Group Project – Milestone II

D7056E, **Predictive Maintenance for Railway Infrastructure Using Image-Based Fault Detection**

D7056E - Predictive Analytics

Group Members: Lars-Erik Englund, Daniel Habte-Medhanie, Victor Rigworo Kebande, Anton Sandin, Jassim Al-Ahmmad

Submission Date: 2025-03-03

Assignment Description

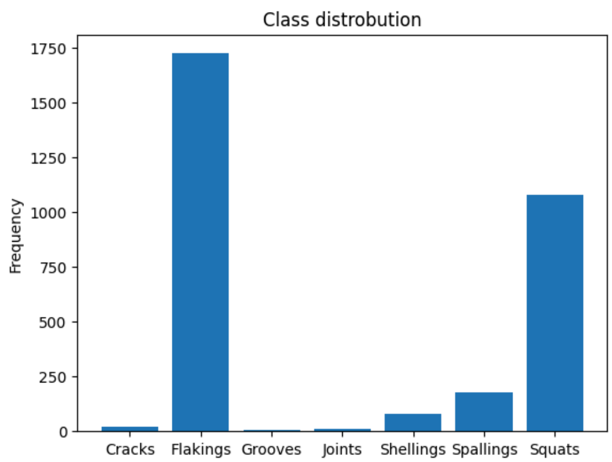
In this milestone, we implement and evaluate two predictive models for railway defect classification. The goal is to compare model performance and determine the best approach for deployment.

The milestone consists of four key steps:

1. **Final dataset preparation and feature selection**
2. **Implementation of two predictive models**
3. **Model evaluation and performance comparison**
4. **Result analysis and model selection**

Step 1: Dataset Summary and Feature Selection

The dataset consists of labeled images of railway track defects, categorized into multiple defect types. This section explains the process of organizing and preprocessing the image data including resizing, normalization, and augmentation to ensure consistency and quality. It also describes how we identify and select key features while addressing challenges such as varying lighting conditions and different image angles for effective model training. This dataset had a varied amount of data points for each maintenance problem which could make it so the model is biased to certain types of railway faults. However it is important to note that these data points represent a real-world distribution of actual faults that happen during railway maintenance which a potential bias could aid in the modelling.



Descriptive Features and Target Feature:

• **Input Features:** Image data (224x224 RGB pixels)

• **Target Feature:** Defect classification (e.g., Cracks, Flakings, Joints, Shellings, Spellings, Squats, Grooves)

Data Characteristics:

• **Data Type:** Image (numeric pixel values)

• **Range of Values:** 0–255 (RGB pixel intensity)

• **Quality Issues:** Variability in lighting conditions, angle differences, lack in variation and high correlation of images in dataset, and class imbalances.

Preprocessing:

• **Image Preprocessing:** Resizing to 224x224, normalization to certain mean and standard deviation, and data augmentation.

Step 2: Model Implementation – Two Approaches

Two predictive models were implemented to classify railway track defects.

The first model involves a Convolutional Neural Network (CNN) that processes the prepared images through layers designed to extract and learn visual features. The second approach utilizes a fine-tuned Residual Neural Network(ResNet) adapted for defect detection. Below we explain the architectural choices, training configurations, and hyperparameters that distinguish each model.

Model 1: Convolutional Neural Network (CNN)

• **Architecture:** Our architecture is a CNN with 3 convolutional layers followed by a flattening layer and finally two feed-forward layers. Between each layer is a ReLU activation function and a Dropout layer for regularisation. In addition to this max pooling layers with 2x2 kernels are added between the convolutional layers.

• **Hyperparameters:** Dropout rate: 50%. L2 weight_decay of 5e-5 for regularization.

