Model Training & Testing Report

McDefect Solutions

Object Detection (RetinaNet with ResNet50 Backbone):

For detecting and segmenting out relevant components (by creating bounding boxes) from in image (or video), we choose to fine-tune the **RetinaNet** (with **ResNet50** Backbone) architecture on our custom dataset(s). This is a transfer learning based approach, where we instantiate & fine-tune the RetinaNet model for catering to the requirements of every different industry that we plan to collaborate with, based on the component images that we obtain from the industries. For the demo, we intend to fine-tune the model for detecting pump impellers in an image.

Dataset Preparation:

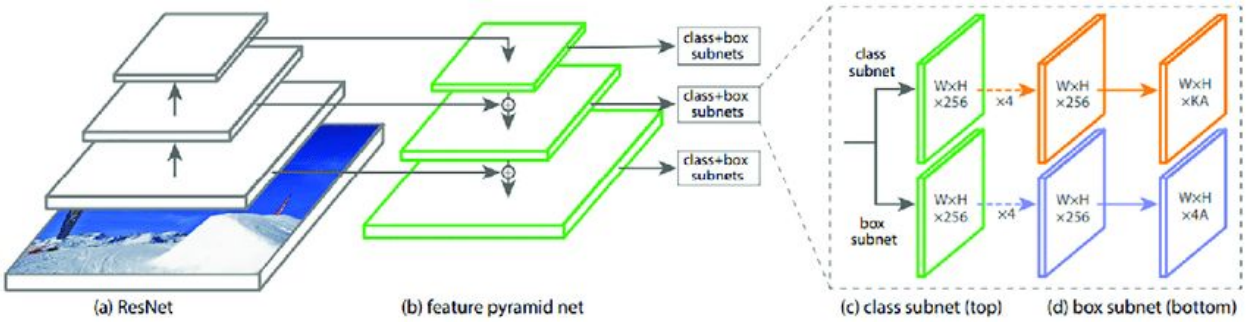
An augmented subset of Submersible Pump Impeller Defect Dataset (so as to include multiple impellers in a single image, for effective detection). A dataset of 12 images (these many are sufficient for object detection, since only 1 class is involved) is prepared, such that each image contains 1 or more pump impellers. All these images are annotated using the annotation tool labelmg.

This dataset is split into training & testing sets. For training, two CSV files are needed. The first one containing the *path*, *bounding box* and *class name* of each image (`train.csv`). The second file only contains the *class name* and their *mapping* (`class.csv`).

The CSV file with annotations contains one annotation per line. Images with multiple bounding boxes should use one row per bounding box. Note that indexing for pixel values starts at 0.

Model Architecture:

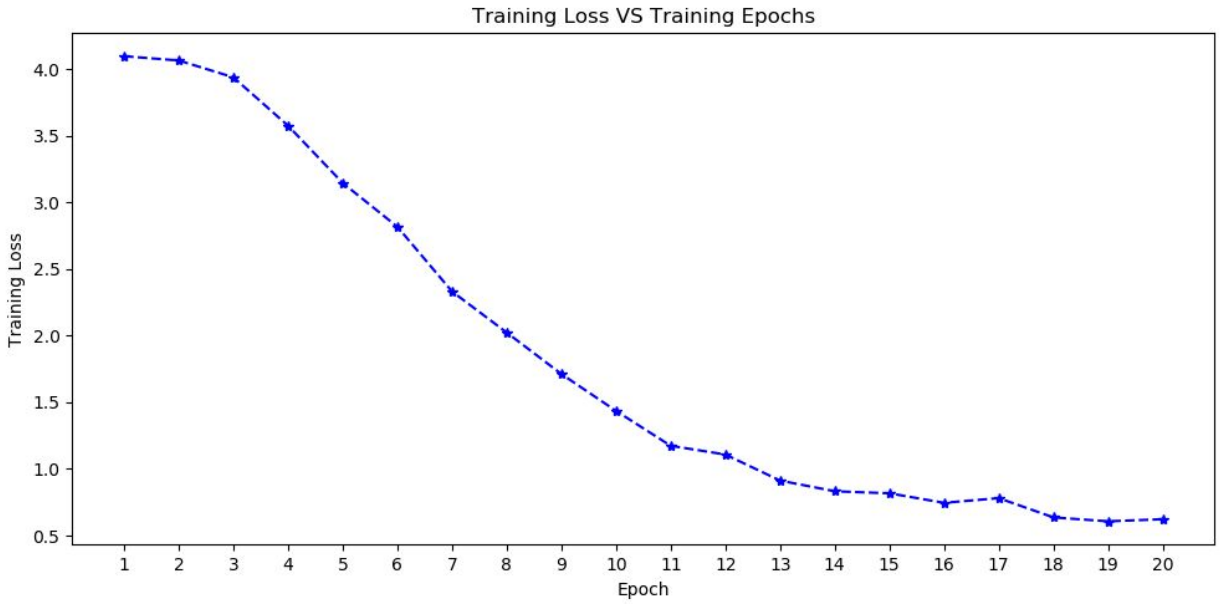
The architecture of the original RetinaNet model (with ResNet50 Backbone) is shown:



