McMaster University | ENGINEERING
W Booth School of Engineering Practice and Technology

SEP 799 – Cyber Physical Systems Part- II

3D Printing Monitoring Tool

Instructor: Dr. Marjan Alavi, Dr. Zhen Gao

Group Members :

Jithin Varghese - 400392674
Xuanyu Liu - 400119930

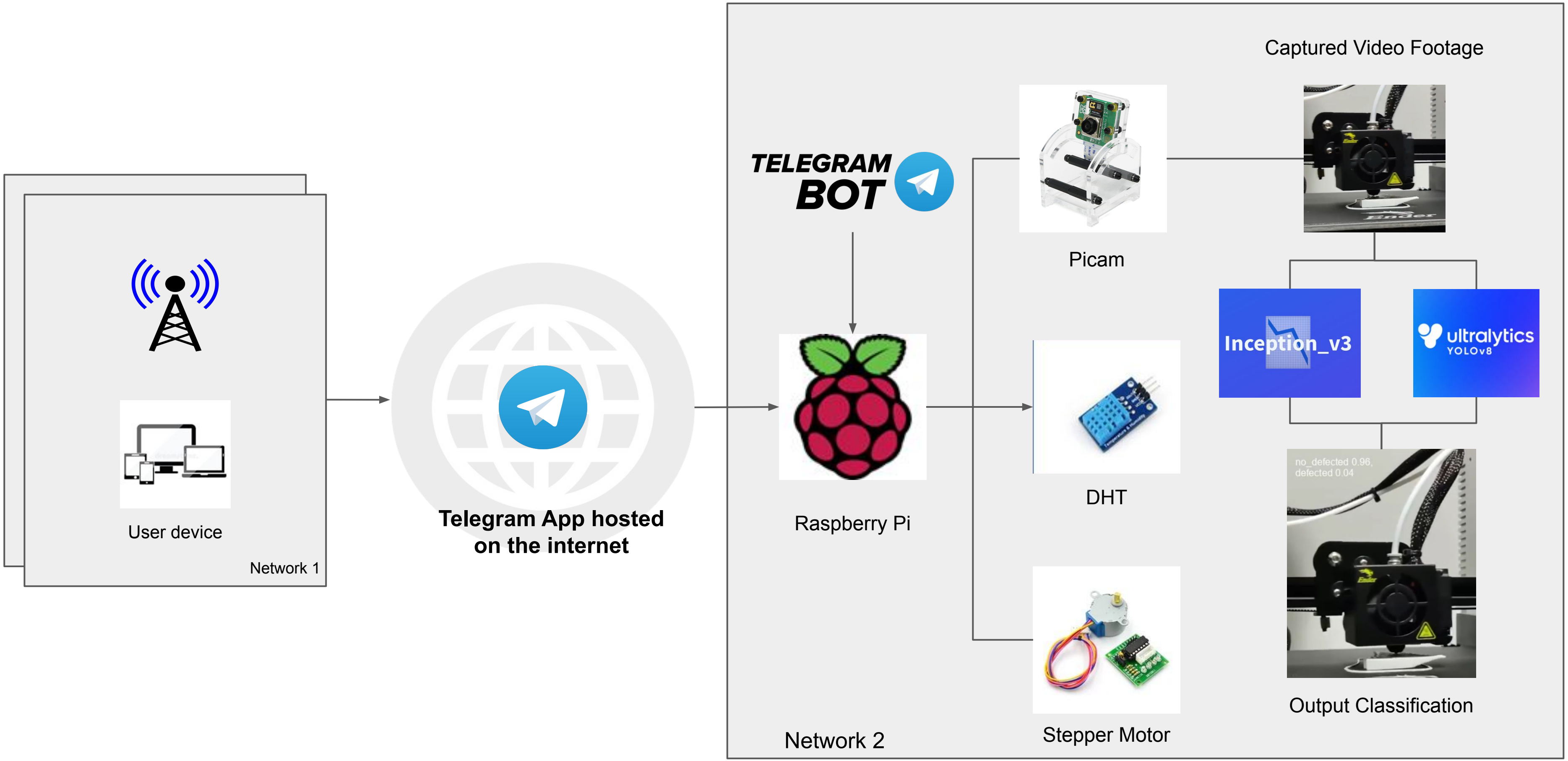
Objective

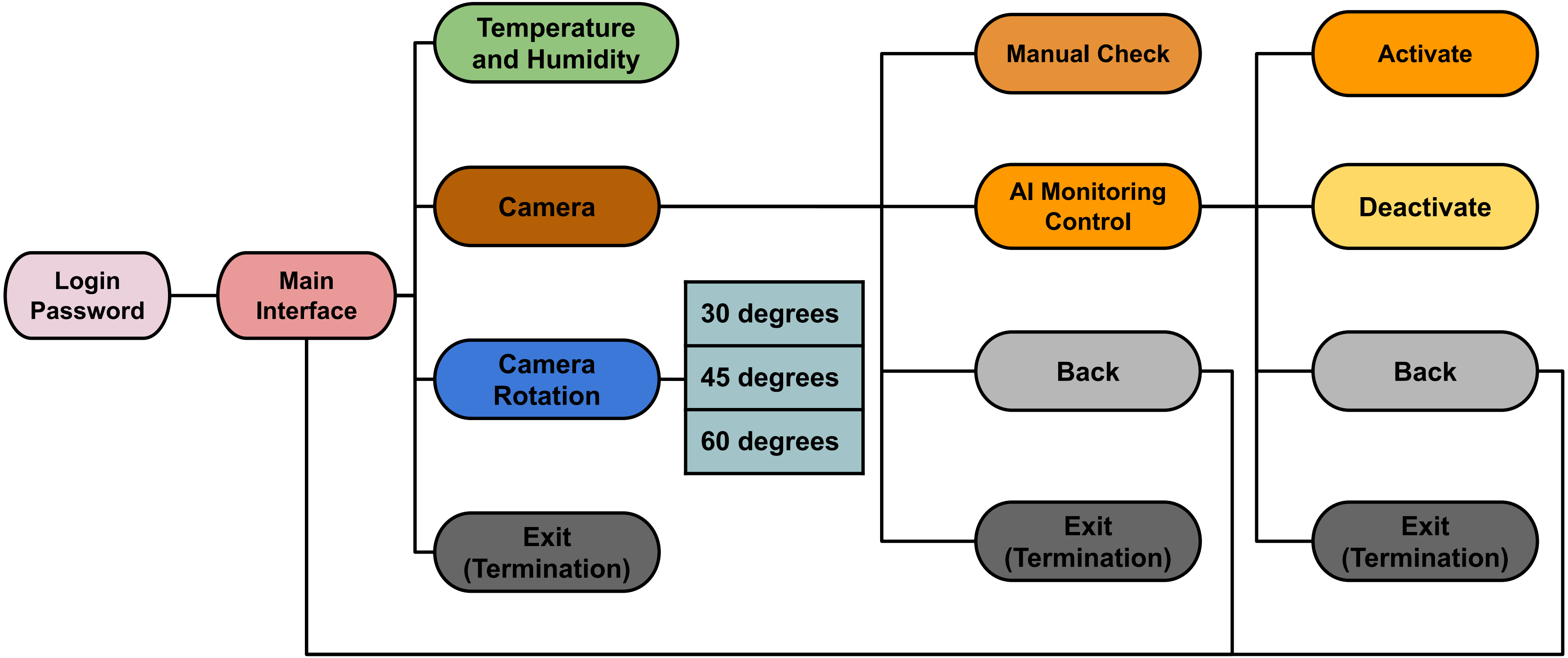
- 3D Printing Process Defect Classification Tool