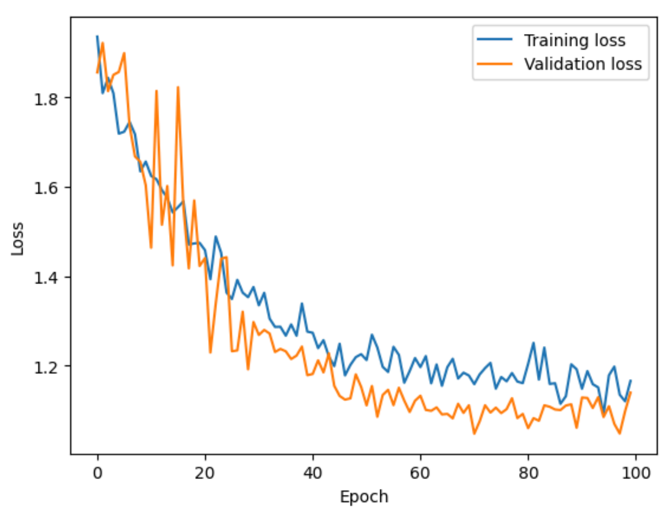
• **Optimizer:** Adam

• **Loss function:** Weighted CrossEntropyLoss to adjust for class imbalance

• **Learning Rate:** 0.0015 with 5.0% learning rate decay each epoch.

• **Batch Size:** 32

• **Epochs:** 100



Model 2: Residual Neural Network (ResNet-18)

• **Architecture:** The Resnet architecture uses the PyTorch implementation which follows the implementation in the “Deep Residual Learning for Image Recognition” (available at https://arxiv.org/abs/1512.03385) however to reduce overfitting we also added multiple dropout layers to the existing architecture. The model was initialized with Pytorchs ResNet18_Weights.IMAGENET1K_V1 weights which are weights trained on the imagenet dataset.

• **Pretrained ResNet-18 with fine-tuning**

• **Adjustments:** Output layers adjusted for defect classification by adding an additional layer from 1000 classes down to 7 classes, these were trained alone for the first 3 epochs to avoid disturbing the pretrained weights in earlier layers.

• **Hyperparameters:** Dropout rate: 3% for convolution part of network and 50% for feed forward part. L2 weight_decay of 75e-6.

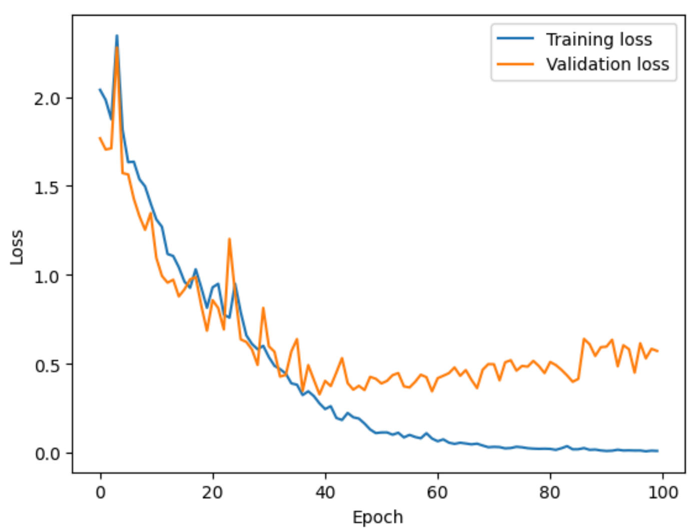
• **Optimizer:** Adam

• **Loss function:** Weighted CrossEntropyLoss to adjust for class imbalance

• **Learning Rate:** 0.0015 with 4.5% learning rate decay each epoch

• **Batch Size:** 32

• **Epochs:** 100



Step 3: Model Evaluation and Performance Metrics

To compare the models, the following evaluation metrics were used:

• **Accuracy:** Measures overall correctness.

