Surface Crack Detection

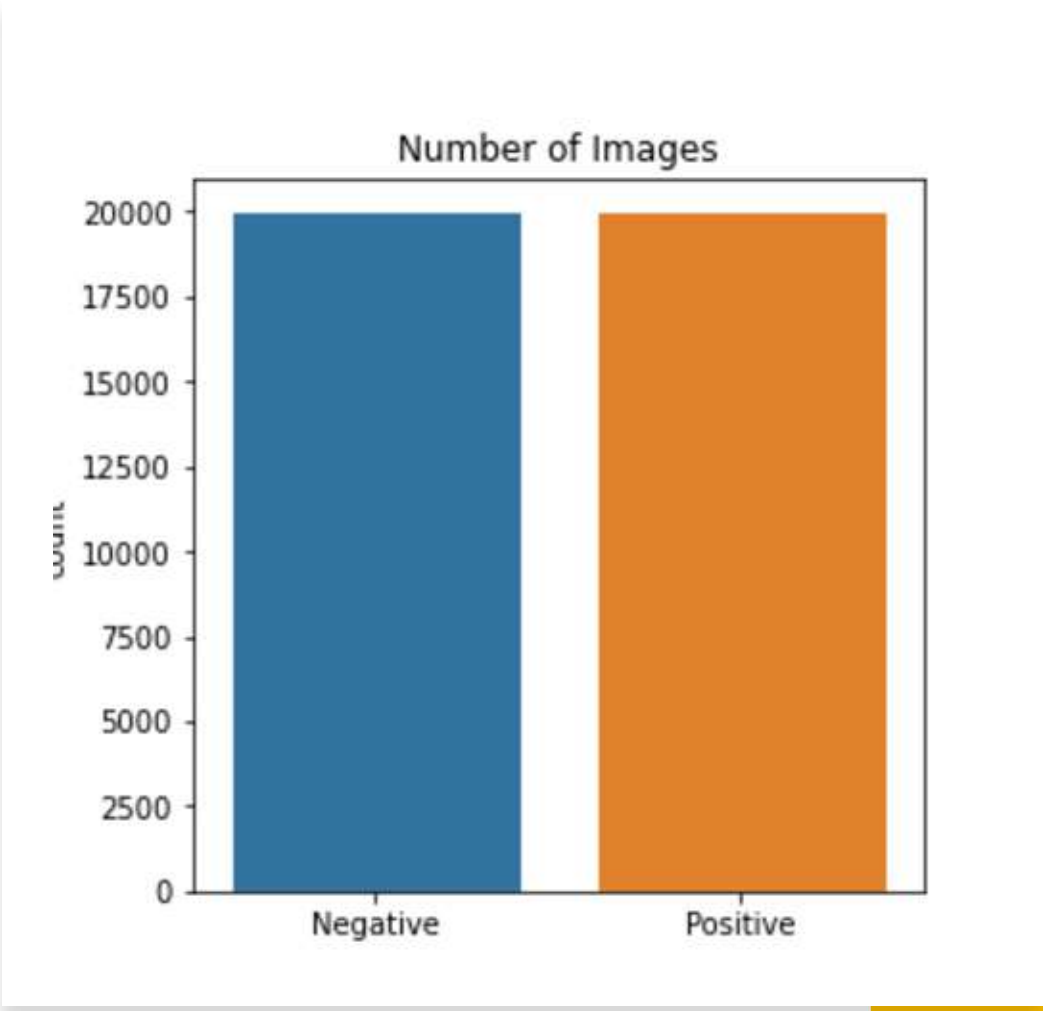
Team Members: Madhumitha Mandyam, Muneendra Magani, Nandhika Rajmanikandan, Ganesh Vannam.

Goal of Study

- The goal of the study is to develop an efficient system for detecting surface cracks in images using convolutional neural networks (CNNs).
- This system aims to address critical issues in industries such as manufacturing and construction by preventing structural failures caused by undetected cracks.
- Additionally, it focuses on leveraging PySpark for large-scale data processing across multiple virtual machines, ensuring scalability and efficiency.

Data Structure

- We collected the data from '**Kaggle'**
- **Total Number of Images**: 40,000 images
- **Classes**: The dataset is divided into **2 classes**:
- **Positive Crack Images**: 20,000 images
- **Negative Crack Images**: 20,000 images



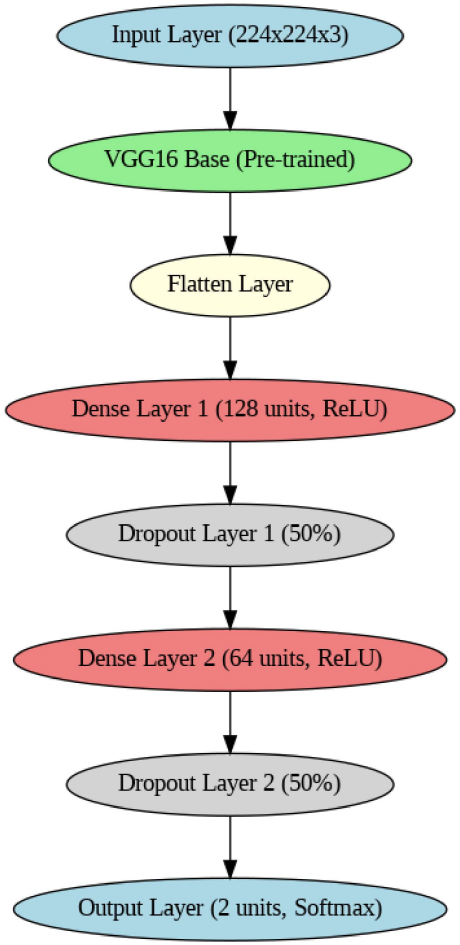
Data Splitting

- **Training Set**: 70% (28,000 images)
- **Validation Set**: 15% (6,000 images)
- **Test Set**: 15% (6,000 images)

