```
import matplotlib.pyplot as plt
import tensorflow as tf
import pandas as pd
import numpy as np

import warnings
warnings.filterwarnings('ignore')

from tensorflow import keras
from keras import layers
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Activation, Dropout, Flatten, Dense
from tensorflow.keras.layers import Conv2D, MaxPooling2D
from tensorflow.keras.utils import image_dataset_from_directory
from tensorflow.keras.preprocessing.image import ImageDataGenerator, load_img
from tensorflow.keras.preprocessing import image_dataset_from_directory

import os
import matplotlib.image as mpimg
```

```
!unzip /content/OriginalDataset.zip
```

```
      inflating: Original Dataset/kids-running/0943.jpg
      inflating: Original Dataset/kids-running/0944.jpg
      inflating: Original Dataset/kids-running/0945.jpg
      inflating: Original Dataset/kids-running/0946.jpg
      inflating: Original Dataset/kids-running/0947.jpg
      inflating: Original Dataset/kids-running/0948.jpg
      inflating: Original Dataset/kids-running/0949.jpg
      inflating: Original Dataset/kids-running/0950.jpg
      inflating: Original Dataset/kids-running/0951.jpg
      inflating: Original Dataset/kids-running/0952.jpg
      inflating: Original Dataset/kids-running/0953.jpg
      inflating: Original Dataset/kids-running/0954.jpg
      inflating: Original Dataset/kids-running/0955.jpg
      inflating: Original Dataset/kids-running/0956.jpg
      inflating: Original Dataset/kids-running/0957.jpg
      inflating: Original Dataset/kids-running/0958.jpg
      inflating: Original Dataset/kids-running/0959.jpg
      inflating: Original Dataset/kids-running/0960.jpg
      inflating: Original Dataset/kids-running/0961.jpg
      inflating: Original Dataset/kids-running/0962.jpg
      inflating: Original Dataset/kids-running/0963.jpg
      inflating: Original Dataset/kids-running/0964.jpg
      inflating: Original Dataset/kids-running/0965.jpg
      inflating: Original Dataset/kids-running/0966.jpg
      inflating: Original Dataset/kids-running/0967.jpg
      inflating: Original Dataset/kids-running/0968.jpg
      inflating: Original Dataset/kids-running/0969.jpg
      inflating: Original Dataset/kids-running/0970.jpg
      inflating: Original Dataset/kids-running/0971.jpg
      inflating: Original Dataset/kids-running/0972.jpg
      inflating: Original Dataset/kids-running/0973.jpg
      inflating: Original Dataset/kids-running/0974.jpg
      inflating: Original Dataset/kids-running/0975.jpg
      inflating: Original Dataset/kids-running/0976.jpg
      inflating: Original Dataset/kids-running/0977.jpg
      inflating: Original Dataset/kids-running/0978.jpg
      inflating: Original Dataset/kids-running/0979.jpg
      inflating: Original Dataset/kids-running/0980.jpg
      inflating: Original Dataset/kids-running/0981.jpg
      inflating: Original Dataset/kids-running/0982.jpg
      inflating: Original Dataset/kids-running/0983.jpg
      inflating: Original Dataset/kids-running/0984.jpg
      inflating: Original Dataset/kids-running/0985.jpg
      inflating: Original Dataset/kids-running/0986.jpg
      inflating: Original Dataset/kids-running/0987.jpg
      inflating: Original Dataset/kids-running/0988.jpg
      inflating: Original Dataset/kids-running/0989.jpg
      inflating: Original Dataset/kids-running/0990.jpg
      inflating: Original Dataset/kids-running/0991.jpg
      inflating: Original Dataset/kids-running/0992.jpg
      inflating: Original Dataset/kids-running/0993.jpg
      inflating: Original Dataset/kids-running/0994.jpg
      inflating: Original Dataset/kids-running/0995.jpg
      inflating: Original Dataset/kids-running/0996.jpg
      inflating: Original Dataset/kids-running/0997.jpg
      inflating: Original Dataset/kids-running/0998.jpg
      inflating: Original Dataset/kids-running/0999.jpg
      inflating: Original Dataset/kids-running/1000.jpg
```

```
path = '/content/Original Dataset'
classes = os.listdir(path)
classes
```

```
    ['kids-running', 'dogs-running']
```

```
base_dir = '/content/Original Dataset'

# Create datasets
train_datagen = image_dataset_from_directory(base_dir,
                                                  image_size=(200,200),
                                                  subset='training',
                                                  seed = 1,
                                                 validation_split=0.1,
                                                  batch_size= 32)
test_datagen = image_dataset_from_directory(base_dir,
                                                  image_size=(200,200),
                                                  subset='validation',
                                                  seed = 1,
                                                 validation_split=0.1,
                                                  batch_size= 32)
```

```
Found 2000 files belonging to 2 classes.
