In [1]:
```
import tensorflow as tf
import keras
from keras.models import Sequential
from keras.layers import Dense,Dropout,Conv2D,Flatten,MaxPooling2D,BatchNormalization
from keras.optimizers import SGD
from keras import regularizers
```

```
Using TensorFlow backend.
```

In [2]:
```
model=Sequential()
```

In [3]:
```
model.add(Conv2D(32,kernel_size=5,input_shape=(64,64,3),activation='relu'))
```

In [4]:
```
model.add(MaxPooling2D(pool_size=(2,2),strides=2))
```

In [5]:
```
model.add(BatchNormalization())
```

In [6]:
```
model.add(Conv2D(64,kernel_size=5,activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2),strides=2))
model.add(Dropout(0.4))
```

In [7]:
```
model.add(Flatten())
```

In [15]:
```
#full Connection
model.add(Dense(output_dim=32,activation='relu'))
model.add(Dense(output_dim=1,activation='softmax'))
```

```
/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:2: UserWarning: Update your `Dense` call to
the Keras 2 API: `Dense(activation="relu", units=32)`

/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:3: UserWarning: Update your `Dense` call to
the Keras 2 API: `Dense(activation="softmax", units=1)`
  This is separate from the ipykernel package so we can avoid doing imports until
```

In [9]:
```
sgd=keras.optimizers.SGD(learning_rate=0.01,momentum=0.9,nesterov=True)
model.compile(optimizer=sgd,loss='binary_crossentropy',metrics=['accuracy'])
```

In [10]:
```
from keras.preprocessing.image import ImageDataGenerator
train_data=ImageDataGenerator(rescale=1./255,
                              shear_range=0.2,
                              zoom_range=0.2,
                              horizontal_flip=True)
```

In [11]:
```
test_data=ImageDataGenerator(rescale=1./255)
```

In [12]:
```
training_set=train_data.flow_from_directory('train',
                                            target_size=(64,64),
                                            batch_size=32,
                                            class_mode='binary')
```

```
Found 40 images belonging to 2 classes.
```

In [13]:
```
test_set=test_data.flow_from_directory('test',
                        target_size=(64,64),
                        batch_size=32,
                        class_mode='binary')
```

```
Found 20 images belonging to 2 classes.
```

In [16]:
```
model.fit_generator(training_set,
                    samples_per_epoch=8000,
                    nb_epoch=5,
                    validation_data=(test_set),
                    nb_val_samples=2000)
```

```
/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:5: UserWarning: The semantics of the Keras
2 argument `steps_per_epoch` is not the same as the Keras 1 argument `samples_per_epoch`. `steps_per_epoch`
is the number of batches to draw from the generator at each epoch. Basically steps_per_epoch = samples_per_e
poch/batch_size. Similarly `nb_val_samples`->`validation_steps` and `val_samples`->`steps` arguments have ch
anged. Update your method calls accordingly.
  """
/opt/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:5: UserWarning: Update your `fit_generator`
call to the Keras 2 API: `fit_generator(<keras.pre..., validation_data=<keras.pre..., steps_per_epoch=250, e
pochs=5, validation_steps=2000)`
  """
/opt/anaconda3/lib/python3.7/site-packages/keras/engine/training.py:297: UserWarning: Discrepancy between tr
ainable weights and collected trainable weights, did you set `model.trainable` without calling `model.compil
e` after ?
  'Discrepancy between trainable weights and collected trainable'

Epoch 1/5
250/250 [==============================] - 258s 1s/step - loss: 7.6552 - val_loss: 7.6246
Epoch 2/5
250/250 [==============================] - 265s 1s/step - loss: 7.7239 - val_loss: 7.6246
Epoch 3/5
250/250 [==============================] - 265s 1s/step - loss: 7.5864 - val_loss: 7.6246
Epoch 4/5
250/250 [==============================] - 264s 1s/step - loss: 7.6552 - val_loss: 7.6246
Epoch 5/5
250/250 [==============================] - 463s 2s/step - loss: 7.7525 - val_loss: 7.6246
```

Out[16]:
```
<keras.callbacks.callbacks.History at 0x634bed690>
```

In [60]:
```
import numpy as np
from keras.preprocessing import image
test_image=image.load_img('101.jpg',target_size=(64,64))
test_image=image.img_to_array(test_image)
test_image=np.expand_dims(test_image,axis=0)
result=model.predict(test_image)
training_set.class_indices
if result[0][0]>=0.5:
    prediction='dog'
else:
    prediction='cat'
print(prediction)
```

```
dog
```

In [ ]:

In [ ]:

In [ ]:

In [ ]: