ML & AI Internship Assignment Report

Objective

The task was to develop a machine learning model to classify images of industrial equipment into two categories:

Defective Non-Defective

Bonus Objectives (Optional)

Identify and classify specific types of defects. Optimize the model for hardware-accelerated inference (GPU/TPU/NPU/FPGA).

Dataset

Large dataset of steel defect images. Original image size: 256×1600 pixels. Labels were provided only for validation set, so custom handling was required.

Approach

Model Selection

Initially considered CNN-based classification models, but due to the large dataset and limited GPU resources, opted for YOLOv5 for segmentation/classification. YOLOv5 allows faster experimentation with less training time while maintaining accuracy.

Training Details

Used YOLOv5s pretrained weights (‘yolov5s.pt‘) as a starting point. Training command used:

python train.py --img 640 --batch 2 --epochs 17 --data /home/extra_space/akhilesh/severstal-steel-defect-detection/data.yaml --weights yolov5s.pt --device 0

Adjusted epochs to 17 (instead of 100) for faster experimentation due to hardware limitations.

Challenges Faced

1. Large Dataset – Training on full-resolution images (256×1600) was computationally expensive. 2. Limited GPU Resources – Used a local GPU instead of cloud computing, which constrained training time and model size. 3. Label Availability – Only validation labels were provided, requiring careful dataset setup. 4. Image Resolution Mismatch – Original images were 256×1600, but resized to 640 for YOLOv5 compatibility. YOLO automatically rescales annotations correctly, so labels remained valid.

Results

Successfully trained YOLOv5 model for defective vs. non-defective classification. Achieved reasonable performance within limited compute and training epochs. Demonstrated that defect detection can be scaled further with larger models and longer training.

Future Improvements

Train longer (50–100 epochs) on higher-resolution images for better accuracy. Try CNN-based classifiers (e.g., EfficientNet, ResNet) for direct binary classification. Explore segmentation models (U-Net, Mask R-CNN) for precise defect localization. Implement cloud-based training to remove local hardware constraints. Optimize model using TensorRT on NVIDIA Jetson Nano for real-time inference.

Deployment

YOLOv5 ‘.pt‘ model can be exported and converted to TensorRT engine:

The optimized TensorRT model can run efficiently on Jetson Nano with CUDA acceleration.

Conclusion

This task demonstrated an end-to-end pipeline:

Dataset preparation Model training with YOLOv5 Handling hardware and data challenges Planning for real-time deployment

The assignment provided valuable hands-on experience in working with defect detection problems in industrial settings.