Rice grain Classification using CNN Model

By Rahul Inchal

Introduction:

Rice is a staple food consumed globally, and its classification is essential for quality control, pricing, and export purposes. In this project, we aim to build a robust image classification system that identifies five varieties of rice grains using CNN model.

1. **Arborio**
2. **Basmati**
3. **Ipsala**
4. **Jasmine**
5. **Karacadag**

Data Set.

Link: https://www.kaggle.com/datasets/muratkokludataset/rice-image-dataset
There are 5 classes of the data which are as follows.

Methodology:

1. Data Augmentation:

IMAGE_SIZE = 224
BATCH_SIZE = 32

- Rescaling -
- Shear Range
- Zoom Range
- Horizontal Flip
- Validation split = 10% of the training data

2. Train Generator and Validation Generator

- Target_size = (IMAGE_SIZE, IMAGE_SIZE)
- Color_mode = ‘grayscale'
- batch_size = BATCH_SIZE
- class_mode = ‘categorical'
- subset = ‘training'
- shuffle = True

Hidden Layers:

1. First layer: Conv2D

- filters = 64
- kernel_size = 3
- strides = 2
- padding = “same"
- activation = ‘relu'
- input_shape = (224,224,3)

2. Second Layer: MaxPool2D

- pool_size = 2
- strides = 2

3. Third Layer: Conv2D

- filters = 16
- kernel_size = 3
- strides = 2
- padding = “same”
- activation = 'relu'

4. Fourth Layer: MaxPool2D

- pool_size = 2
- strides = 2

5. Fifth Layer: Conv2D

- filters = 32
- kernel_size = 3
- strides = 2
- padding = “same"
- activation = 'relu'

6. Sixth Layer: MaxPool2D

- pool_size = 2

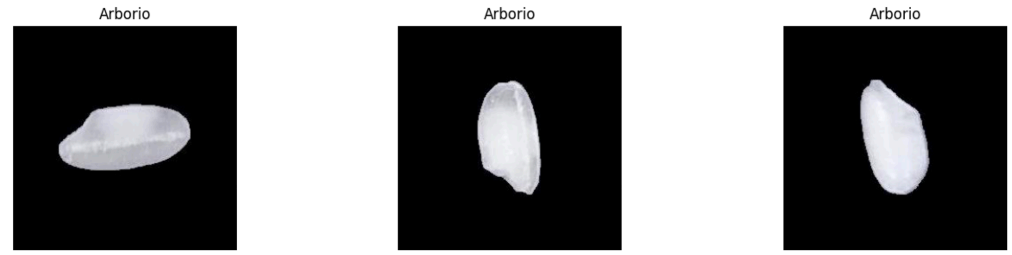
7. Seventh Layer: Flatten: Converts 3D features to 2D

8. Eighth Layer: Dense

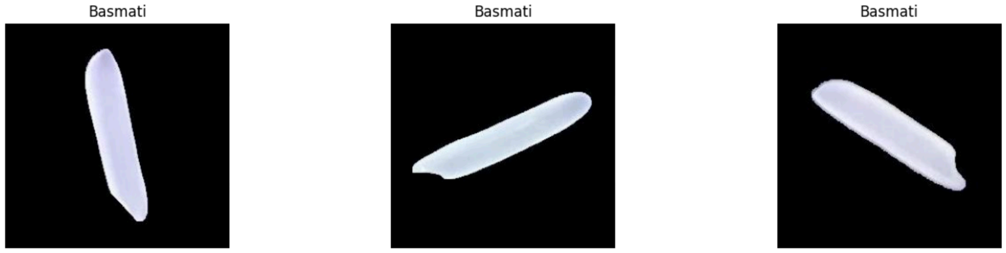
- Unit = 8, as there are 5 classes
- Activation function = Softmax

Classes:

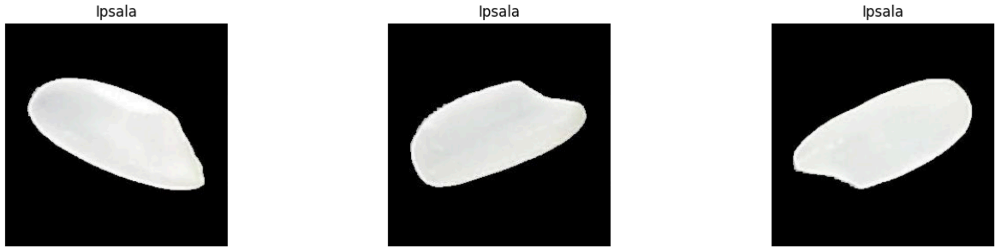
1. Arborio



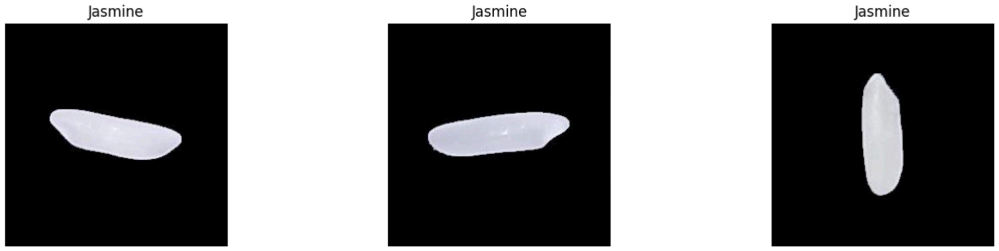
2. Basmati



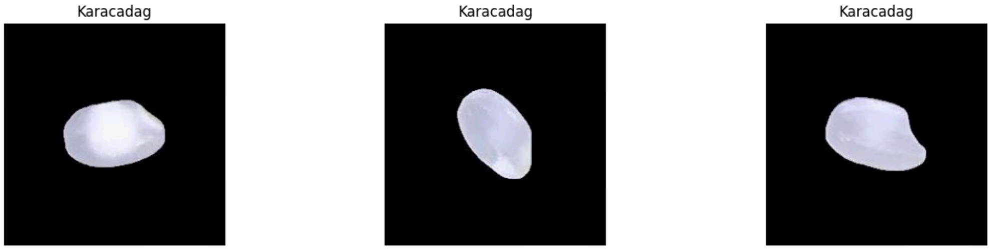
3. Ipsala



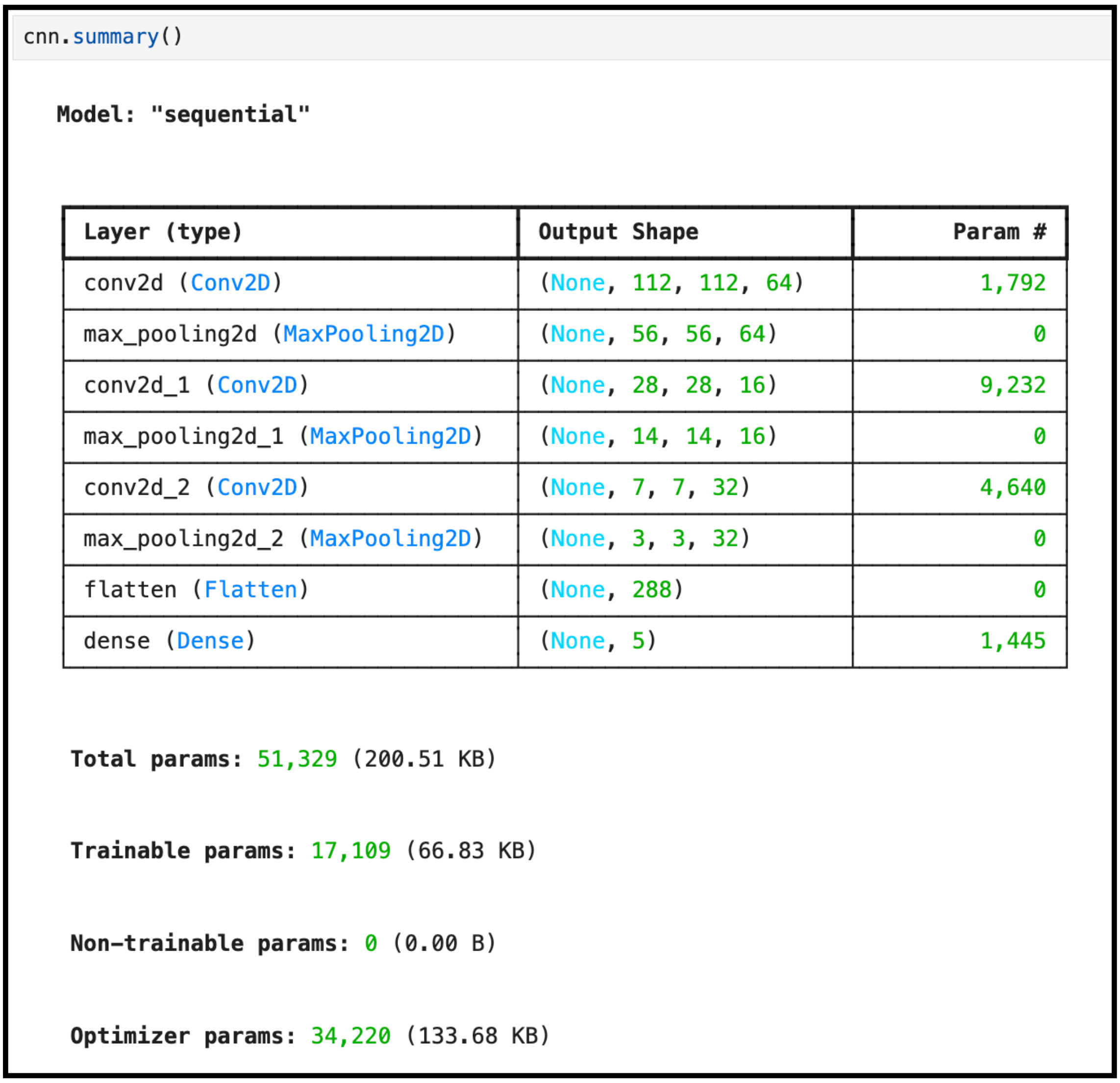
4. Jasmine



5. Karacadag



Summary:



```
cnn.summary()
```

```
Model: "sequential"

┌─────────────────────────────────┬────────────────────────┬───────────────┐
│ Layer (type)                    │ Output Shape           │       Param # │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ conv2d (Conv2D)                 │ (None, 112, 112, 64)   │         1,792 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ max_pooling2d (MaxPooling2D)    │ (None, 56, 56, 64)     │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ conv2d_1 (Conv2D)               │ (None, 28, 28, 16)     │         9,232 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ max_pooling2d_1 (MaxPooling2D)  │ (None, 14, 14, 16)     │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ conv2d_2 (Conv2D)               │ (None, 7, 7, 32)       │         4,640 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ max_pooling2d_2 (MaxPooling2D)  │ (None, 3, 3, 32)       │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ flatten (Flatten)               │ (None, 288)            │             0 │
├─────────────────────────────────┼────────────────────────┼───────────────┤
│ dense (Dense)                   │ (None, 5)              │         1,445 │
└─────────────────────────────────┴────────────────────────┴───────────────┘

 Total params: 51,329 (200.51 KB)

 Trainable params: 17,109 (66.83 KB)

 Non-trainable params: 0 (0.00 B)

 Optimizer params: 34,220 (133.68 KB)
```

