**DEFECT DETECTION**

**U-net Architecture**

We perform the task of defect detection through segmentation i.e., we label every pixel of the input as no defect or defect of a specific type. To perform image segmentation, we use the **U-net** architecture. This architecture was first proposed for biomedical image segmentation. The main idea is to supplement a usual contracting network by successive layers, where pooling operations are replaced by up-sampling operators. Hence these layers increase the resolution of the output. It is also a popular architecture in a class of DNNs called Fully-Convolutional Nets (FCNs).

The FCNs or U-net comprises of 2 parts:

* The first part is similar to image classification CNNs, that have a sequence of CONV/ReLU/MaxPool layers

* In the second part, instead of classification layers, transpose convolution is performed to gradually upsize the features and get the final segment map for each class. Transpose convolution also uses feature from the first part of the model when up-sampling the image in order to preserve the resolution.

This gives the U-shape to the architecture.

**Model Architecture specifics**

The U-net model used for this comprises of 33 layers of which, 23 are convolution (Conv2D) layers, 5 are Maxpool layers and 5 are transpose convolution (Conv2DTranspose) layers. The total number of trainable parameters int the network is 600,612.

The U-net model used for this project is shown below.

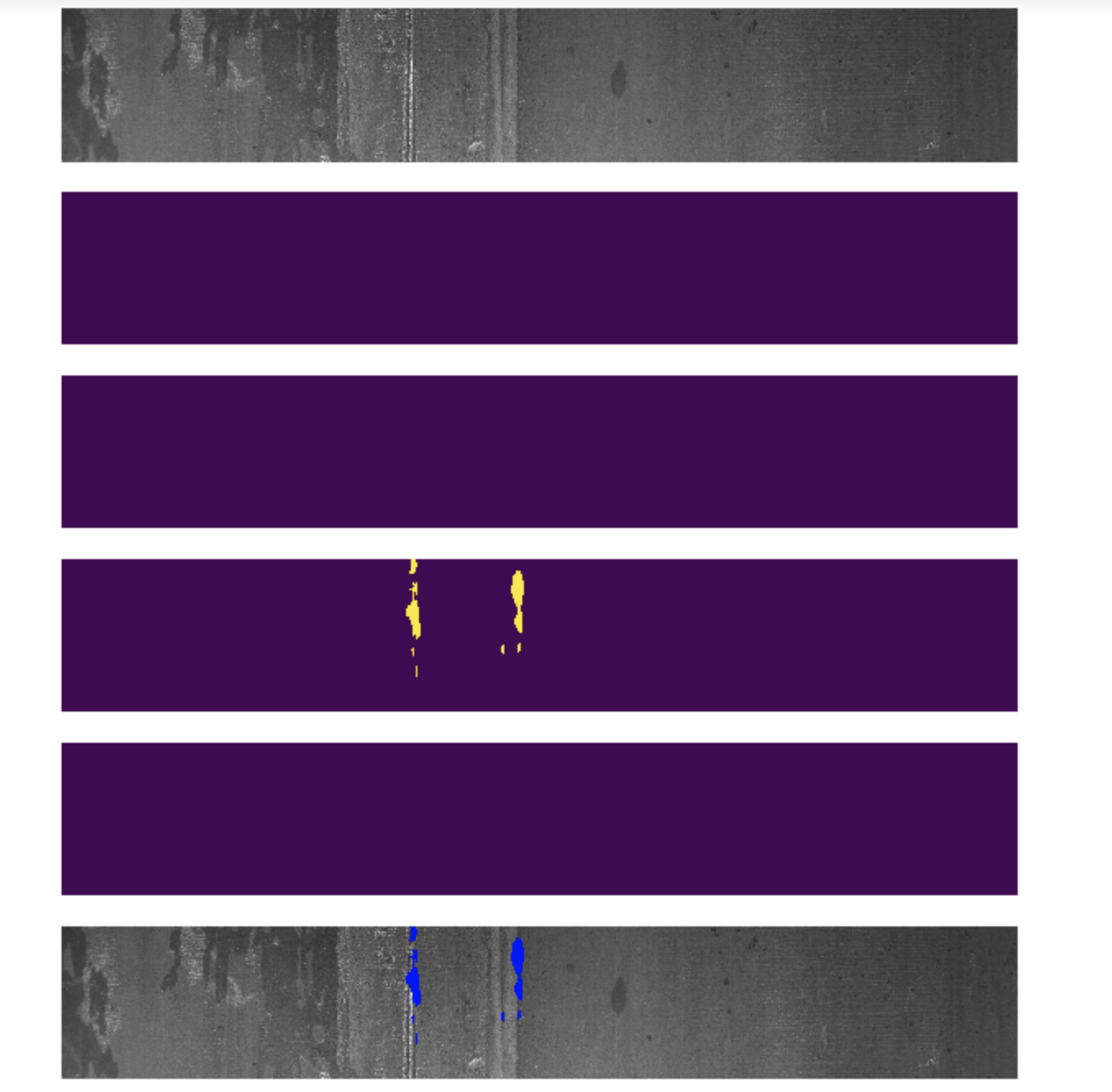
**U-net architecture**



We pass an image of size 256\*1600 with 3 channels as the input to the model. The annotations on the side shows how the image size decreases and the number channels increases as we pass through the maxpool layers. In the second half of the network, the image is up-sampled by using transpose convolution and we finally get an output segmentation map for each image with 4 channels, one for each defect type.

