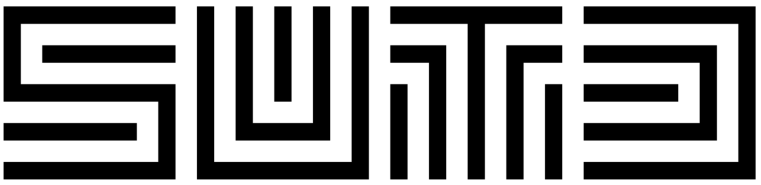


SINGAPORE UNIVERSITY OF TECHNOLOGY AND DESIGN

50.039 Theory and Practice of Deep Learning

Breast Ultrasound Image Segmentation Project Report

Group 22

Name	Student ID
Isaac Tay Eng Hian	1006327
Vidhi Mahajan	1006157
Natalie Ang Zi Yi	1006131

Table of Contents

1. Introduction

Early detection of breast tumours is crucial for successful treatment of breast cancer. The objective of our project is to create a semantic image segmentation model that automatically analyses breast ultrasound images, segmenting the image into tumour and non-tumour regions. The tumour regions are highlighted in the mask. This is presented here as a binary segmentation problem, where each pixel in the image is classified as either belonging to a tumour region or the background.

2. Data Preparation

2.1 Dataset

The dataset [1] was retrieved from a kaggle source [2] (“Breast Ultrasound Images Dataset”) and was collected in 2018. The data collected at baseline consists of greyscale breast ultrasound images of 600 female patients aged between 25 and 75 years old. It contains a total of 1560 images – 780 grayscale ultrasound samples, along with 780 corresponding black and white ground truth (mask) images. All images are formatted in PNG and are categorised into 3 classes: normal (133 samples), benign (437 samples) and malignant (210 samples).

While noting that the dataset is relatively small for an image segmentation task, we assessed that it was a reasonable size for the task at hand, binary image segmentation.

2.2 Pre-processing

Different image pre-processing techniques were implemented to counter the potential issue of having insufficient samples, in order to enhance the data quality and introduce variability for optimised model training. Special attention was paid to the data augmentation to artificially increase data diversity and to avoid possible overfitting.

2.2.1 Data Augmentation and Transformation

A series of v2 image transformations were implemented as follows:

- **Grayscale conversion**: convert images to grayscale with a single output channel
- **Image conversion**: images converted to PIL format to facilitate subsequent transformations
- **Colour jitter**: make random adjustments to brightness, contrast, saturation and hue
- **Random resized crop**: random cropping and resizing to a uniform size of 512x512 pixels while maintaining aspect ratio
- **Random flipping**: apply random horizontal and vertical flips with 50% probability
- **Datatype conversion**: images were converted to floats and scaled for model training

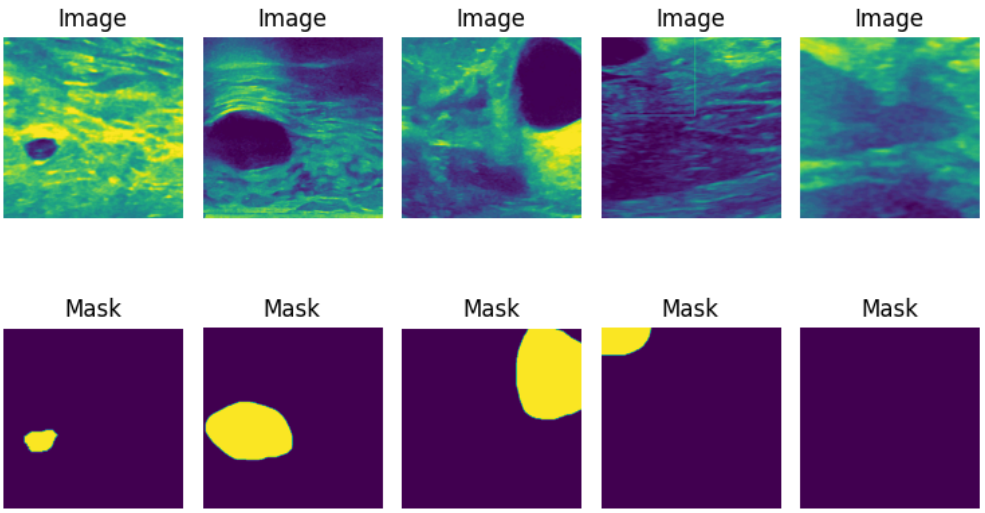


Figure 1: Five transformed ultrasound samples and corresponding mask images

3. Methodology

3.1 Data Splitting

For training, the dataset was split randomly in a 8:1:1 ratio corresponding to the training, validation and test sets, with a fixed seed. Dataloader objects were created for each set, with a shuffle being applied to the training set only. The training data loader had a batch size of 4, while the validation and test sets had a batch size of 32.

3.2 Loss Function and Optimizer

For the loss function, we used a composite loss function that combines two different losses: **Binary Cross-Entropy with Logits Loss** and **Dice Loss**. The first provides a smooth gradient for optimisation of binary classification tasks, while the latter measures the overlap between the predicted and ground truth masks. This combination would leverage the advantages that come with both approaches. Both loss functions were calculated as shown below:

$$Dice\ Loss\ =\ \frac{1-2\cdot intersection}{union}$$

$$BCEWithLogitsLoss(y_{pred},\ y_{true})\ =\ -\frac{1}{N}\Sigma^{N}_{i=1}\ [y_{true,\ i}\ log(y_{pred,i})\ +\ (1-y_{true,i})$$

To achieve efficient neural network training, we decided to use the **Adam optimizer**, after evaluating its effectiveness against other common optimizers.