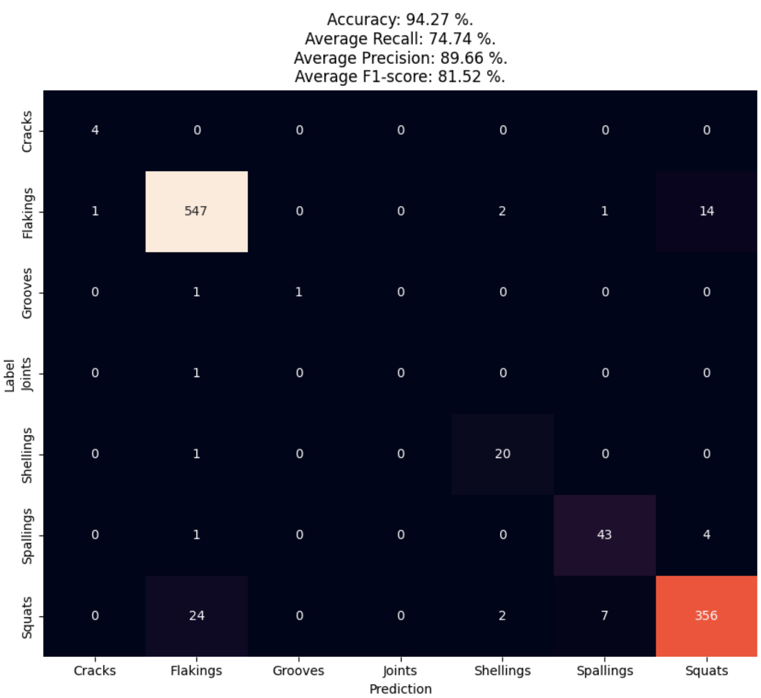
• **Precision & Recall:** Ensures correct defect classification and minimizes false negatives and false positives.

• **Macro Averaged F1-score:** Balances precision and recall for multi-class problems.

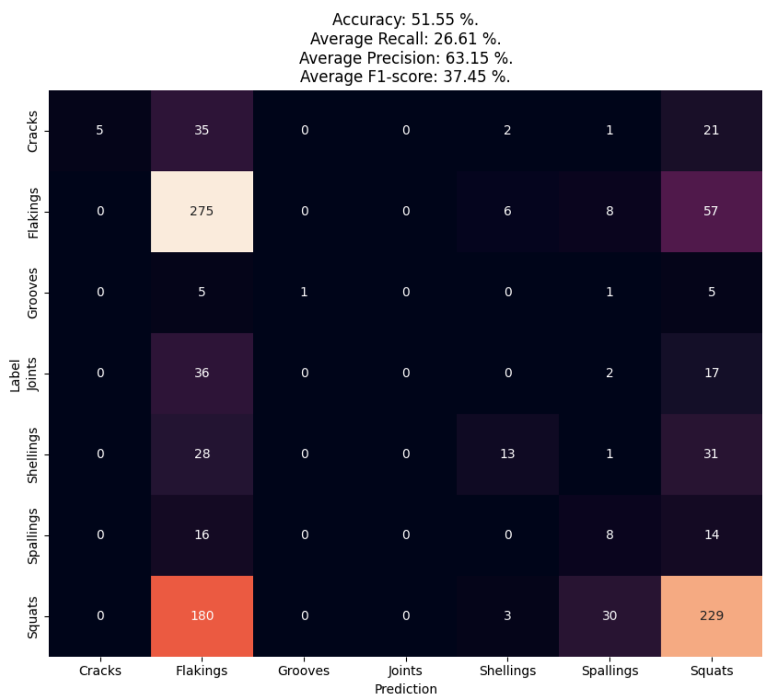
• **Confusion Matrix:** Visualizes classification performance across defect categories.

Both models were tested on a 20% test dataset and evaluated using the metrics above.

ResNet-18



CNN



Step 4: Results and Model Selection

Key Observations:

• **CNN:** Faster to train and faster inference but much lower accuracy compared to ResNet. The CNN has a strong bias to only predict Flakings or Squats.

• **ResNet:** Much more accurate but requires more computational power. ResNet in this case was more robust to the biased dataset that was used for our model training which led it to have a better result compared to the CNN. By analysing the final loss of when training the ResNet we see that the training loss is close to zero while the validation loss is a bit higher around 0.5, this suggests that the ResNet could be improved further by tweaking the hyperparameters to add more regularization.

Final Model Selection:

ResNet was chosen due to higher accuracy and better generalization. Despite higher training costs, it offers superior performance in detecting subtle railway defects.