Installing Libraries

```
pip install keras
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: keras in /usr/local/lib/python3.9/dist-packages (2.11.0)
```

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from keras import applications
import os
from tensorflow import keras
from tensorflow.keras.preprocessing.image import ImageDataGenerator, load_img
from sklearn.model_selection import train_test_split


Image_Width=128
Image_Height=128
Image_Size=(Image_Width,Image_Height)
Image_Channels=3
```

File Reading

```
filenames_train=os.listdir("/content/drive/MyDrive/DevTown/train")
filenames_test=os.listdir("/content/drive/MyDrive/DevTown/test")


#data splitting
categories_train=[]
for f_name in filenames_train:
    category=f_name.split('.')[0]
    if category=='dog':
        categories_train.append(1)
    else:
        categories_train.append(0)

df_train=pd.DataFrame({'filename':filenames_train,'category':categories_train})

categories_test=[]
for f_name in filenames_test:
    category=f_name.split('.')[0]
    if category=='dog':
        categories_test.append(1)
    else:
        categories_test.append(0)

df_test=pd.DataFrame({'filename':filenames_test,'category':categories_test})


from keras.models import Sequential
from keras.layers import Conv2D,MaxPooling2D,\
     Dropout,Flatten,Dense,Activation,\
     BatchNormalization

model=Sequential()

model.add(Conv2D(32,(3,3),activation='relu',input_shape=(Image_Width,Image_Height,Image_Channels)))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(64,(3,3),activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Conv2D(128,(3,3),activation='relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512,activation='relu'))
model.add(BatchNormalization())
model.add(Dropout(0.5))
```

```
model.add(Dense(2,activation='softmax'))

model.compile(loss='categorical_crossentropy',
  optimizer='rmsprop',metrics=['accuracy'])


model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 126, 126, 32)      896

 batch_normalization (BatchN  (None, 126, 126, 32)     128
 ormalization)

 max_pooling2d (MaxPooling2D  (None, 63, 63, 32)       0
 )

 dropout (Dropout)           (None, 63, 63, 32)        0

 conv2d_1 (Conv2D)           (None, 61, 61, 64)        18496

 batch_normalization_1 (Batc  (None, 61, 61, 64)       256
 hNormalization)

 max_pooling2d_1 (MaxPooling  (None, 30, 30, 64)       0
 2D)

 dropout_1 (Dropout)         (None, 30, 30, 64)        0

 conv2d_2 (Conv2D)           (None, 28, 28, 128)       73856

 batch_normalization_2 (Batc  (None, 28, 28, 128)      512
 hNormalization)

 max_pooling2d_2 (MaxPooling  (None, 14, 14, 128)      0
 2D)

 dropout_2 (Dropout)         (None, 14, 14, 128)       0

 flatten (Flatten)           (None, 25088)             0

 dense (Dense)               (None, 512)               12845568

 batch_normalization_3 (Batc  (None, 512)              2048
 hNormalization)

 dropout_3 (Dropout)         (None, 512)               0

 dense_1 (Dense)             (None, 2)                 1026

=================================================================
Total params: 12,942,786
Trainable params: 12,941,314
Non-trainable params: 1,472
_________________________________________________________________
```

```
from keras.callbacks import EarlyStopping, ReduceLROnPlateau
earlystop = EarlyStopping(patience = 10)
learning_rate_reduction = ReduceLROnPlateau(monitor = 'val_acc',patience = 2,verbose = 1,factor = 0.5,min_lr = 0.00001)
callbacks = [earlystop,learning_rate_reduction]
```

Model Training / Testing

