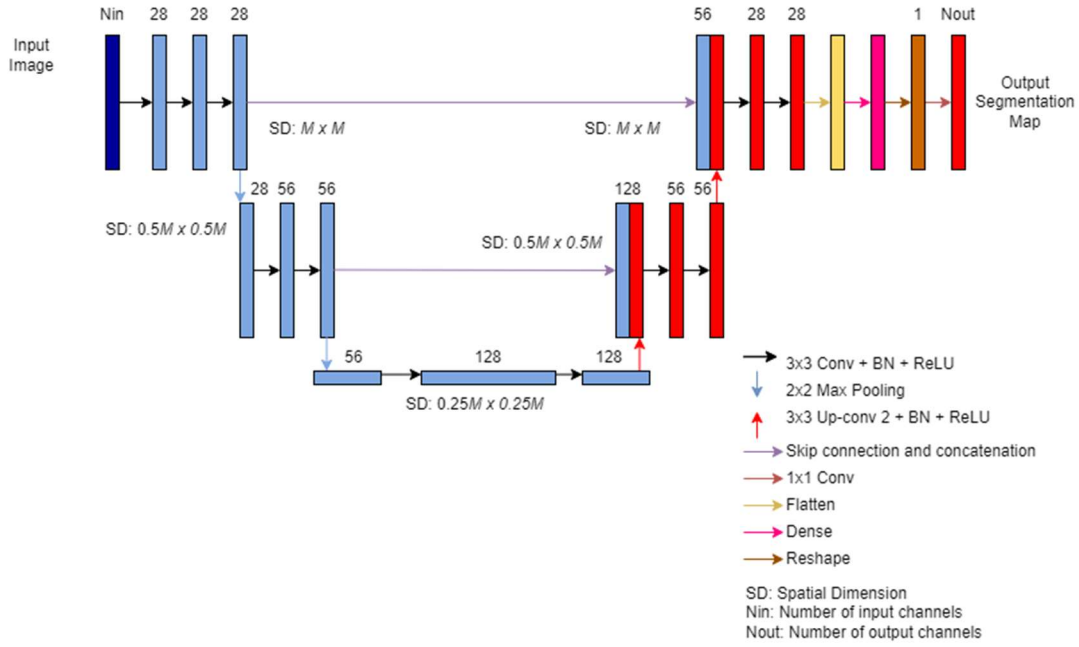

Fig 8: Second proposed U-Net architecture

Another architecture was also considered to get the output image to the shape 28x28x1. One extra decoder block was added on to the expansive path to get the shape from 24x24x1 to 48x48x1, followed by a crop layer to evenly crop it down to 28x28x1. However, the Richmond Solver algorithm was updated to generate output image as a 1-by-1 mapping of EMNIST database, increase the size of the EMNIST characters and fix their location in the domain field. This method created the problem of cropping out some important features, and therefore, affect the model evaluation. In the end, only the two architectures in (Fig 5) and (Fig 6) were used in this project.

d) Model Training Setup:

The dataset generated from the EMNIST database contains 630 images which were created using frequency 1e9, target object permittivity value of 2. It was split into training and testing set with the ratio of 80:20. Then the training set was further split into training and validation set with the same ratio.

The research paper [1] proposed using Stochastic Gradient Descent optimizer with momentum to helps with faster learning time [14]. However, some studies [14], [15] suggest a superior performance in terms of minizine the error rate and increase the accuracy rate when using Adam optimizer (Equation 4). Therefore, Adam optimizer is chosen for this project. The parameters for Adam optimizer include η: initial learning rate, g_t: gradient at time t along w_j, x_t: exponential average of gradient along w_j, y_t: exponential average of squares of gradient along w_j, σ_1 and σ_2: hyperparameters.

