```
# 13 Write a program to implement CNN

#implement  CNN

# This Python 3 environment comes with many helpful analytics libraries installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

# You can write up to 20GB to the current directory (/kaggle/working/) that gets preserved as output when you create a version u
# You can also write temporary files to /kaggle/temp/, but they won't be saved outside of the current session
```

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Mounted at /content/drive
```

```
!unzip drive/My\ Drive/data/dog-vs-cat.zip
```

```
Streaming output truncated to the last 5000 lines.
  inflating: train/train/dog.5499.jpg
  inflating: train/train/dog.55.jpg
  inflating: train/train/dog.550.jpg
  inflating: train/train/dog.5500.jpg
  inflating: train/train/dog.5501.jpg
  inflating: train/train/dog.5502.jpg
  inflating: train/train/dog.5503.jpg
  inflating: train/train/dog.5504.jpg
  inflating: train/train/dog.5505.jpg
  inflating: train/train/dog.5506.jpg
  inflating: train/train/dog.5507.jpg
  inflating: train/train/dog.5508.jpg
  inflating: train/train/dog.5509.jpg
  inflating: train/train/dog.551.jpg
  inflating: train/train/dog.5510.jpg
  inflating: train/train/dog.5511.jpg
  inflating: train/train/dog.5512.jpg
  inflating: train/train/dog.5513.jpg
  inflating: train/train/dog.5514.jpg
  inflating: train/train/dog.5515.jpg
  inflating: train/train/dog.5516.jpg
  inflating: train/train/dog.5517.jpg
  inflating: train/train/dog.5518.jpg
  inflating: train/train/dog.5519.jpg
  inflating: train/train/dog.552.jpg
  inflating: train/train/dog.5520.jpg
  inflating: train/train/dog.5521.jpg
  inflating: train/train/dog.5522.jpg
  inflating: train/train/dog.5523.jpg
  inflating: train/train/dog.5524.jpg
  inflating: train/train/dog.5525.jpg
  inflating: train/train/dog.5526.jpg
  inflating: train/train/dog.5527.jpg
  inflating: train/train/dog.5528.jpg
  inflating: train/train/dog.5529.jpg
  inflating: train/train/dog.553.jpg
  inflating: train/train/dog.5530.jpg
  inflating: train/train/dog.5531.jpg
  inflating: train/train/dog.5532.jpg
  inflating: train/train/dog.5533.jpg
  inflating: train/train/dog.5534.jpg
  inflating: train/train/dog.5535.jpg
  inflating: train/train/dog.5536.jpg
  inflating: train/train/dog.5537.jpg
  inflating: train/train/dog.5538.jpg
  inflating: train/train/dog.5539.jpg
  inflating: train/train/dog.554.jpg
  inflating: train/train/dog.5540.jpg
  inflating: train/train/dog.5541.jpg
  inflating: train/train/dog.5542.jpg
  inflating: train/train/dog.5543.jpg
  inflating: train/train/dog.5544.jpg
  inflating: train/train/dog.5545.jpg
```

```
    inflating: train/train/dog.5546.jpg
    inflating: train/train/dog.5547.jpg
    inflating: train/train/dog.5548.jpg
    inflating: train/train/dog.5549.jpg
```

```
os.listdir('/content/train/train')
#os.listdir('/kaggle/input/dogs-vs-cats/')
```

```
['cat.5159.jpg',
 'cat.4869.jpg',
 'cat.6287.jpg',
 'cat.4140.jpg',
 'cat.7473.jpg',
 'cat.7708.jpg',
 'cat.9746.jpg',
 'dog.10648.jpg',
 'cat.9488.jpg',
 'dog.6925.jpg',
 'cat.2023.jpg',
 'cat.2067.jpg',
 'cat.4734.jpg',
 'dog.3496.jpg',
 'dog.7765.jpg',
 'dog.1560.jpg',
 'dog.174.jpg',
 'cat.10510.jpg',
 'cat.6041.jpg',
 'cat.10383.jpg',
 'dog.429.jpg',
 'dog.1104.jpg',
 'dog.6480.jpg',
 'dog.7422.jpg',
 'cat.8339.jpg',
 'dog.8490.jpg',
 'dog.1128.jpg',
 'dog.11297.jpg',
 'cat.3558.jpg',
 'cat.4831.jpg',
 'cat.4311.jpg',
 'cat.3760.jpg',
 'dog.6678.jpg',
 'cat.2635.jpg',
 'dog.10873.jpg',
 'cat.552.jpg',
 'dog.6220.jpg',
 'cat.4452.jpg',
 'dog.5030.jpg',
 'dog.3626.jpg',
 'cat.8091.jpg',
 'dog.9545.jpg',
 'cat.2848.jpg',
 'cat.791.jpg',
 'cat.4522.jpg',
 'cat.2638.jpg',
 'dog.9447.jpg',
 'cat.6560.jpg',
 'cat.7394.jpg',
 'cat.5153.jpg',
 'cat.7299.jpg',
 'cat.8475.jpg',
 'dog.6026.jpg',
 'cat.8148.jpg',
 'cat.117.jpg',
 'cat.1445.jpg',
 'cat.11986.jpg',
 'cat.7351.jpg',
```