3.3 Evaluation Metrics

To assess and monitor the training process, 3 evaluation metrics were used and recorded throughout the training epochs. After training, their average values on the test set samples were calculated to evaluate its performance and generalisation to unseen values. All 3 metrics range between 0 to 1, where the value being close to 1 is optimal. The metrics consist of the following:

a) Pixel F1

$$Pixel\ F1\ = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

The pixel f1 measures a balance between the precision and recall if we treat the segmentation as a binary classification task. This provides a more balanced view of the models performance compared to accuracy as the classes are imbalanced for the segmentation task.

b) Intersection over Union (IoU)

$$IoU\ = \frac{Intersection}{Union}$$

The IoU measures the overlap between the output and ground truth, specifically the ratio of the intersected area of the predicted mask and ground truth mask to their combined area. It takes into account both true positives (correctly classified tumour pixels) and false positives/ negatives (incorrect classifications), thus being an informative evaluation metric.

c) Dice Coefficient

$$Dice\ Coefficient\ = \frac{2 \times Intersection}{Sum\ of\ pixels\ in\ predicted\ and\ ground\ truth\ masks}$$

The Dice coefficient measures the similarity between the predicted mask and ground truth mask. It accounts for both precision and recall. As we are working with an imbalance dataset (background class dominates), this metric is useful as it penalises both false positives and false negatives equally.

4. Model

4.1 Model Architecture

The model is defined as a subclass of the PyTorch nn.Module class. It contains two main components, an encoder and a decoder, that are both sequential containers of layers.

Encoder Component

The encoder consists of a series of convolutional layers with a combination of interleaved ELU activation functions, batch normalisation layers, and dropout layers (with probability 0.5). This aims to progressively downsample the input images, reducing its spatial dimensions while increasing the depth of the features.

The encoder can be further broken down into 3 blocks:
Block 1: Transforms the input image from 1 to 64 channels with downsampling
Block 2: Processes the output of block 1, downsamples and increases depth to 128 channels
Block 3: Further processes output of block 2, downsamples and increases depth to 256 channels

Decoder Component

The decoder component traditionally reverses the process of the encoder, progressively upsampling features maps to their original spatial dimensions while simultaneously reducing the number of channels. However, our model instead uses 3 layers to progressively compress the channels, before doing a final upscale using bilinear interpolation. We chose this implementation as it is much more computationally efficient, while only losing minor accuracies in the exact edges of the segment. This allows the model to focus more on the detection rather than finer shape details, and allow for better generalisation.

Block 1: Compresses features, from 256 to 128 channels
Block 2: Compresses features from 128 to 64 channels
Block 3: Compresses features from 64 to 32 channels
Final Block: Converts features from 32 to 1 channel (the original input dimension), for single class prediction, using bilinear interpolation

4.2 Training

As a relatively simple model, training per epoch was efficient but it took several training epochs to begin seeing convergence. The model was trained for 2000 epochs on the training set while being evaluated on the validation set. The model appeared to start overfitting at epoch 1600 (approximately) where validation values started to dip despite the training values beginning to accelerate. The validation loss also starts to increase around the same time.

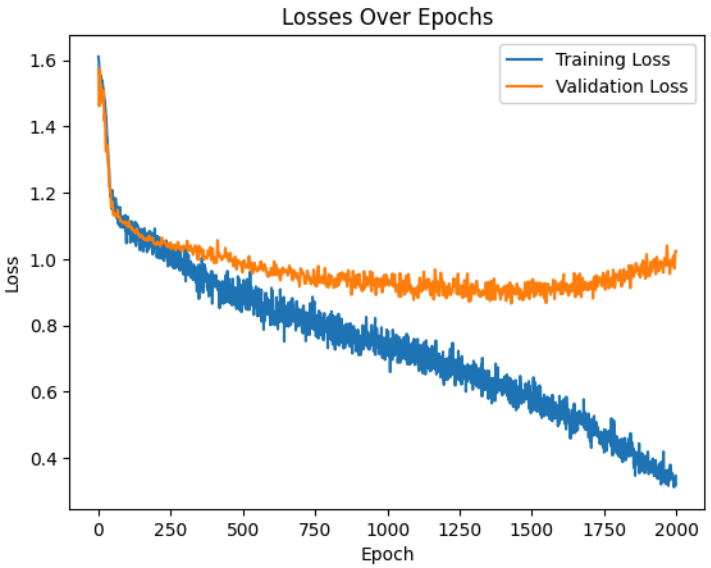


Figure 2: Training and validation losses plot

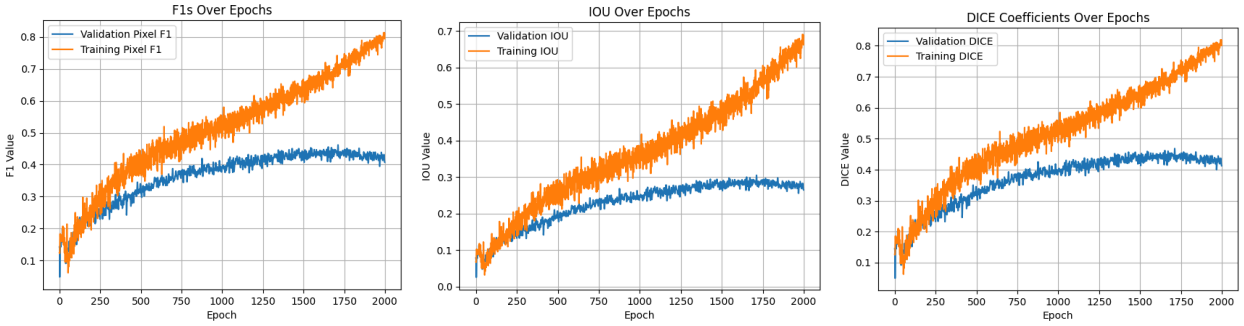


Figure 3: Comparing training and validation metrics

All 3 evaluation metrics can be seen to follow very similar trends. More relevant graphs can be found in the Appendix (section 10.1).

5. Results and Discussion

5.1 Model Results

Average evaluation metrics on test dataset:

Metric	Average Value
Pixel F1	0.8949948648496733
IoU	0.8150839455947413
DICE Coefficient	0.9039819517449006

Interestingly, the metric values on the test set were substantially higher than the maximum validation values observed during training.
Breaking down these values, both the average pixel F1 score and DICE Coefficient are relatively high at ~90%, indicating the model's ability to correctly classify the pixels and to produce masks with relatively high similarity to the ground truth masks. The average IoU score is about 9% lower at 81.5%, but remains quite high.

Some simple visualisations showed that tumour localizations were generally well found, but the shapes of the tumours had different extents of inaccuracies.

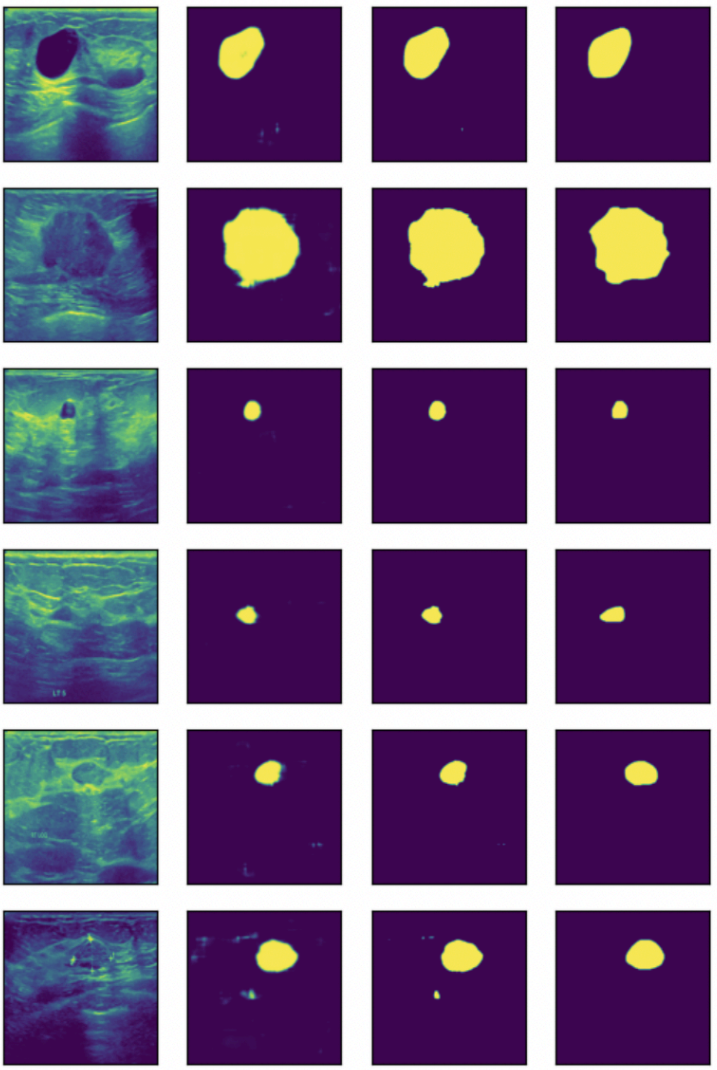


Figure 4: Input image, predicted output, binary output and ground truth mask for 6 test samples for model

More comparisons of the output and ground truth masks can be found in the Appendix (section 10.2).

Based on these results, we evaluated that the model was relatively promising and robust in distinguishing tumour from non-tumour pixels.

5.2 Limitations

The model was observed to perform less well on larger tumours and tumours with more complex shapes, and would often perform the segmentation incompletely or in inaccurate shapes.

This could be due to several reasons. Since these tumours may exhibit more complex shapes and boundaries compared to the smaller tumours and tumours with less complex shapes, the model architecture may not be sufficiently complex or lacks the capacity to handle the variations in these samples. There is also a possibility that there is not enough representation of these tumours in the dataset.

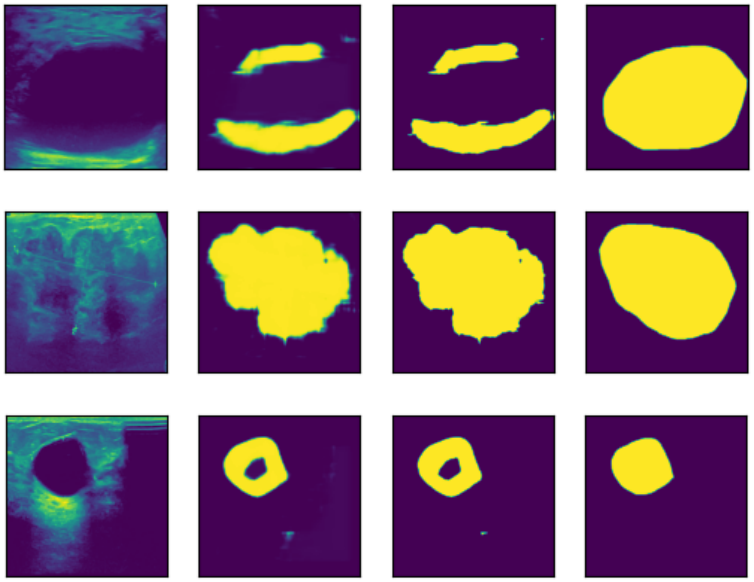


Figure 5: Selected samples with incomplete or inaccurate segmentation

It is also noted that our model results are still far from qualifying for professional medical standards. There are potential areas for improvement if given a larger dataset to work with, and with further fine tuning to the training parameters and model's architecture.

5.3 State-of-the-art Models

A few commonly used state-of-the-art image segmentation models were tested to provide a brief comparison to our model's performance.

5.3.1 U-Net

The default UNet model involves a U-shaped architecture with contracting encoder and expansive decoder pathways. The encoder captures information by downsampling the image through repeated convolutional layers. The decoder relies on skip connections and directly feeds high-resolution features from the encoder to corresponding decoder levels,

thus preserving spatial information that is important for accurate segmentation (especially of boundaries).

A UNet model from a public source [3] with no modifications was trained for 200 epochs for a brief comparison to our model. The evaluation metrics on the test dataset are as follows:

Test evaluation metrics after 200 epochs

Metric	Average Value
Pixel F1	0.6783902473003808
IoU	0.5247418363880215
DICE Coefficient	0.6835218094473774

Briefly comparing this to our model performance, while convergence for UNet occurred much earlier in training than our model, the validation values also plateaued early and more instability was observed. This could possibly be mitigated with some countermeasures. Visualisation shows that the segmentation is relatively effective but would perform poorly with larger tumours (similar to our model), and occasionally neglect large areas.

More relevant figures for the U-Net can be found in the Appendix (section 10.3).

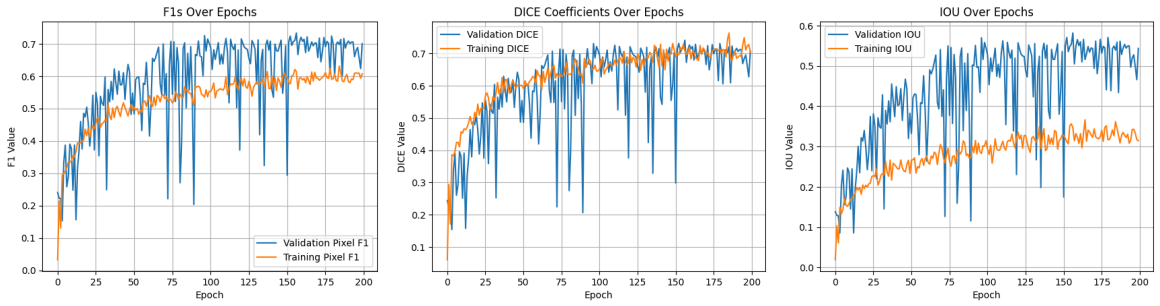
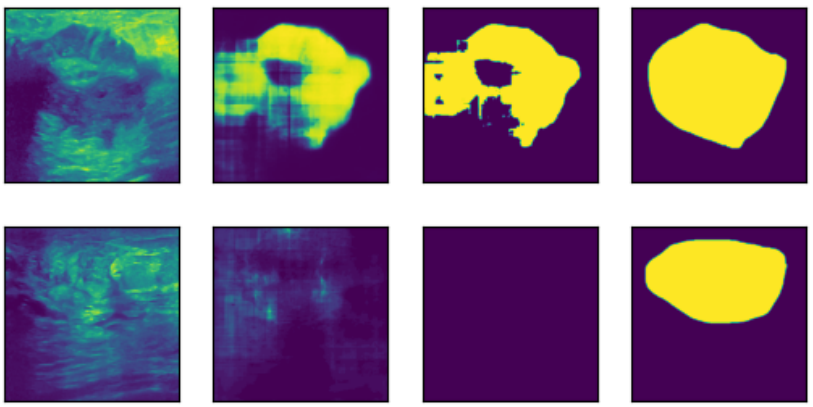


Figure 6: Comparing validation and training metrics for U-Net



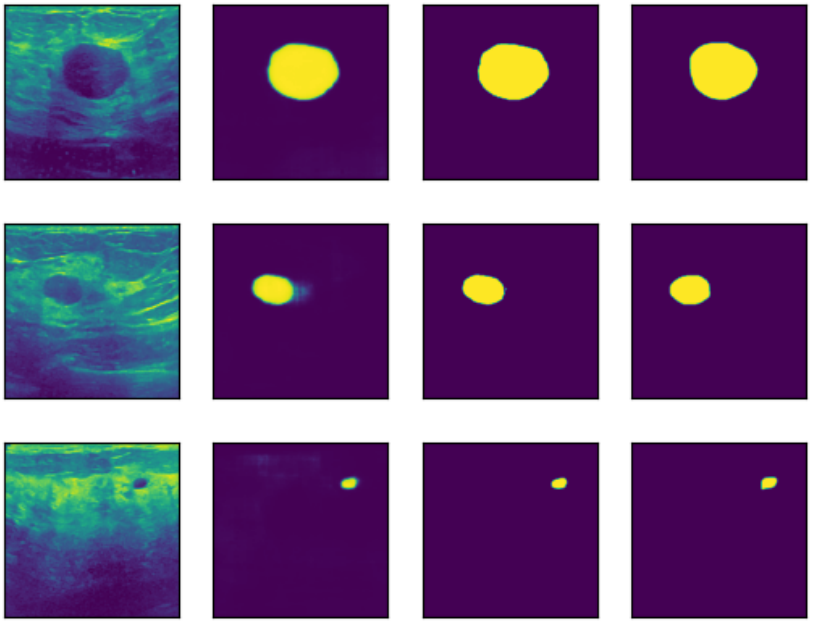


Figure 7: Input image, predicted output, binary output and ground truth mask for 5 test samples for U-Net

5.3.2 U-Net++

While sharing a similar overall architecture with U-Net, U-Net++ was chosen as it is an extension of U-Net that supposedly improves segmentation performance by introducing nested skip pathways and better feature aggregation [4]. Therefore, it is projected to have a higher potential to capture richer contextual information and is expected to possibly outperform U-Net.

The pre-built U-Net++ from the segmentation_models_pytorch library was used with a pre-trained ResNet34 encoder. The final layer was modified to match our output channels.

Test evaluation metrics after 200 epochs

Metric	Average Value
Pixel F1	0.754655526607476
IoU	0.6292740606195765
DICE Coefficient	0.7614706063783977