Models

- VGG16
- CNN
- RestNet50
- LeNet

VGG 16


Input Layer (224x224x3)
VGG16 Base (Pre-trained)
Flatten Layer
Dense Layer 1 (128 units, ReLU)
Dropout Layer 1 (50%)
Dense Layer 2 (64 units, ReLU)
Dropout Layer 2 (50%)
Output Layer (2 units, Softmax)

Model: "functional_1"

Layer (type)	Output Shape	Param #
input_layer_1 (InputLayer)	(None, 34, 34, 3)	0
block1_conv1 (Conv2D)	(None, 34, 34, 64)	1,792
block1_conv2 (Conv2D)	(None, 34, 34, 64)	36,928
block1_pool (MaxPooling2D)	(None, 17, 17, 64)	0
block2_conv1 (Conv2D)	(None, 17, 17, 128)	73,856
block2_conv2 (Conv2D)	(None, 17, 17, 128)	147,584
block2_pool (MaxPooling2D)	(None, 8, 8, 128)	0
block3_conv1 (Conv2D)	(None, 8, 8, 256)	295,168
block3_conv2 (Conv2D)	(None, 8, 8, 256)	590,080
block3_conv3 (Conv2D)	(None, 8, 8, 256)	590,080
block3_pool (MaxPooling2D)	(None, 4, 4, 256)	0
block4_conv1 (Conv2D)	(None, 4, 4, 512)	1,180,160
block4_conv2 (Conv2D)	(None, 4, 4, 512)	2,359,808
block4_conv3 (Conv2D)	(None, 4, 4, 512)	2,359,808
block4_pool (MaxPooling2D)	(None, 2, 2, 512)	0
block5_conv1 (Conv2D)	(None, 2, 2, 512)	2,359,808
block5_conv2 (Conv2D)	(None, 2, 2, 512)	2,359,808
block5_conv3 (Conv2D)	(None, 2, 2, 512)	2,359,808
block5_pool (MaxPooling2D)	(None, 1, 1, 512)	0
flatten_1 (Flatten)	(None, 512)	0
dense_3 (Dense)	(None, 128)	65,664
dropout_2 (Dropout)	(None, 128)	0
dense_4 (Dense)	(None, 64)	8,256
dropout_3 (Dropout)	(None, 64)	0
dense_5 (Dense)	(None, 2)	130

Total params: 14,788,738 (56.41 MB)
Trainable params: 74,050 (289.26 KB)
Non-trainable params: 14,714,688 (56.13 MB)

