Double-click (or enter) to edit

New section

```
from google.colab import drive
drive. mount('/content/drive')
!unzip drive/My\Drive/dataset.zip
```

```
Mounted at /content/drive
Archive:  drive/MyDrive/dataset.zip
  inflating: dataset/readme.txt
   creating: dataset/test_set/
   creating: dataset/test_set/Cyclone/
  inflating: dataset/test_set/Cyclone/867.jpg
  inflating: dataset/test_set/Cyclone/868.jpg
  inflating: dataset/test_set/Cyclone/869.jpg
  inflating: dataset/test_set/Cyclone/870.jpg
  inflating: dataset/test_set/Cyclone/871.jpg
  inflating: dataset/test_set/Cyclone/872.jpg
  inflating: dataset/test_set/Cyclone/873.jpg
  inflating: dataset/test_set/Cyclone/874.jpg
  inflating: dataset/test_set/Cyclone/875.jpg
  inflating: dataset/test_set/Cyclone/876.jpg
  inflating: dataset/test_set/Cyclone/877.jpg
  inflating: dataset/test_set/Cyclone/878.jpg
  inflating: dataset/test_set/Cyclone/879.jpg
  inflating: dataset/test_set/Cyclone/880.jpg
  inflating: dataset/test_set/Cyclone/881.jpg
  inflating: dataset/test_set/Cyclone/882.jpg
  inflating: dataset/test_set/Cyclone/883.jpg
  inflating: dataset/test_set/Cyclone/884.jpg
  inflating: dataset/test_set/Cyclone/885.jpg
  inflating: dataset/test_set/Cyclone/886.jpg
  inflating: dataset/test_set/Cyclone/887.jpg
  inflating: dataset/test_set/Cyclone/888.jpg
  inflating: dataset/test_set/Cyclone/889.jpg
  inflating: dataset/test_set/Cyclone/890.jpg
  inflating: dataset/test_set/Cyclone/891.jpg
  inflating: dataset/test_set/Cyclone/892.jpg
  inflating: dataset/test_set/Cyclone/893.jpg
  inflating: dataset/test_set/Cyclone/894.jpg
  inflating: dataset/test_set/Cyclone/895.jpg
  inflating: dataset/test_set/Cyclone/896.jpg
  inflating: dataset/test_set/Cyclone/897.jpg
  inflating: dataset/test_set/Cyclone/898.jpg
  inflating: dataset/test_set/Cyclone/899.jpg
  inflating: dataset/test_set/Cyclone/900.jpg
  inflating: dataset/test_set/Cyclone/901.jpg
  inflating: dataset/test_set/Cyclone/902.jpg
  inflating: dataset/test_set/Cyclone/903.jpg
```

```
    inflating: dataset/test_set/Cyclone/904.jpg
    inflating: dataset/test_set/Cyclone/905.jpg
    inflating: dataset/test_set/Cyclone/906.jpg
    inflating: dataset/test_set/Cyclone/907.jpg
    inflating: dataset/test_set/Cyclone/908.jpg
    inflating: dataset/test_set/Cyclone/909.jpg
    inflating: dataset/test_set/Cyclone/910.jpg
    inflating: dataset/test_set/Cyclone/911.jpg
    inflating: dataset/test_set/Cyclone/912.jpg
    inflating: dataset/test_set/Cyclone/913.jpg
    inflating: dataset/test_set/Cyclone/914.jpg
    inflating: dataset/test_set/Cyclone/915.jpg
    inflating: dataset/test_set/Cyclone/916.jpg
    inflating: dataset/test_set/Cyclone/917.jpg
    inflating: dataset/test_set/Cyclone/918.jpg
    inflating: dataset/test_set/Cyclone/919.jpg
```

data augmentation

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


#image Data Agumentation

#setting parameter for Image Data agumentation to the traing data

train_datagen = ImageDataGenerator (rescale=1./255, shear_range=0.2,zoom_range=0.2, h

#Image Data agumentation to the testing data
test_datagen=ImageDataGenerator(rescale=1./255)


#Loading our data and performing data agumentation
#performing data agumentation to train data

x_train = train_datagen.flow_from_directory('/content/dataset/train_set',target_size=

#performing data agumentation to test data

x_test = test_datagen.flow_from_directory('/content/dataset/test_set',target_size=(64
```

```
    Found 742 images belonging to 4 classes.
    Found 198 images belonging to 4 classes.
```

Train test and save model