```
#splitting data for testing
df_train["category"] = df_train["category"].replace({0:'cat',1:'dog'})
df_test["category"] = df_test["category"].replace({0:'cat',1:'dog'})
train_df,validate_df = train_test_split(df_train,test_size=0.20,
  random_state=42)

train_df = train_df.reset_index(drop=True)
validate_df = validate_df.reset_index(drop=True)

total_train=train_df.shape[0]
total_validate=validate_df.shape[0]
batch_size=15


df_test.category[df_test.category=='dog']
```

```
Series([], Name: category, dtype: object)
```

```
train_datagen = ImageDataGenerator(rotation_range=15,
                                   rescale=1./255,
                                   shear_range=0.1,
                                   zoom_range=0.2,
                                   horizontal_flip=True,
                                   width_shift_range=0.1,
                                   height_shift_range=0.1
                                   )

train_generator = train_datagen.flow_from_dataframe(train_df,
                                                    "/content/drive/MyDrive/DevTown/train",x_col='filename',y_col='category',
                                                    target_size=Image_Size,
                                                    class_mode='categorical',
                                                    batch_size=batch_size)

validation_datagen = ImageDataGenerator(rescale=1./255)
validation_generator = validation_datagen.flow_from_dataframe(
    validate_df,
    "/content/drive/MyDrive/DevTown/train",
    x_col='filename',
    y_col='category',
    target_size=Image_Size,
    class_mode='categorical',
    batch_size=batch_size
)
```

```
Found 20040 validated image filenames belonging to 2 classes.
Found 5010 validated image filenames belonging to 2 classes.
Found 0 validated image filenames belonging to 0 classes.
/usr/local/lib/python3.9/dist-packages/keras/preprocessing/image.py:1137: UserWarning: Found 20040 invalid image filenam
  warnings.warn(
```

```
test_filenames = os.listdir("/content/drive/MyDrive/DevTown/test")
test_df = pd.DataFrame({'filename': test_filenames})
```

+ Code + Text

```
test_df.head()
```

	filename
0	9090.jpg
1	9089.jpg
2	9102.jpg
3	910.jpg
4	9094.jpg

```
test_datagen = ImageDataGenerator(rotation_range=15,
                                  rescale=1./255,
                                  shear_range=0.1,
                                  zoom_range=0.2,
                                  horizontal_flip=True,
                                  width_shift_range=0.1,
                                  height_shift_range=0.1)

test_generator = test_datagen.flow_from_dataframe(df_test,
                                                  "/content/drive/MyDrive/DevTown/test",x_col='filename',y_col='category',
                                                  target_size=Image_Size,
                                                  class_mode='categorical',
                                                  batch_size=batch_size)
```

```
Found 12500 validated image filenames belonging to 1 classes.
```

```
epochs=10
history = model.fit_generator(
    train_generator,
    epochs=epochs,
    validation_data=validation_generator,
    validation_steps=total_validate//batch_size,
    steps_per_epoch=total_train//batch_size,
    callbacks=callbacks
)
```

```
<ipython-input-18-f7b6d0c9c94b>:2: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future vers
  history = model.fit_generator(
Epoch 1/10
1336/1336 [==============================] - ETA: 0s - loss: 0.7358 - accuracy: 0.6386WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 6668s 5s/step - loss: 0.7358 - accuracy: 0.6386 - val_loss: 0.6920 - val_ac
```

```
Epoch 2/10
1336/1336 [==============================] - ETA: 0s - loss: 0.5648 - accuracy: 0.7118WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 1411s 1s/step - loss: 0.5648 - accuracy: 0.7118 - val_loss: 0.7173 - val_ac
Epoch 3/10
1336/1336 [==============================] - ETA: 0s - loss: 0.5053 - accuracy: 0.7598WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 1394s 1s/step - loss: 0.5053 - accuracy: 0.7598 - val_loss: 0.6256 - val_ac
Epoch 4/10
1336/1336 [==============================] - ETA: 0s - loss: 0.4700 - accuracy: 0.7833WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 1377s 1s/step - loss: 0.4700 - accuracy: 0.7833 - val_loss: 0.5397 - val_ac
Epoch 5/10
1336/1336 [==============================] - ETA: 0s - loss: 0.4358 - accuracy: 0.8020WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 1462s 1s/step - loss: 0.4358 - accuracy: 0.8020 - val_loss: 0.3891 - val_ac
Epoch 6/10
1336/1336 [==============================] - ETA: 0s - loss: 0.4187 - accuracy: 0.8123WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 1413s 1s/step - loss: 0.4187 - accuracy: 0.8123 - val_loss: 0.5027 - val_ac
Epoch 7/10
1336/1336 [==============================] - ETA: 0s - loss: 0.3988 - accuracy: 0.8206WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 1383s 1s/step - loss: 0.3988 - accuracy: 0.8206 - val_loss: 0.3451 - val_ac
Epoch 8/10
1336/1336 [==============================] - ETA: 0s - loss: 0.3838 - accuracy: 0.8291WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 1386s 1s/step - loss: 0.3838 - accuracy: 0.8291 - val_loss: 0.3332 - val_ac
Epoch 9/10
1336/1336 [==============================] - ETA: 0s - loss: 0.3794 - accuracy: 0.8325WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 1391s 1s/step - loss: 0.3794 - accuracy: 0.8325 - val_loss: 0.4925 - val_ac
Epoch 10/10
1336/1336 [==============================] - ETA: 0s - loss: 0.3695 - accuracy: 0.8392WARNING:tensorflow:Learning rate r
1336/1336 [==============================] - 1362s 1s/step - loss: 0.3695 - accuracy: 0.8392 - val_loss: 0.4176 - val_ac
```

