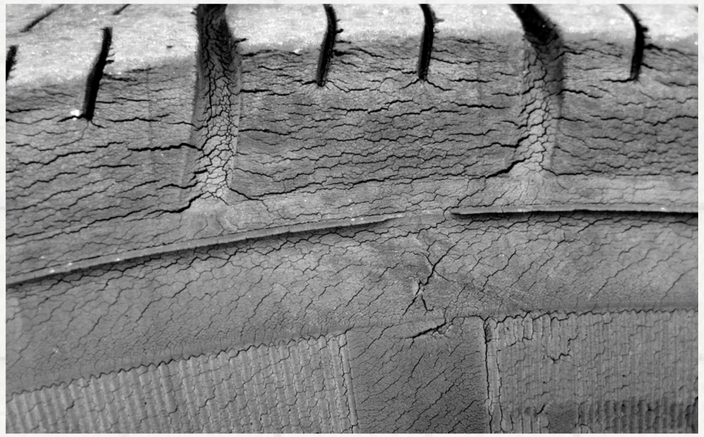
Tire Condition Predictor

Suppakit Laomahamek 6438232221

Origin of the project

I once experienced a tire blowout on my way to Chula, likely caused by the condition of the tire. This incident inspired me to develop a model that can distinguish between good and defective tires.