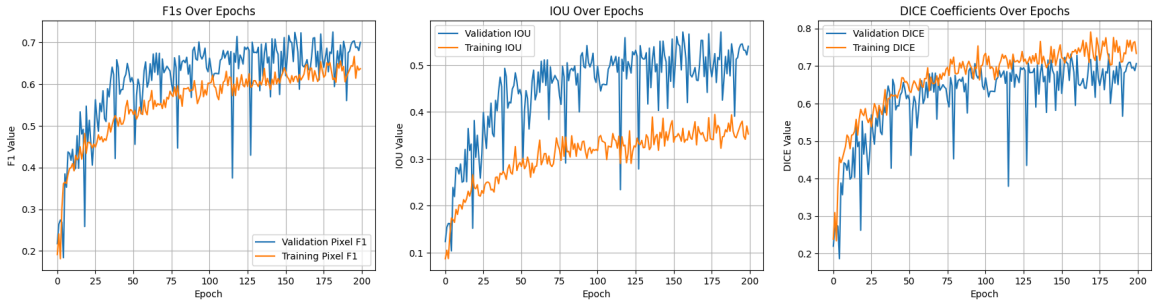


Figure 8: Comparing validation and training metrics for U-Net++

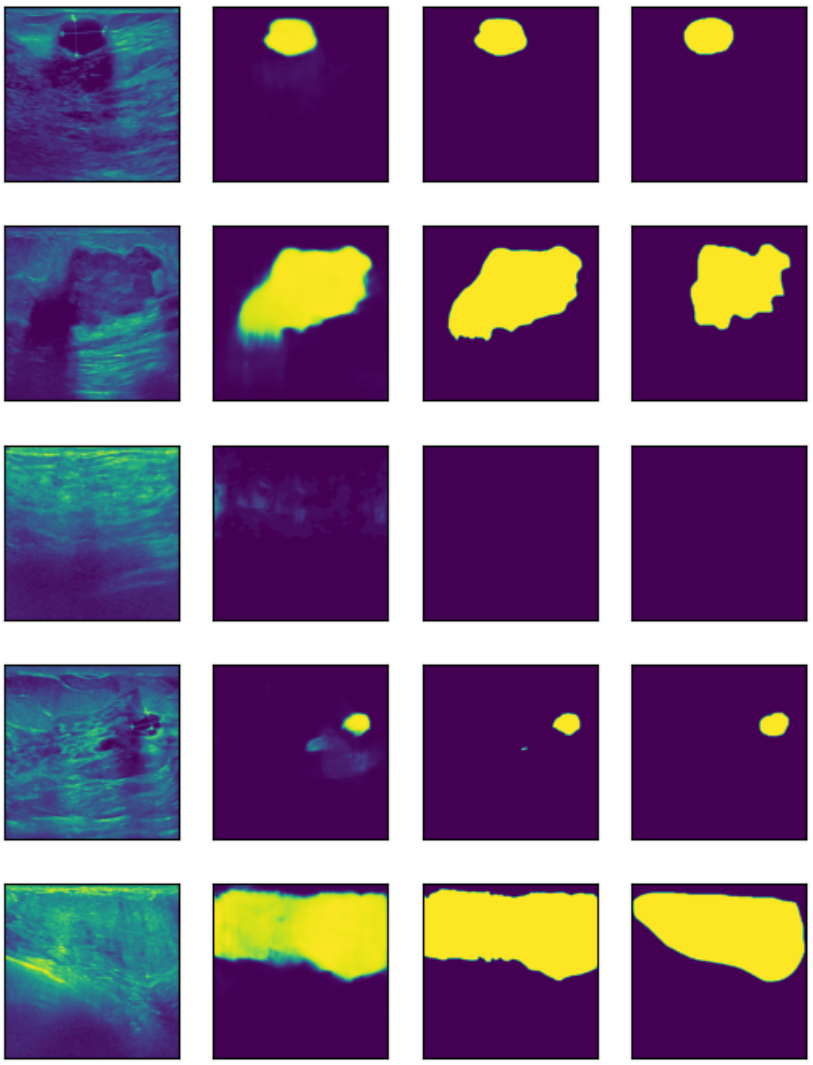


Figure 9: Input image, predicted output, binary output and ground truth mask for 5 test samples for U-Net++

More relevant figures for the U-Net++ can be found in the Appendix (section 10.4).

In comparison to our model, the U-Net++ model reached convergence much earlier for both training and validation datasets. However, the U-Net++ model validation dataset experienced significant fluctuations, while our model validation dataset experienced milder fluctuations.

Upon observation of the visualisations, it can be noted that segmentation is more effective for smaller and non tumour areas (similar to our model). For larger tumour areas, the U-Net++ model seems to predict a tumour area that is larger than the tumour area in the ground truth mask.

6. Conclusion / Future Work

Our semantic image segmentation model developed for breast tumour detection in ultrasound images showed promising results, with relatively high pixel F1, IoU and DICE coefficient values, despite using a relatively simple model and a small-sized dataset, which is a considerably positive outcome. However, the model faced challenges in handling larger and complex-shaped tumours and occasionally produced incomplete or inaccurate segmentations, therefore indicating there is room for improvement in the model's performance.

For future work, we could explore and experiment with different model architectures such as Attention U-Net, fully convolutional networks (FCNs) and other segmentation models to find the one best suited for the task. Performing cross-validation across different data subsets would also help improve model stability and reliability. Through continued research and development, the goal of creating more accurate and reliable models for early detection of breast tumours can be achieved.

7. Code Instructions

The code and relevant files can be found at our public GitHub repository:
https://github.com/IsaacTay/Segmentation_Project

Additional checkpoints too large for github can be found in:
https://drive.google.com/drive/folders/1xmHW979pR3a5vmzKegH904Yle15j1a0G?usp=sharing

7.1 Setup

The following packages are required for the notebooks to run:
- pytorch >= 2.0
- torchvision
- numpy
- scikit-learn
- pillow
- matplotlib
- tqdm
- icecream
- jupyter (for the notebooks)

7.2 Recreating the Model & Figures

Notebooks:
- `Final_Model_2000epochs.ipynb`.
- `UNet_SOTA.ipynb`
- `UNetPlusPlus.ipynb`

All notebooks are self-contained. They will contain all the training & figure creation code for their respective model. The visualised image transformations can also be recreated by running any of the notebooks.

8. Workload Distribution

Isaac – Data processing, model architecture, model training, report
Vidhi – State-of-the-art models, model training, report
Natalie – Data visualisation, evaluation & metrics, report

9. References

[1] Al-Dhabyani W, Gomaa M, Khaled H, Fahmy A. Dataset of breast ultrasound images. Data in Brief. 2020 Feb;28:104863. DOI: 10.1016/j.dib.2019.104863.

