```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/dri
```

```
import os
import numpy as np
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.optimizers import Adam
from sklearn.model_selection import train_test_split
import matplotlib.pyplot as plt
```

```
train_dir = '/content/drive/MyDrive/split_data/train'
val_dir = '/content/drive/MyDrive/split_data/val'
test_dir = '/content/drive/MyDrive/split_data/test'
```

```
train_gen = ImageDataGenerator(rescale=1./255)
val_gen = ImageDataGenerator(rescale=1./255)
test_gen = ImageDataGenerator(rescale=1./255)
```

```
train_data = train_gen.flow_from_directory(train_dir, target_size=(224, 224), batch_size=32, class_mode='cat
val_data = val_gen.flow_from_directory(val_dir, target_size=(224, 224), batch_size=32, class_mode='categoric
```

```
Found 4295 images belonging to 8 classes.
Found 904 images belonging to 8 classes.
```

```
# Base model: VGG16 (without top/fully connected layers)
base_model = tf.keras.applications.VGG16(weights="imagenet", include_top=False, input_shape=(224, 224, 3))
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/vgg16/vgg16_weights_
58889256/58889256 ━━━━━━━━━━━━━━━━━━━━ 0s 0us/step
```

```
base_model.summary()
```

```
Model: "vgg16"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                         ┃ Output Shape                ┃         Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩
│ input_layer (InputLayer)             │ (None, 224, 224, 3)         │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block1_conv1 (Conv2D)                │ (None, 224, 224, 64)        │           1,792 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block1_conv2 (Conv2D)                │ (None, 224, 224, 64)        │          36,928 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block1_pool (MaxPooling2D)           │ (None, 112, 112, 64)        │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block2_conv1 (Conv2D)                │ (None, 112, 112, 128)       │          73,856 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block2_conv2 (Conv2D)                │ (None, 112, 112, 128)       │         147,584 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block2_pool (MaxPooling2D)           │ (None, 56, 56, 128)         │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block3_conv1 (Conv2D)                │ (None, 56, 56, 256)         │         295,168 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block3_conv2 (Conv2D)                │ (None, 56, 56, 256)         │         590,080 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block3_conv3 (Conv2D)                │ (None, 56, 56, 256)         │         590,080 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block3_pool (MaxPooling2D)           │ (None, 28, 28, 256)         │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block4_conv1 (Conv2D)                │ (None, 28, 28, 512)         │       1,180,160 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block4_conv2 (Conv2D)                │ (None, 28, 28, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block4_conv3 (Conv2D)                │ (None, 28, 28, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block4_pool (MaxPooling2D)           │ (None, 14, 14, 512)         │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block5_conv1 (Conv2D)                │ (None, 14, 14, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block5_conv2 (Conv2D)                │ (None, 14, 14, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block5_conv3 (Conv2D)                │ (None, 14, 14, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block5_pool (MaxPooling2D)           │ (None, 7, 7, 512)           │               0 │
└──────────────────────────────────────┴─────────────────────────────┴─────────────────┘
 Total params: 14,714,688 (56.13 MB)
```

```
# Freeze all layers in the base model- weights of these will not be updated during training. so preserves
for layer in base_model.layers:
    layer.trainable = False
```

```
x = base_model.output
x = Flatten()(x)  #convert 3d output to 1d vector, for dense layers
x = Dense(512, activation='relu')(x)  #fully connected layer with 512 neurons
x = Dropout(0.5)(x)  #drop 50% neurons to avoid overfitting
output = Dense(8, activation='softmax')(x)  # 8 output classes for 8 blood groups
```

```
model = tf.keras.Model(inputs=base_model.input, outputs=output)
```

```
# Compile the model
model.compile(optimizer=Adam(learning_rate=0.0001), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
model.summary()
```

```
Model: "functional"
┏━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━━━━━━━━━━━━━┳━━━━━━━━━━━━━━━━━┓
┃ Layer (type)                         ┃ Output Shape                ┃         Param # ┃
┡━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━━━━━━━━━━━━━╇━━━━━━━━━━━━━━━━━┩