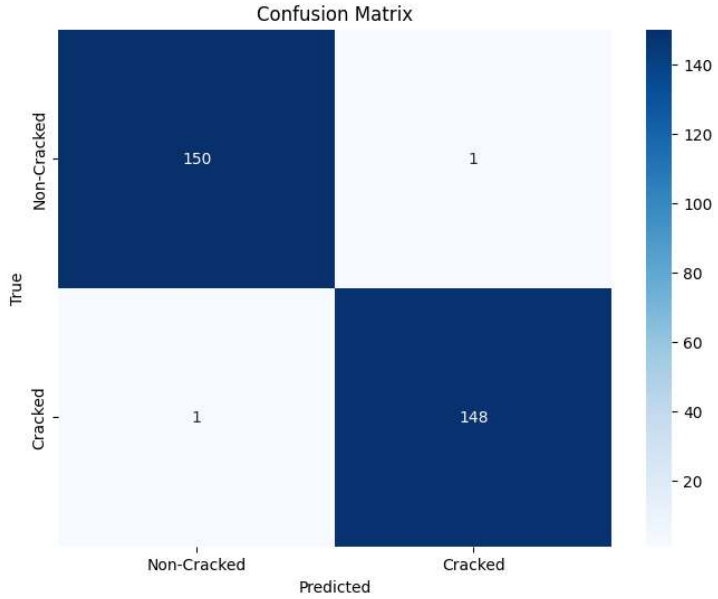
Hyper Parameters

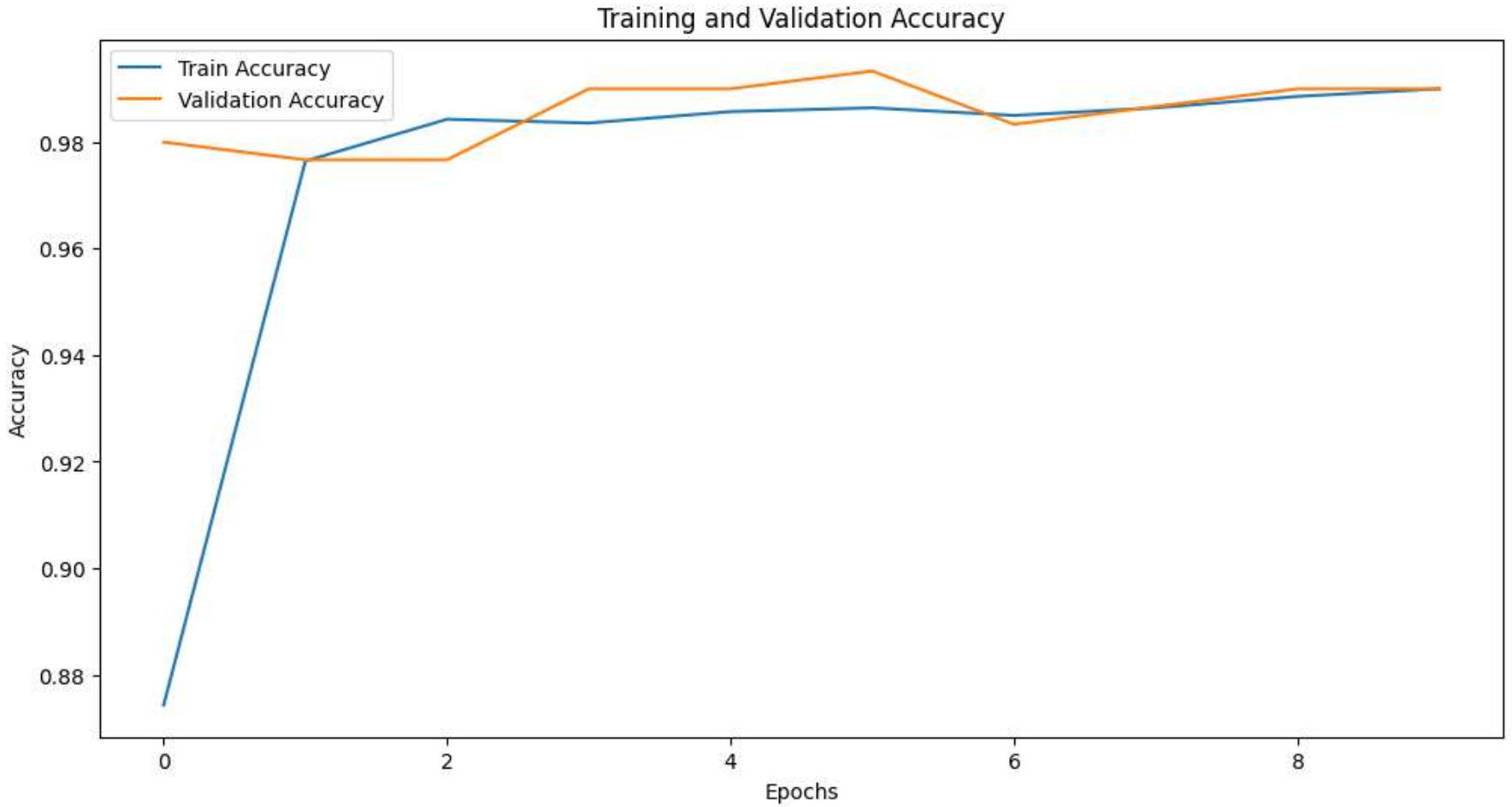
- **Learning Rate**: Start at 0.0001 and use scheduling to decrease it over time for smoother convergence.
- **Grid Search**: Explore different combinations of learning rate, dropout rate, and batch size to find the best configuration.
- **Batch Size**: A batch size of 32 will be used to balance training time and memory usage.
- **Optimizer**: Apply Adam for adaptive learning rates, ensuring faster and more stable convergence.
- **Dropout Rate:** Apply a dropout rate of 0.5 to prevent overfitting during training

Metrics

Classification Report:

	precision	recall	f1-score	support
Non-Cracked	0.99	0.99	0.99	151
Cracked	0.99	0.99	0.99	149
accuracy			0.99	300
macro avg	0.99	0.99	0.99	300
weighted avg	0.99	0.99	0.99	300

