PROJECT REPORT
On
ROCK IDENTIFICATION
Using
DEEP CONVOLUTIONAL NEURAL NETWORK

Submitted By:

Sahil Khandual

ABSTRACT:

The automatic identification of rock type in the field would aid geological surveying, education, and automatic mapping. Deep learning is receiving significant research attention for pattern recognition and machine learning. Its application here has effectively identified rock types from images captured in the field. This paper proposes an accurate approach for identifying rock types in the field based on image analysis using deep convolutional neural networks. The proposed approach can identify three common rock types with an overall classification accuracy of 96.6%, thus outperforming other established deep-learning models and a linear model. The results show that the proposed approach based on deep learning represents an improvement in intelligent rock-type identification and solves several difficulties facing the automated identification of rock types in the field.

1. INTRODUCTION:

Rocks are a fundamental component of Earth. They contain the raw materials for virtually all modern construction and manufacturing and are thus indispensable to almost all the endeavors of an advanced society. In addition to the direct use of rocks, mining, drilling, and excavating provide the material sources for metals, plastics, and fuels. Natural rock types have a variety of origins and uses. The three major groups of rocks (igneous, sedimentary, and metamorphic) are further divided into sub-types according to various characteristics. Rock type identification is a basic part of geological surveying and research, and mineral resources exploration.

Rocks are a fundamental component of Earth. The automatic identification of rock type in the field would aid geological surveying, education, and automatic mapping. It is a basic part of geological surveying and research, and mineral resources exploration. The automatic identification of rock type in the field would aid geological surveying, education, and automatic mapping. Working conditions in the field generally limit identification to visual methods, including using a magnifying glass for fine-grained rocks. Visual inspection assesses properties such as colour, composition, grain size, and structure. The attributes of rocks reflect their mineral and chemical composition, formation environment, and genesis. The colour of rock reflects its chemical composition. But these analysis is time taken process to identify the rocks.Its application here has effectively identified rock types from images captured in the field. This paper proposes an accurate approach for identifying rock types in the field based on image analysis using deep convolutional neural networks.

1.1 Overview:

Rock identification using traditional method is time taken process to identify the rocks. Using deep convolutional neural networks we can accurately identify the rock types based on image analysis. In this project, we construct a new convolution neural network for rock classification collecting different rock images.

This project is capable of classifying images based on the type of rock. A convolutional neural network (CNN) convolves an input image which identifies the type of rock. There will be a web application through which user can give their input image then they can check the predicted output and this application is integrated with trained CNN model.

1.2 Purpose:

Its application here has effectively identified rock types from images captured in the field. This project proposes an accurate approach for identifying rock types in the field based on image analysis using deep convolutional neural networks.
This project is capable of classifying images based on the type of rock. A convolutional neural network (CNN) convolves an input image which identifies the type of rock. There will be a web application through which user can give their input image then they can check the predicted output and this application is integrated with trained CNN model.

2. LITERATURE SURVEY

2.1 Existing Problem

Working conditions in the field generally limit identification to visual methods, including using a magnifying glass for fine-grained rocks. Visual inspection assesses properties such as colour, composition, grain size, and structure. The attributes of rocks reflect their mineral and chemical composition, formation environment, and genesis. The colour of rock reflects its chemical composition. But these analysis is time taken process to identify the rocks.

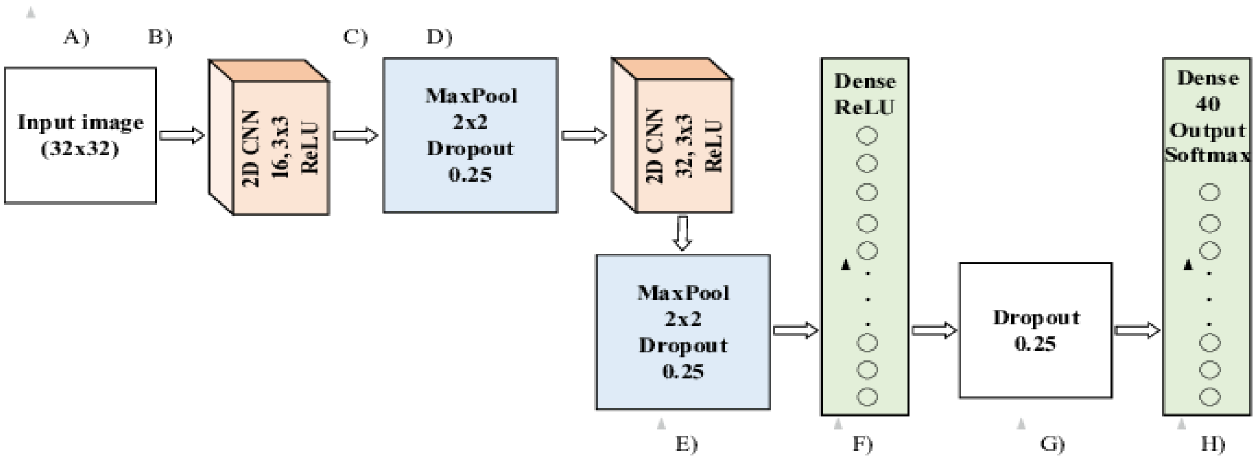
2.2 Proposed Solution:

The identification of rock type by the naked eye is effectively an image recognition task based on knowledge of rock classification. The rapid development of image acquisition and computer image pattern recognition technology has thus allowed the development of automatic systems to identify rocks from images taken in the field. These systems will greatly assist geologists by improving identification accuracy and efficiency and will also help student and newly qualified geologists practice rock-type identification.

First module is optimization model from which the maxima and minima pixel values are selected and is gives to next module. Second module is similarity measures from where the pixel similarity calculated based on location or color. Third module is computational complexity from which data and time complexity measured. Fourth one is the application to collection of images in which we not only taking single image segmentation, our method implements to multiple images. Fifth module is deep learning neural network (DLNN) which provides complete tuning of image and predicting the type of rock. In this module training the image with labeled data and un labeled data because of Deep learning could satisfies to both supervised segmentation and un-supervised segmentation. Last module is the performance evaluation from this module CPU performance time for running each image and accuracy is calculated.