$$x_t = \delta_1 * x_{t-1} - (1 - \delta_1) * g_t$$

$$y_t = \delta_2 * y_{t-1} - (1 - \delta_2) * g_t^2$$

$$\triangle \omega_t = -\eta \frac{x_t}{\sqrt{y_t + \epsilon}} * g_t$$

$$\omega_{t+1} = \omega_t + \triangle \omega_t$$

Equation 4: Mathematical notation for Adam optimizer

The cost function is binary cross-entropy (Equation 5) since our task is simplified to binary classification of 0 (free space) or 1 (target object). The parameters for binary cross-entropy includes N: total number of outputs, y_i: targets/labels and $p(y_i)$: probability of y_i being equal to 1.

$$H = \frac{-1}{N}\sum_{i=1}^{N}(y_i.\log(p(y_i)) + (1 - y_i).\log(1 - p(y_i)))$$

Equation 5: Binary cross-entropy

The initial learning rate is 2.5e-5, for the first model, and 2e-4, for the second one, and the maximum number of epochs is 80. The batch size was set to 8, for the first model, and 10, for the second one. K-fold cross-validation were implemented with 5 folds for training and validation. If the validation loss does not change after 10 consecutive epochs, for the first model, and 8, for the second model, the learning rate is reduced at the rate of 0.2. If the training loss does not change after 15 consecutive epochs, for the first model, and 10, for the second model, training is stopped. The full model hyperparameters are included in Table 5. Along with training and validation losses, training and validation accuracies were recorded.

Training was done on personal computer with 11th Gen Intel(R) Core(TM) i7-1165G7 @ 2.80GHz and 8.00 GB RAM.

Parameter's Name	**Model 1 Values**	**Model 2 Values**
Input channels	1	
Output channels	1	
Optimizer	Adam	
Batch size	8	10
Learning rate	2e-5	2e-4
Loss type	Binary Cross-Entropy	
Maximum number of epochs	80	
Epochs patience for early stopping	15	10
Epochs patience for changing the learning rate	10	8
Learning factor	0.2	
Number of folds	5	

Table 5: Hyperparameter values

e. Results and Evaluation:

i. Training and Validation:

For model 1, training took approximately 45 minutes with training and validation losses for each epoch in each fold plotted in (Fig 9). As can be observed, the model was trained for at least 70 epochs and the performance becomes saturated after around 40 to 50 epochs. Training and validation accuracies also becomes saturated after the same number of epochs. Training accuracy reached 96% and validation accuracy reached 88%. Thus, the model does not overfit and converges well.