```
#Importing Neccessary Libraries

import numpy as np #used for numerical analysis
import tensorflow #open source used for both ML and DL for computation

from tensorflow.keras.models import Sequential #it is a plain stack of Layers

from tensorflow.keras import layers #A Layer consists of a tensor-in tensor-out compu

#Dense layer is the regular deeply connected neural network Layer

from tensorflow.keras.layers import Dense, Flatten

#Faltten-used fot flattening the input or change the dimension

from tensorflow.keras.layers import Conv2D, MaxPooling2D #Convolutional Layer

#MaxPooling2O-for downsampling the image

from keras.preprocessing.image import ImageDataGenerator
```

Initializing the model

```
classifier=Sequential()

# First convolution layer and pooling

classifier.add(Conv2D(32, (3, 3), input_shape=(64, 64, 3), activation='relu'))
classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Second convolution layer and pooling

classifier.add(Conv2D(32, (3, 3), activation='relu'))

# input_shape is going to be the pooled feature maps from the previous convolution I

classifier.add(MaxPooling2D(pool_size=(2, 2)))

# Flattening the Layers

classifier.add(Flatten())
```

Adding dense layers cnn

```
# Adding a fully connected Layer
classifier.add(Dense (units=128, activation='relu'))
classifier.add(Dense (units=4, activation='softmax'))
# softmax for more than 2
classifier. summary()
```

```
    Model: "sequential_8"
```

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_10 (Conv2D)          (None, 62, 62, 32)        896

 max_pooling2d_6 (MaxPooling  (None, 31, 31, 32)       0
 2D)

 conv2d_11 (Conv2D)          (None, 29, 29, 32)        9248

 max_pooling2d_7 (MaxPooling  (None, 14, 14, 32)       0
 2D)

 flatten_3 (Flatten)         (None, 6272)              0

 dense_4 (Dense)             (None, 128)               802944

 dense_5 (Dense)             (None, 4)                 516

=================================================================
Total params: 813,604
Trainable params: 813,604
Non-trainable params: 0
_________________________________________________________________
```

```
#Compiling the model

# Compiling the CNN

# categorical_crossentropy for more than 2

classifier.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accura


#fitting the model

classifier.fit_generator( generator=x_train, steps_per_epoch = len(x_train), epochs=2(
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: UserWarning: `Mo
  This is separate from the ipykernel package so we can avoid doing imports unti
Epoch 1/20
149/149 [==============================] - 28s 179ms/step - loss: 1.1876 - accur
Epoch 2/20
149/149 [==============================] - 26s 176ms/step - loss: 0.8671 - accur
Epoch 3/20
149/149 [==============================] - 26s 178ms/step - loss: 0.7304 - accur
Epoch 4/20
149/149 [==============================] - 27s 179ms/step - loss: 0.7039 - accur
Epoch 5/20
149/149 [==============================] - 28s 190ms/step - loss: 0.5969 - accur
Epoch 6/20
149/149 [==============================] - 26s 175ms/step - loss: 0.5413 - accur
Epoch 7/20
```

```
    149/149 [==============================] - 26s 177ms/step - loss: 0.5225 - accur
    Epoch 8/20
    149/149 [==============================] - 28s 190ms/step - loss: 0.4258 - accur
    Epoch 9/20
    149/149 [==============================] - 27s 179ms/step - loss: 0.4013 - accur
    Epoch 10/20
    149/149 [==============================] - 30s 201ms/step - loss: 0.3676 - accur
    Epoch 11/20
    149/149 [==============================] - 26s 177ms/step - loss: 0.4074 - accur
    Epoch 12/20
    149/149 [==============================] - 27s 180ms/step - loss: 0.3413 - accur
    Epoch 13/20
    149/149 [==============================] - 26s 176ms/step - loss: 0.3183 - accur
    Epoch 14/20
    149/149 [==============================] - 26s 177ms/step - loss: 0.2462 - accur
    Epoch 15/20
    149/149 [==============================] - 29s 198ms/step - loss: 0.3194 - accur
    Epoch 16/20
    149/149 [==============================] - 26s 177ms/step - loss: 0.2228 - accur
    Epoch 17/20
    149/149 [==============================] - 26s 177ms/step - loss: 0.1697 - accur
    Epoch 18/20
    149/149 [==============================] - 26s 176ms/step - loss: 0.1958 - accur
    Epoch 19/20
    149/149 [==============================] - 26s 176ms/step - loss: 0.2352 - accur
    Epoch 20/20
    149/149 [==============================] - 26s 176ms/step - loss: 0.1357 - accur
    <keras.callbacks.History at 0x7fb853040910>
```