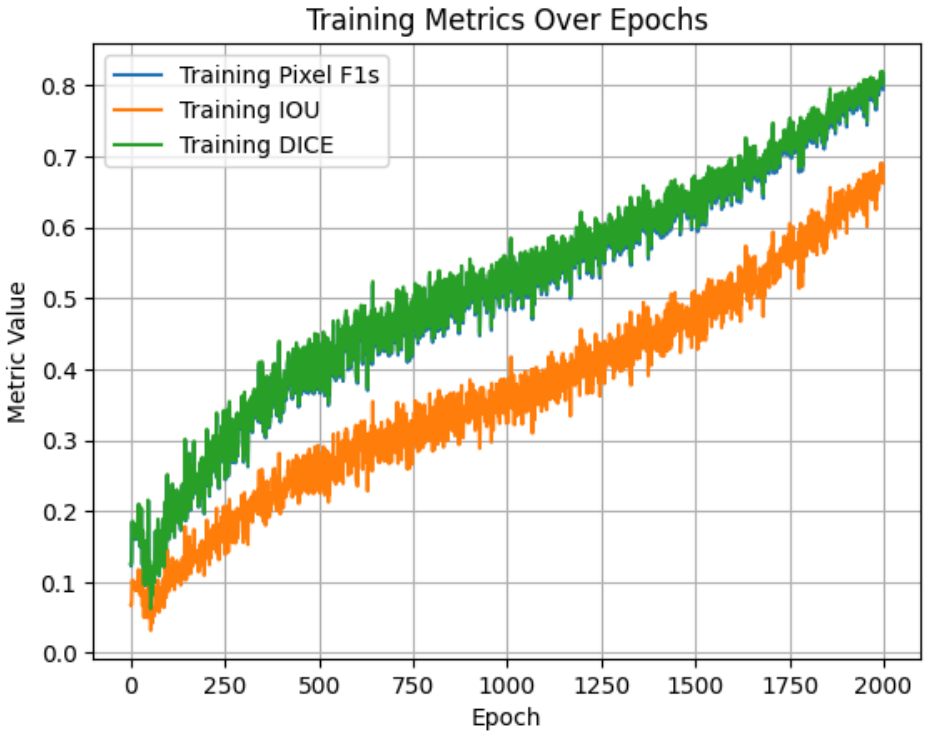
[2] Arya Shah, “Breast Ultrasound Images Dataset.” *Kaggle*, https://www.kaggle.com/datasets/aryashah2k/breast-ultrasound-images-dataset.

[3] Aladdin Person, Machine-Learning-Collection, (2023), GitHub repository, https://github.com/aladdinpersson/Machine-Learning-Collection/blob/master/ML/Pytorch/image_segmentation/semantic_segmentation_unet/model.py

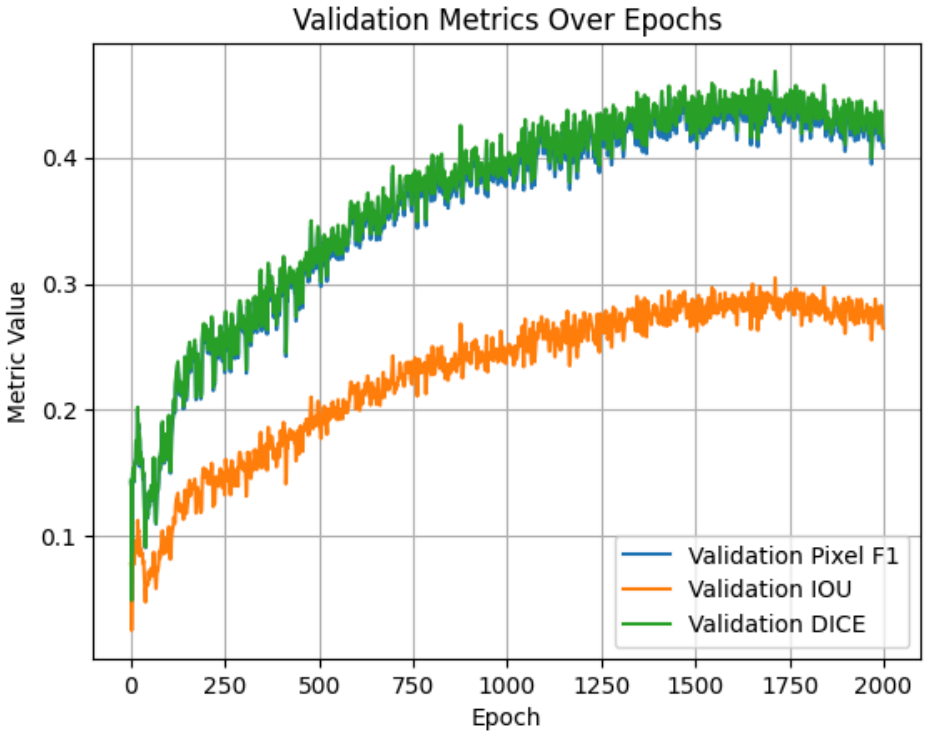
[4] Hesaraki, S. (2023, October 18), *UNET++*, Medium, https://medium.com/@saba99/unet-443b429ae0be#:~:text=In%20summary%2C%20UNet%2B%2B%20is,vision%20tasks%2C%20especially%20semantic%20segmentation