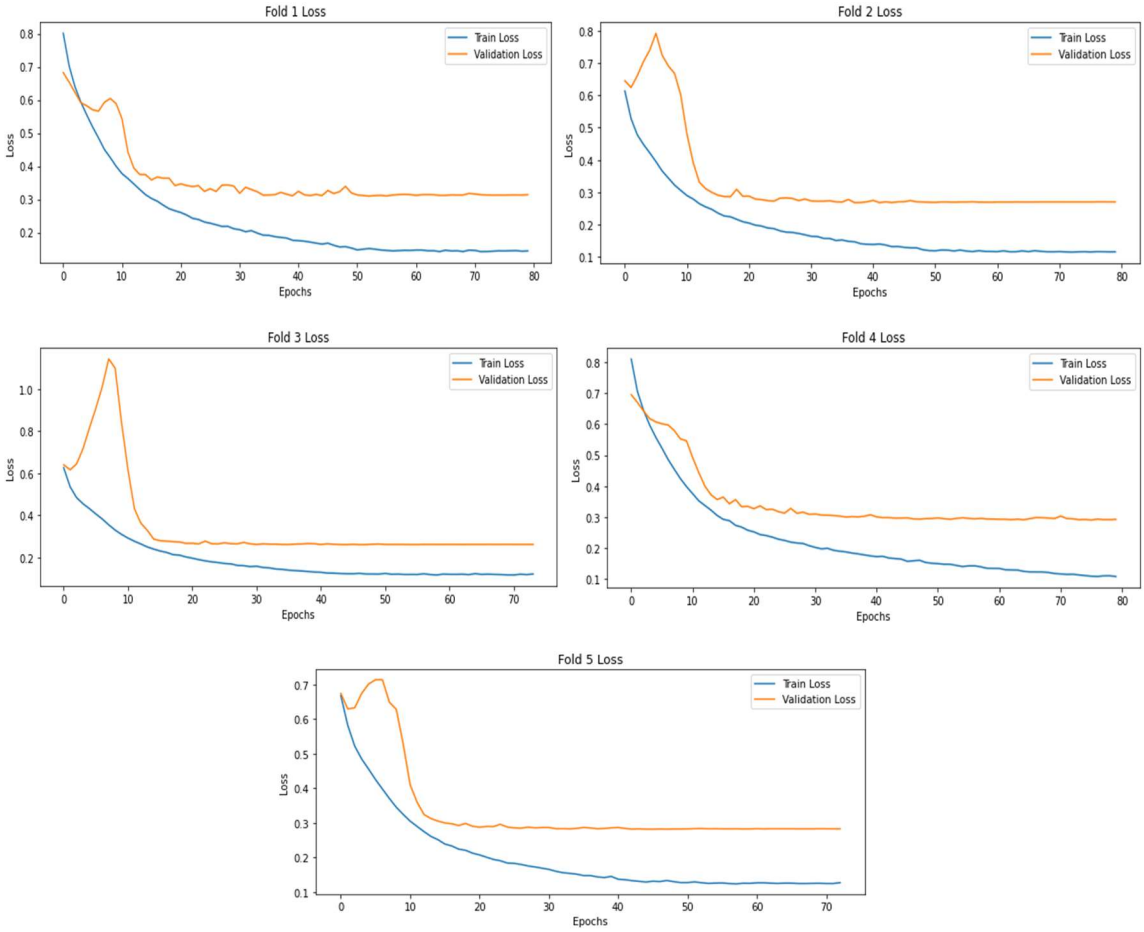

Fig 9: Training and Validation Losses for Model 1

For model 2, training was a bit faster at about 30 minutes due to early stopping. Training and validation losses for each epoch in each fold are plotted in (Fig 10). As can be observed, the model was trained for at least 40 epochs and the performance becomes saturated after around 15 epochs except from fold 2 having 80 epochs in total and performance saturation was reached after 65 epochs. Training and validation accuracies observed the same behavior concerning performance saturation. The highest values for training and validation accuracies are 91% and 84%, respectively. Thus, the model does not overfit and converges well.