- Software
  - Deep learning model training, dataset comparison
  - Model deployment, output integration
  - Telegram interactive bot

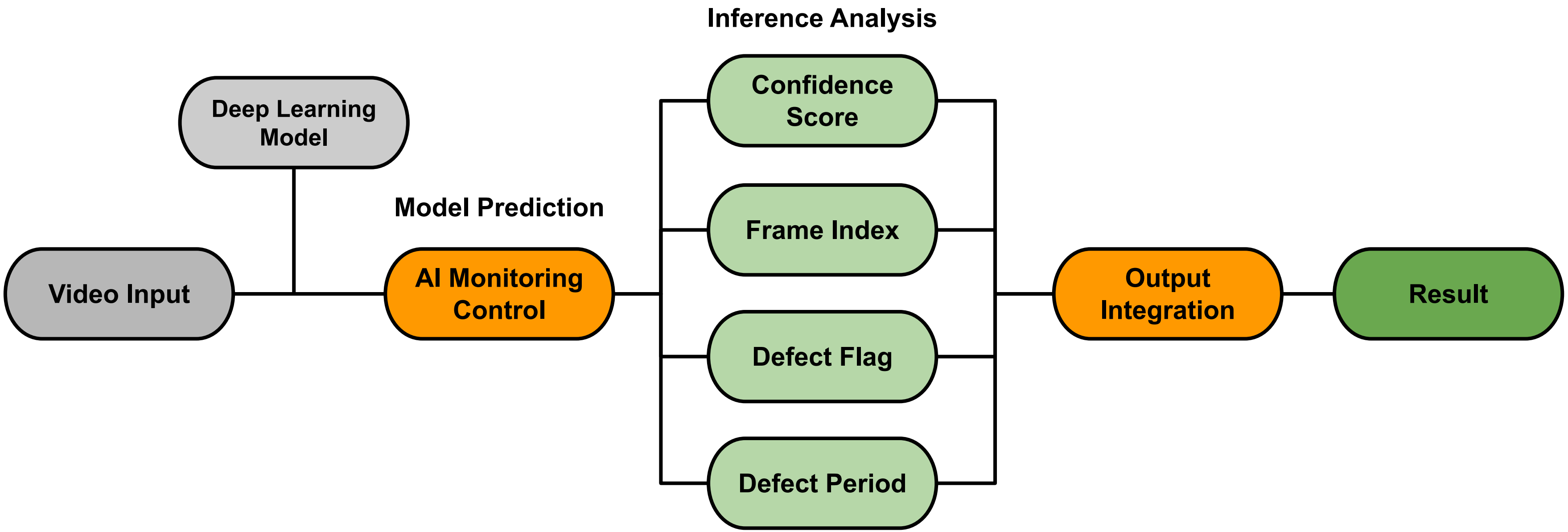
- Hardware
  - Raspberry Pi hardware configuration

The main objective is to deliver a fully functional product for the user to interact and perform defect monitoring while researching and implementing solutions to optimize the prediction process


User device
Network 1
Telegram App hosted on the internet
TELEGRAM BOT
Raspberry Pi
Picam
DHT
Stepper Motor
Captured Video Footage
Inception_v3
ultralytics YOLOv8
no_defected 0.96, defected 0.04
Output Classification
Network 2


Login Password
Main Interface
Temperature and Humidity
Camera
Camera Rotation
Exit (Termination)
30 degrees
45 degrees
60 degrees
Manual Check
AI Monitoring Control
Back
Exit (Termination)
Activate
Deactivate
Back
Exit (Termination)

AI Monitoring Tool

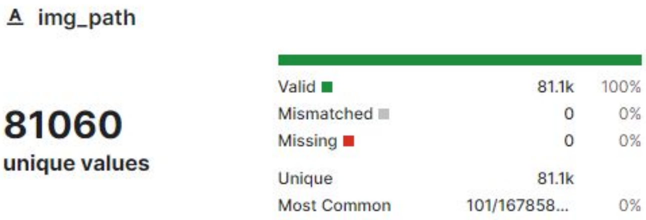


Dataset

- Kaggle competition: Early detection of 3D printing issues
- Binary classification for under extrusion (defect) vs. no defect
- 80,000+ labelled dataset
- Image captured from 7 different printer

Early detection of 3D printing issues

Early detection of 3D printing issues



img_path

81060
unique values

Valid 🟩	81.1k	100%
Mismatched ▨	0	0%
Missing 🟥	0	0%
Unique	81.1k	
Most Common	101/167858...	0%

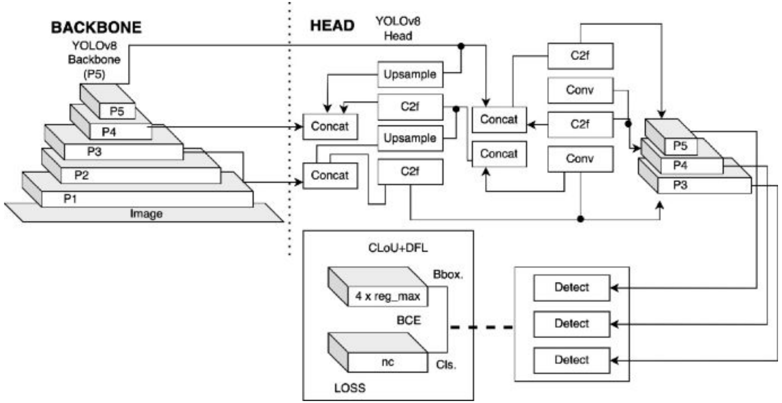
AI Monitoring Control - YOLO-v8

McMaster University

ENGINEERING
W Booth School of Engineering
Practice and Technology

YOLO-v8 Model

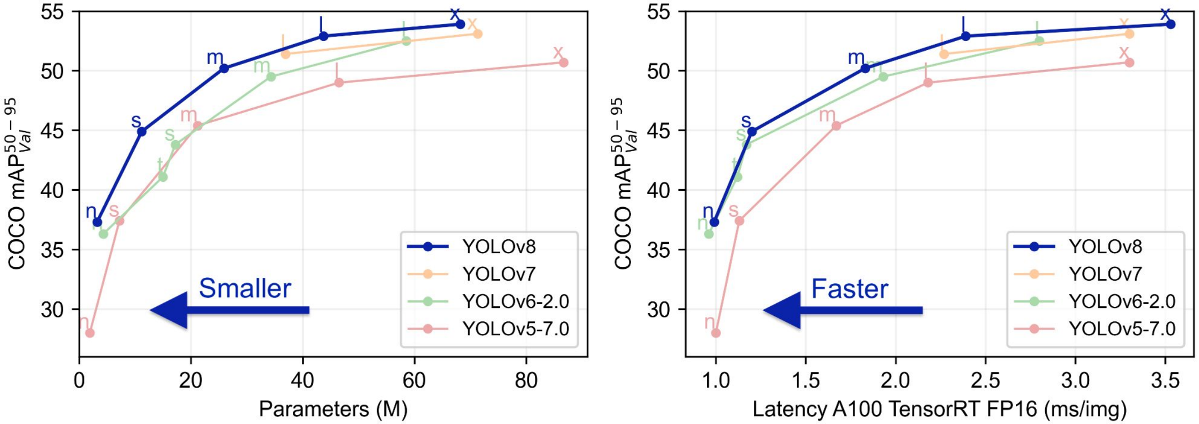
- Iterative Product of the previous model
- Compared to YOLO-v5
  - Replace C3 with C2f module
  - Replace 6*6 Conv with 3*3 in Backbone
  - Remove Bottleneck by replacing 1*1 Conv with 3*3 Cov
- Boost performance and flexibility