Model Design



01. Able to tell good and defective tires apart.
The main focus is to distinguish tires that have cracking surfaces.

02. Able to receive input picture of any orientation and lighting condition of the tire surfaces.

Process

01

Read Image, Label
Resize to 224x224

02

Convert to grayscale for model simplicity
Normalize for faster training

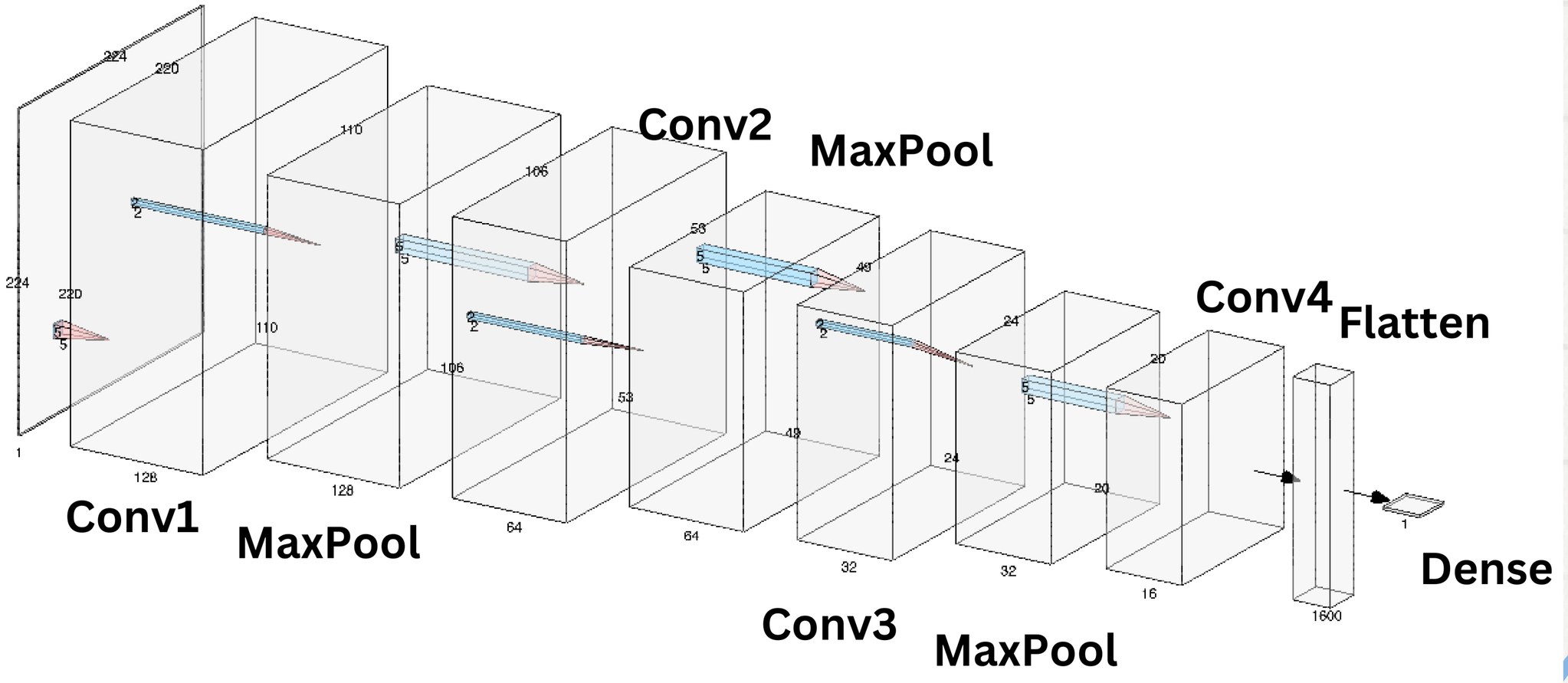
03

Convolutional Neural Network

04

Output as probability of being a good tire

Model Design



Data Collection

Kaggle Dataset

Google Image

Taking Real Photos

Training Process

```python
earlystopping=tf.keras.callbacks.EarlyStopping(
    monitor='accuracy',
    min_delta=0.0001,
    patience=5,
    restore_best_weights=True,
    start_from_epoch=5,
    verbose=1,
)

chkpt = tf.keras.callbacks.ModelCheckpoint(
    'best_checkpoint.keras',
    monitor='accuracy',
    save_best_only=True,
    mode='max',
    save_freq='epoch',
    verbose=1
```

