TRAINING A CNN MODEL—

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense

from tensorflow.keras.preprocessing.image import ImageDataGenerator

# Preprocessing and loading images using ImageDataGenerator

train\_datagen = ImageDataGenerator(rescale=1./255, validation\_split=0.2)

train\_generator = train\_datagen.flow\_from\_directory(

'data/train/', # directory containing training images

target\_size=(150, 150),

batch\_size=32,

class\_mode='binary',

subset='training'

)

validation\_generator = train\_datagen.flow\_from\_directory(

'data/train/', # same directory

target\_size=(150, 150),

batch\_size=32,

class\_mode='binary',

subset='validation'

)

# Create a CNN model

model = Sequential([

Conv2D(32, (3, 3), activation='relu', input\_shape=(150, 150, 3)),

MaxPooling2D(2, 2),

Conv2D(64, (3, 3), activation='relu'),

MaxPooling2D(2, 2),

Flatten(),

Dense(128, activation='relu'),

Dense(1, activation='sigmoid') # Output layer: 1 for crack/no-crack classification

])

model.compile(optimizer='adam', loss='binary\_crossentropy', metrics=['accuracy'])

# Train the model

model.fit(

train\_generator,

steps\_per\_epoch=train\_generator.samples // train\_generator.batch\_size,

epochs=10,

validation\_data=validation\_generator,

validation\_steps=validation\_generator.samples // validation\_generator.batch\_size

)

# Save the model

model.save('crack\_detection\_model.h5')