│ input_layer (InputLayer)             │ (None, 224, 224, 3)         │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block1_conv1 (Conv2D)                │ (None, 224, 224, 64)        │           1,792 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block1_conv2 (Conv2D)                │ (None, 224, 224, 64)        │          36,928 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block1_pool (MaxPooling2D)           │ (None, 112, 112, 64)        │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block2_conv1 (Conv2D)                │ (None, 112, 112, 128)       │          73,856 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block2_conv2 (Conv2D)                │ (None, 112, 112, 128)       │         147,584 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block2_pool (MaxPooling2D)           │ (None, 56, 56, 128)         │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block3_conv1 (Conv2D)                │ (None, 56, 56, 256)         │         295,168 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block3_conv2 (Conv2D)                │ (None, 56, 56, 256)         │         590,080 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block3_conv3 (Conv2D)                │ (None, 56, 56, 256)         │         590,080 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block3_pool (MaxPooling2D)           │ (None, 28, 28, 256)         │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block4_conv1 (Conv2D)                │ (None, 28, 28, 512)         │       1,180,160 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block4_conv2 (Conv2D)                │ (None, 28, 28, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block4_conv3 (Conv2D)                │ (None, 28, 28, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block4_pool (MaxPooling2D)           │ (None, 14, 14, 512)         │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block5_conv1 (Conv2D)                │ (None, 14, 14, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block5_conv2 (Conv2D)                │ (None, 14, 14, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block5_conv3 (Conv2D)                │ (None, 14, 14, 512)         │       2,359,808 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ block5_pool (MaxPooling2D)           │ (None, 7, 7, 512)           │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ flatten (Flatten)                    │ (None, 25088)               │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dense (Dense)                        │ (None, 512)                 │      12,845,568 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dropout (Dropout)                    │ (None, 512)                 │               0 │
├──────────────────────────────────────┼─────────────────────────────┼─────────────────┤
│ dense_1 (Dense)                      │ (None, 8)                   │           4,104 │
└──────────────────────────────────────┴─────────────────────────────┴─────────────────┘
 Total params: 27,564,360 (105.15 MB)
```

```
from tensorflow.keras.callbacks import EarlyStopping, ModelCheckpoint

# Early stopping configuration
early_stopping = EarlyStopping(monitor='val_loss', patience=5, restore_best_weights=True)

# Model checkpoint to save the best model
model_checkpoint = ModelCheckpoint('best_model.keras', save_best_only=True, monitor='val_loss', mode='min

history = model.fit(                                #training the model
    train_data,
    validation_data=val_data,
    epochs=50,
    steps_per_epoch=train_data.samples // 32,
    validation_steps=val_data.samples // 32,
    callbacks=[early_stopping, model_checkpoint]
```

```
Epoch 1/50
/usr/local/lib/python3.10/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.py:121: Use
  self._warn_if_super_not_called()
134/134 ━━━━━━━━━━━━━━━━━━━━ 1817s 13s/step - accuracy: 0.5084 - loss: 1.3805 - val_accuracy: 0.8214 -
Epoch 2/50
  1/134 ━━━━━━━━━━━━━━━━━━━━ 20s 155ms/step - accuracy: 0.8438 - loss: 0.4818/usr/lib/python3.10/conte
  self.gen.throw(typ, value, traceback)
134/134 ━━━━━━━━━━━━━━━━━━━━ 8s 59ms/step - accuracy: 0.8438 - loss: 0.4818 - val_accuracy: 1.0000 - va
Epoch 3/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 24s 169ms/step - accuracy: 0.8017 - loss: 0.5312 - val_accuracy: 0.8415 -
Epoch 4/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 0s 946us/step - accuracy: 0.7812 - loss: 0.5135 - val_accuracy: 0.7500 - v
Epoch 5/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 25s 176ms/step - accuracy: 0.8389 - loss: 0.4384 - val_accuracy: 0.8549 -
Epoch 6/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 0s 473us/step - accuracy: 0.9062 - loss: 0.2782 - val_accuracy: 0.3750 - v
Epoch 7/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 28s 197ms/step - accuracy: 0.8648 - loss: 0.3620 - val_accuracy: 0.8616 -
Epoch 8/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 2s 11ms/step - accuracy: 0.9062 - loss: 0.3013 - val_accuracy: 1.0000 - va
Epoch 9/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 35s 171ms/step - accuracy: 0.8871 - loss: 0.3137 - val_accuracy: 0.8661 -
Epoch 10/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 0s 536us/step - accuracy: 0.9062 - loss: 0.2880 - val_accuracy: 1.0000 - v
Epoch 11/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 23s 168ms/step - accuracy: 0.9122 - loss: 0.2466 - val_accuracy: 0.8739 -
Epoch 12/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 0s 451us/step - accuracy: 0.8125 - loss: 0.3283 - val_accuracy: 0.8750 - v
Epoch 13/50
134/134 ━━━━━━━━━━━━━━━━━━━━ 23s 168ms/step - accuracy: 0.9032 - loss: 0.2589 - val_accuracy: 0.8605 -
```

```
val_loss, val_accuracy = model.evaluate(val_data) #model evalution on val data

print(f'Validation Loss: {val_loss}')
print(f'Validation Accuracy: {val_accuracy}')
```

```
29/29 ━━━━━━━━━━━━━━━━━━━━ 4s 120ms/step - accuracy: 0.8483 - loss: 0.3794
Validation Loss: 0.3704761862754822
Validation Accuracy: 0.855088472366333
```

```
model.save('/content/drive/MyDrive/Docs/blood_group_classifier.keras')
```

Start coding or generate with AI.