Note:
C3: CSP Bottleneck with 3 convolutions
C2f: Coarse to Find

McMaster University
ENGINEERING
W Booth School of Engineering Practice and Technology

YOLO Version

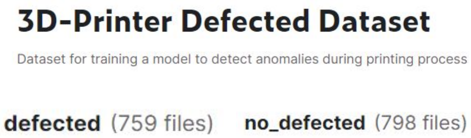


- YOLO-v8 offers smaller and efficient model with the best performance indicated by the highest mAP score
- YOLO-v8 uses new convolutional layers to make the classification more accurate

YOLO-v8 Defect Classification

- Initial and secondary stage for development
- Dataset & Data Processing
- Training & Testing
- Model Implementation
- Deployment

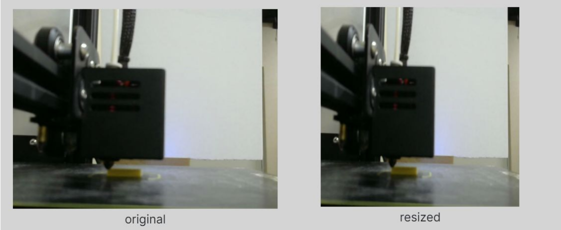
Dataset & Data Processing (Initial Stage)

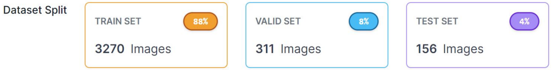


3D-Printer Defected Dataset

Dataset for training a model to detect anomalies during printing process

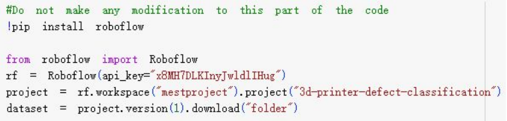
defected (759 files) **no_defected** (798 files)



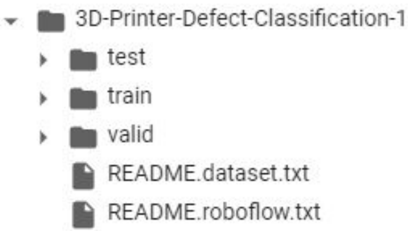


Dataset Split

TRAIN SET 88%	VALID SET 8%	TEST SET 4%
3270 Images	311 Images	156 Images



```
#Do not make any modification to this part of the code
!pip install roboflow

from roboflow import Roboflow
rf = Roboflow(api_key="x8MH7DLKInyJwldlIHug")
project = rf.workspace("mestproject").project("3d-printer-defect-classification")
dataset = project.version(1).download("folder")
```



```
▾ 3D-Printer-Defect-Classification-1
    ▸ test
    ▸ train
    ▸ valid
      README.dataset.txt
      README.roboflow.txt
```

- Kaggle 3D-printer Defect Dataset uploaded to Roboflow to initialize the classification project
- Data Pre-Processing and Augmentation with dataset export to work with YOLO-v8 model

YOLO-v8 Hyperparameter (Initial Stage)

Hyperparameter variable

- 15 Epoches, 16 Batches
- Automatic Optimizer with 0.01 learning rate
- Autodetected Adam Optimizer
- 0.000714 learning rate, 0.9 momentum

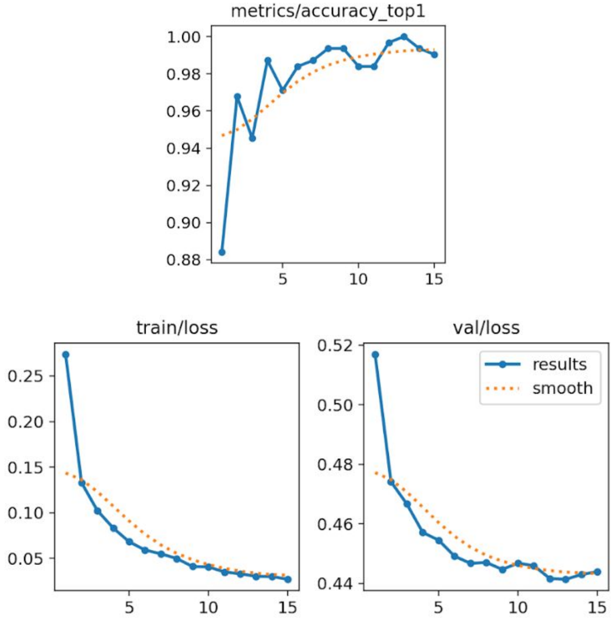
Hyperparameter - Function

- Categorical Cross-Entropy Loss
- Activation function:
  - Output layer: Sigmoid
  - Hidden layer: ReLU

McMaster University | ENGINEERING
W Booth School of Engineering Practice and Technology

YOLO-v8 Accuracy & Loss ROC curve (Initial stage)

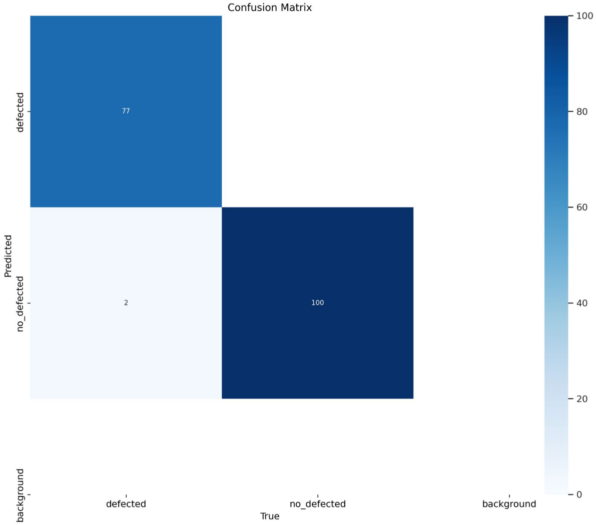
- Total training time: 0.925 hours with CPU
- Training accuracy
  - Steady increase with fluctuation
- Training loss
  - Continuous convergence
- Test loss fluctuation & overfitting
  - Adaptive learning rate
  - Gradient estimation noise



McMaster University

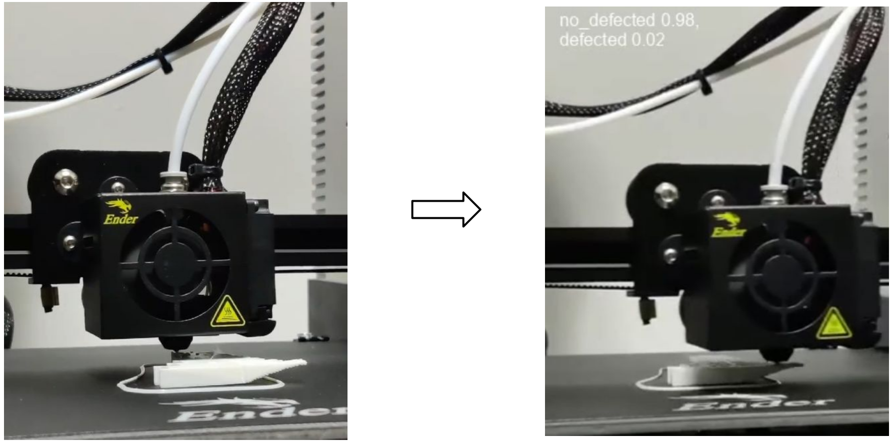
ENGINEERING
W Booth School of Engineering Practice and Technology