```
model.save("model_cats-dogs.h5") #saving the model
```

Visualisation

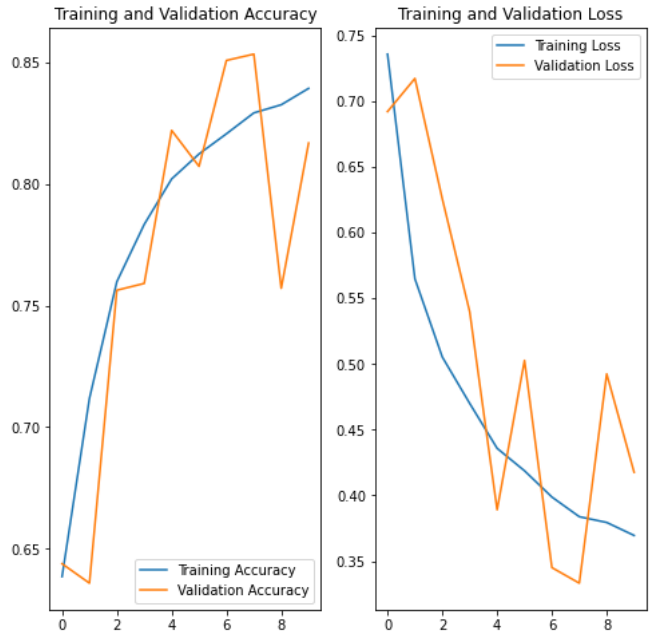
```
acc = history.history['accuracy']
val_acc = history.history['val_accuracy']

loss = history.history['loss']
val_loss = history.history['val_loss']

epochs_range = range(epochs)

plt.figure(figsize=(8, 8))
plt.subplot(1, 2, 1)
plt.plot(epochs_range, acc, label='Training Accuracy')
plt.plot(epochs_range, val_acc, label='Validation Accuracy')
plt.legend(loc='lower right')
plt.title('Training and Validation Accuracy')

plt.subplot(1, 2, 2)
plt.plot(epochs_range, loss, label='Training Loss')
plt.plot(epochs_range, val_loss, label='Validation Loss')
plt.legend(loc='upper right')
plt.title('Training and Validation Loss')
plt.show()
```



```
nb_samples = test_df.shape[0]
```

Running testing data on the trained Model

```
predict = model.predict_generator(test_generator, steps=np.ceil(nb_samples/batch_size))
```

```
<ipython-input-53-52619fd72ccc>:1: UserWarning: `Model.predict_generator` is deprecated and will be removed in a future
  predict = model.predict_generator(test_generator, steps=np.ceil(nb_samples/batch_size))
```

```
test_df['category'] = np.argmax(predict, axis=-1)

label_map = dict((v,k) for k,v in train_generator.class_indices.items())
test_df['category'] = test_df['category'].replace(label_map)

#test_df['category'] = test_df['category'].replace({ 'dog': 1, 'cat': 0 })
```

```
test_df.head()
```

	filename	category
0	9090.jpg	cat
1	9089.jpg	dog
2	9102.jpg	cat
3	910.jpg	dog
4	9094.jpg	cat

Extracting pickle file

```
test_df.to_pickle("Dog-Cat Prediction.pkl")
```