3.THEORITICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware / Software designing:

For hardware we would need ICT tools such as digital camera**.** Here we used both Digital Camera for capturing the rock image and collected the images from internet which is given as input to the web page.

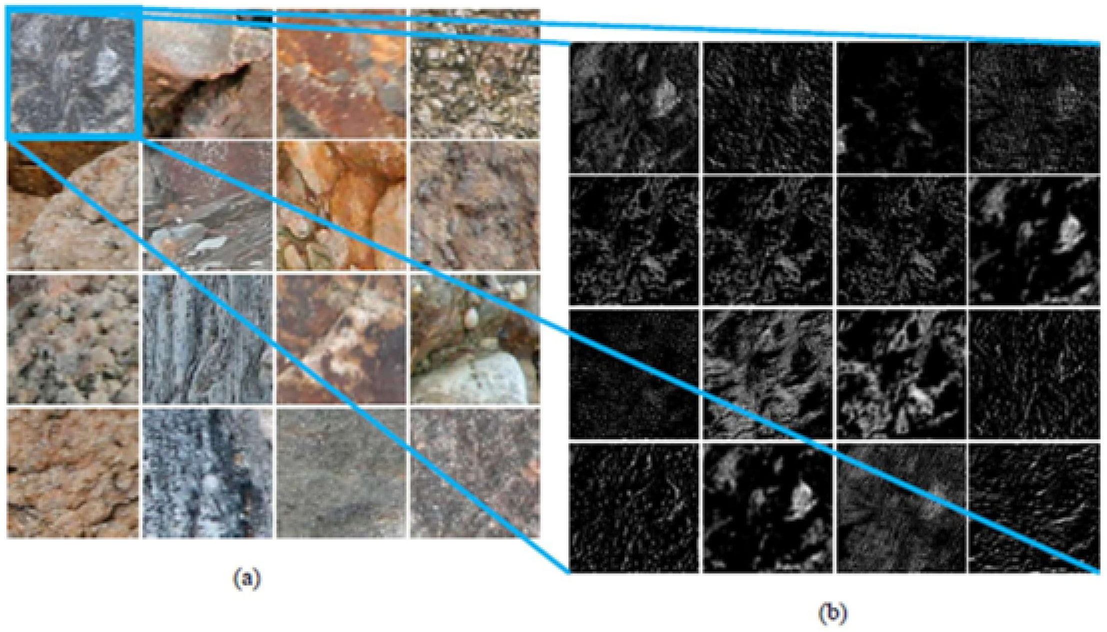
For software we would need a compatible operating system for python, java script and HTML.

Software needed are:

1. Tensorflow
2. OpenCV
3. Keras
4. Flask

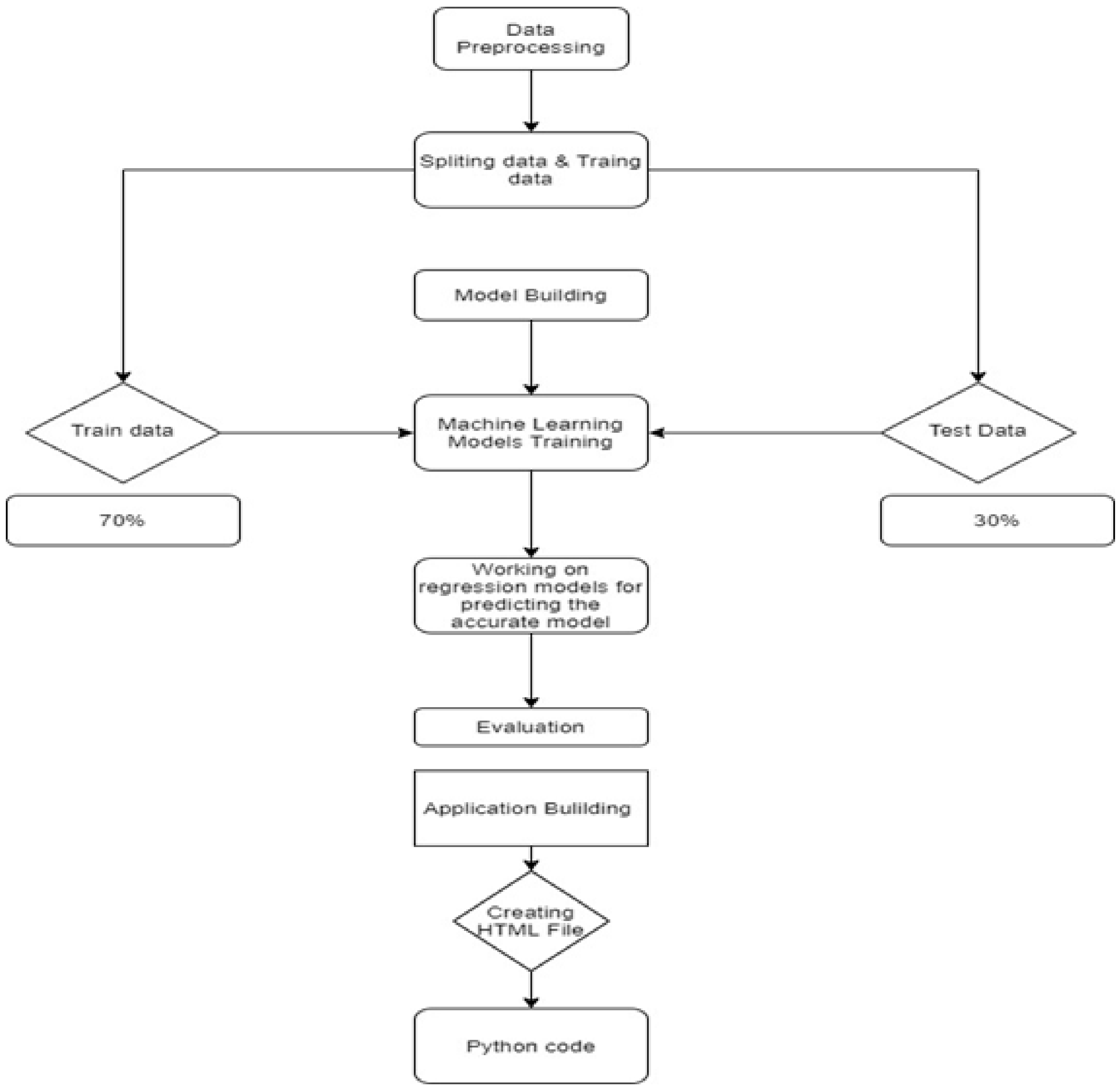
4. EXPERIMENTAL INVESTIGATION

A convolution layer extracts the features of the input images by convolution and outputs the feature maps. It is composed of a series of fixed size filters, known as convolution kernels, which are used to perform convolution operations on image data to produce the feature maps.



(**a**) Input patched field rock sample images. (**b**) Outputted feature maps partly after the first convolution of the input image

5. FLOWCHAT


Data Preprocessing
Spliting data & Traing data
Model Building
Train data
Machine Learning Models Training
Test Data
70%
30%
Working on regression models for predicting the accurate model
Evaluation
Application Bulilding
Creating HTML File
Python code

6. RESULT

We got an accuracy of 0.96 which is good measure for a Convolution Neural Network. The Model predicts the type of rock with good efficiency.

7. ADVANTAGES & DISADVANTAGES

Advantages:

- Effective and predictsaccurately.
- Predicts the type of rock using color, shape andsize.
- Efficiency is maintained by updating the datasetfrequently.

Disadvantages:

- The input image must have clarity to predict the correctoutput.
- Cost of the digital camera ishigh.
- High computationalcost.

8. APPLICATIONS

The traditional method for rock classification is a manual work with many problems such as time-consuming and low accuracy. Hence we use this project to predict the type of rock quickly andaccurately.

- Can be used to identify type of rocks in mines
- Implementable on the website

9. CONCLUSION

The continuing development of CNNs has made them suitable for application in many fields. A deep CNNs model with optimized parameters is proposed here for the accurate identification of rock types from images taken in the field. Novelly, we sliced and patched the original obtained photographic images to increase their suitability for training the model. The sliced samples clearly retain the relevant features of the rock and augment the training dataset. Finally, the proposed deep CNNs model was trained and tested using 624 sample rock images belonging to 3 classes and achieved an overall accuracy of 96.96%. , thereby signifying that the model represents an advance in the automated identification of rock types in the field. The identification of rock type using a deep CNN is quick and easily applied in the field, making this approach useful for geological surveying and for students of geoscience. Meanwhile, the method of identifying rock types proposed in the paper can be applied to the identification of other textures after retraining the corresponding parameters, such as rock thin section images, sporopollen fossil images and so on.

10. FUTURE SCOPE

Although CNNs have helped to identify and classify rock types in the field, some challenges remain. First, the recognition accuracy still needs to be improved. The accuracy of 96.96% achieved using the proposed model meant that 89 images were misidentified in the testing dataset. The model attained relatively low identification accuracy for Igneous rock type, which is attributed to the small grain size and similar colors of these rocks. Furthermore, only a narrow range of sample types (Three rock types overall) was considered in this study. The three main rock groups (igneous, sedimentary, and metamorphic) can be divided into hundreds of types (and subtypes) according to mineral composition. Therefore, our future work will combine the deep learning model with a knowledge library, containing more rock knowledge and relationships among different rock types, to classify more rock types and improve both the accuracy and the range of rock-type identification in the field. In addition, each field photograph often contains more than one rock type, but the proposed model can classify each image into only one category, stressing the importance of the quality of the original image capture.

11. BIBILOGRAPHY

1. Singh N., Singh T.N., Tiwary A., Sarkar M.K. Textural identification of basaltic rock mass using image processing and neural network.

2. Młynarczuk M., Górszczyk A., Ślipek B. The application of pattern recognition in the automatic classification of microscopic rock images.

3. Ślipek B., Młynarczuk M. Application of pattern recognition methods to automatic identification of microscopic images of rocks registered under different polarization and lighting conditions. .

4. Thesmartbridgeteachable.com

12. APPENDIX

A. Source Code

Rock Classification

Steps:

step 1: import the libraries

step 2: Initialize the model

step 3: Add the convolution 2-D layer (Size_of_image,how many feature detected)

step 4: Add the MaxPolling layer(size of the max polling)

step 5: Add Flattening layer (empty) - this acts as an input to the ANN

step 6: Add hidden layer on ann (dimention ,init , activation)

step 7: Add output layer on cnn (dimention =1 , init, activation= sigmoid)

step 8: Import the dataset

step 9: You have to apply Image processing techniques

step 10: Compile (loss=binary_cross_entropy/cetegorical ,metrices=['accuracy'])

step 11: fit (x,y,epochs)

step 12: predict

step 13: Save the model

```
#importing the libraries
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.preprocessing.image import ImageDataGenerator
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizont
al_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)
x_train = train_datagen.flow_from_directory(r'C:\Users\sahil\Desktop\Rock
Classification\Dataset\trainset',target_size=(64,64),batch_size=16,class_mode='categorical')
x_test = train_datagen.flow_from_directory(r'C:\Users\sahil\Desktop\Rock
Classification\Dataset\testset',target_size=(64,64),batch_size=16,class_mode='categorical')
print(x_train.class_indices)
#initialize the model
model = Sequential()
model.add(Convolution2D(32,3,3,input_shape=(64,64,3),activation='relu'))
```