Save the model as. h5

```
# Save the model

classifier.save('disaster.h5')

model_json = classifier.to_json()

with open("model-bw.json", "w") as json_file:

 json_file. write(model_json)

    WARNING:tensorflow:Compiled the loaded model, but the compiled metrics have yet
```

Testing the model

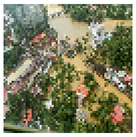
```
 from tensorflow.keras.models import load_model
from keras.preprocessing import image
model = load_model("disaster.h5") #Loading the mode
```

Taking the image as input and checking the result

By using the model we are predicting the output for the given input image. The predicted class index name will be printed here.

```
from tensorflow.keras.preprocessing import image

import numpy as np
img = image.load_img('/content/dataset/test_set/Flood/1009.jpg', target_size=(64,64))
img
#Loading of the image
```



```
x= image.img_to_array(img)
x
#image to array
```

```
    array([[[115., 137., 135.],
            [107., 141., 142.],
            [121., 151., 153.],
            ...,
            [106., 120.,  84.],
            [ 86., 101.,  80.],
            [ 71.,  86.,  63.]],

           [[124., 142., 142.],
            [102., 130., 133.],
            [109., 139., 139.],
            ...,
            [103., 115., 101.],
            [120., 115.,  93.],
            [ 93., 101.,  77.]],

           [[139., 146., 154.],
            [ 99., 114., 119.],
            [106., 130., 130.],
            ...,
            [157., 156., 138.],
            [180., 172., 159.],
            [114., 125.,  91.]],

           ...,

           [[ 63.,  77.,  44.],
            [ 81.,  96.,  57.],
            [106., 115.,  60.],
            ...,
            [ 76.,  71.,  51.],
            [ 62.,  66.,  43.],
            [ 60.,  57.,  38.]],

           [[ 17.,  35.,  21.],
            [  9.,  28.,   9.],
            [ 12.,  27.,   8.],
            ...,
            [131., 113.,  67.],
            [ 92.,  86.,  62.],
            [ 95.,  92.,  75.]],

           [[106., 133., 114.],
            [ 94., 109.,  90.],
            [ 77.,  94.,  75.],
            ...,
            [ 88.,  66.,  16.],
            [157., 134.,  67.],
            [ 89.,  82.,  56.]]], dtype=float32)
```

```
x = np.expand_dims(x,axis = 0)
x
#changing the shape
```

```
     array([[[[[115., 137., 135.],
               [107., 141., 142.],
               [121., 151., 153.],
               ...,
               [106., 120.,  84.],
               [ 86., 101.,  80.],
               [ 71.,  86.,  63.]]],


             [[[124., 142., 142.],
               [102., 130., 133.],
               [109., 139., 139.],
               ...,
               [103., 115., 101.],
               [120., 115.,  93.],
               [ 93., 101.,  77.]]],


             [[[139., 146., 154.],
               [ 99., 114., 119.],
               [106., 130., 130.],
               ...,
               [157., 156., 138.],
               [180., 172., 159.],
               [114., 125.,  91.]]],


             ...,


             [[[ 63.,  77.,  44.],
               [ 81.,  96.,  57.],
               [106., 115.,  60.],
               ...,
               [ 76.,  71.,  51.],
               [ 62.,  66.,  43.],
               [ 60.,  57.,  38.]]],


             [[[ 17.,  35.,  21.],
               [  9.,  28.,   9.],
               [ 12.,  27.,   8.],
               ...,
               [131., 113.,  67.],
               [ 92.,  86.,  62.],
               [ 95.,  92.,  75.]]],


             [[[106., 133., 114.],
               [ 94., 109.,  90.],
               [ 77.,  94.,  75.],
               ...,
               [ 88.,  66.,  16.],
               [157., 134.,  67.],
               [ 89.,  82.,  56.]]]]], dtype=float32)
```

```
from tensorflow.keras.preprocessing import image

import numpy as np
img = image.load_img('/content/dataset/test_set/Flood/1009.jpg', target_size=(64,64))
x= image.img_to_array(img)
x = np.expand_dims(x,axis = 0)
pred = np.argmax(model.predict(x))
Output=['earthquake','cyclone','flood','wildfire']
Output[pred]
#predicting the class
```

```
    1/1 [==============================] - 0s 20ms/step
    'flood'
```