YOLO-v8 Confusion Matrix (Initial Stage)



- Test Accuracy is observed from the confusion matrix to be (77+100)/(77+100+2) = 98.88%
- Overfitting and lack of generalization is a concern from the dataset perspective

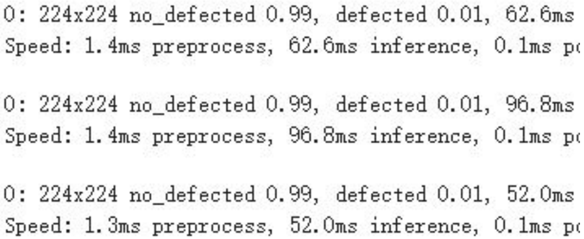
YOLO-v8 Model Implementation (Initial Stage)



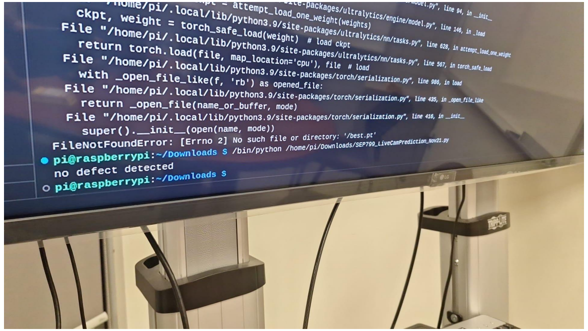
- Implementing OpenCV to predict, annotate each frame for a test video
- Output video with frame annotation representing the classification result

YOLO-v8 Output Integration (Unified Scaling)

- Confidence score threshold
  - Over 0.5 for true positive identified as Defect
- Frame confidence score result tracking
  - Defect vs. No Defect Counter
  - Frame Index Array
- Periodically Checking
  - Iteratively check the last 30 frames
  - Standardize 25 defect detected out of 30

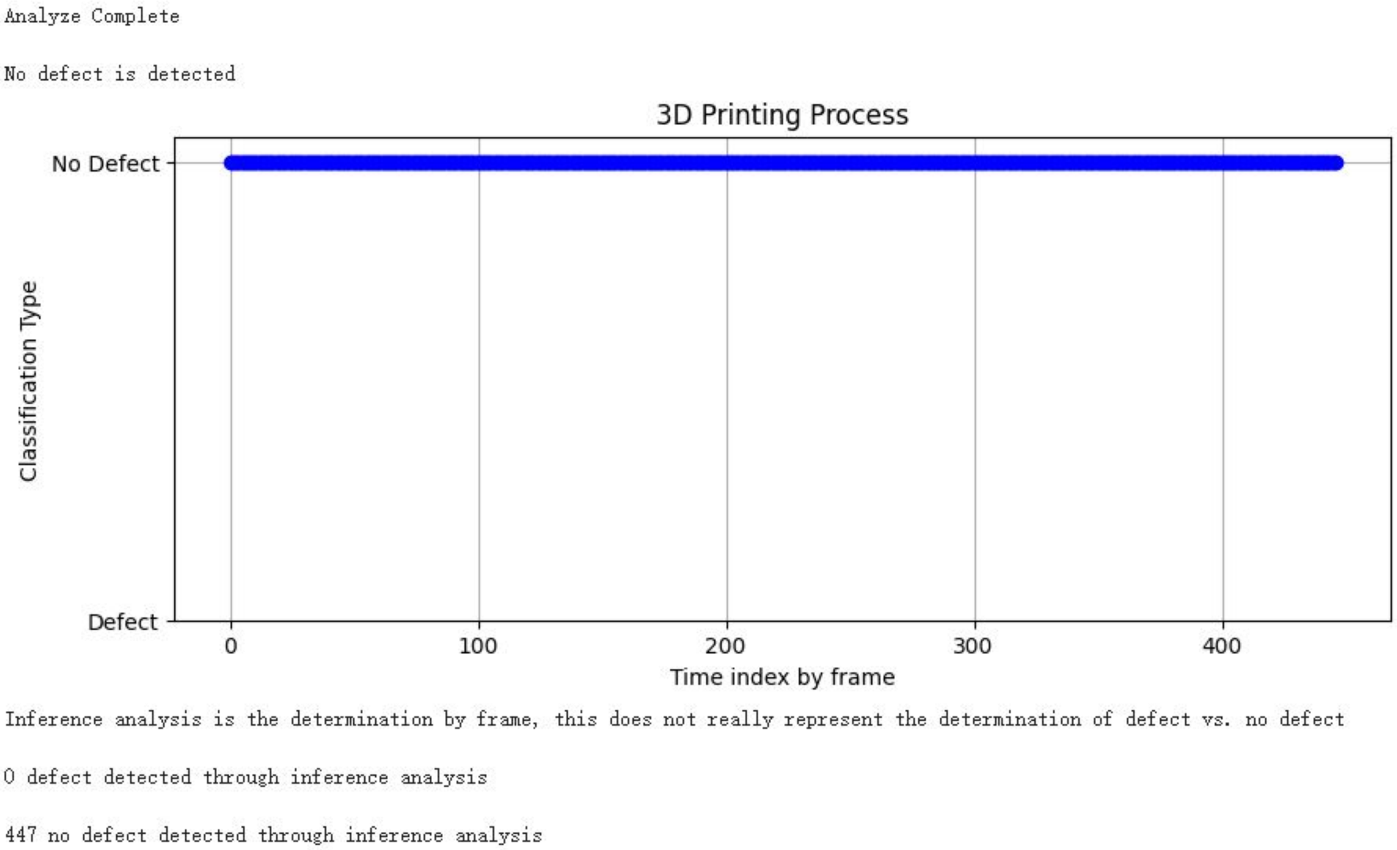


```
0: 224x224 no_defected 0.99, defected 0.01, 62.6ms
Speed: 1.4ms preprocess, 62.6ms inference, 0.1ms pc

0: 224x224 no_defected 0.99, defected 0.01, 96.8ms
Speed: 1.4ms preprocess, 96.8ms inference, 0.1ms pc

0: 224x224 no_defected 0.99, defected 0.01, 52.0ms
Speed: 1.3ms preprocess, 52.0ms inference, 0.1ms pc
```



```
  /home/pi/.local/lib/python3.9/site-packages/ultralytics/engine/model.py", line 146, in _load
    pt = attempt_load_one_weight(weights)
  File "/home/pi/.local/lib/python3.9/site-packages/ultralytics/nn/tasks.py", line 628, in attempt_load_one_weight
    ckpt, weight = torch_safe_load(weight)  # load ckpt
  File "/home/pi/.local/lib/python3.9/site-packages/ultralytics/nn/tasks.py", line 567, in torch_safe_load
    return torch.load(file, map_location='cpu'), file  # load
  File "/home/pi/.local/lib/python3.9/site-packages/torch/serialization.py", line 986, in load
    with _open_file_like(f, 'rb') as opened_file:
  File "/home/pi/.local/lib/python3.9/site-packages/torch/serialization.py", line 435, in _open_file_like
    return _open_file(name_or_buffer, mode)
  File "/home/pi/.local/lib/python3.9/site-packages/torch/serialization.py", line 416, in __init__
    super().__init__(open(name, mode))
FileNotFoundError: [Errno 2] No such file or directory: '/best.pt'
● pi@raspberrypi:~/Downloads $ /bin/python /home/pi/Downloads/SEP799_LiveCamPrediction_Nov21.py
no defect detected
○ pi@raspberrypi:~/Downloads $
```

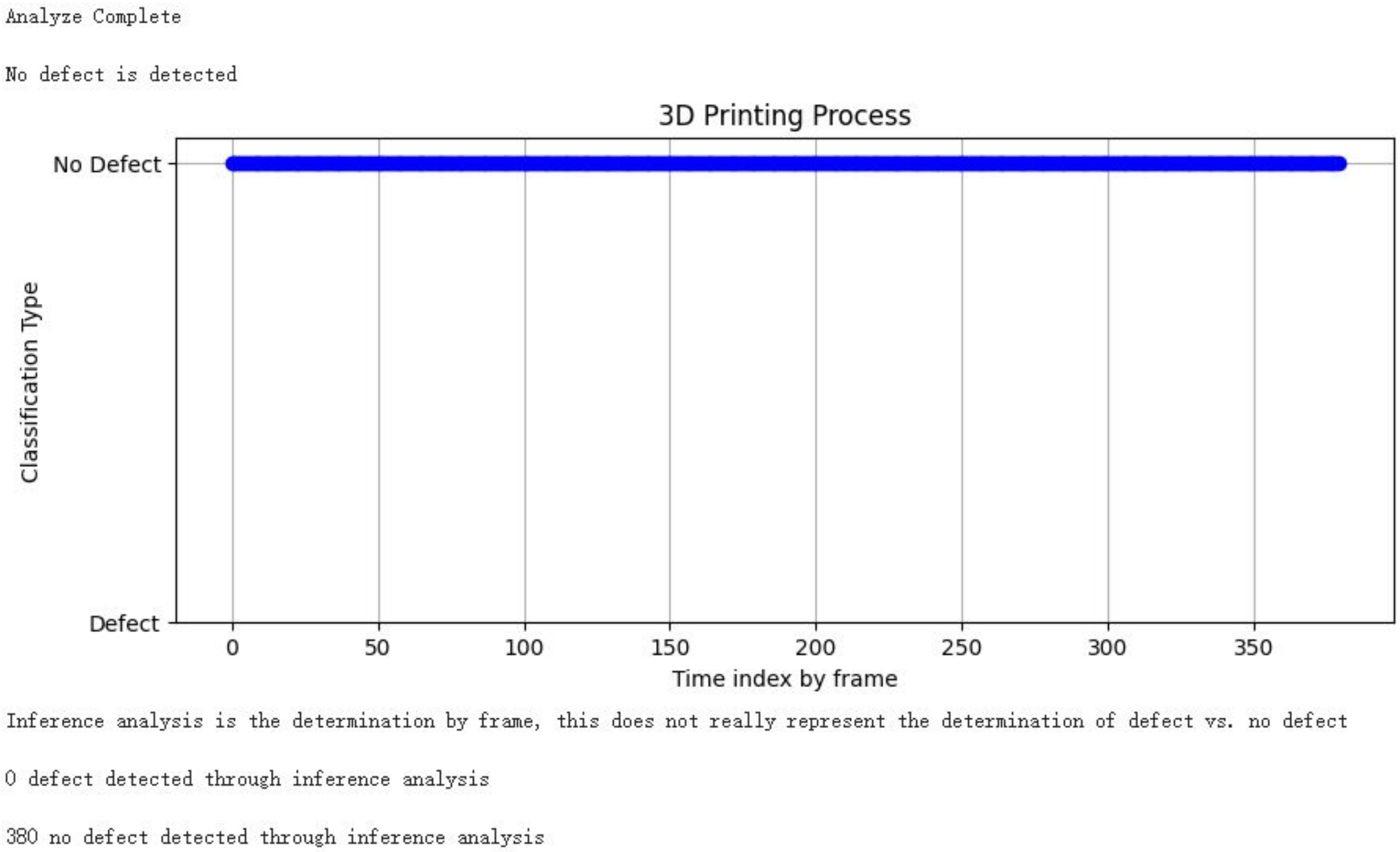
YOLO-v8 Feedback Analysis for Defect (First stage)



```
Analyze Complete

No defect is detected
```

```
Inference analysis is the determination by frame, this does not really represent the determination of defect vs. no defect

0 defect detected through inference analysis

447 no defect detected through inference analysis
```

YOLO-v8 Feedback Analysis for No Defect (First stage)



```
Analyze Complete

No defect is detected
```

```
Inference analysis is the determination by frame, this does not really represent the determination of defect vs. no defect

0 defect detected through inference analysis

380 no defect detected through inference analysis
```

YOLO-v8 Second Stage Development

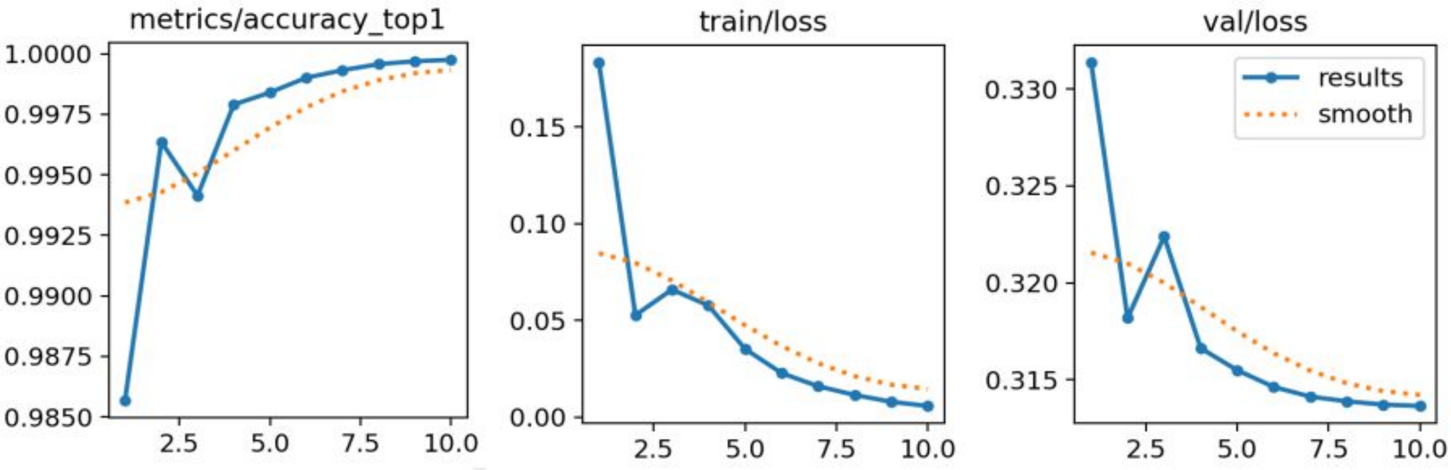
- Dataset Expansion
- Hyperparameter Improvement
- Training Result Analysis
- Output Integration

YOLO-v8 Hyperparameter (Second Stage)

Hyperparameter variable

- 10 Epoches, 8 Batches
- Automatic Optimizer with 0.01 learning rate
- Autodetected SGD Optimizer
- 0.01 learning rate, 0.9 momentum
- 0.3 dropout rate