#(32,(3,3)) - 32 defines no of features , (3,3) - size of Convolution

# (64,64,3) - as our dataset contain colored images so the input channel is 3,if it is b&w then 1 , (64,64) -size of image

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

#(2,2) - size of pooling

model.add(Flatten())

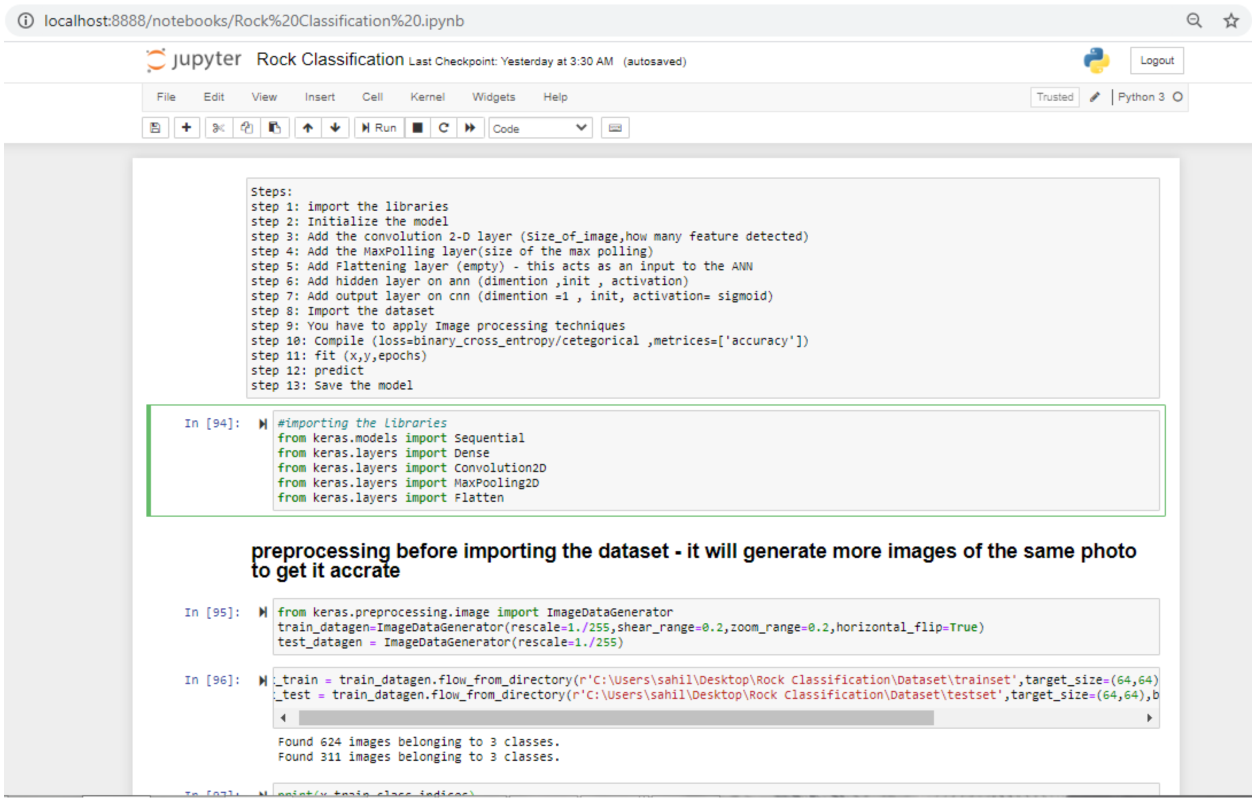
model.add(Dense(output_dim=128,activation='relu',init='random_uniform'))

model.add(Dense(output_dim=3,activation='softmax',init='random_uniform'))

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

model.fit_generator(x_train,steps_per_epoch=39,validation_data=x_test,validation_steps=20,epochs=20)

model.save('rock_classification.h5')


localhost:8888/notebooks/Rock%20Classification%20.ipynb
jupyter Rock Classification Last Checkpoint: Yesterday at 3:30 AM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted Python 3
Run Code

```
Steps:
step 1: import the libraries
step 2: Initialize the model
step 3: Add the convolution 2-D layer (Size_of_image,how many feature detected)
step 4: Add the MaxPolling layer(size of the max polling)
step 5: Add Flattening layer (empty) - this acts as an input to the ANN
step 6: Add hidden layer on ann (dimention ,init , activation)
step 7: Add output layer on cnn (dimention =1 , init, activation= sigmoid)
step 8: Import the dataset
step 9: You have to apply Image processing techniques
step 10: Compile (loss=binary_cross_entropy/cetegorical ,metrices=['accuracy'])
step 11: fit (x,y,epochs)
step 12: predict
step 13: Save the model
```

