```
from keras.layers import Input, Lambda, Dense, Flatten, Dropout
from keras.models import Model
from keras.applications.vgg16 import VGG16
from keras.applications.vgg16 import preprocess_input
from keras.preprocessing import image
from keras.preprocessing.image import ImageDataGenerator
from keras.models import Sequential
import numpy as np
from glob import glob
import matplotlib.pyplot as plt

import warnings
warnings.filterwarnings("ignore", category=FutureWarning)


IMAGE_SIZE = [224, 224]
vgg = VGG16(input_shape=IMAGE_SIZE + [3], weights='imagenet', include_top=False)
vgg.input
```

```
<KerasTensor: shape=(None, 224, 224, 3) dtype=float32 (created by layer 'input_1')>
```

```
x = Flatten()(vgg.output)
x = Dropout(0.25)(x)
prediction = Dense(58, activation='softmax')(x)
model = Model(inputs=vgg.input, outputs=prediction)
model.summary()
```

```
Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 block1_conv1 (Conv2D)       (None, 224, 224, 64)      1792

 block1_conv2 (Conv2D)       (None, 224, 224, 64)      36928

 block1_pool (MaxPooling2D)  (None, 112, 112, 64)      0

 block2_conv1 (Conv2D)       (None, 112, 112, 128)     73856

 block2_conv2 (Conv2D)       (None, 112, 112, 128)     147584

 block2_pool (MaxPooling2D)  (None, 56, 56, 128)       0

 block3_conv1 (Conv2D)       (None, 56, 56, 256)       295168

 block3_conv2 (Conv2D)       (None, 56, 56, 256)       590080

 block3_conv3 (Conv2D)       (None, 56, 56, 256)       590080

 block3_pool (MaxPooling2D)  (None, 28, 28, 256)       0

 block4_conv1 (Conv2D)       (None, 28, 28, 512)       1180160

 block4_conv2 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_conv3 (Conv2D)       (None, 28, 28, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 14, 14, 512)       0

 block5_conv1 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv2 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_conv3 (Conv2D)       (None, 14, 14, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0

 flatten (Flatten)           (None, 25088)             0

 dropout (Dropout)           (None, 25088)             0

 dense (Dense)               (None, 58)                1455162

=================================================================
Total params: 16,169,850
Trainable params: 16,169,850
Non-trainable params: 0
_________________________________________________________________
```

```
train_generator = ImageDataGenerator(validation_split=0.20,rescale=1./255, shear_range=0.1,
    zoom_range=0.1,preprocessing_function = preprocess_input)
```

```
image_generator = ImageDataGenerator(preprocessing_function = preprocess_input)


# Create Data Loaders
train_loader = train_generator.flow_from_directory(
    directory = "/kaggle/input/training-handwritten-marathi-character-fusion/Data_set",
    target_size = (224, 224),
    batch_size = 32,
    shuffle = True,
    class_mode='categorical',
    subset='training',
    seed=42
)

valid_loader = train_generator.flow_from_directory(
    directory = "/kaggle/input/training-handwritten-marathi-character-fusion/Data_set",
    target_size = (224, 224),
    batch_size = 32,
    shuffle = True,
    class_mode='categorical',
    subset='validation',
    seed=42
)

test_loader = image_generator.flow_from_directory(
    directory = "/kaggle/input/handwritten-marathi-character-dataset-of-4-people/Testing_Data",
    target_size = (224, 224),
    batch_size = 32,
    shuffle = True,
    seed=42
)
```

```
Found 13348 images belonging to 58 classes.
Found 3305 images belonging to 58 classes.
Found 232 images belonging to 58 classes.
```

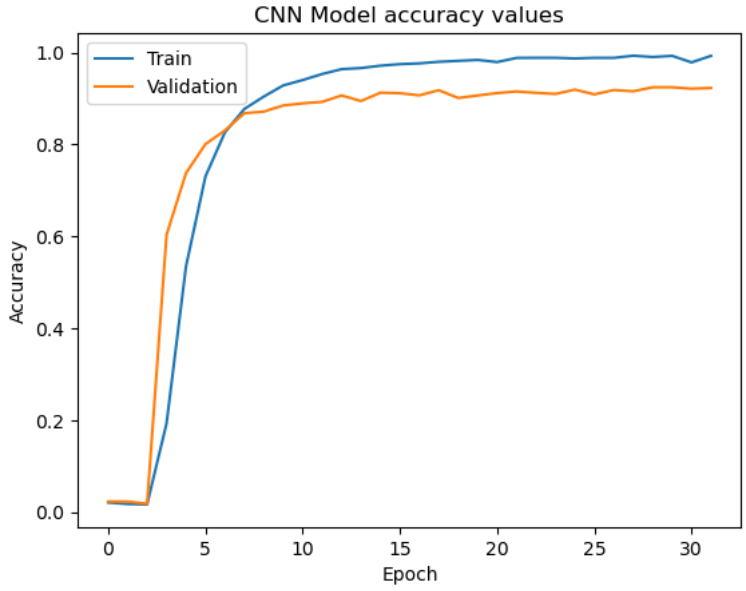
```
from keras import optimizers


adam = optimizers.Adam()
model.compile(loss='categorical_crossentropy',
              optimizer=adam,
              metrics=['accuracy'])
# Train and Test Steps
steps_train = len(train_loader)    # = round(num_train_images / batch_size)
steps_valid = len(valid_loader)
```

```
model_history=model.fit_generator(
  train_loader,
  validation_data=valid_loader,
  epochs=32,
  steps_per_epoch=steps_train,
  validation_steps=steps_valid,
)
```

```
418/418 [==============================] - 339s 810ms/step - loss: 0.0921 - accuracy: 0.0170 - val_loss: 0.0872 - val_accuracy: 0