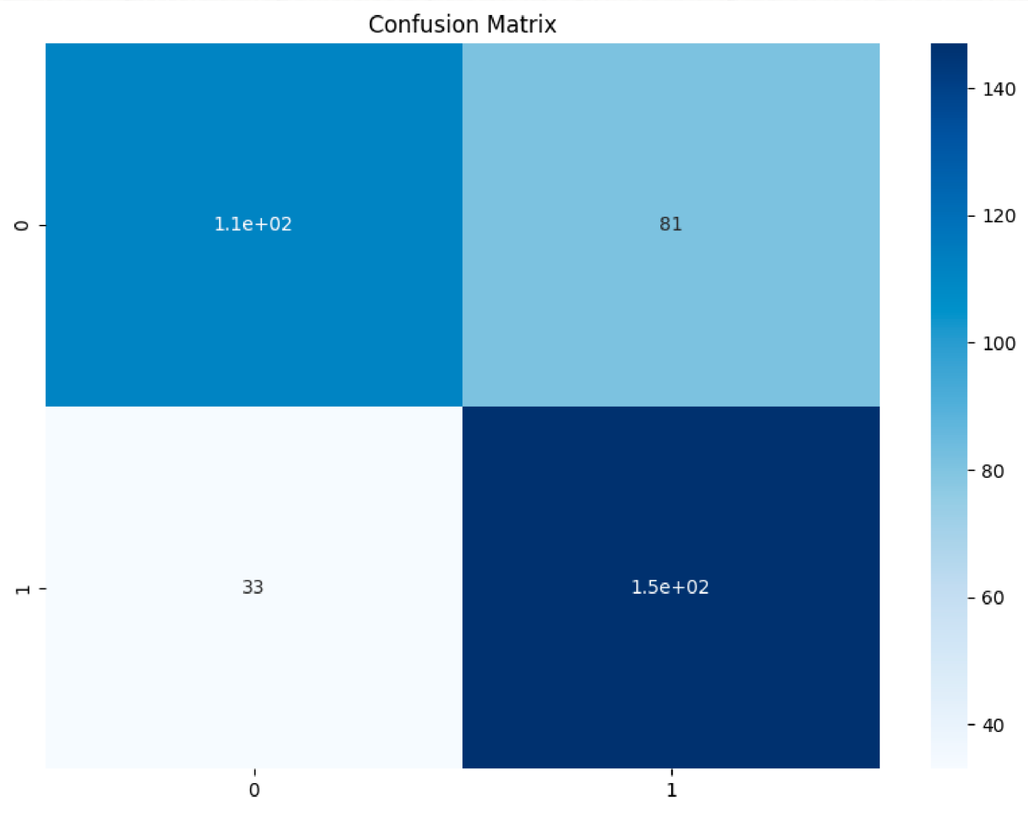
```
model.fit(x_train,y_train,epochs=50,callbacks=[earlystopping,chkpt])
✓ 10m 18.5s

Epoch 1/50
47/47 ━━━━━━━━━━━━━━━━━━━━ 0s 329ms/step - accuracy: 0.6070 - loss: 2.1221
Epoch 1: accuracy improved from -inf to 0.63208, saving model to best_checkpoint.keras
47/47 ━━━━━━━━━━━━━━━━━━━━ 19s 331ms/step - accuracy: 0.6075 - loss: 2.1194
Epoch 2/50
47/47 ━━━━━━━━━━━━━━━━━━━━ 0s 318ms/step - accuracy: 0.7015 - loss: 1.6135
Epoch 2: accuracy improved from 0.63208 to 0.69474, saving model to best_checkpoint.keras
47/47 ━━━━━━━━━━━━━━━━━━━━ 15s 319ms/step - accuracy: 0.7014 -
Epoch 3/50
47/47 ━━━━━━━━━━━━━━━━━━━━ 0s 318ms/step - accuracy: 0.7179 -
Epoch 3: accuracy improved from 0.69474 to 0.71361, saving model to best_checkpoint.keras
47/47 ━━━━━━━━━━━━━━━━━━━━ 15s 319ms/step - accuracy: 0.7178 - loss: 1.3304
Epoch 4/50
47/47 ━━━━━━━━━━━━━━━━━━━━ 0s 319ms/step - accuracy: 0.7291 - loss: 1.1656
Epoch 4: accuracy improved from 0.71361 to 0.73720, saving model to best_checkpoint.keras
47/47 ━━━━━━━━━━━━━━━━━━━━ 15s 320ms/step - accuracy: 0.7293 - loss: 1.1649
Epoch 5/50
47/47 ━━━━━━━━━━━━━━━━━━━━ 0s 318ms/step - accuracy: 0.7332 - loss: 1.0361
Epoch 5: accuracy improved from 0.73720 to 0.74933, saving model to best_checkpoint.keras
47/47 ━━━━━━━━━━━━━━━━━━━━ 15s 319ms/step - accuracy: 0.7335 - loss: 1.0353
Epoch 6/50
47/47 ━━━━━━━━━━━━━━━━━━━━ 0s 317ms/step - accuracy: 0.7751 - loss: 0.9044
Epoch 6: accuracy improved from 0.74933 to 0.76415, saving model to best_checkpoint.keras
47/47 ━━━━━━━━━━━━━━━━━━━━ 15s 318ms/step - accuracy: 0.7749 - loss: 0.9041
Epoch 7/50
...
Epoch 41: accuracy did not improve from 0.97035
47/47 ━━━━━━━━━━━━━━━━━━━━ 15s 314ms/step - accuracy: 0.9745 - loss: 0.2506
Epoch 41: early stopping
Restoring model weights from the end of the best epoch: 36.
```

checkpoint

earlystop

Testing Results



69.4%
Accuracy

.721
F1 Score

.644
Precision

.817
Recall

Testing Results

Thank you very much!