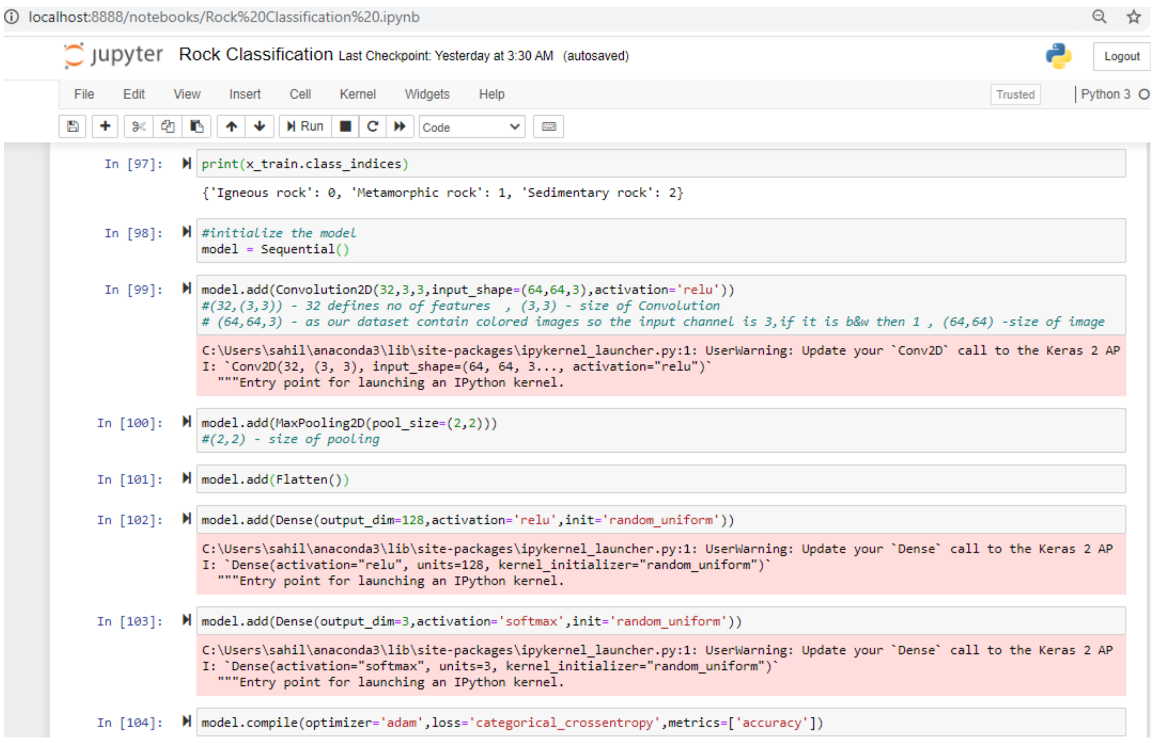
In [94]:
```
#importing the libraries
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
```

preprocessing before importing the dataset - it will generate more images of the same photo to get it accrate

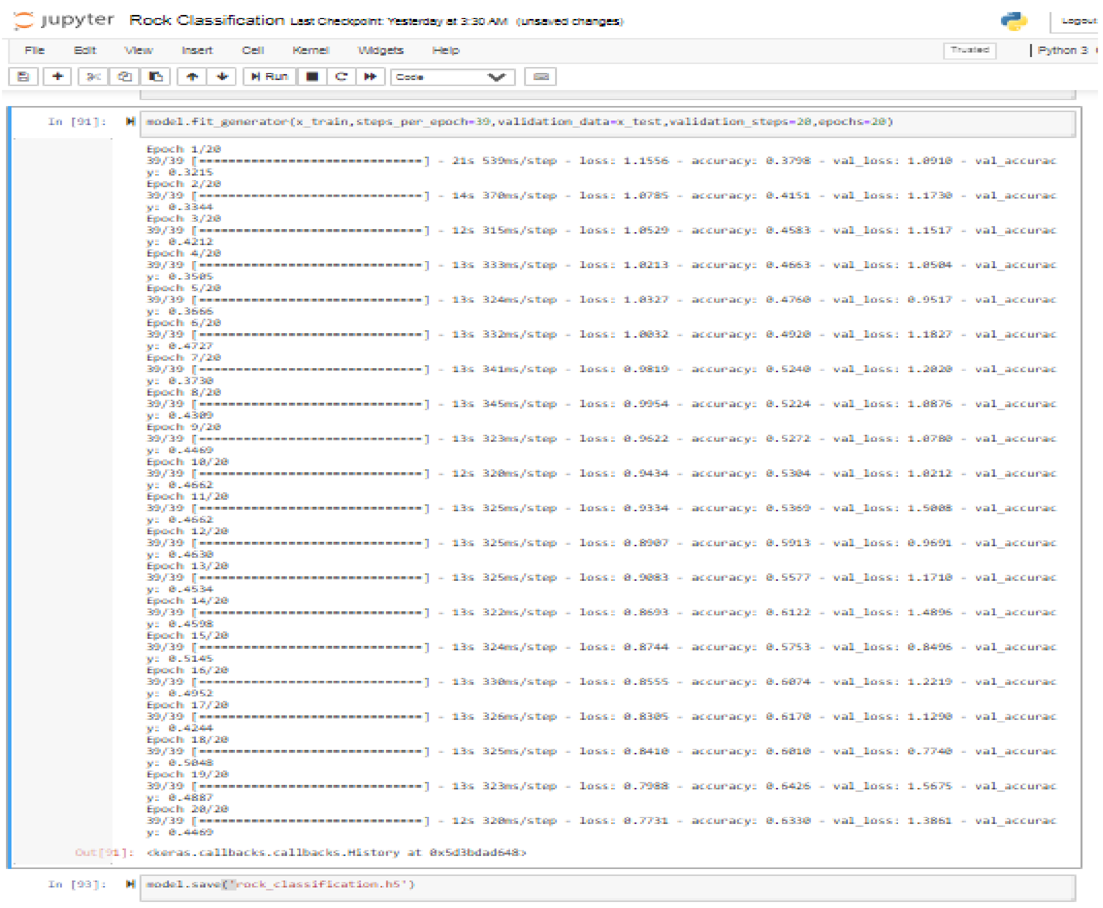
In [95]:
```
from keras.preprocessing.image import ImageDataGenerator
train_datagen=ImageDataGenerator(rescale=1./255,shear_range=0.2,zoom_range=0.2,horizontal_flip=True)
test_datagen = ImageDataGenerator(rescale=1./255)
```

In [96]:
```
_train = train_datagen.flow_from_directory(r'C:\Users\sahil\Desktop\Rock Classification\Dataset\trainset',target_size=(64,64)
_test = train_datagen.flow_from_directory(r'C:\Users\sahil\Desktop\Rock Classification\Dataset\testset',target_size=(64,64),b
```

```
Found 624 images belonging to 3 classes.
Found 311 images belonging to 3 classes.
```


localhost:8888/notebooks/Rock%20Classification%20.ipynb
jupyter Rock Classification Last Checkpoint: Yesterday at 3:30 AM (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

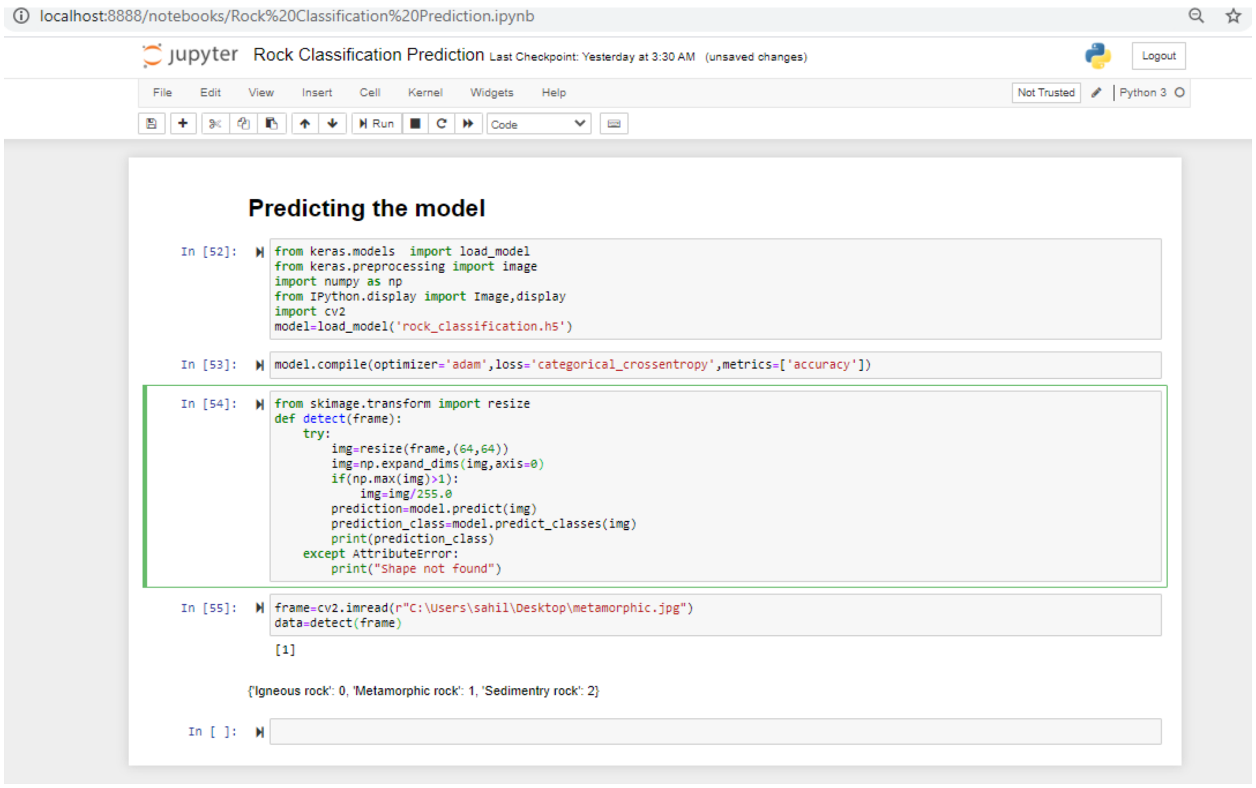
```
In [97]:  print(x_train.class_indices)

          {'Igneous rock': 0, 'Metamorphic rock': 1, 'Sedimentary rock': 2}

In [98]:  #initialize the model
          model = Sequential()

In [99]:  model.add(Convolution2D(32,3,3,input_shape=(64,64,3),activation='relu'))
          #(32,(3,3)) - 32 defines no of features  , (3,3) - size of Convolution
          # (64,64,3) - as our dataset contain colored images so the input channel is 3,if it is b&w then 1 , (64,64) -size of image

          C:\Users\sahil\anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Conv2D` call to the Keras 2 AP
          I: `Conv2D(32, (3, 3), input_shape=(64, 64, 3..., activation="relu")`
            """Entry point for launching an IPython kernel.

In [100]: model.add(MaxPooling2D(pool_size=(2,2)))
          #(2,2) - size of pooling

In [101]: model.add(Flatten())

In [102]: model.add(Dense(output_dim=128,activation='relu',init='random_uniform'))

          C:\Users\sahil\anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 AP
          I: `Dense(activation="relu", units=128, kernel_initializer="random_uniform")`
            """Entry point for launching an IPython kernel.

In [103]: model.add(Dense(output_dim=3,activation='softmax',init='random_uniform'))

          C:\Users\sahil\anaconda3\lib\site-packages\ipykernel_launcher.py:1: UserWarning: Update your `Dense` call to the Keras 2 AP
          I: `Dense(activation="softmax", units=3, kernel_initializer="random_uniform")`
            """Entry point for launching an IPython kernel.

In [104]: model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
```