10. Appendix

10.1 Model Training Graphs

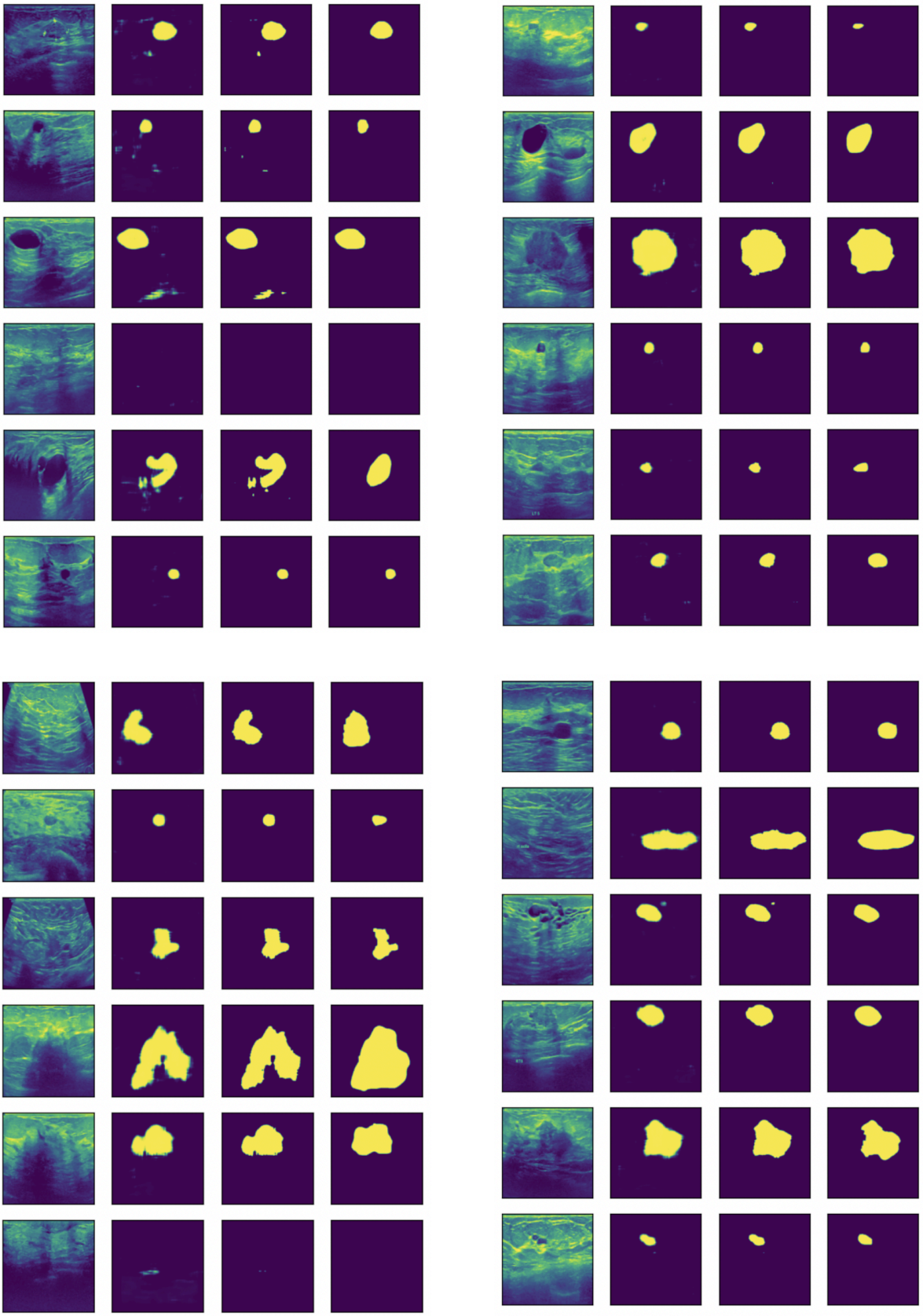


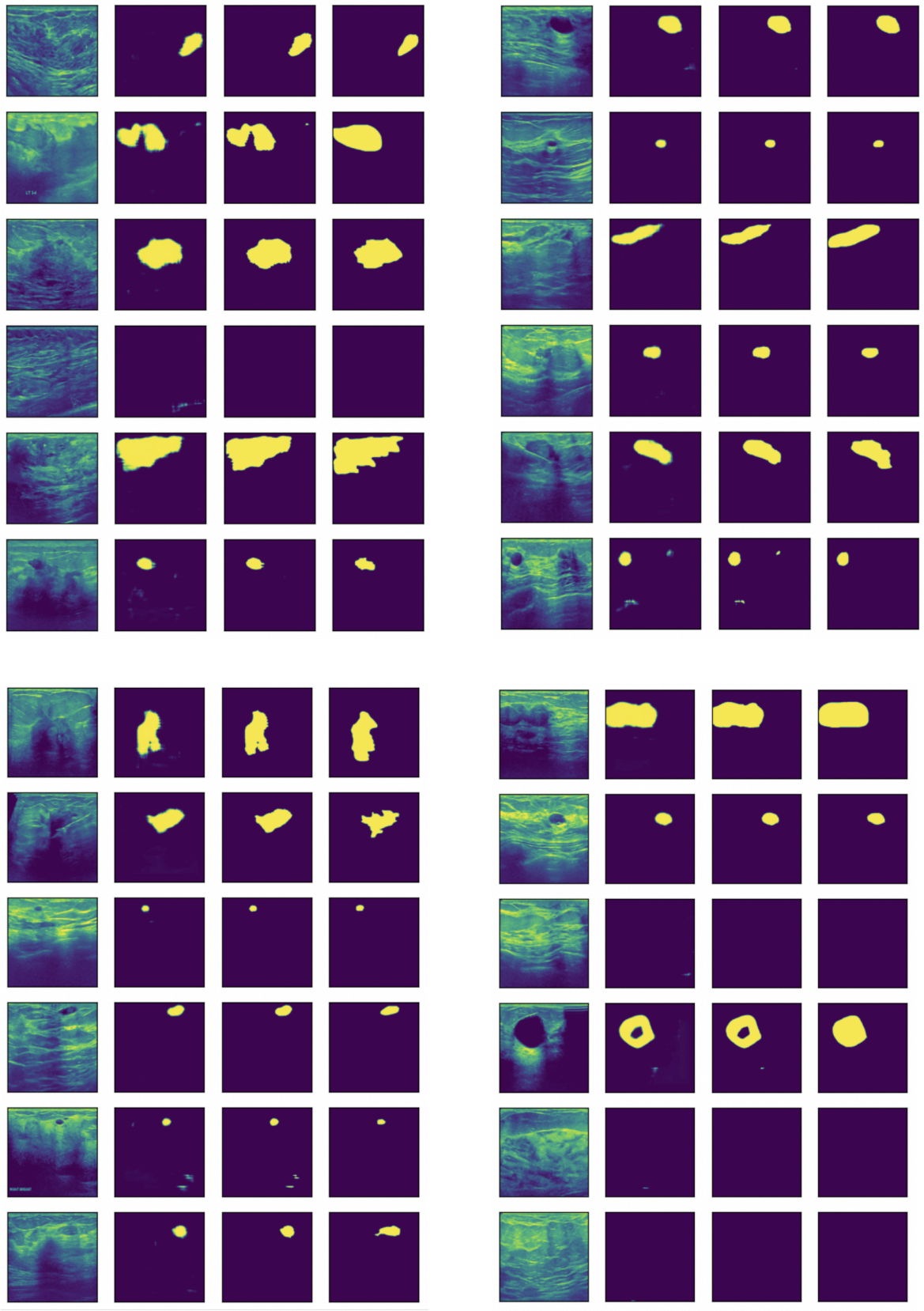
Training Metrics of Model Over Epochs



Validation Metrics Over Epochs

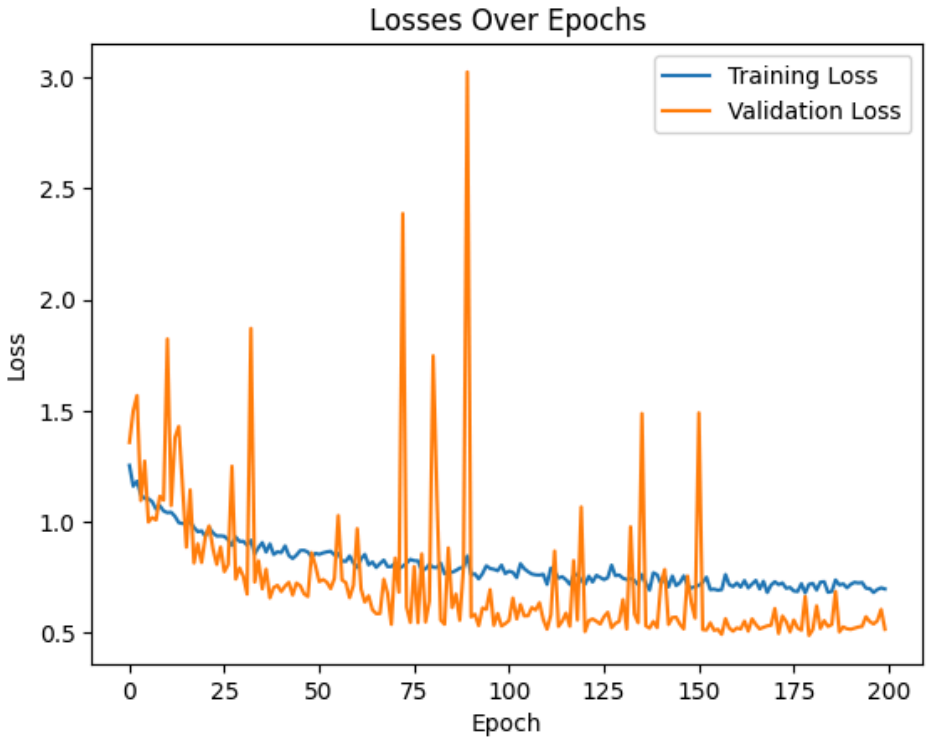
10.2 Model Predicted Output Comparisons