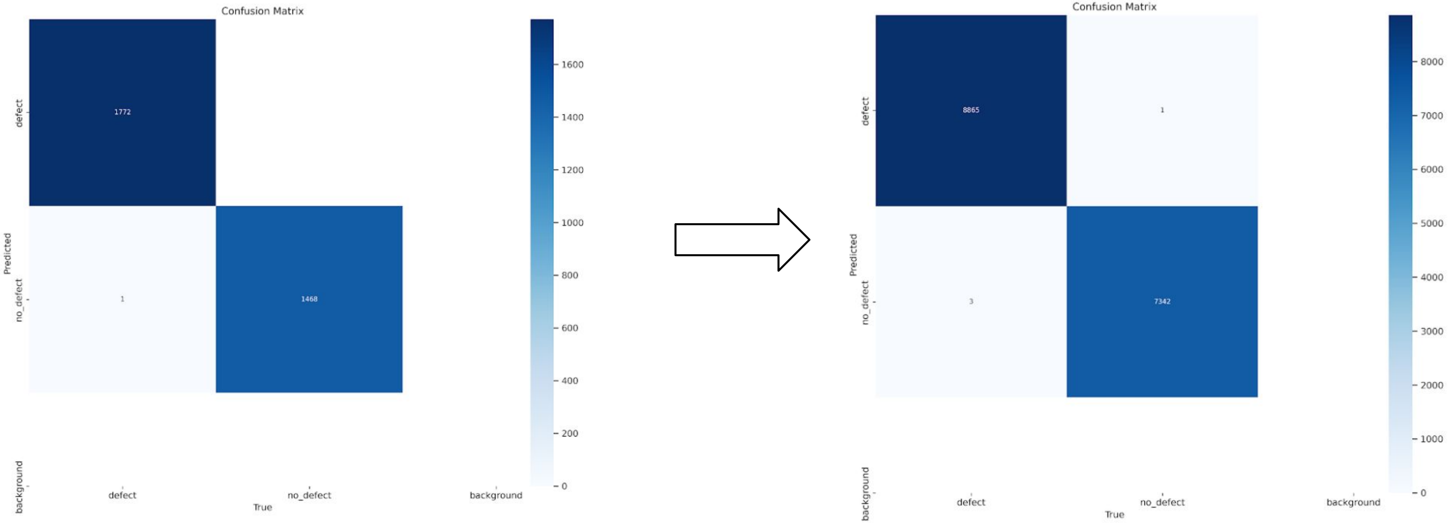
YOLO-v8 Accuracy & Loss ROC curve (Second stage)



McMaster University

ENGINEERING
W Booth School of Engineering Practice and Technology

YOLO-v8 Confusion Matrix (Second Stage)

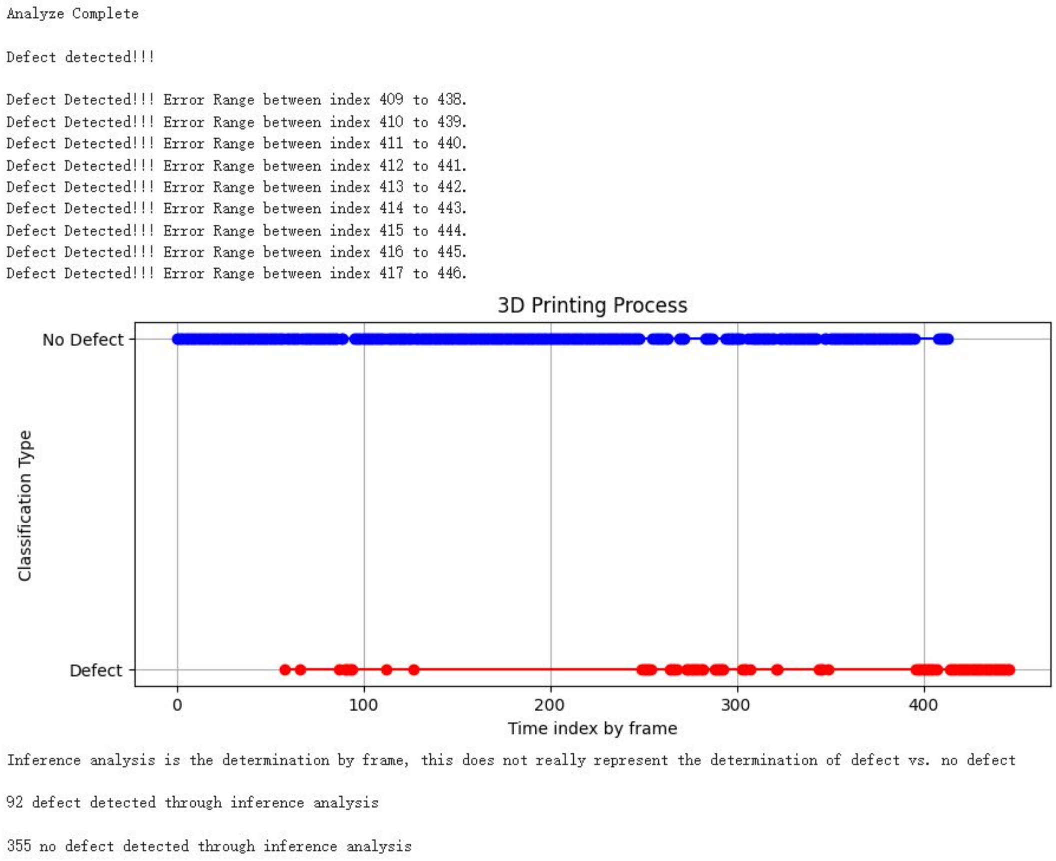


- Test Accuracy is observed from the first confusion matrix to be (8865+7342)/(8865+7342+3+1) = 99.97%
- Despite the near perfection score, overfitting is still an unresolved issue.

YOLO-v8 Output Integration (Second Stage)

- Inference Analysis for confidence score
- Defect Classifier Array
- Defect Error Message for continuous detection
- Defect vs. No Defect counter/plot