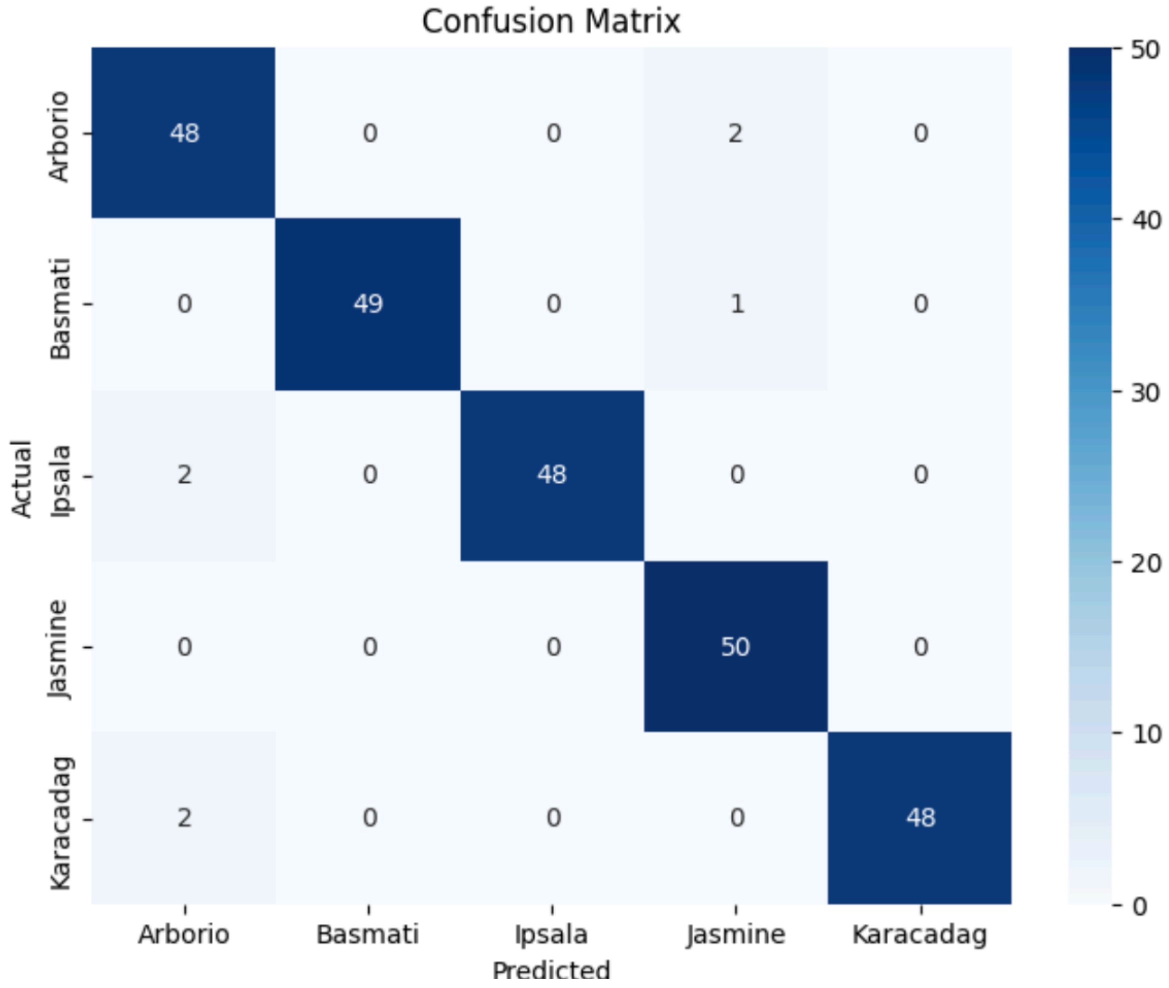
Observation:

1. Feature Extraction:
   - The model starts with convolutional layers to extract spatial features (edges, shapes, textures).
   - Pooling layers reduce spatial size to make the model more efficient and robust to position changes.

2. Flattening:
   - The final convolutional output (a 3D volume) is flattened to a 1D vector to feed into dense layers.

3. Classification:
   - A dense (fully connected) layer outputs a vector of 5 values (one for each class).
   - With a softmax activation (usually), this becomes a probability distribution over the 5 classes.

Evaluation Matrix:

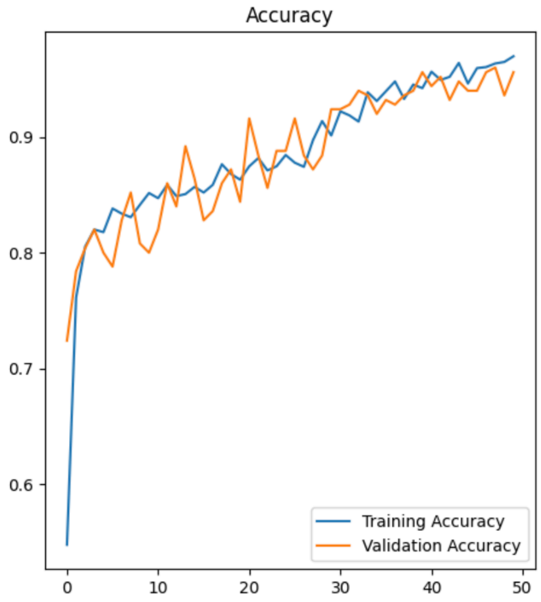


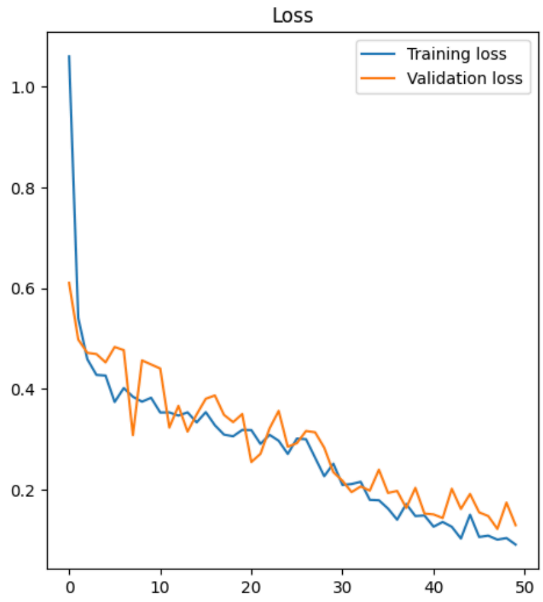
Confusion Matrix

Actual \ Predicted	Arborio	Basmati	Ipsala	Jasmine	Karacadag
Arborio	48	0	0	2	0
Basmati	0	49	0	1	0
Ipsala	2	0	48	0	0
Jasmine	0	0	0	50	0
Karacadag	2	0	0	0	48

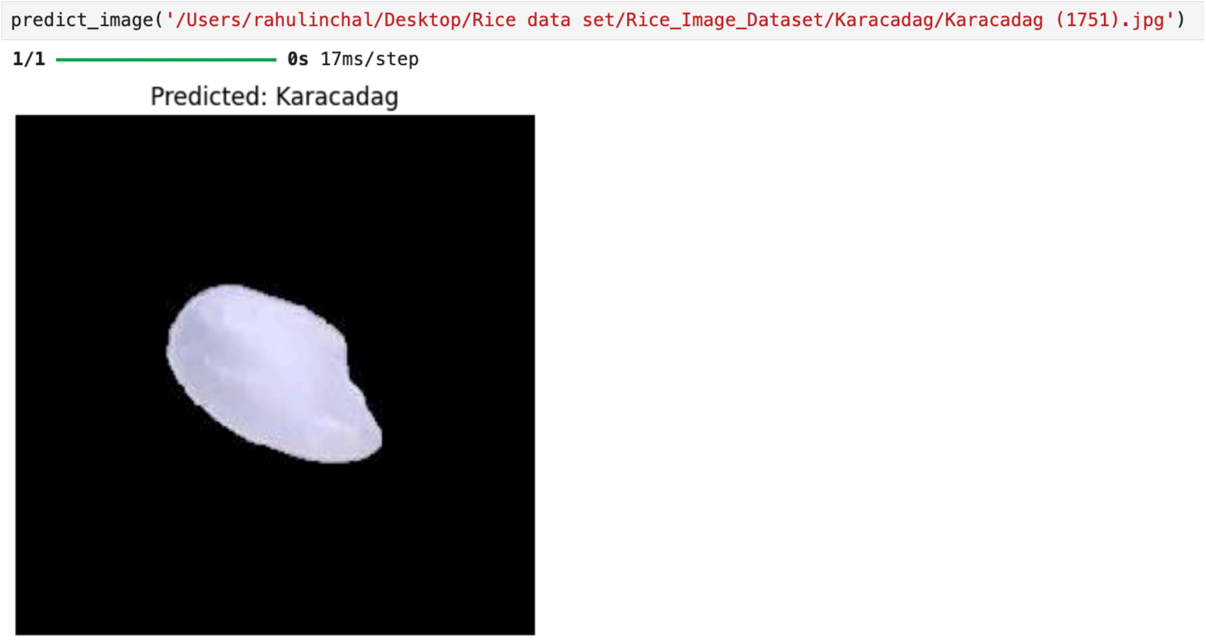
Model performance:

1. Accuracy: 96.98 %
2. Validation Accuracy: 95.6 %
3. Loss: 0.09
4. Validation Loss: 0.13





Prediction:



```
predict_image('/Users/rahulinchal/Desktop/Rice data set/Rice_Image_Dataset/Karacadag/Karacadag (1751).jpg')
```

```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 17ms/step
```

Predicted: Karacadag

Observation:

We can observe that the model was asked to predict the Rice grain class Karacadag and it has accurately predicted.
Also from the evaluation matrix we can see that there are very few misclassified labels are presents.
The model is predicting very good With very less errors and hence can go to the production.