Epoch 4/32
418/418 [==============================] - 338s 809ms/step - loss: 0.0736 - accuracy: 0.1933 - val_loss: 0.0392 - val_accuracy: 0
Epoch 5/32
418/418 [==============================] - 341s 817ms/step - loss: 0.0286 - accuracy: 0.7306 - val_loss: 0.0223 - val_accuracy: 0
Epoch 7/32
418/418 [==============================] - 344s 822ms/step - loss: 0.0206 - accuracy: 0.8264 - val_loss: 0.0193 - val_accuracy: 0
Epoch 8/32
418/418 [==============================] - 346s 826ms/step - loss: 0.0157 - accuracy: 0.8774 - val_loss: 0.0167 - val_accuracy: 0
Epoch 9/32
418/418 [==============================] - 344s 823ms/step - loss: 0.0129 - accuracy: 0.9044 - val_loss: 0.0152 - val_accuracy: 0
Epoch 10/32
418/418 [==============================] - 341s 816ms/step - loss: 0.0104 - accuracy: 0.9287 - val_loss: 0.0145 - val_accuracy: 0
Epoch 11/32
418/418 [==============================] - 340s 813ms/step - loss: 0.0090 - accuracy: 0.9404 - val_loss: 0.0155 - val_accuracy: 0
Epoch 12/32
418/418 [==============================] - 342s 819ms/step - loss: 0.0078 - accuracy: 0.9535 - val_loss: 0.0138 - val_accuracy: 0
```

```
418/418 [==============================] - 337s 807ms/step - loss: 0.0047 - accuracy: 0.9750 - val_loss: 0.0159 - val_accuracy: 0
Epoch 17/32
418/418 [==============================] - 339s 810ms/step - loss: 0.0047 - accuracy: 0.9766 - val_loss: 0.0127 - val_accuracy: 0
Epoch 18/32
418/418 [==============================] - 339s 812ms/step - loss: 0.0040 - accuracy: 0.9801 - val_loss: 0.0137 - val_accuracy: 0
Epoch 19/32
418/418 [==============================] - 341s 815ms/step - loss: 0.0038 - accuracy: 0.9821 - val_loss: 0.0153 - val_accuracy: 0
Epoch 20/32
418/418 [==============================] - 342s 818ms/step - loss: 0.0035 - accuracy: 0.9842 - val_loss: 0.0163 - val_accuracy: 0
Epoch 21/32
418/418 [==============================] - 339s 812ms/step - loss: 0.0043 - accuracy: 0.9795 - val_loss: 0.0131 - val_accuracy: 0
Epoch 22/32
418/418 [==============================] - 382s 914ms/step - loss: 0.0028 - accuracy: 0.9885 - val_loss: 0.0132 - val_accuracy: 0
Epoch 23/32
418/418 [==============================] - 344s 823ms/step - loss: 0.0029 - accuracy: 0.9888 - val_loss: 0.0149 - val_accuracy: 0
Epoch 24/32
418/418 [==============================] - 340s 812ms/step - loss: 0.0026 - accuracy: 0.9888 - val_loss: 0.0131 - val_accuracy: 0
Epoch 25/32
418/418 [==============================] - 340s 814ms/step - loss: 0.0029 - accuracy: 0.9873 - val_loss: 0.0133 - val_accuracy: 0
Epoch 26/32
418/418 [==============================] - 339s 811ms/step - loss: 0.0026 - accuracy: 0.9887 - val_loss: 0.0181 - val_accuracy: 0
Epoch 27/32
418/418 [==============================] - 338s 809ms/step - loss: 0.0028 - accuracy: 0.9886 - val_loss: 0.0144 - val_accuracy: 0
Epoch 28/32
418/418 [==============================] - 339s 811ms/step - loss: 0.0021 - accuracy: 0.9934 - val_loss: 0.0139 - val_accuracy: 0
Epoch 29/32
418/418 [==============================] - 339s 812ms/step - loss: 0.0024 - accuracy: 0.9905 - val_loss: 0.0150 - val_accuracy: 0
Epoch 30/32
418/418 [==============================] - 340s 814ms/step - loss: 0.0021 - accuracy: 0.9930 - val_loss: 0.0142 - val_accuracy: 0
Epoch 31/32
418/418 [==============================] - 340s 815ms/step - loss: 0.0051 - accuracy: 0.9788 - val_loss: 0.0158 - val_accuracy: 0
Epoch 32/32
```

```
_# Plot training & validation loss values
plt.plot(model_history.history['accuracy'])
plt.plot(model_history.history['val_accuracy'])
plt.title('CNN Model accuracy values')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Validation'], loc='upper left')
plt.show()
```




```
test_loader = image_generator.flow_from_directory(
    directory = "/kaggle/input/handwritten-marathi-character-dataset-of-4-people/Testing_Data/",
    target_size = (224, 224),
    batch_size = 32,
    shuffle = True,
    seed=42
)

model.evaluate(test_loader)
```

```
Found 232 images belonging to 58 classes.
8/8 [==============================] - 3s 348ms/step - loss: 9.4408 - accuracy: 0.6207
[9.440812110900879, 0.6206896305084229]
```