We use our dataset prepared above to fine-tune this model

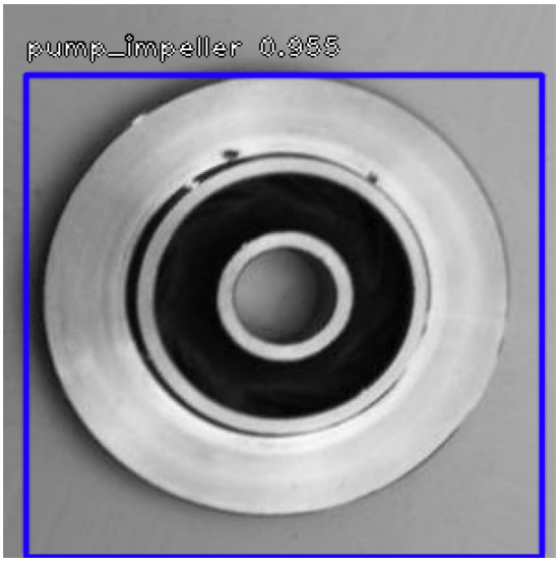
Training:

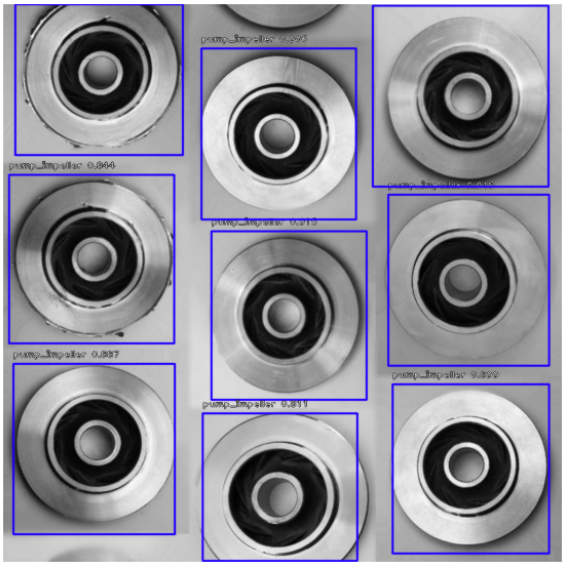
Our model is instantiated & trained for **20 epochs** with the images from our augmented dataset. Following is the plot of the training loss with each epoch of training:



Testing & Detection:

For detecting pump impellers in an image, we tune the **detection confidence threshold** to **0.8** (this value gave us the best results on our dataset) for creating bounding boxes around the detected impellers in an image. Following are some of the results of our detector:





Defect Detection (Fine-tuned VGG16 Model):

For the core task of defect detection, we select the **VGG16** architecture (since it was the better performing model compared to ResNet50 for our tasks) as our base model for the transfer learning process.

Dataset Preparation:

We use the Submersible Pump Impeller Defect Dataset to demonstrate the working of this model. To achieve robustness, instead of using the images of pump impellers as they are, we preprocess them to introduce some *rescaling*, *shearing*, *rotation* & *zooming* (which could happen during actual usage due to the camera positioning, etc).

The dataset contains **6633 training images** & **715 test images**.