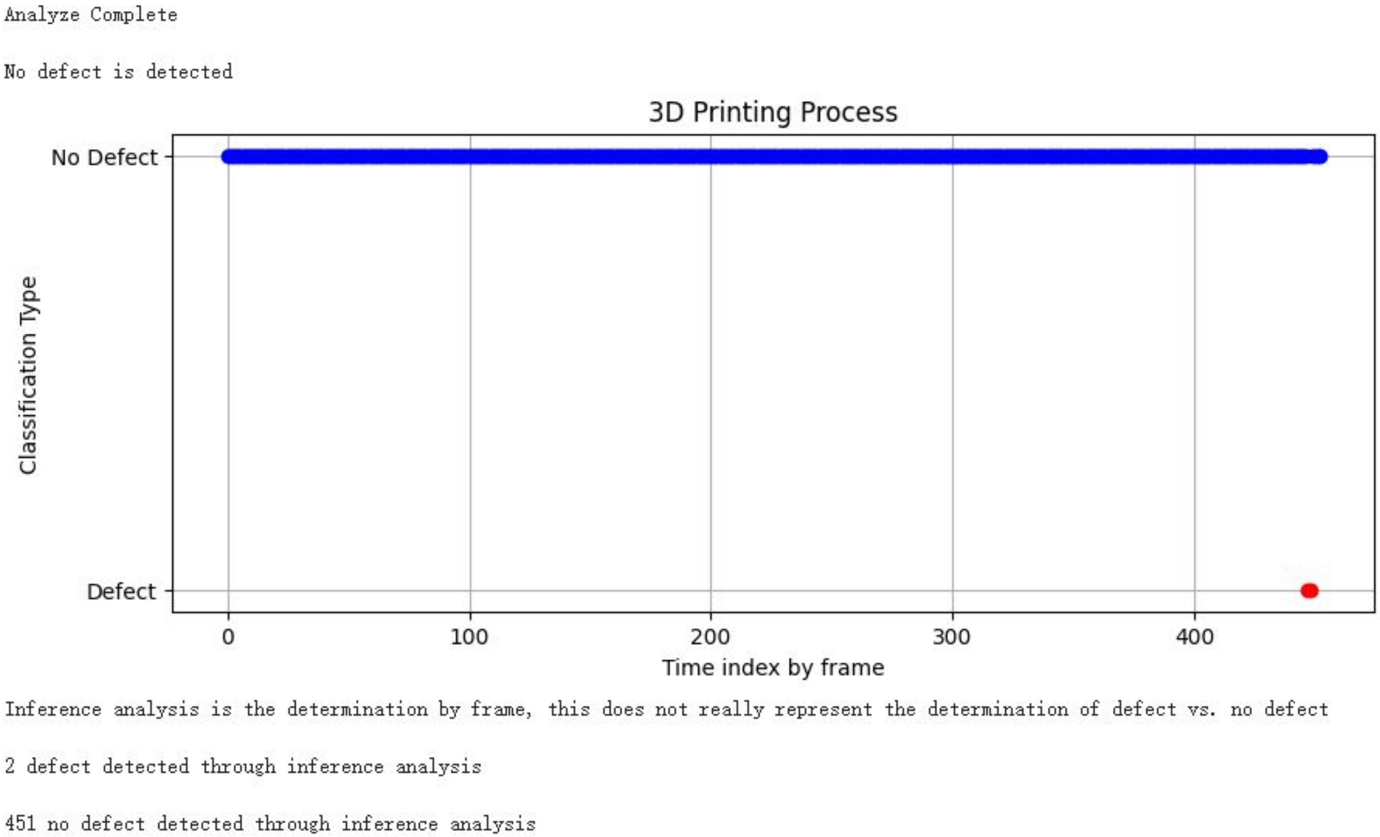
YOLO-v8 Feedback Analysis for Defect (Second stage)



```
Analyze Complete

Defect detected!!!

Defect Detected!!! Error Range between index 409 to 438.
Defect Detected!!! Error Range between index 410 to 439.
Defect Detected!!! Error Range between index 411 to 440.
Defect Detected!!! Error Range between index 412 to 441.
Defect Detected!!! Error Range between index 413 to 442.
Defect Detected!!! Error Range between index 414 to 443.
Defect Detected!!! Error Range between index 415 to 444.
Defect Detected!!! Error Range between index 416 to 445.
Defect Detected!!! Error Range between index 417 to 446.
```

```
Inference analysis is the determination by frame, this does not really represent the determination of defect vs. no defect

92 defect detected through inference analysis

355 no defect detected through inference analysis
```

YOLO-v8 Feedback Analysis for No Defect (Second stage)



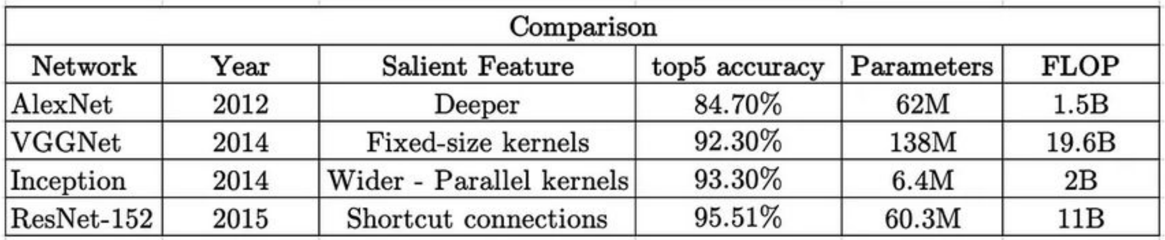
```
Analyze Complete

No defect is detected
```

```
Inference analysis is the determination by frame, this does not really represent the determination of defect vs. no defect

2 defect detected through inference analysis

451 no defect detected through inference analysis
```

Inception V3 Model

Inception V3 Model

- Why Inception?

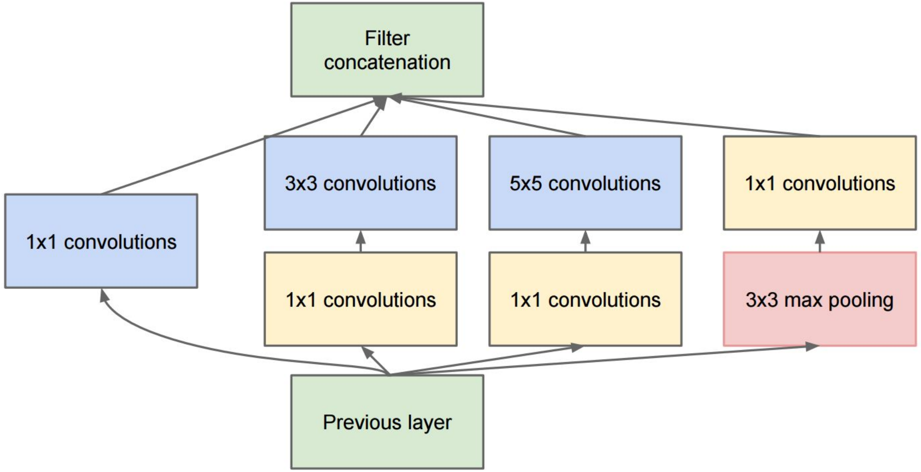


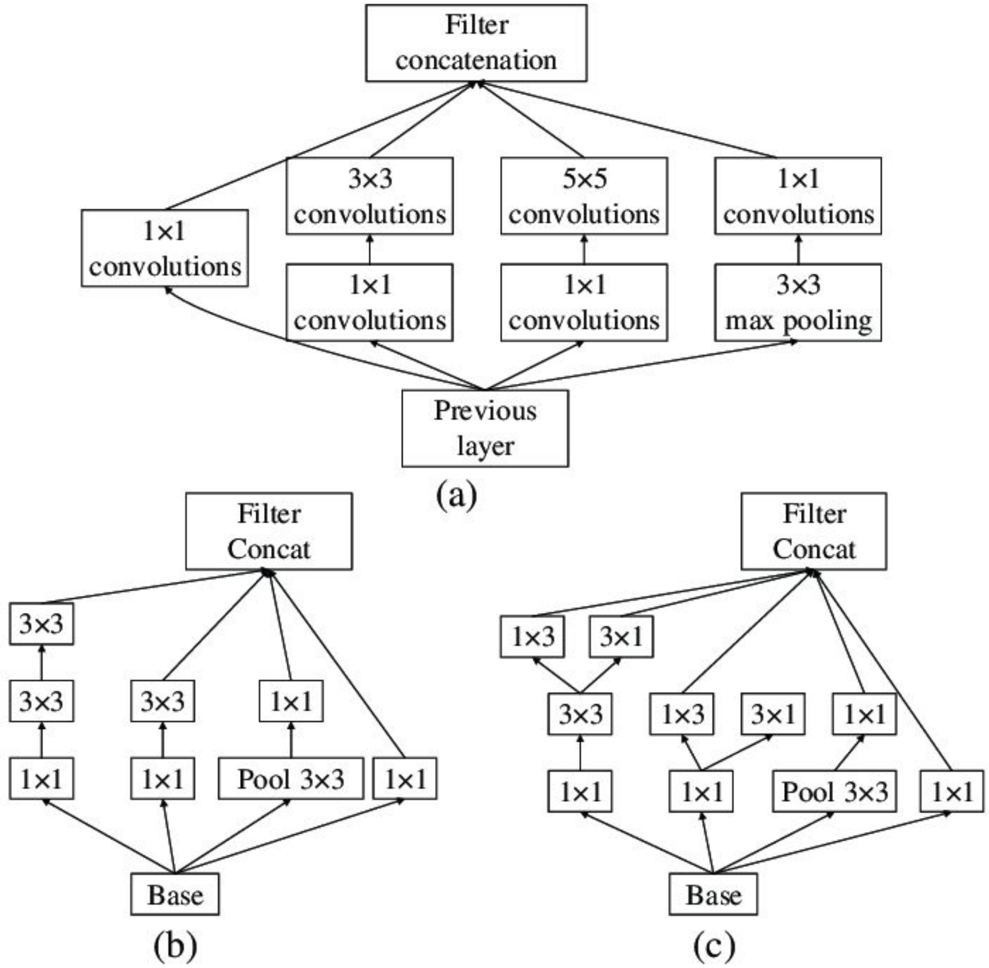
Comparison					
Network	Year	Salient Feature	top5 accuracy	Parameters	FLOP
AlexNet	2012	Deeper	84.70%	62M	1.5B
VGGNet	2014	Fixed-size kernels	92.30%	138M	19.6B
Inception	2014	Wider - Parallel kernels	93.30%	6.4M	2B
ResNet-152	2015	Shortcut connections	95.51%	60.3M	11B