jupyter Rock Classification Last Checkpoint: Yesterday at 3:30 AM (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

```
In [91]:  model.fit_generator(x_train,steps_per_epoch=39,validation_data=x_test,validation_steps=20,epochs=20)

          Epoch 1/20
          39/39 [==============================] - 21s 539ms/step - loss: 1.1556 - accuracy: 0.3798 - val_loss: 1.0910 - val_accurac
          y: 0.3215
          Epoch 2/20
          39/39 [==============================] - 14s 370ms/step - loss: 1.0785 - accuracy: 0.4151 - val_loss: 1.1730 - val_accurac
          y: 0.3344
          Epoch 3/20
          39/39 [==============================] - 12s 315ms/step - loss: 1.0529 - accuracy: 0.4583 - val_loss: 1.1517 - val_accurac
          y: 0.4212
          Epoch 4/20
          39/39 [==============================] - 13s 333ms/step - loss: 1.0213 - accuracy: 0.4663 - val_loss: 1.0504 - val_accurac
          y: 0.3505
          Epoch 5/20
          39/39 [==============================] - 13s 324ms/step - loss: 1.0327 - accuracy: 0.4760 - val_loss: 0.9517 - val_accurac
          y: 0.3666
          Epoch 6/20
          39/39 [==============================] - 13s 332ms/step - loss: 1.0032 - accuracy: 0.4920 - val_loss: 1.1827 - val_accurac
          y: 0.4727
          Epoch 7/20
          39/39 [==============================] - 13s 341ms/step - loss: 0.9819 - accuracy: 0.5240 - val_loss: 1.2020 - val_accurac
          y: 0.3730
          Epoch 8/20
          39/39 [==============================] - 13s 345ms/step - loss: 0.9954 - accuracy: 0.5224 - val_loss: 1.0876 - val_accurac
          y: 0.4309
          Epoch 9/20
          39/39 [==============================] - 13s 323ms/step - loss: 0.9622 - accuracy: 0.5272 - val_loss: 1.0780 - val_accurac
          y: 0.4469
          Epoch 10/20
          39/39 [==============================] - 12s 320ms/step - loss: 0.9434 - accuracy: 0.5304 - val_loss: 1.0212 - val_accurac
          y: 0.4662
          Epoch 11/20
          39/39 [==============================] - 13s 325ms/step - loss: 0.9334 - accuracy: 0.5369 - val_loss: 1.5008 - val_accurac
          y: 0.4662
          Epoch 12/20
          39/39 [==============================] - 13s 325ms/step - loss: 0.8907 - accuracy: 0.5913 - val_loss: 0.9691 - val_accurac
          y: 0.4630
          Epoch 13/20
          39/39 [==============================] - 13s 325ms/step - loss: 0.9083 - accuracy: 0.5577 - val_loss: 1.1710 - val_accurac
          y: 0.4534
          Epoch 14/20
          39/39 [==============================] - 13s 322ms/step - loss: 0.8693 - accuracy: 0.6122 - val_loss: 1.4896 - val_accurac
          y: 0.4598
          Epoch 15/20
          39/39 [==============================] - 13s 324ms/step - loss: 0.8744 - accuracy: 0.5753 - val_loss: 0.8496 - val_accurac
          y: 0.5145
          Epoch 16/20
          39/39 [==============================] - 13s 330ms/step - loss: 0.8555 - accuracy: 0.6074 - val_loss: 1.2219 - val_accurac
          y: 0.4952
          Epoch 17/20
          39/39 [==============================] - 13s 326ms/step - loss: 0.8305 - accuracy: 0.6170 - val_loss: 1.1290 - val_accurac
          y: 0.4244
          Epoch 18/20
          39/39 [==============================] - 13s 325ms/step - loss: 0.8410 - accuracy: 0.6010 - val_loss: 0.7740 - val_accurac
          y: 0.5048
          Epoch 19/20
          39/39 [==============================] - 13s 323ms/step - loss: 0.7988 - accuracy: 0.6426 - val_loss: 1.5675 - val_accurac
          y: 0.4887
          Epoch 20/20
          39/39 [==============================] - 12s 320ms/step - loss: 0.7731 - accuracy: 0.6330 - val_loss: 1.3861 - val_accurac
          y: 0.4469

Out[91]:  <keras.callbacks.callbacks.History at 0x5d3bdad648>

In [93]:  model.save('rock_classification.h5')
```

Rock Classification Prediction

```
from keras.models  import load_model
from keras.preprocessing import image
import numpy as np
from IPython.display import Image,display
import cv2
model=load_model('rock_classification.h5')

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

from skimage.transform import resize
def detect(frame):
    try:
        img=resize(frame,(64,64))
        img=np.expand_dims(img,axis=0)
        if(np.max(img)>1):
            img=img/255.0
        prediction=model.predict(img)
        print(prediction)
        prediction_class=model.predict_classes(img)
        print(prediction_class)
    except AttributeError:
        print("Shape not found")
frame=cv2.imread(r"C:\Users\sahil\Desktop\metamorphic.jpg")
data=detect(frame)
```