Comparisons between transformed input, predicted output, binary output and ground truth images (48 random samples)

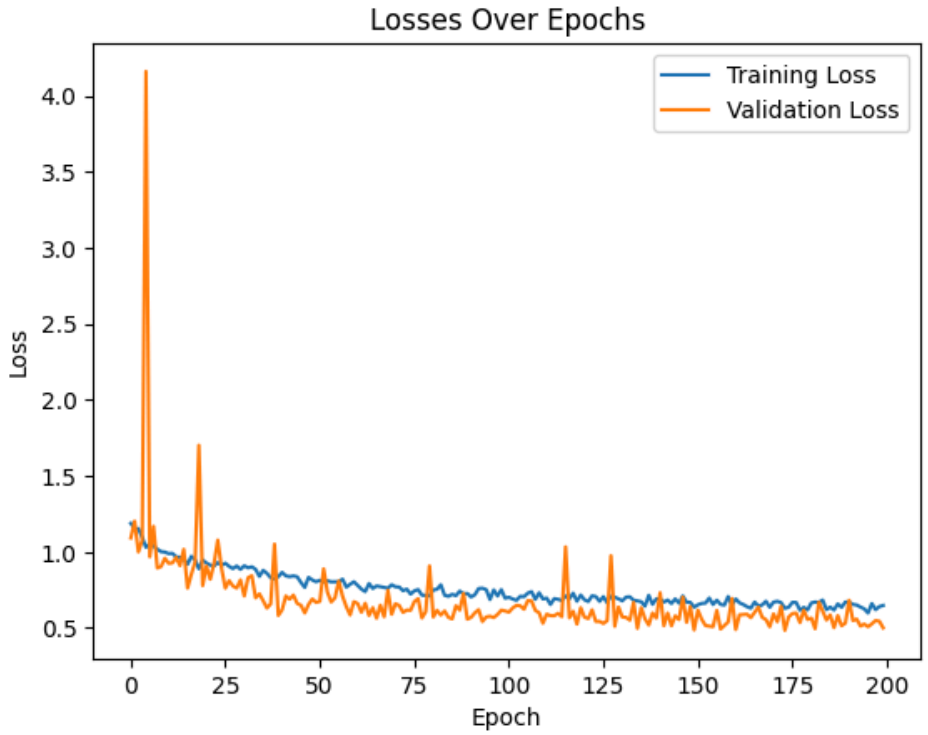
10.3 U-Net Training Graphs



Training and Validation Losses Over Epochs

Training Metrics Over Epochs

10.4 U-Net++ Training Graphs



Training and Validation Losses Over Epochs

Training Metrics Over Epochs