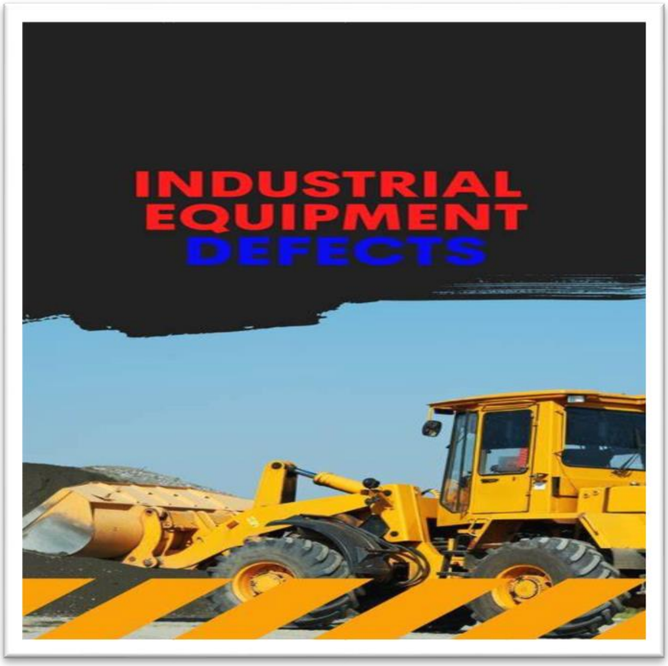
Industry Equipment Defect Detection Using CNN: Project Report

1. Introduction

The Industry Equipment Defect Detection project aims to automate the identification of defects in industrial equipment using image analysis powered by Convolutional Neural Networks (CNNs). This approach helps to enhance quality control, minimize human inspection errors, and improve operational efficiency by reducing downtime. By classifying equipment images into "defective" and "non-defective," this solution supports faster and more accurate defect detection, ensuring reliable industrial equipment performance.



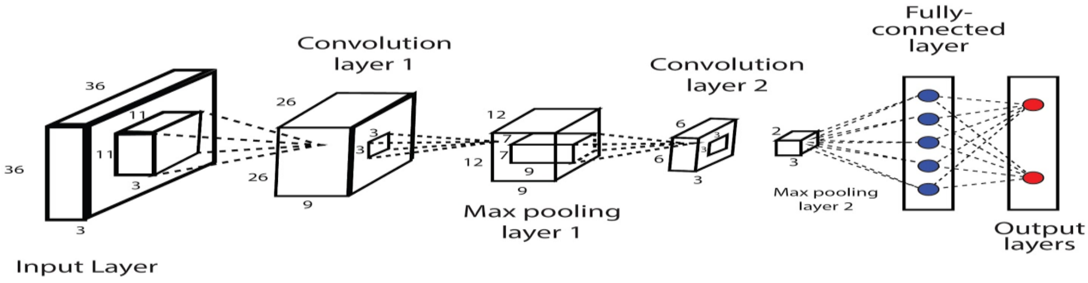
2. Methodology

Data Preparation and Pre-processing

- **Dataset**: The project uses labeled images of industrial equipment categorized as *defective* and *non-defective*. We only had the original data; we didn't have the train and test data, so we are created it.
- **Image Pre-processing**:
    - **Resizing**: Images were resized to a standard 150x150 pixels to ensure consistent input dimensions for the model.
    - **Normalization**: Pixel values were scaled to a [0, 1] range to improve model training.
    - **Data Splitting**: The dataset was split into training and testing sets with an 80:20 ratio to evaluate the model's performance on unseen data.

Industry Equipment Defect Detection Using CNN: Project Report

Model Architecture



- **Convolutional Layers**: Three convolutional layers were used, each followed by max-pooling layers to down-sample feature maps and extract essential patterns.
- **Dense Layers**: Two fully connected layers with ReLU activation functions capture complex relationships in the features.
- **Output Layer**: A sigmoid-activated layer produces binary output for classifying images as *defective* or *non-defective*.

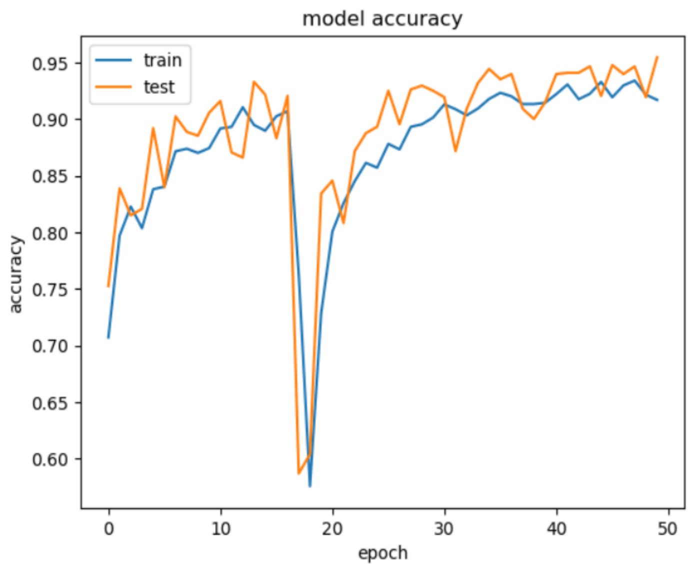
Training and Evaluation

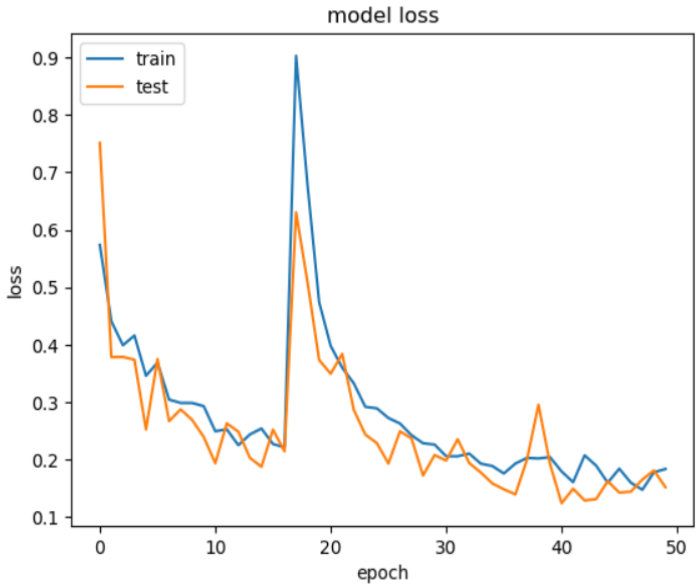
The model was trained using binary cross-entropy loss, optimized with the Adam optimizer. Performance was validated on a separate test dataset.

```
result = cnn_model.fit(training_set,
                       epochs=50,
                       verbose=1,
                       validation_data= test_set)
```

```
Epoch 1/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 157s 1s/step - accuracy: 0.6143 - loss: 0.6861 - val_accuracy: 0.7526 - val_loss: 0.7508
Epoch 2/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 106s 928ms/step - accuracy: 0.7895 - loss: 0.4825 - val_accuracy: 0.8388 - val_loss: 0.3784
Epoch 3/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 103s 901ms/step - accuracy: 0.8273 - loss: 0.3985 - val_accuracy: 0.8150 - val_loss: 0.3788
Epoch 4/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 101s 873ms/step - accuracy: 0.7972 - loss: 0.4335 - val_accuracy: 0.8207 - val_loss: 0.3740
Epoch 5/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 102s 890ms/step - accuracy: 0.8168 - loss: 0.3915 - val_accuracy: 0.8922 - val_loss: 0.2527
Epoch 6/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 102s 883ms/step - accuracy: 0.8495 - loss: 0.3589 - val_accuracy: 0.8400 - val_loss: 0.3752
Epoch 7/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 101s 877ms/step - accuracy: 0.8568 - loss: 0.3221 - val_accuracy: 0.9024 - val_loss: 0.2672
Epoch 8/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 100s 870ms/step - accuracy: 0.8865 - loss: 0.2774 - val_accuracy: 0.8888 - val_loss: 0.2875
Epoch 9/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 106s 924ms/step - accuracy: 0.8694 - loss: 0.2976 - val_accuracy: 0.8854 - val_loss: 0.2692
Epoch 10/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 101s 881ms/step - accuracy: 0.8773 - loss: 0.2883 - val_accuracy: 0.9058 - val_loss: 0.2396
Epoch 11/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 102s 888ms/step - accuracy: 0.8852 - loss: 0.2548 - val_accuracy: 0.9160 - val_loss: 0.1939
Epoch 12/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 103s 890ms/step - accuracy: 0.9047 - loss: 0.2363 - val_accuracy: 0.8706 - val_loss: 0.2632
Epoch 13/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 102s 884ms/step - accuracy: 0.9037 - loss: 0.2247 - val_accuracy: 0.8661 - val_loss: 0.2488
Epoch 14/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 99s 869ms/step - accuracy: 0.9042 - loss: 0.2270 - val_accuracy: 0.9330 - val_loss: 0.2032
Epoch 15/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 99s 861ms/step - accuracy: 0.8975 - loss: 0.2534 - val_accuracy: 0.9217 - val_loss: 0.1878
Epoch 16/50
110/110 ━━━━━━━━━━━━━━━━━━━━ 99s 861ms/step - accuracy: 0.9153 - loss: 0.2099 - val_accuracy: 0.8831 - val_loss: 0.2523
```

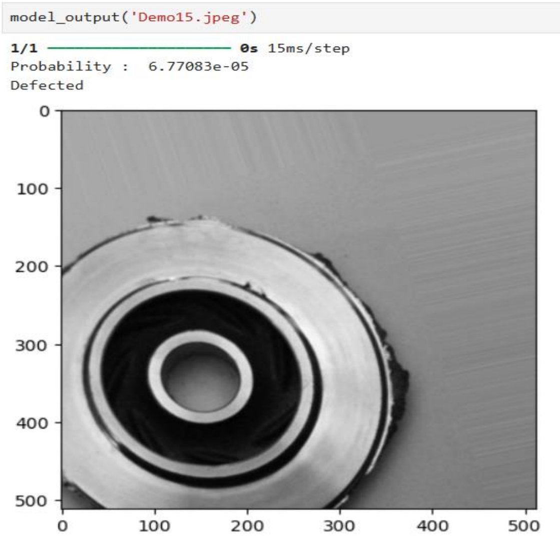
3. Model Performance

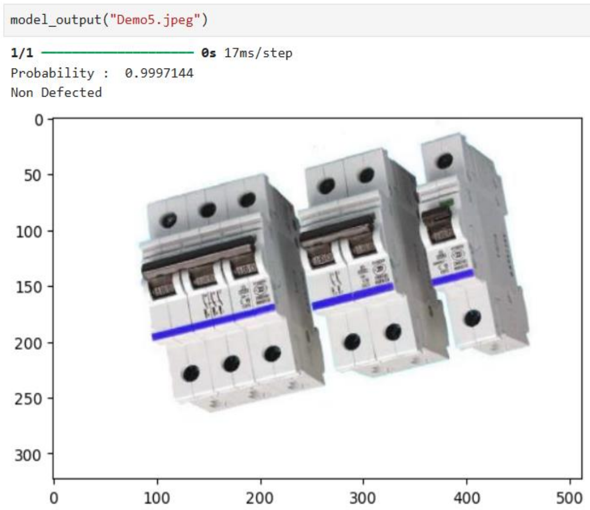




Results

- **Accuracy**: The model achieved an accuracy score of approximately 95%, demonstrating strong predictive capabilities for classifying defective and non-defective equipment.





4. Conclusion

This CNN-based defect detection model successfully classifies industrial equipment images into *defective* and *non-defective* categories with an accuracy of 95%. The project demonstrates the potential of deep learning in automating defect detection, enabling faster and more reliable quality control processes in industrial settings. Future improvements in data quality, data diversity, and model complexity could further enhance performance, making this a scalable solution for defect detection across different industrial applications.