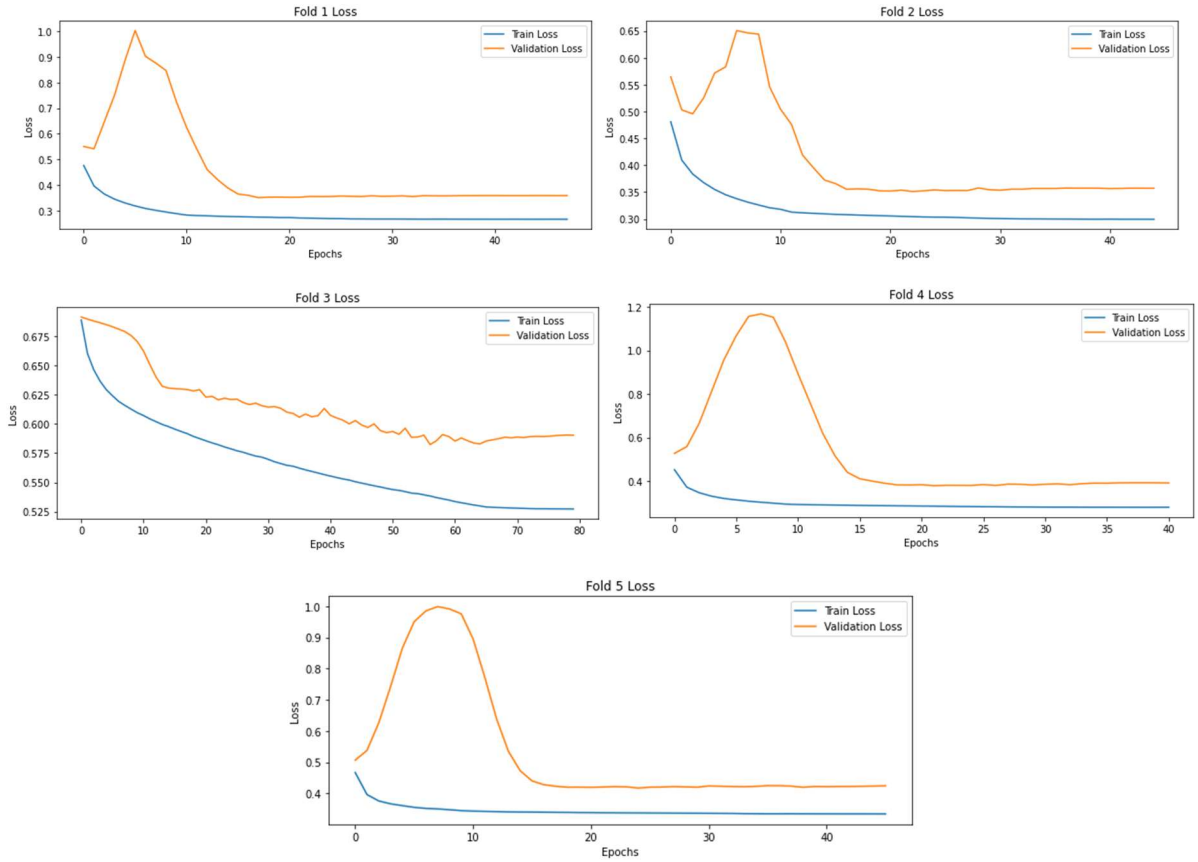

Fig 10: Training and Validation Losses for Model 2

ii. Prediction:

For model 1, when doing prediction with the test sets, we achieved an accuracy score of 88.77% for the field images with fixed target permittivity (sample results in Fig 11) but only 57.37% for the field images with randomly added target permittivity (sample results in Fig 12) which is understandable since our activation and loss functions are specifically chosen for binary classification task. It was also seen that the model provided a better image reconstruction for certain targets depends on its geometric complexity (for example, better image for letter “p” compared to “k” or better for letter “r” compared to “g”). This applies to both types of testing datasets (fixed and non-fixed permittivity).

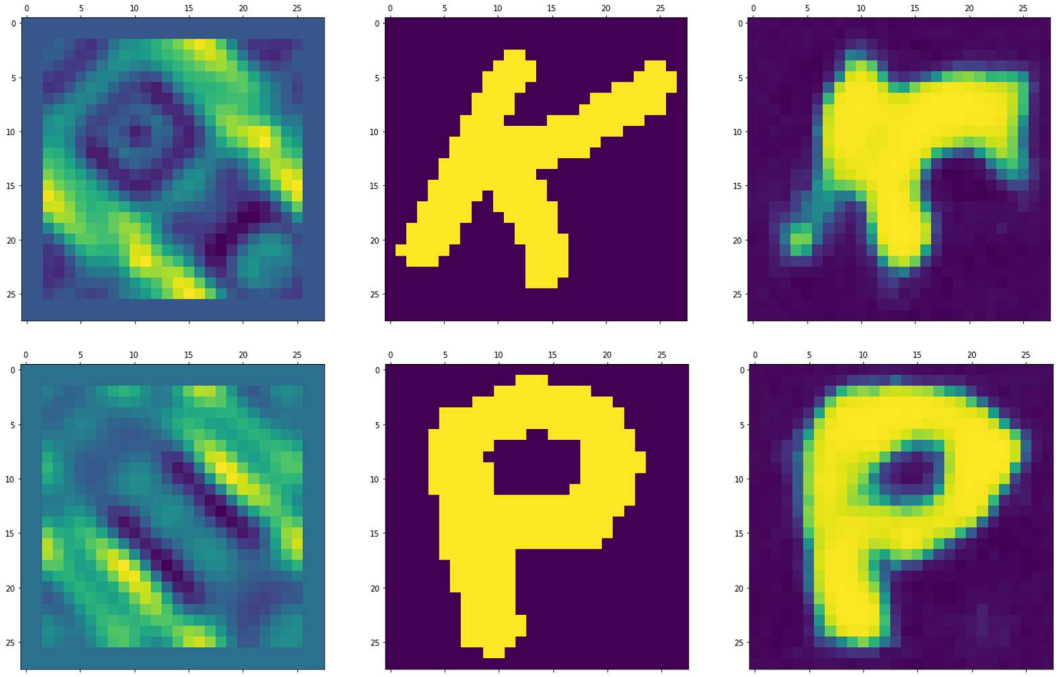

Fig 11: Padded field (left), true target (center) and reconstructed target (right) from model 1 with fixed permittivity

