```
#import keras libraries
import numpy as np
import tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from keras.layers import Dense
from keras.layers import Conv2D
from keras.layers import MaxPooling2D,Dropout
from keras.layers import Flatten
```

```
model=Sequential()
```

```
# add Convolutional layer
model.add(Conv2D(32, (3,3), activation = "relu", input_shape = (64,64,3) ))
#1St parameter =no of features detectors 2nd& 3rd =Size of feature detector,
#4th input image size,5 th parameter is channel for color=3 gray scale=1,6 th to avoid negati
```

```
model.add(MaxPooling2D(Pool_size=(2,2)))
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
<ipython-input-5-2989700f946c> in <module>
----> 1 model.add(MaxPooling2D(Pool_size=(2,2)))

                              4 frames
/usr/local/lib/python3.7/dist-packages/keras/utils/generic_utils.py in
validate_kwargs(kwargs, allowed_kwargs, error_message)
   1172   for kwarg in kwargs:
   1173     if kwarg not in allowed_kwargs:
-> 1174       raise TypeError(error_message, kwarg)
   1175
   1176

TypeError: ('Keyword argument not understood:', 'Pool_size')
```

SEARCH STACK OVERFLOW

```
# add flatten layer
model.add(Flatten())
```

```
model.add(Dense(units=128, activation='relu'))
```

```
model.add(Dense(units=46, activation='softmax'))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896

 conv2d_1 (Conv2D)           (None, 60, 60, 32)        9248

 flatten (Flatten)           (None, 115200)            0

 dense (Dense)               (None, 128)               14745728

 dense_1 (Dense)             (None, 46)                5934

=================================================================
Total params: 14,761,806
Trainable params: 14,761,806
Non-trainable params: 0
_________________________________________________________________
```

```
# configure the learning process
model.compile(optimizer='adam',loss='sparse_categorical_crossentropy',metrics=['accuracy'])
```

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

```
train_datagen = ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_f
test_datagen = ImageDataGenerator(rescale=1./255)
```

```
x_train = train_datagen.flow_from_directory(r"/content/drive/MyDrive/Data Collection/training
x_test = test_datagen.flow_from_directory(r"/content/drive/MyDrive/Data Collection/testing",t
```

```
Found 436 images belonging to 2 classes.
Found 121 images belonging to 2 classes.
```

```
model.fit(x_train, epochs=10, steps_per_epoch=len(x_train))
```

```
Epoch 1/10
14/14 [==============================] - 68s 5s/step - loss: 1.3579 - accuracy: 0.6950
Epoch 2/10
14/14 [==============================] - 20s 1s/step - loss: 0.3046 - accuracy: 0.8716
Epoch 3/10
14/14 [==============================] - 20s 1s/step - loss: 0.2812 - accuracy: 0.8830
Epoch 4/10
14/14 [==============================] - 20s 1s/step - loss: 0.3038 - accuracy: 0.8853
Epoch 5/10
14/14 [==============================] - 20s 1s/step - loss: 0.1819 - accuracy: 0.9174
Epoch 6/10
14/14 [==============================] - 20s 1s/step - loss: 0.1349 - accuracy: 0.9518
Epoch 7/10
14/14 [==============================] - 20s 1s/step - loss: 0.1404 - accuracy: 0.9312
```

```
Epoch 8/10
14/14 [==============================] - 20s 1s/step - loss: 0.1181 - accuracy: 0.9472
Epoch 9/10
14/14 [==============================] - 20s 1s/step - loss: 0.1097 - accuracy: 0.9610
Epoch 10/10
14/14 [==============================] - 20s 1s/step - loss: 0.1814 - accuracy: 0.9312
<keras.callbacks.History at 0x7fab4f252250>
```

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

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.moun
```

```
model.save("forestfire13.h5")
```

```
# import load_model from keras.model
from keras.models import load_model
# import image class from keras
from tensorflow.keras.preprocessing import image
# import numpy
import numpy as np
# import cv2
import cv2
```

```
model = load_model("forestfire13.h5")
```

```
img = image.load_img(r'/content/drive/MyDrive/Data Collection/training/Forest with fire/with
x = image.img_to_array(img)
res = cv2.resize(x,dsize=(128,128),interpolation=cv2.INTER_CUBIC)
```

```
img
```

```
x = np.expand_dims(x,axis = 0)
pred = model.predict(x_train)
pred
```

```
14/14 [==============================] - 17s 1s/step
array([[6.4124724e-06, 9.9999350e-01, 7.5694399e-12, ..., 3.9510457e-12,
         2.1328671e-10, 1.2458858e-10],
       [3.8031963e-06, 9.9999613e-01, 1.6327113e-11, ..., 9.4911900e-12,
         5.5777788e-10, 4.7620452e-10],
       [2.6398324e-03, 9.9735844e-01, 6.9926578e-08, ..., 2.6108628e-08,
         2.8292567e-07, 1.3121816e-07],
       ...,
       [7.3989264e-05, 9.9992597e-01, 3.5352973e-10, ..., 1.4010611e-10,
         3.9017349e-09, 2.2922026e-09],
       [2.0621063e-05, 9.9997938e-01, 1.6054391e-10, ..., 9.9064562e-11,
         4.5884576e-09, 1.6456229e-09],
       [2.8511005e-02, 9.7147030e-01, 1.1196929e-06, ..., 3.5170328e-07,
         2.2374422e-06, 7.1359256e-07]], dtype=float32)
```

```
img = image.load_img(r'/content/drive/MyDrive/Data Collection/testing/Forest without fire/0.4
x = image.img_to_array(img)
res = cv2.resize(x,dsize=(128,128),interpolation=cv2.INTER_CUBIC)
```

```
img
```

```
pred
```

```
array([[1.1587713e-03, 9.9883813e-01, 4.0071235e-08, ..., 9.8882033e-08,
         2.0025149e-07, 1.2349827e-07],
        [4.4410473e-07, 9.9999946e-01, 2.4467886e-12, ..., 2.2597707e-11,
         4.9190919e-11, 1.0035421e-11],
        [4.8205187e-03, 9.9517596e-01, 5.1223079e-08, ..., 8.8770307e-08,
         1.7347064e-07, 1.5321189e-07],
        ...,
        [7.0153046e-01, 2.9846936e-01, 8.0397404e-09, ..., 1.8101799e-09,
         1.5282405e-08, 3.4656555e-08],
        [9.8358423e-01, 1.6415808e-02, 3.3958215e-11, ..., 3.5911565e-12,
         4.9828641e-11, 2.2245601e-10],
        [2.5552115e-04, 9.9974447e-01, 1.1333734e-11, ..., 2.3387781e-11,
         9.8332419e-11, 5.7065221e-11]], dtype=float32)
```