In [49]:

```
import numpy as np
import cv2  # converting image into array
import random
import matplotlib.pyplot as plt
import pickle
import pandas as pd
import zipfile
import os
from sklearn.model_selection import train_test_split
```

In [18]:

```
image_width = 128
image_height  = 128
image_size = (image_height,image_width)
image_channel = 3
```

In [21]:

```
filename = os.listdir(r"C:\Users\INDIA\Documents\Jupyter Files\Dog vs Cat image CNN\train")
categories = []

for f_name in filename:
    category = f_name.split(".")[0]
    if category =='dog':
        categories.append(1)
    else:
        categories.append(0)
```

In [23]:

```
data = pd.DataFrame({'Filename': filename , 'Category':categories})
```

In [27]:

```
data
```

Out[27]:

| | Filename | Category |
|---|---|---|
| **0** | cat.0.jpg | 0 |
| **1** | cat.1.jpg | 0 |
| **2** | cat.10.jpg | 0 |
| **3** | cat.100.jpg | 0 |
| **4** | cat.1000.jpg | 0 |
| **...** | ... | ... |
| **24995** | dog.9995.jpg | 1 |
| **24996** | dog.9996.jpg | 1 |
| **24997** | dog.9997.jpg | 1 |
| **24998** | dog.9998.jpg | 1 |
| **24999** | dog.9999.jpg | 1 |

25000 rows × 2 columns

In [29]:

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

In [43]:

```
model = Sequential()

model.add(Conv2D(32,(5,5),activation = 'relu' , input_shape = (128,128,3)))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size = (2,2) , strides = 2))
model.add(Dropout(0.25))

model.add(Conv2D(64,(5,5),activation = 'relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size = (2,2) , strides = 2))
model.add(Dropout(0.25))

model.add(Conv2D(128,(5,5),activation = 'relu'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size = (2,2) , strides = 2))
model.add(Dropout(0.25))

model.add(Flatten())
model.add(Dense(512,activation = 'relu'))
model.add(BatchNormalization())
model.add(Dropout(0.25))
model.add(Dense(2,activation = 'softmax'))

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

In [45]:

```
model.summary()
```

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d_3 (Conv2D)           (None, 124, 124, 32)      2432      
                                                                 
 batch_normalization_4 (Batc  (None, 124, 124, 32)     128       
 hNormalization)                                                 
                                                                 
 max_pooling2d_3 (MaxPooling  (None, 62, 62, 32)       0         
 2D)                                                             
                                                                 
 dropout_4 (Dropout)         (None, 62, 62, 32)        0         
                                                                 
 conv2d_4 (Conv2D)           (None, 58, 58, 64)        51264     
                                                                 
 batch_normalization_5 (Batc  (None, 58, 58, 64)       256       
 hNormalization)                                                 
                                                                 
 max_pooling2d_4 (MaxPooling  (None, 29, 29, 64)       0         
 2D)                                                             
                                                                 
 dropout_5 (Dropout)         (None, 29, 29, 64)        0         
                                                                 
 conv2d_5 (Conv2D)           (None, 25, 25, 128)       204928    
                                                                 
 batch_normalization_6 (Batc  (None, 25, 25, 128)      512       
 hNormalization)                                                 
                                                                 
 max_pooling2d_5 (MaxPooling  (None, 12, 12, 128)      0         
 2D)                                                             
                                                                 
 dropout_6 (Dropout)         (None, 12, 12, 128)       0         
                                                                 
 flatten_1 (Flatten)         (None, 18432)             0         
                                                                 
 dense_2 (Dense)             (None, 512)               9437696   
                                                                 
 batch_normalization_7 (Batc  (None, 512)              2048      
 hNormalization)                                                 
 dropout_7 (Dropout)         (None, 512)               0

 dense_3 (Dense)             (None, 2)                 1026

=================================================================
Total params: 9,700,290
Trainable params: 9,698,818
Non-trainable params: 1,472
_________________________________________________________________
```

In [48]:

```
from tensorflow.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 = 0.000010)
callbacks = [earlystop,learning_rate_reduction]
```

In [58]:

```
data["Category"] = data["Category"].replace({0:'cat',1:'dog'})
train_df,validate_df = train_test_split(data , test_size = 0.2,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
```

In [59]:

```
from keras.preprocessing.image import ImageDataGenerator,load_img
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,
                                                 r"C:\Users\INDIA\Documents\Jupyter Files
\Dog vs Cat image CNN\train",
                                                    x_col='Filename',y_col='Category',
                                                 target_size=image_size,
                                                 class_mode='categorical',
                                                 batch_size=batch_size)
```

```
Found 20000 validated image filenames belonging to 2 classes.
```