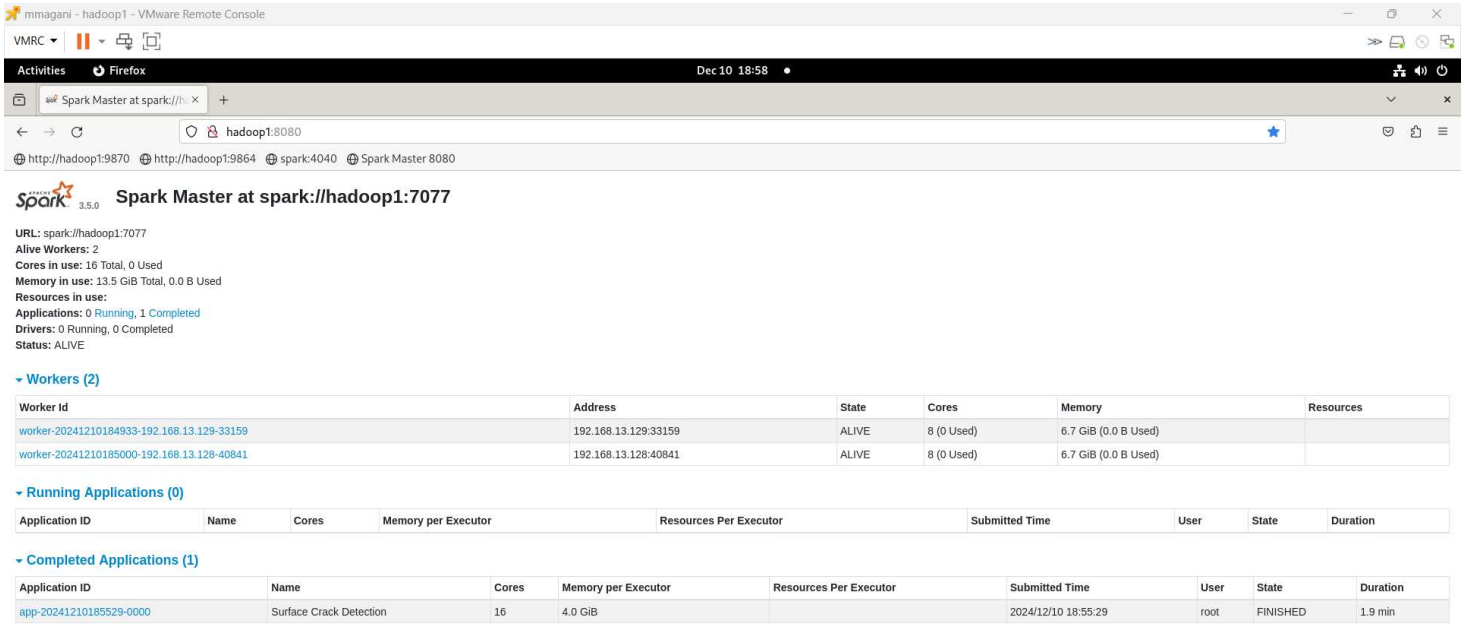




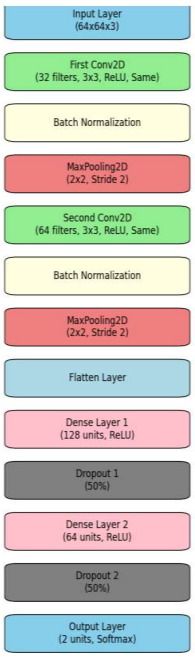
Michigan Tech

VGG 16

Computational Time



CNN



Layer (type)	Output Shape	Param #
input_layer (InputLayer)	(None, 64, 64, 3)	0
conv2d (Conv2D)	(None, 64, 64, 32)	896
batch_normalization (BatchNormalization)	(None, 64, 64, 32)	128
max_pooling2d (MaxPooling2D)	(None, 32, 32, 32)	0
conv2d_1 (Conv2D)	(None, 32, 32, 64)	18,496
batch_normalization_1 (BatchNormalization)	(None, 32, 32, 64)	256
conv2d_1 (Conv2D)	(None, 32, 32, 64)	18,496
batch_normalization_1 (BatchNormalization)	(None, 32, 32, 64)	256
max_pooling2d_1 (MaxPooling2D)	(None, 16, 16, 64)	0
flatten (Flatten)	(None, 16384)	0
dense (Dense)	(None, 128)	2,097,280
dropout (Dropout)	(None, 128)	0
dense_1 (Dense)	(None, 64)	8,256
dropout_1 (Dropout)	(None, 64)	0
dense_2 (Dense)	(None, 2)	130

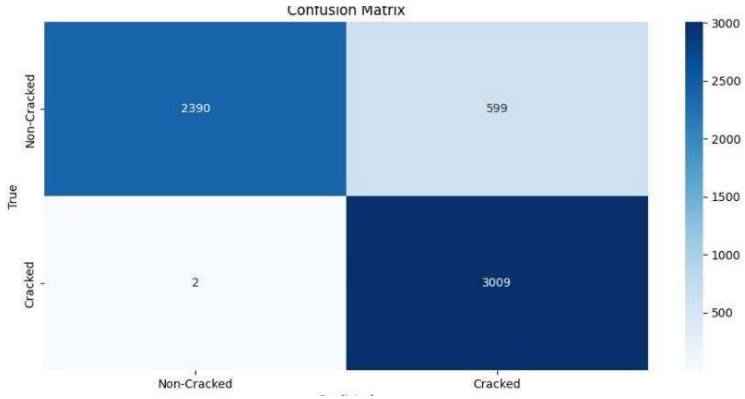
```
Total params: 2,125,442 (8.11 MB)
Trainable params: 2,125,250 (8.11 MB)
Non-trainable params: 192 (768.00 B)
/usr/local/lib/python3.11/site-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: User
Warning: Your `PyDataset` class should call `super().__init__(**kwargs)` in its constructor. `**kwargs`
can include `workers`, `use_multiprocessing`, `max_queue_size`. Do not pass these arguments to `fit()`,
as they will be ignored.
  self._warn_if_super_not_called()
```

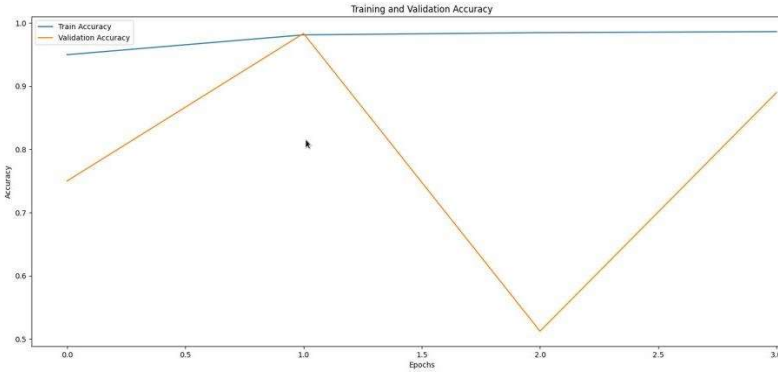
Hyperparameters

- **Learning Rate**: Default value of the Adam optimizer: **0.001** (not explicitly set in the code).
- **Batch Size**: **16** (used in datagen.flow for training).
- **Epochs**: 4 (number of complete passes through the dataset).
- **Dropout Rate**: **0.5** (used after each dense layer to reduce overfitting).
- This rate specifies the fraction of neurons to drop during training.
- **Optimizer**: **Adam** (used for optimization).

Model Performance:

- The model achieved following results:
- Accuracy: 0.98
- Test accuracy:0.98
- Precision:0.96
- Recall:1.00
- F1 Score:0.98