Using 1800 files for training.
Found 2000 files belonging to 2 classes.
Using 200 files for validation.
```

```
model = tf.keras.models.Sequential([
    layers.Conv2D(32, (3, 3), activation='relu', input_shape=(200, 200, 3)),
    layers.MaxPooling2D(2, 2),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D(2, 2),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D(2, 2),
    layers.Conv2D(64, (3, 3), activation='relu'),
    layers.MaxPooling2D(2, 2),

    layers.Flatten(),
    layers.Dense(512, activation='relu'),
    layers.BatchNormalization(),
    layers.Dense(512, activation='relu'),
    layers.Dropout(0.1),
    layers.BatchNormalization(),
    layers.Dense(512, activation='relu'),
    layers.Dropout(0.2),
    layers.BatchNormalization(),
    layers.Dense(1, activation='sigmoid')
])
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 198, 198, 32)      896

 max_pooling2d (MaxPooling2D  (None, 99, 99, 32)       0
 )

 conv2d_1 (Conv2D)           (None, 97, 97, 64)        18496

 max_pooling2d_1 (MaxPooling  (None, 48, 48, 64)       0
 2D)

 conv2d_2 (Conv2D)           (None, 46, 46, 64)        36928

 max_pooling2d_2 (MaxPooling  (None, 23, 23, 64)       0
 2D)

 conv2d_3 (Conv2D)           (None, 21, 21, 64)        36928

 max_pooling2d_3 (MaxPooling  (None, 10, 10, 64)       0
 2D)

 flatten (Flatten)           (None, 6400)              0

 dense (Dense)               (None, 512)               3277312

 batch_normalization (BatchN  (None, 512)              2048
 ormalization)

 dense_1 (Dense)             (None, 512)               262656

 dropout (Dropout)           (None, 512)               0

 batch_normalization_1 (Batc  (None, 512)              2048
 hNormalization)
 dense_2 (Dense)             (None, 512)               262656

 dropout_1 (Dropout)         (None, 512)               0

 batch_normalization_2 (Batc  (None, 512)              2048
 hNormalization)

 dense_3 (Dense)             (None, 1)                 513

=================================================================
Total params: 3,902,529
Trainable params: 3,899,457
Non-trainable params: 3,072
_________________________________________________________________
```

```
model.compile(
    loss='binary_crossentropy',
    optimizer='adam',
    metrics=['accuracy']
)
```

```
history = model.fit(train_datagen,
          epochs=10,
          validation_data=test_datagen)
```

```
Epoch 1/10
57/57 [==============================] - 137s 2s/step - loss: 0.8243 - accuracy: 0.5783 - val_loss: 6.4838 - val_accuracy: 0.5450
Epoch 2/10
57/57 [==============================] - 132s 2s/step - loss: 0.7011 - accuracy: 0.6161 - val_loss: 1.6462 - val_accuracy: 0.5050
Epoch 3/10
57/57 [==============================] - 133s 2s/step - loss: 0.6403 - accuracy: 0.6633 - val_loss: 1.1517 - val_accuracy: 0.6050
Epoch 4/10
57/57 [==============================] - 133s 2s/step - loss: 0.6211 - accuracy: 0.6733 - val_loss: 0.6577 - val_accuracy: 0.6550
Epoch 5/10
57/57 [==============================] - 132s 2s/step - loss: 0.5847 - accuracy: 0.7061 - val_loss: 0.9850 - val_accuracy: 0.5550
Epoch 6/10
57/57 [==============================] - 132s 2s/step - loss: 0.6037 - accuracy: 0.6950 - val_loss: 4.7834 - val_accuracy: 0.4550
Epoch 7/10
57/57 [==============================] - 132s 2s/step - loss: 0.6040 - accuracy: 0.6939 - val_loss: 0.7120 - val_accuracy: 0.6450
Epoch 8/10
57/57 [==============================] - 133s 2s/step - loss: 0.6069 - accuracy: 0.6922 - val_loss: 0.9547 - val_accuracy: 0.6150
Epoch 9/10
57/57 [==============================] - 132s 2s/step - loss: 0.5618 - accuracy: 0.7161 - val_loss: 0.9793 - val_accuracy: 0.5050
Epoch 10/10
57/57 [==============================] - 132s 2s/step - loss: 0.5261 - accuracy: 0.7439 - val_loss: 0.8377 - val_accuracy: 0.5250
```