Despite being a powerful model we can see that Inception is quite small and suited for execution on edge IOT devices such as Raspberry Pi, Google Coral board etc.This unique advantage along with readily available library modules for Inception made it possible to quickly train and deploy.

Features of Inception :

- Area covered by the desired objects in each image varies from an image to image. So, it becomes difficult to determine "the size of the kernel".Hence the approach to apply convolutional operations on an input with different sizes of kernel (1x1, 3x3, 5x5) along with max pooling at the same level to extract all kind of information.





(a)

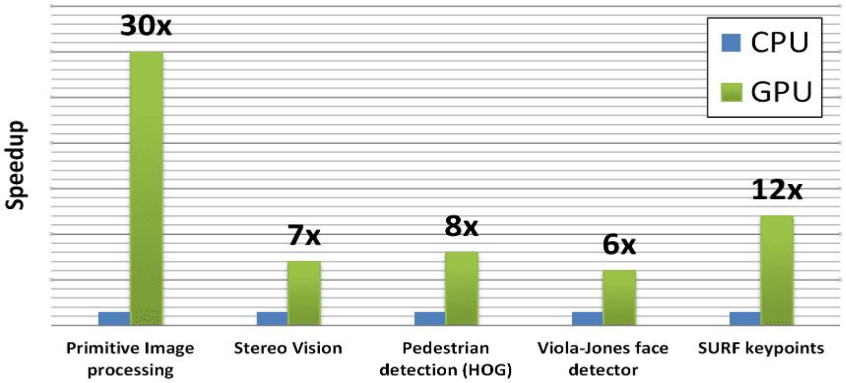
(b)

(c)

- For defect identification in 3d objects, we have trained most advanced version (V3) of inception model that consist core inception block at various level but along with factorized convolution.
  (Image showcasing V1,V2,V3 and their differences)

GPU Accelerated Training

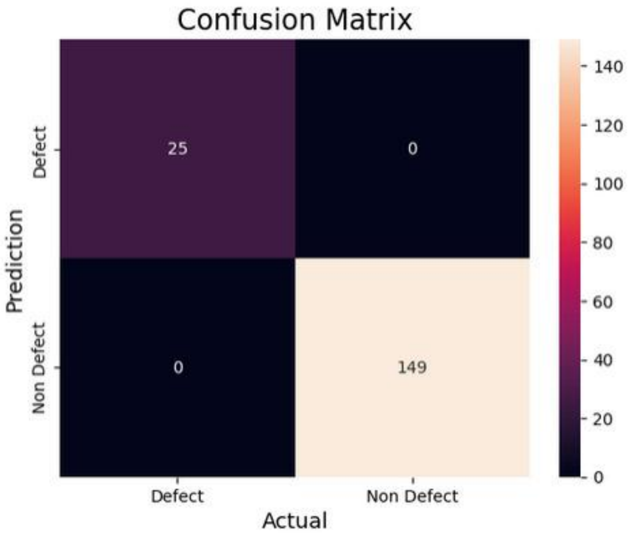
To train our inception model we used a Nvidia 3070 GPU with 8gb of VRAM to perform the calculations for weights.The time taken on average to fine tune last few layers from “Mixed 6” to “Fully Connected layer” was around 30 mins. In case of overfitting result we trained the network for about an hour. Under fitting was achieved in 15 to 20 mins of training.The best fit trained model was then exported to be used in the Pi for inference.



McMaster University | ENGINEERING
W Booth School of Engineering Practice and Technology

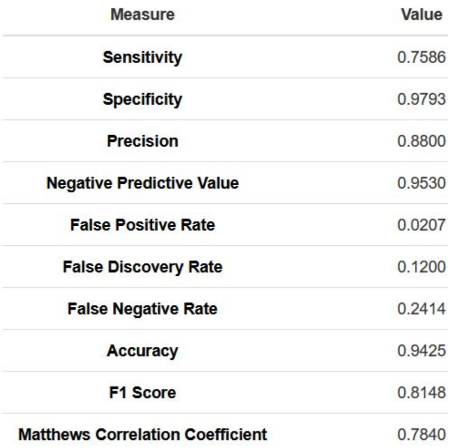
InceptionV3 Confusion Matrix (Results)

Overfitting Confusion Matrix

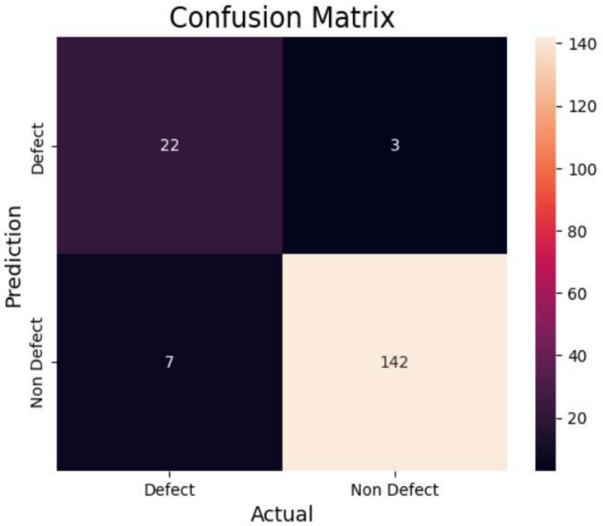


Over Fit confusion matrix
(160 steps 70 epochs - 100%)
training time 1 hr

Bestfitting Confusion Matrix



Measure	Value
Sensitivity	0.7586
Specificity	0.9793
Precision	0.8800
Negative Predictive Value	0.9530
False Positive Rate	0.0207
False Discovery Rate	0.1200
False Negative Rate	0.2414
Accuracy	0.9425
F1 Score	0.8148
Matthews Correlation Coefficient	0.7840



Best Fit confusion matrix
(140 steps 35 epochs - 94%)
training time 30 min

McMaster University ENGINEERING
W Booth School of Engineering Practice and Technology

Live Demo

Any Questions?

Thank you

McMaster University

ENGINEERING
W Booth School of Engineering Practice and Technology