```
filenames=os.listdir('/content/train/train')
```

```
len(filenames)
```

```
25000
```

```
filenames[:10]
```

```
['cat.5159.jpg',
 'cat.4869.jpg',
 'cat.6287.jpg',
 'cat.4140.jpg',
 'cat.7473.jpg',
 'cat.7708.jpg',
 'cat.9746.jpg',
 'dog.10648.jpg',
 'cat.9488.jpg',
 'dog.6925.jpg']
```

```
df=pd.DataFrame({'filename':filenames})
df.head()
```

	filename
0	cat.5159.jpg
1	cat.4869.jpg
2	cat.6287.jpg
3	cat.4140.jpg
4	cat.7473.jpg

```
df['class']=df['filename'].apply(lambda X:X[:3])
```

```
df.head()
```

	filename	class
0	cat.5159.jpg	cat
1	cat.4869.jpg	cat
2	cat.6287.jpg	cat
3	cat.4140.jpg	cat
4	cat.7473.jpg	cat

```
from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
data_gen=ImageDataGenerator(zoom_range=0.2,shear_range=0.2,horizontal_flip=True,rescale=1/255)
```

```
train_data=data_gen.flow_from_dataframe(df,'/content/train/train',X='filename',y='class',target_size=(224,224))
```

```
Found 25000 validated image filenames belonging to 2 classes.
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Conv2D,MaxPool2D,Flatten,Dense
```

```
model=Sequential()
model.add(Conv2D(16,(3,3),activation='relu',input_shape=(224,224,3)))
model.add(MaxPool2D())
model.add(Conv2D(32,(3,3),activation='relu'))
model.add(MaxPool2D())
model.add(Conv2D(64,(3,3),activation='relu'))
model.add(MaxPool2D())
model.add(Conv2D(64,(5,5),activation='relu'))
model.add(MaxPool2D())
model.add(Conv2D(128,(3,3),activation='relu'))
model.add(MaxPool2D())
model.add(Flatten())
model.add(Dense(2,activation='softmax'))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d (Conv2D)             (None, 222, 222, 16)      448       
                                                                 
 max_pooling2d (MaxPooling2D  (None, 111, 111, 16)     0         
 )                                                               
                                                                 
 conv2d_1 (Conv2D)           (None, 109, 109, 32)      4640      
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 54, 54, 32)       0         
 2D)                                                             
                                                                 
 conv2d_2 (Conv2D)           (None, 52, 52, 64)        18496     
                                                                 
 max_pooling2d_2 (MaxPooling  (None, 26, 26, 64)       0         
 2D)                                                             
                                                                 
 conv2d_3 (Conv2D)           (None, 22, 22, 64)        102464    
                                                                 
 max_pooling2d_3 (MaxPooling  (None, 11, 11, 64)       0         
 2D)

 conv2d_4 (Conv2D)           (None, 9, 9, 128)         73856

 max_pooling2d_4 (MaxPooling  (None, 4, 4, 128)        0
 2D)

 flatten (Flatten)           (None, 2048)              0

 dense (Dense)               (None, 2)                 4098

=================================================================
Total params: 204,002
Trainable params: 204,002
Non-trainable params: 0
_________________________________________________________________
```

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

```
model.fit_generator(train_data,epochs=5)
```

```
<ipython-input-49-fd4c89a97472>:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please
  model.fit_generator(train_data,epochs=5)
Epoch 1/5
782/782 [==============================] - 1472s 2s/step - loss: 0.2344 - accuracy: 0.9004
Epoch 2/5
782/782 [==============================] - 1460s 2s/step - loss: 0.2147 - accuracy: 0.9092
Epoch 3/5
331/782 [===========>..................] - ETA: 14:03 - loss: 0.1974 - accuracy: 0.9192
```

```
import cv2
def get_class(img_path):
    img=cv2.imread(img_path)
    img=cv2.resize(img,(224,224))
    img=img/255
    op=model.predict(img.reshape(1,224,224,3)).argmax()
    return 'cat' if op==0 else 'dog'
```

```
train_data.class_mode
```

```
'categorical'
```

```
get_class('model.fit_generator(train_data,epochs=5)')
```

```
---------------------------------------------------------------------------
error                                     Traceback (most recent call last)
<ipython-input-48-d9060c6ec3ad> in <module>
----> 1 get_class('model.fit_generator(train_data,epochs=5)')

<ipython-input-46-132abfc3b56d> in get_class(img_path)
      2 def get_class(img_path):
      3     img=cv2.imread(img_path)
----> 4     img=cv2.resize(img,(224,224))
      5     img=img/255
      6     op=model.predict(img.reshape(1,224,224,3)).argmax()

error: OpenCV(4.6.0) /io/opencv/modules/imgproc/src/resize.cpp:4052: error:
(-215:Assertion failed) !ssize.empty() in function 'resize'
```

SEARCH STACK OVERFLOW