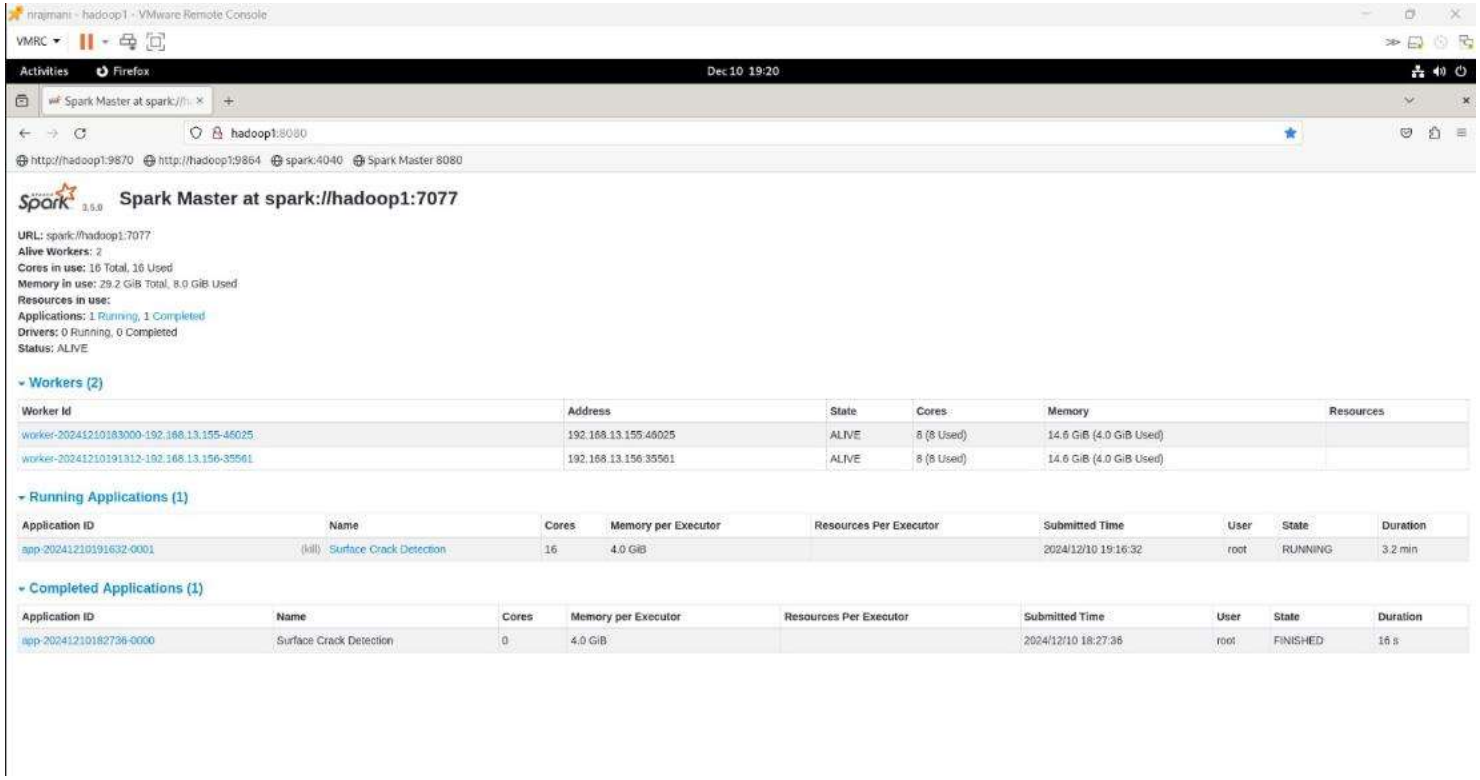


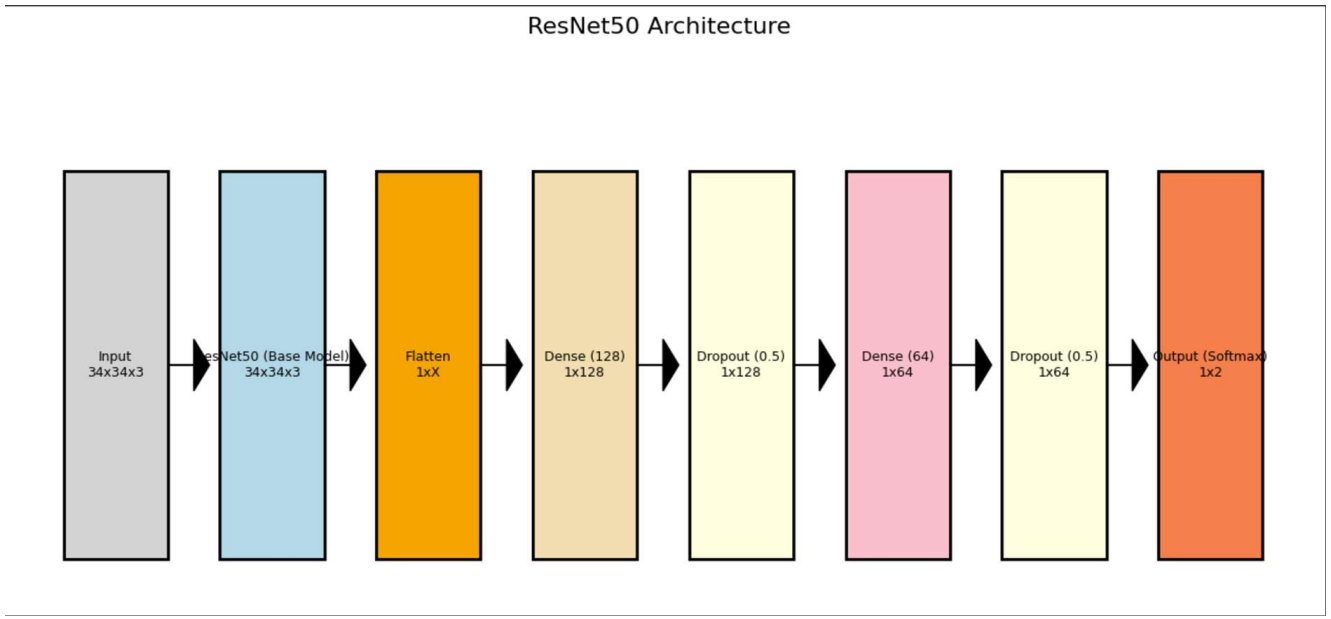


lassification Report:

	precision	recall	f1-score	sup
Non-Cracked	1.00	0.23	0.38	
Cracked	0.57	1.00	0.72	
accuracy			0.62	
macro avg	0.78	0.62	0.55	
eighted avg	0.78	0.62	0.55	