In [113]:

```
validation_datagen = ImageDataGenerator(rescale=1./255)
validation_generator = validation_datagen.flow_from_dataframe(
    validate_df,
     r"C:\Users\INDIA\Documents\Jupyter Files\Dog vs Cat image CNN\train",
    x_col='Filename',
    y_col='Category',
    target_size=image_size,
    class_mode='categorical',
    batch_size=batch_size
)
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_data = train_datagen.flow_from_dataframe(tr_df,
                                        directory=r"C:\Users\INDIA\Documents\Jupyter Files
\Dog vs Cat image CNN\test1",
                                        x_col='Filename',
                                        y_col='Category',
                                        class_mode='categorical',
                                        target_size=image_size,
                                        shuffle=True,
                                        seed=17
                                       )
```

```
Found 5000 validated image filenames belonging to 2 classes.
Found 20000 validated image filenames belonging to 2 classes.
```

In [63]:

```
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
)
```

```
Epoch 1/10
```

```
C:\Users\INDIA\AppData\Local\Temp/ipykernel_14404/2528005629.py:2: UserWarning: `Model.fit_
generator` is deprecated and will be removed in a future version. Please use `Model.fit`, w
hich supports generators.
  history = model.fit_generator(
```

```
1333/1333 [==============================] - ETA: 0s - loss: 0.6737 - accuracy: 0.6336WARNI
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 865s 649ms/step - loss: 0.6737 - accuracy: 0.6
336 - val_loss: 0.7585 - val_accuracy: 0.6118 - lr: 0.0010
Epoch 2/10
1333/1333 [==============================] - ETA: 0s - loss: 0.5478 - accuracy: 0.7248WARNI
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 880s 660ms/step - loss: 0.5478 - accuracy: 0.7
248 - val_loss: 0.9730 - val_accuracy: 0.6158 - lr: 0.0010
Epoch 3/10
1333/1333 [==============================] - ETA: 0s - loss: 0.4908 - accuracy: 0.7665WARNI
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 911s 684ms/step - loss: 0.4908 - accuracy: 0.7
665 - val_loss: 1.2231 - val_accuracy: 0.6470 - lr: 0.0010
Epoch 4/10
1333/1333 [==============================] - ETA: 0s - loss: 0.4537 - accuracy: 0.7890WARNI
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 939s 704ms/step - loss: 0.4537 - accuracy: 0.7
890 - val_loss: 0.7436 - val_accuracy: 0.6707 - lr: 0.0010
Epoch 5/10
1333/1333 [==============================] - ETA: 0s - loss: 0.4228 - accuracy: 0.8091WARNI
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 921s 691ms/step - loss: 0.4228 - accuracy: 0.8
091 - val_loss: 0.6244 - val_accuracy: 0.7560 - lr: 0.0010
Epoch 6/10
1333/1333 [==============================] - ETA: 0s - loss: 0.3951 - accuracy: 0.8231WARNI
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 936s 702ms/step - loss: 0.3951 - accuracy: 0.8
231 - val_loss: 0.3110 - val_accuracy: 0.8665 - lr: 0.0010
Epoch 7/10
1333/1333 [==============================] - ETA: 0s - loss: 0.3775 - accuracy: 0.8332WARNI
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 966s 724ms/step - loss: 0.3775 - accuracy: 0.8
332 - val_loss: 0.3604 - val_accuracy: 0.8468 - lr: 0.0010
Epoch 8/10
1333/1333 [==============================] - ETA: 0s - loss: 0.3624 - accuracy: 0.8393WARNI
```

```
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 943s 707ms/step - loss: 0.3624 - accuracy: 0.8
393 - val_loss: 0.4199 - val_accuracy: 0.8084 - lr: 0.0010
Epoch 9/10
1333/1333 [==============================] - ETA: 0s - loss: 0.3539 - accuracy: 0.8452WARNI
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 893s 670ms/step - loss: 0.3539 - accuracy: 0.8
452 - val_loss: 0.4082 - val_accuracy: 0.8158 - lr: 0.0010
Epoch 10/10
1333/1333 [==============================] - ETA: 0s - loss: 0.3458 - accuracy: 0.8507WARNI
NG:tensorflow:Learning rate reduction is conditioned on metric `val_acc` which is not avail
able. Available metrics are: loss,accuracy,val_loss,val_accuracy,lr
1333/1333 [==============================] - 897s 673ms/step - loss: 0.3458 - accuracy: 0.8
507 - val_loss: 0.3538 - val_accuracy: 0.8679 - lr: 0.0010
```