Model Architecture:

We enable Transfer Learning by freezing weights of the base VGG16 Model & adding the following layers to the top of the base model for the detection task with 2 classes (*Defective* & *OK*):

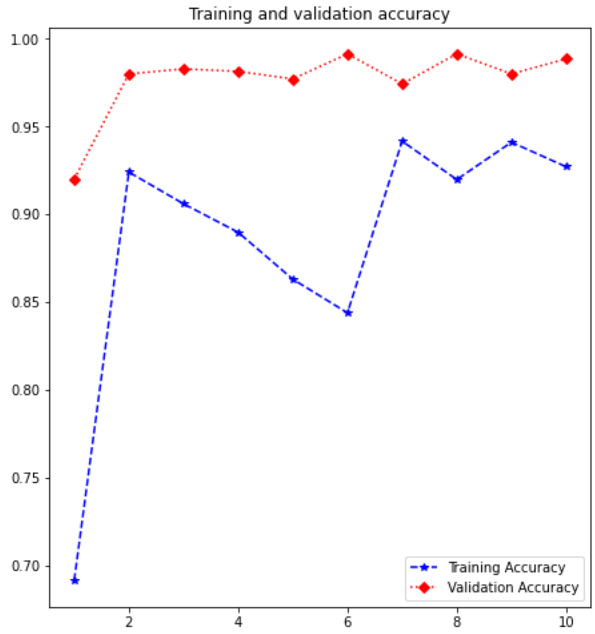
Layer Type	Output Shape	Number of Parameters
VGG (base model)	(BATCH_SIZE, 7, 7, 512)	14714688
Flatten	(BATCH_SIZE, 25088)	0

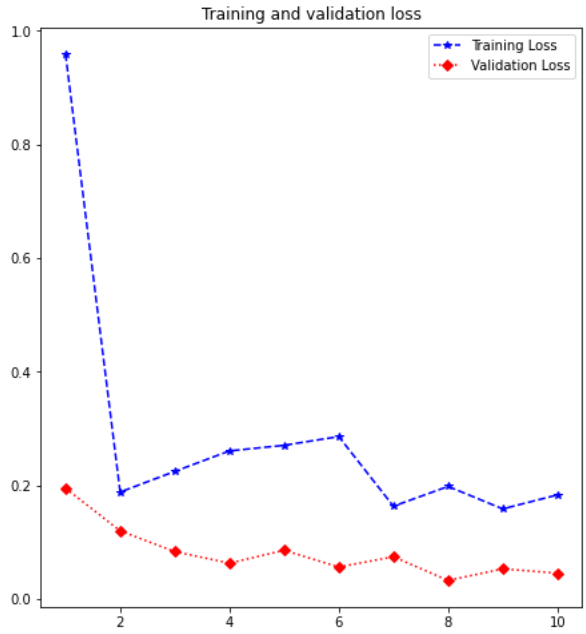
Fully Connected with Dropout	(BATCH_SIZE, 1024)	25691136
Fully Connected with Dropout	(BATCH_SIZE, 256)	262400
Fully Connected with Dropout	(BATCH_SIZE, 2)	514

Out of the 40,668,738 total parameters, we have **25,954,050 trainable parameters**.

Training & Testing:

We use **BATCH_SIZE = 50** & an **Adam Optimizer** (with **learning rate = 0.001**) to train the model for **10 epochs**. We achieved a **test accuracy** of **98.88 %**. The plot for the performance of the model over training period is presented below:





The confusion matrix for this model on the test images is shown below:

		Actual Label	
		OK	**Defective**
Predicted Label	**OK**	445	8
	Defective	0	262

The Classification Report for this model obtained is shown here:

	Precision	Recall	F1 Score	Support
OK	1.00	0.99	0.99	453
Defective	0.97	1.00	0.98	262
Accuracy			0.99	715
Macro Average	0.99	0.99	0.99	715
Weighted Average	0.99	0.99	0.99	715

Defect Classification (Fine-tuned VGG16 Model):

For the core task of defect classification too, we select the **VGG16** architecture (since it was the better performing model compared to ResNet50 for our tasks) as our base model for the transfer learning process. Based on the industry's requirements, the number of output classes of our final model would vary. For the demo, we have **6 types** of defects.

Dataset Preparation:

We use the Metal Surface Defect Dataset to demonstrate the working of this model. To achieve robustness, instead of using the images of metal surfaces as they are, we preprocess them to introduce some *rescaling*, *shearing*, *rotation* & *zooming* (which could happen during actual usage due to the camera positioning, etc).

The dataset contains **1656 training images**, **72 validation images** & **72 test images** (equal number of images per class).

Model Architecture:

We enable Transfer Learning by freezing weights of the base VGG16 Model & adding the following layers to the top of the base model for the classification task with 6 classes (*Crazing, Inclusion, Patches, Pitted, Rolled* & *Scratches*):

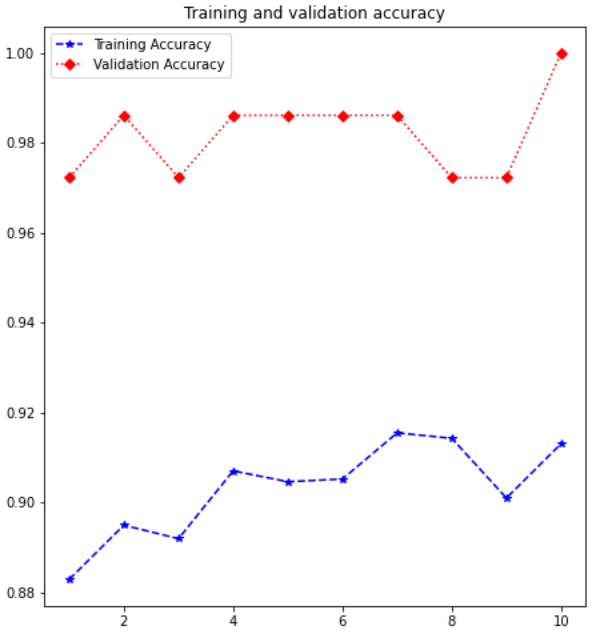
Layer Type	Output Shape	Number of Parameters
VGG (base model)	(BATCH_SIZE, 7, 7, 512)	14714688
Flatten	(BATCH_SIZE, 25088)	0
Fully Connected with Dropout	(BATCH_SIZE, 1024)	25691136

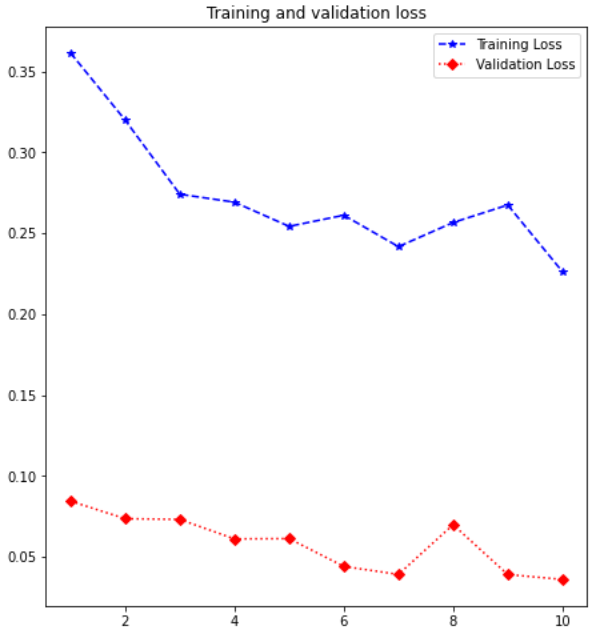
Fully Connected with Dropout	(BATCH_SIZE, 256)	262400
Fully Connected with Dropout	(BATCH_SIZE, 6)	1542

Out of the 40,669,766 total parameters, we have **25,955,078 trainable parameters**.

Training & Testing:

We use **BATCH_SIZE = 36** & an **Adam Optimizer** (with **learning rate = 0.001**) to train the model for **10 epochs**. We achieved a **test accuracy** of **100 %**. The plot for the performance of the model over training period is presented below:





The **confusion matrix** for this model on the test images is shown below:

		Actual Label					
		Crazing	**Inclusion**	**Patches**	**Pitted**	**Rolled**	**Scratches**
Predicted Label	**Crazing**	12	0	0	0	0	0
	Inclusion	0	12	0	0	0	0
	Patches	0	0	12	0	0	0
	Pitted	0	0	0	12	0	0
	Rolled	0	0	0	0	12	0
	Scratches	0	0	0	0	0	12

The **Classification Report** for this model obtained is shown here:

	Precision	Recall	F1 Score	Support
Crazing	1.00	1.00	1.00	12
Inclusion	1.00	1.00	1.00	12
Patches	1.00	1.00	1.00	12
Pitted	1.00	1.00	1.00	12
Rolled	1.00	1.00	1.00	12
Scratches	1.00	1.00	1.00	12
Crazing	1.00	1.00	1.00	12
Inclusion	1.00	1.00	1.00	12
Accuracy			1.00	72
Macro Average	1.00	1.00	1.00	72
Weighted Average	1.00	1.00	1.00	72

Team

Tezan Sahu 170100035
Suchit Sharma 170040041
Chinmay VG 193109018
Chinmay Naik 203190016
Ayush Pandey 193300007