Computational Time




ResNet50 Architecture
Input
34x34x3
esNet50 (Base Model)
34x34x3
Flatten
1xX
Dense (128)
1x128
Dropout (0.5)
1x128
Dense (64)
1x64
Dropout (0.5)
1x64
Output (Softmax)
1x2

RestNet50

Model: "functional"

Layer (type)	Output Shape	Param #	Connected to
input_layer (InputLayer)	(None, 34, 34, 3)	0	-
conv1_pad (ZeroPadding2D)	(None, 40, 40, 3)	0	input_layer[0][0]
conv1_conv (Conv2D)	(None, 17, 17, 64)	9,472	conv1_pad[0][0]
conv1_bn (BatchNormalization)	(None, 17, 17, 64)	256	conv1_conv[0][0]
conv1_relu (Activation)	(None, 17, 17, 64)	0	conv1_bn[0][0]
pool1_pad (ZeroPadding2D)	(None, 19, 19, 64)	0	conv1_relu[0][0]
pool1_pool (MaxPooling2D)	(None, 9, 9, 64)	0	pool1_pad[0][0]
conv2_block1_1_conv (Conv2D)	(None, 9, 9, 64)	4,160	pool1_pool[0][0]
conv2_block1_1_bn (BatchNormalization)	(None, 9, 9, 64)	256	conv2_block1_1_conv[0…
conv2_block1_1_relu (Activation)	(None, 9, 9, 64)	0	conv2_block1_1_bn[0][…
conv2_block1_2_conv (Conv2D)	(None, 9, 9, 64)	36,928	conv2_block1_1_relu[0…
conv5_block3_2_relu (Activation)	(None, 2, 2, 512)	0	conv5_block3_2_bn[0][…
conv5_block3_3_conv (Conv2D)	(None, 2, 2, 2048)	1,050,624	conv5_block3_2_relu[0…
conv5_block3_3_bn (BatchNormalization)	(None, 2, 2, 2048)	8,192	conv5_block3_3_conv[0…
conv5_block3_add (Add)	(None, 2, 2, 2048)	0	conv5_block2_out[0][0… conv5_block3_3_bn[0][…
conv5_block3_out (Activation)	(None, 2, 2, 2048)	0	conv5_block3_add[0][0]
flatten (Flatten)	(None, 8192)	0	conv5_block3_out[0][0]
dense (Dense)	(None, 128)	1,048,704	flatten[0][0]
dropout (Dropout)	(None, 128)	0	dense[0][0]
dense_1 (Dense)	(None, 64)	8,256	dropout[0][0]
dropout_1 (Dropout)	(None, 64)	0	dense_1[0][0]
dense_2 (Dense)	(None, 2)	130	dropout_1[0][0]

Total params: 24,644,802 (94.01 MB)
Trainable params: 1,057,090 (4.03 MB)
Non-trainable params: 23,587,712 (89.98 MB)

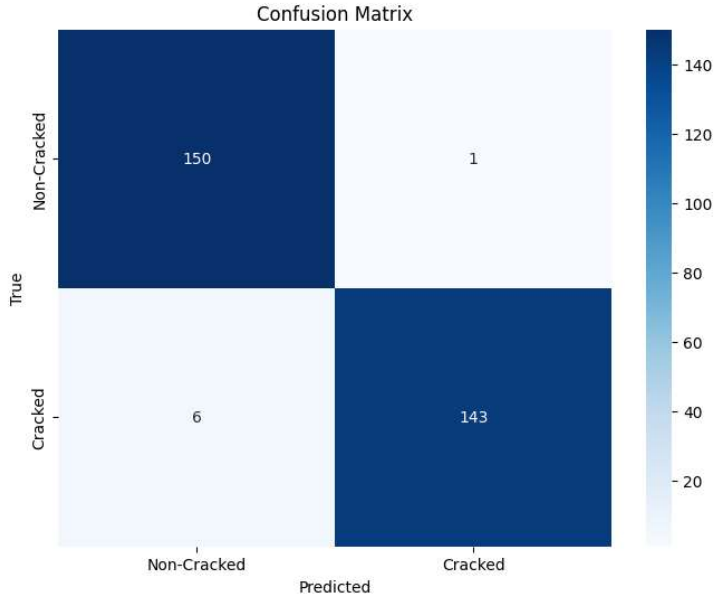
Hyper Parameters

- **Learning Rate**: Default value of the Adam optimizer: **0.001**
- **Batch Size**: **16** (a balance between training speed and stability).
- **Epochs**: **10** (To balance the computational time).
- **Dropout Rate**: **0.5** (used after each dense layer to reduce overfitting).
- **Optimizer**: **Adam** (used for optimization).

Metrics

Classification Report:

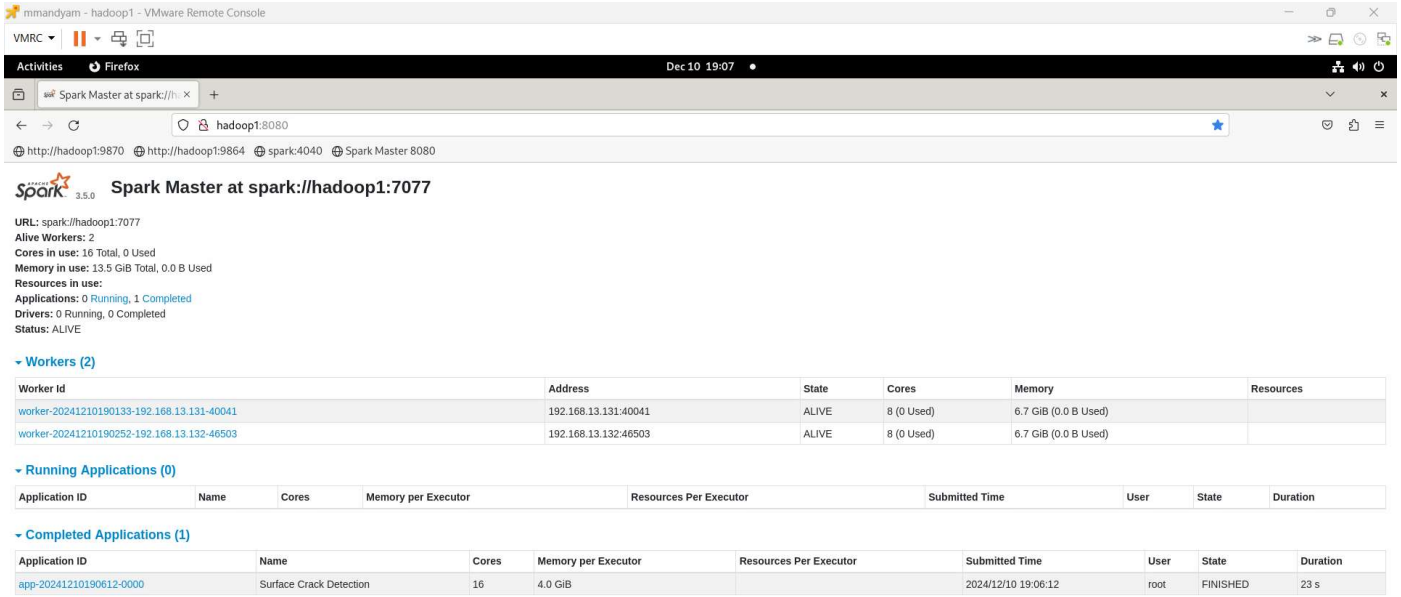
	precision	recall	f1-score	support
Non-Cracked	0.96	0.99	0.98	151
Cracked	0.99	0.96	0.98	149
accuracy			0.98	300
macro avg	0.98	0.98	0.98	300
weighted avg	0.98	0.98	0.98	300




Training and Validation Accuracy
Accuracy
Epochs
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
0
2
4
6
8
Train Accuracy
Validation Accuracy

RestNet50

Computational Time

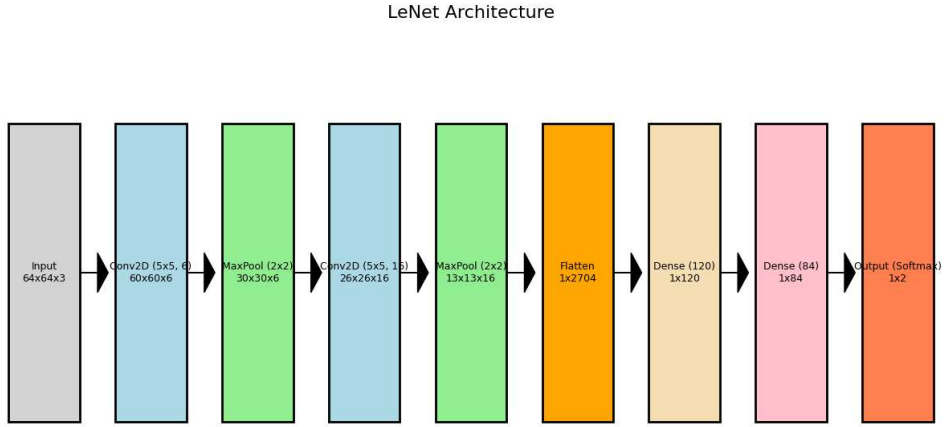


LeNet Model

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 60, 60, 6)	456
max_pooling2d (MaxPooling2D)	(None, 30, 30, 6)	0
conv2d_1 (Conv2D)	(None, 26, 26, 16)	2,416
max_pooling2d_1 (MaxPooling2D)	(None, 13, 13, 16)	0
flatten (Flatten)	(None, 2704)	0
dense (Dense)	(None, 120)	324,600
dense_1 (Dense)	(None, 84)	10,164
dense_2 (Dense)	(None, 2)	170

Total params: 337,806 (1.29 MB)
Trainable params: 337,806 (1.29 MB)
Non-trainable params: 0 (0.00 B)

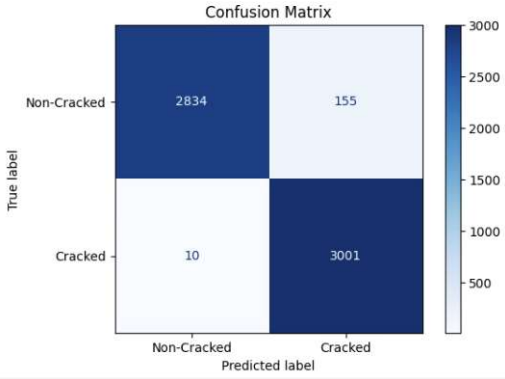


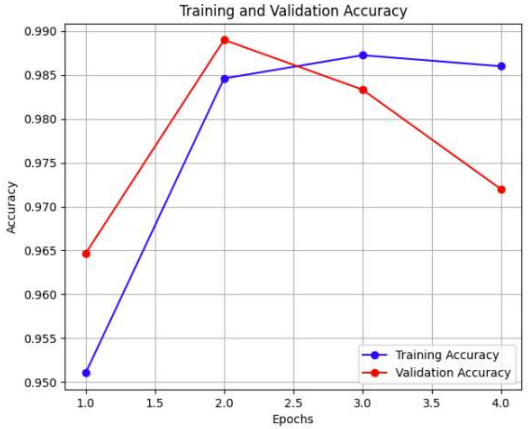
Hyper Parameters

- **Batch Size:** A batch size of 64 is chosen, balancing training time and memory usage for effective learning.
- **Learning Rate:** Default learning rate of the Adam optimizer is used, allowing adaptive learning. Fine-tuning or scheduling could further improve convergence.
- **Epochs:** The model is trained for 4 epochs to balance computational time with performance, though this can be extended for improved results.
- **Optimizer:** The Adam optimizer is employed to leverage adaptive learning rates, ensuring faster and stable convergence during training.