**RESULTS**

The output segment maps from the model for a particular image is shown below.



Original image

Segment map- type 1

Segment map- type 2

Segment map- type 3

Segment map- type 4

Final Output

From the segment maps we can understand that, this image has a defect of type 3. The defect region is masked using a masking function based on the location of defect given to us. The final output is formed by super-imposing the 4 segment maps on top of the original image and color coding the defects based on its defect type.

The training and validation loss of the model is showing below as a graph. The training loss and validation loss decreases as the number of epochs increase. The graph shows the values for the losses for 5 epochs.

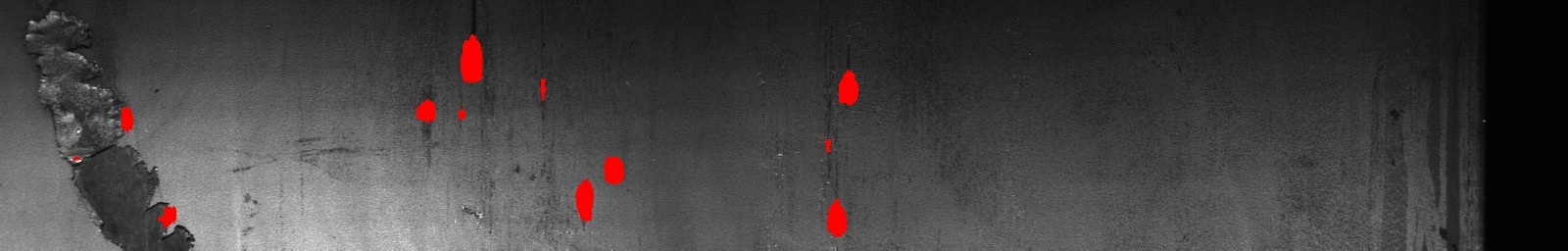
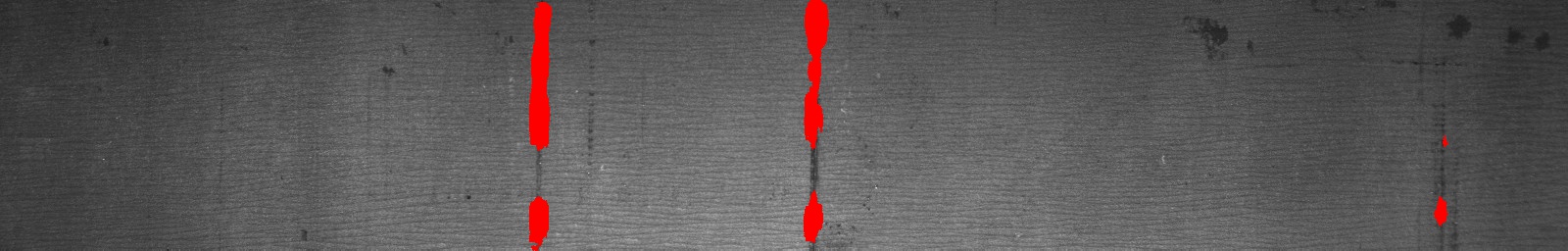
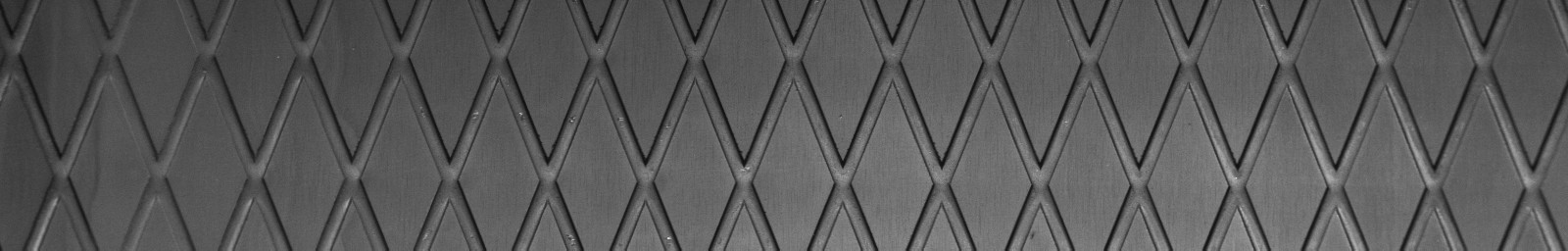
Training vs. Validation Loss

Loss

Epochs

**OUTPUT**

Our model was able to clearly distinguish between images with no defects and images with defects. For the images with defects, the model was able to detect the defect regions well, but it classified the defect as defect of type 3.



No defect

Defect Images

This is mainly because of the class imbalance in the dataset. I have color coded the defect classes as Red for type 3, green for type 4, blue for type 1 and yellow for type 2. Few output samples from the predictions on the test dataset is shown below.

Since the model was taking very long time to train, we could not try training the U-net model with the augmented images. Instead, we used a subset of the data with more balanced images in each class and trained the u-net model. The model little better results but again the model was biased towards the defect class with more number of images.