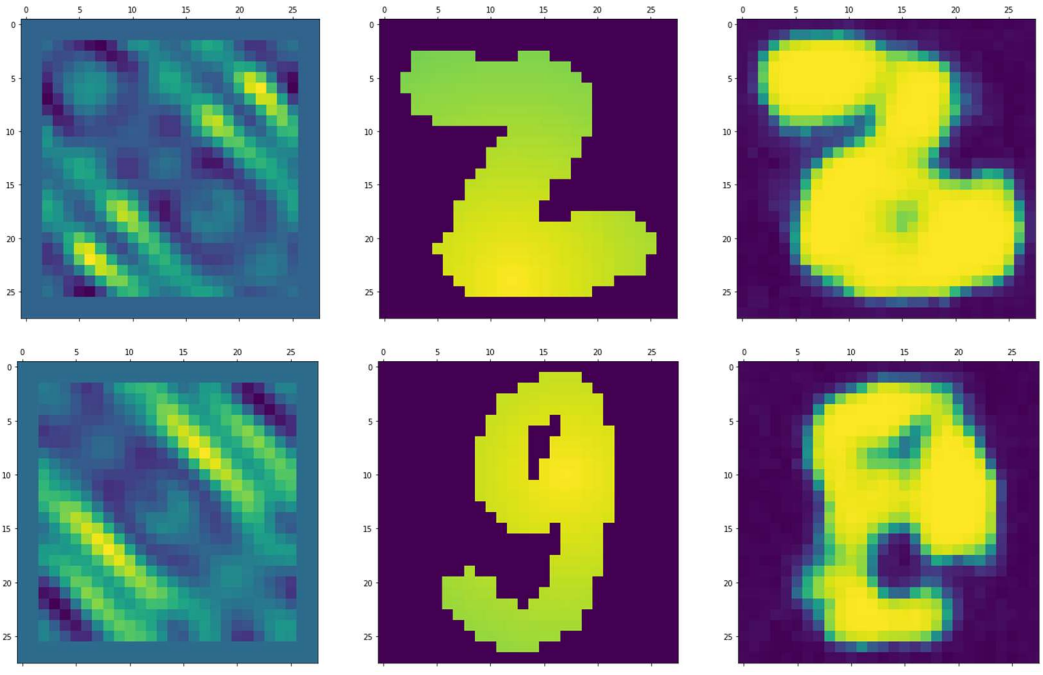

Fig 12: Padded field (left), true target (center) and reconstructed target (right) from model 1 with randomly added permittivity

For model 2, the accuracy score of 84.28% for the field images with fixed target permittivity (sample results in Fig 13) but only 56.79% for the field images with randomly added target permittivity (sample results in Fig 14). The predicted images have a lot of noise, especially around the border of the target. This applies to both types of testing datasets (fixed and non-fixed permittivity).