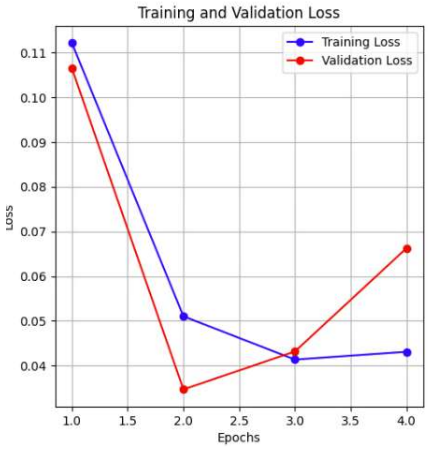
Model Performance:

- The model achieved following results:
  - Accuracy: 0.96
  - Test accuracy:0.97
  - Precision:0.95
  - Recall:0.97
  - F1 Score:0.97

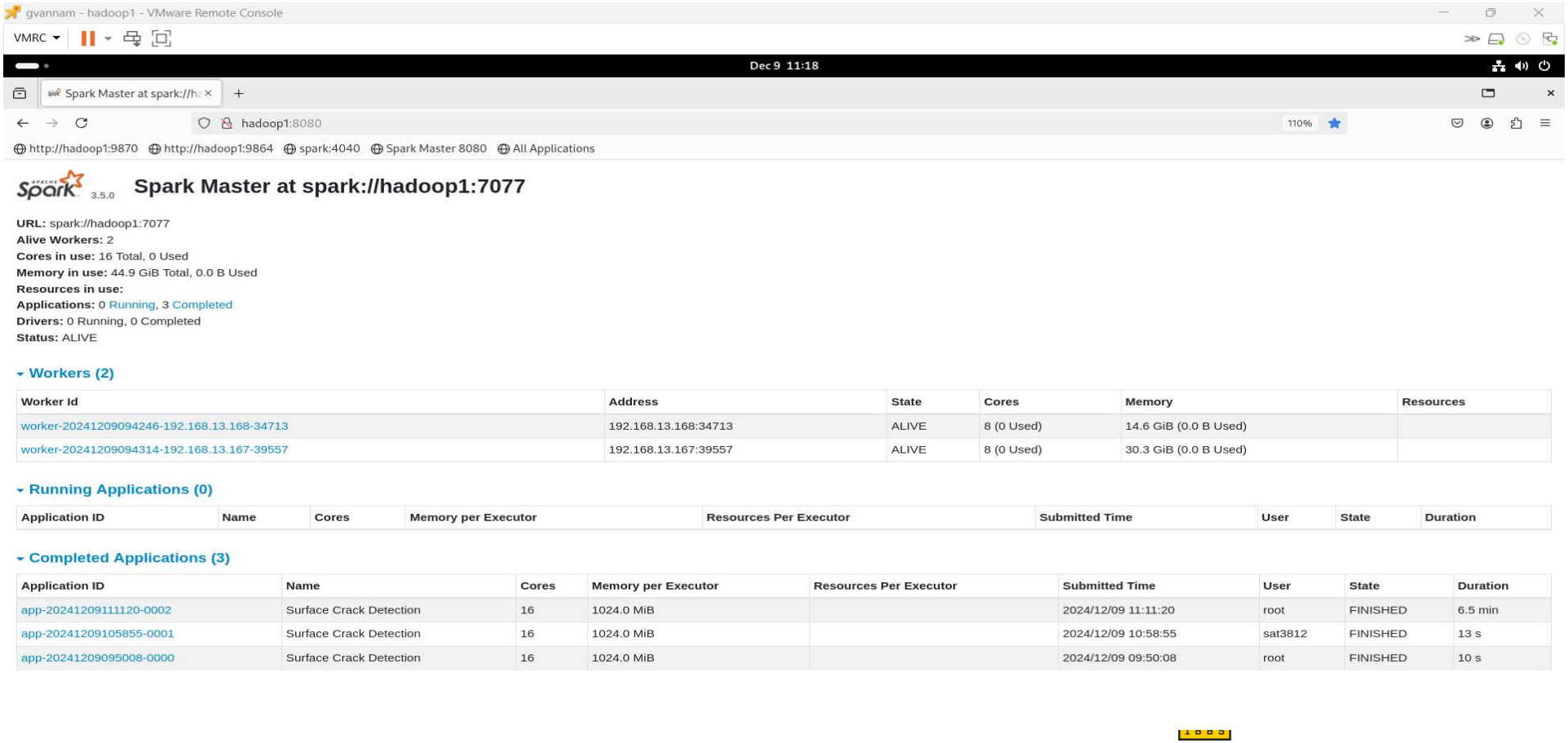




	precision	recall	f1-score	support
Non-Cracked	1.00	0.96	0.98	2989
Cracked	0.96	1.00	0.98	3011
accuracy			0.98	6000
macro avg	0.98	0.98	0.98	6000
weighted avg	0.98	0.98	0.98	6000



Computational Time



Best Model

- **Best Model: VGG16**
- **Reason**:
  - **Fatal Errors**: Only 1 cracked surface was misclassified as non-cracked.
  - **Metrics**:
    - Recall for cracked surfaces is **0.99**, meaning it identified almost all cracks.
    - Precision is also **0.99**, indicating high reliability.
  - **Confusion Matrix**: The misclassification rate is minimal, with just 1 error for cracked surfaces.
- **Recall for "Cracked"**: 0.95 (missed 7 cracked surfaces).
  - **Precision for "Cracked"**: 0.95.
  - Overall accuracy: **0.97**.