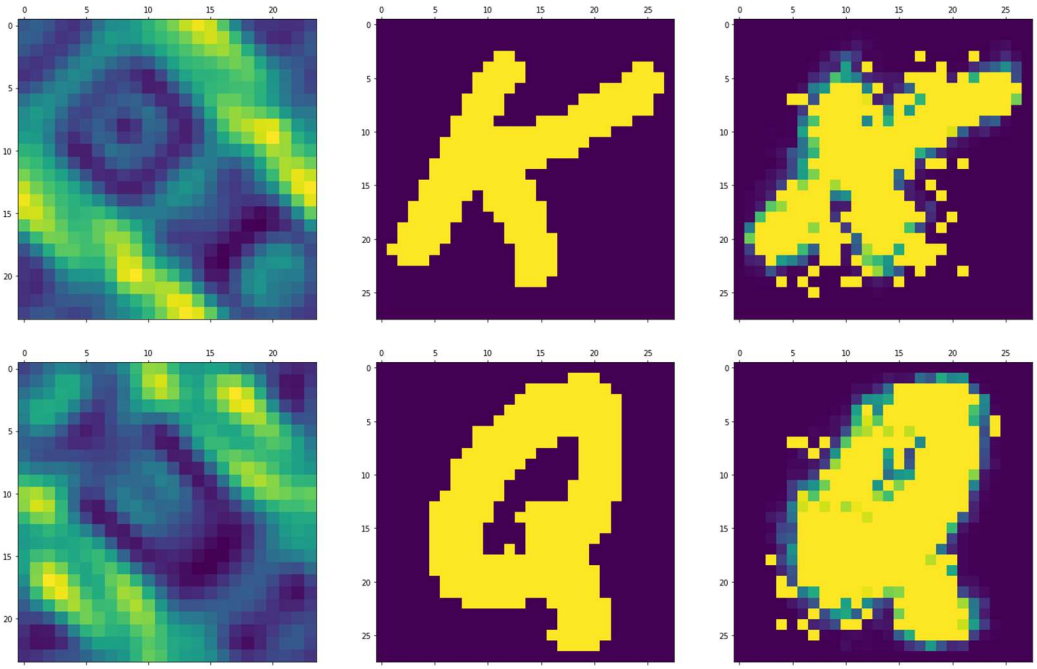

Fig 13: Field (left), true target (center) and reconstructed target (right) from model 2 with fixed permittivity

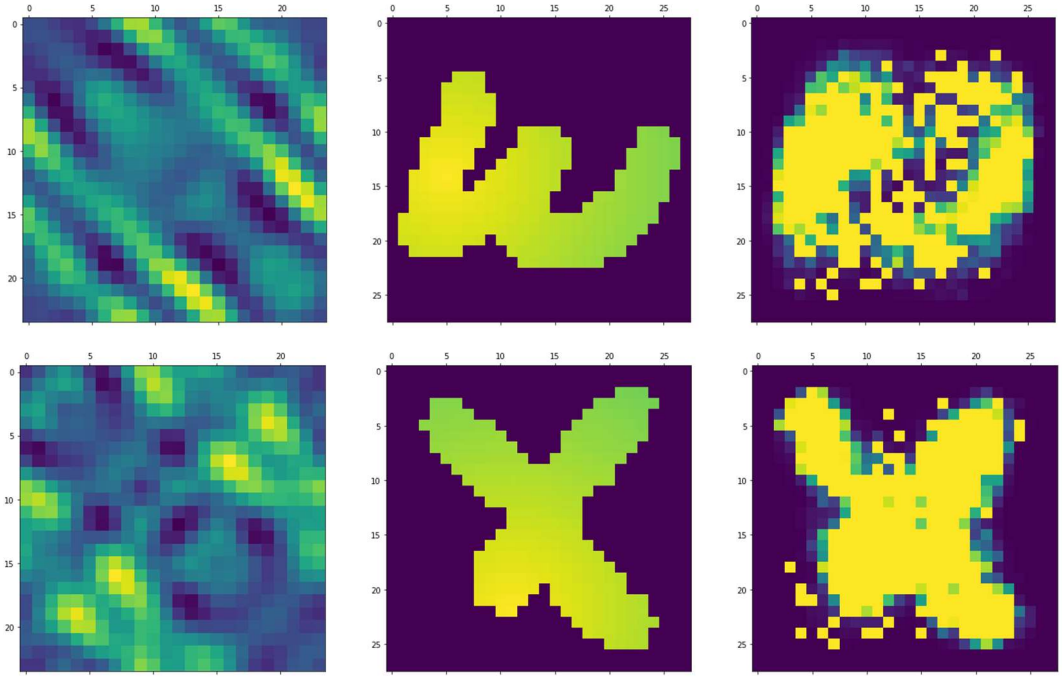

Fig 14: Padded field (left), true target (center) and reconstructed target (right) from model 2 with randomly added permittivity

V. Conclusion:

This project aimed to explore the microwave-imaging dataset with the help of machine learning. The two datasets of interest are generated using the Richmond Solver algorithm with shape targets and EMNIST character targets.

A k-means clustering algorithm was implemented in testing for similarity in the scattered field images of the same object or images created using the same frequency. The number of clusters parameter were set to 2, to hopefully separate the dataset into two cluster of 2 shape objects, and were set to 3, for three different frequencies. This algorithm was fitted using the field images of shape objects (circles, triangles and squares) generated by three frequencies (1e-9, 2e-9 and 3e-9). When fitting for circle and square field images, we found that squares images with frequency 3e-9 are might look significantly different and images of two frequencies 1e-9 and 3e-9 might not share similar features. But low Silhouette scores of under 0.20 suggests that the algorithm could not produce a good separation of the dataset from just the scatter field and separate these two objects or the frequencies used. However, a better result was obtained when changing to circle and triangle field images and number of clusters of 2. We achieved clusters containing only triangle images generated using frequency of 1e-9 both when using 2 and 3 as the number of clusters. Even

though the score was higher for number of clusters of 2 compared to a value of 3, it was still quite low with a Silhouette score of approximately 0.3547. The number of images were only 700 which is quite small. For further work, we would like to generate more data for training or implement data augmentation. Moreover, a recent paper working with MRI image data suggested using mini batch K-Means clustering and CNN to create a brain tumor prediction system for detecting the tumor disease [6]. Thus, exploring with deep clustering network is another possible step.

Two U-Net CNN applications were developed to deal with mismatch field and target image pixels (24x24x1 and 28x28x1, respectively). Both of them have the same contracting and expansive paths but one needs a zero-padding preprocessing step to change the input shape while the other one implements a Flatten layer, a Dense layer and a Reshape layer to change the output shape. These U-Net's were trained on 630 field and target images of EMNIST database targets generated using frequency of 1e-9 and a fixed target permittivity, a physical property of how a material responds to an electric field. With a fixed permittivity, this backscattering problem turns into a binary classification problem. The first U-Net achieves 88.77% testing accuracy while the second U-Net attains a slightly lower but still significant testing accuracy of 84.28%. The loss and accuracy were closely monitored during training and validation to avoid overfitting. Testing for field images with randomly added permittivity based on the location of the object was also carried out but failed to achieve considerable accuracy (approximately 57%) due to the specific choice of activation function and loss function for binary classification task. For further development, we would like to try different preprocessing techniques (for instance, copying pixels that are close to the edge of the original input field), continue hyperparameter tuning, modify the current models in order to work with added permittivity (for instance, changing the activation and loss function) and explore other U-Net architectures.

Some limitations and challenges arose from understanding the data, the Richmond Solver algorithm and the physics aspects of the project (frequency, wavelength, permittivity, etc.) as well as the extensive data generation and model training time. However, they can all be overcome with having more time and possibly a more powerful computer.

VI. Acknowledgment:

I would like to express my sincere gratitude to the following individuals and organizations for their invaluable support throughout this capstone project. To the domain expert, Dr. Ian

Jeffreyy, my deepest thanks go to Dr. Jeffrey for their guidance, mentorship, and continuous encouragement throughout this project. Their expertise in data science and insightful feedback were instrumental in shaping this work and allowed me to learn and grow. To the course organizers and advisors, Dr. Margherita Ferrari, Dr. Vitalii Akimenko and Dr. Carson Leung, the seminars you organized really opened my eye to different industry applications of data science and your feedback helps shaping this project. To the Data Science program from the Faculty of Science, I am grateful to be a part of the program that provided such a stimulating and supportive learning environment and allowed me to develop my data science skills.

VII. References:

1. Z. Wei and X. Chen, "Deep-Learning Schemes for Full-Wave Nonlinear Inverse Scattering Problems," in IEEE Transactions on Geoscience and Remote Sensing, vol. 57, no. 4, pp. 1849-1860, April 2019, doi: 10.1109/TGRS.2018.2869221.
2. Center for Devices and Radiological Health. (2023, October 12). Microwave ovens. U.S. Food And Drug Administration. https://www.fda.gov/radiation-emitting-products/resources-you-radiation-emitting-products/microwave-ovens#:~:text=Microwaves%20are%20non%2Dionizing%20radiation,cause%20skin%20burns%20or%20cataracts
3. Hossain, A., Islam, M. T., Rahman, T., Chowdhury, M. E. H., Tahir, A., Kıranyaz, S., Mat, K., Beng, G. K., & Soliman, M. S. (2023). Brain Tumor Segmentation and Classification from Sensor-Based Portable Microwave Brain Imaging System Using Lightweight Deep Learning Models. Biosensors (Basel), 13(3), 302. https://doi.org/10.3390/bios13030302
4. Song, J., Shen, T., & Wang, Q. (2023). An image Post-Processing approach based on fully dense U-NET for microwave induced Thermo-Acoustic tomography. IEEE Journal of Electromagnetics, RF and Microwaves in Medicine and Biology, 7(1), 59–64. https://doi.org/10.1109/jerm.2022.3223806
5. Anjit, T. A., Benny, R., Cherian, P., & Mythili, A. P. (2021). NON-ITERATIVE MICROWAVE IMAGING SOLUTIONS FOR INVERSE PROBLEMS USING DEEP LEARNING. Progress in Electromagnetics Research M. Pier M, 102, 53–63. https://doi.org/10.2528/pierm21021304
6. Ganapathy, S., Thoidingjam, V., & Sen, A. K. (2024). A brain tumor prediction system for detecting the tumor disease using mini batch K-Means clustering and CNN. Multimedia Tools and Applications. https://doi.org/10.1007/s11042-024-18790-z
7. Islam, A., Kiourti, A., & Volakis, J. L. (2020). A novel method to mitigate Real–Imaginary image imbalance in microwave tomography. *IEEE Transactions on Biomedical Engineering (Print)*, *67*(5), 1328–1337. https://doi.org/10.1109/tbme.2019.2936125
8. Cohen, G., Afshar, S., Tapson, J., & van Schaik, A. (2017). EMNIST: an extension of MNIST to handwritten letters. Retrieved from http://arxiv.org/abs/1702.05373

9. Shiran, G., & Weinshall, D. (2019). Multi-Modal deep clustering: unsupervised partitioning of images. *arXiv (Cornell University)*. http://export.arxiv.org/pdf/1912.02678
10. K. Peng, V. C. M. Leung and Q. Huang, "Clustering Approach Based on Mini Batch Kmeans for Intrusion Detection System over Big Data", *IEEE Access*, vol. 6, pp. 11897-11906, March 2018.
11. . H. Jin, M. T. McCann, E. Froustey, and M. Unser, "Deep convolutional neural network for inverse problems in imaging," IEEE Transactionson Image Processing, vol. 26, no. 9, pp. 4509–4522, 2017.
12. Johnson, J., Alahi, A., & Li, F. (2016). Perceptual losses for Real-Time style transfer and Super-Resolution. In Lecture Notes in Computer Science (pp. 694–711). https://doi.org/10.1007/978-3-319-46475-6_43
13. Ioffe, S., & Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv (Cornell University). http://export.arxiv.org/pdf/1502.03167
14. Yaqub, M., Feng, J., Zia, M., Arshid, K., Jia, K., Rehman, Z. U., & Mehmood, A. (2020). State-of-the-Art CNN Optimizer for brain tumor segmentation in magnetic resonance images. Brain Sciences, 10(7), 427. https://doi.org/10.3390/brainsci10070427
15. Irfan, D., Gunawan, T. S., & Wanayumini, W. (2023). Comparison of SGD, RMSPROP, and ADAM Optimation in animal classification using CNNs. Proceeding International Conference on Information Science and Technology Innovation, 2(1), 45–51. https://doi.org/10.35842/icostec.v2i1.35