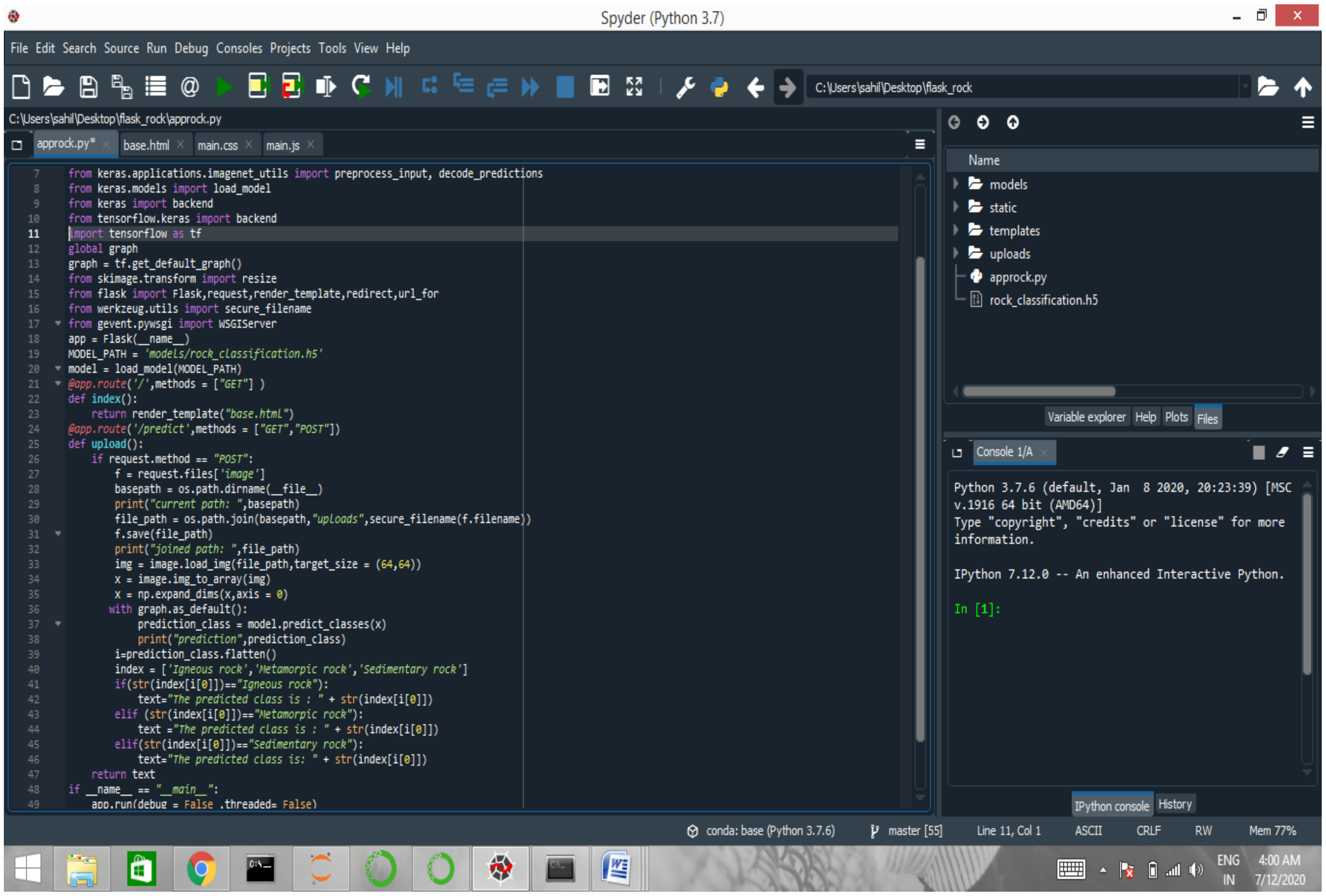

localhost:8888/notebooks/Rock%20Classification%20Prediction.ipynb
jupyter Rock Classification Prediction Last Checkpoint: Yesterday at 3:30 AM (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3
Run
Code

Predicting the model

```
In [52]: from keras.models  import load_model
         from keras.preprocessing import image
         import numpy as np
         from IPython.display import Image,display
         import cv2
         model=load_model('rock_classification.h5')

In [53]: model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])

In [54]: from skimage.transform import resize
         def detect(frame):
             try:
                 img=resize(frame,(64,64))
                 img=np.expand_dims(img,axis=0)
                 if(np.max(img)>1):
                     img=img/255.0
                 prediction=model.predict(img)
                 prediction_class=model.predict_classes(img)
                 print(prediction_class)
             except AttributeError:
                 print("Shape not found")

In [55]: frame=cv2.imread(r"C:\Users\sahil\Desktop\metamorphic.jpg")
         data=detect(frame)

         [1]

{'Igneous rock': 0, 'Metamorphic rock': 1, 'Sedimentry rock': 2}

In [ ]:
```

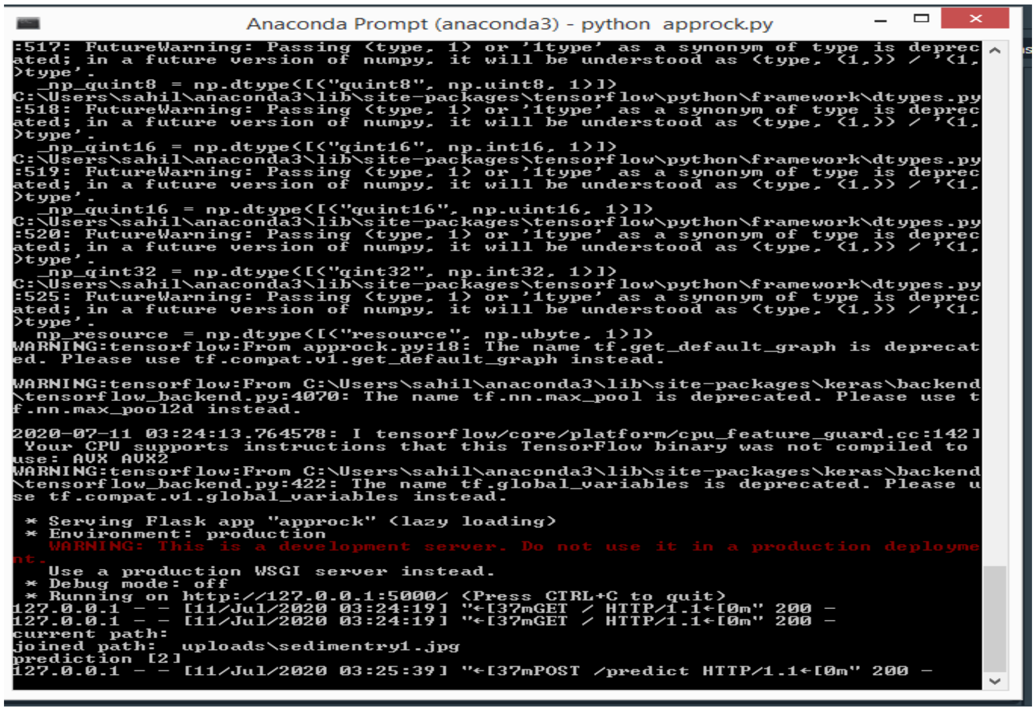
FLASK

```
from __future__ import division , print_function
import sys
import os
import glob
import numpy as np
from keras.preprocessing import image
from keras.applications.imagenet_utils import preprocess_input, decode_predictions

from keras.models import load_model
from keras import backend
from tensorflow.keras import backend




import tensorflow as tf
global graph
graph = tf.get_default_graph()
from skimage.transform import resize
from flask import Flask,request,render_template,redirect,url_for
from werkzeug.utils import secure_filename
from gevent.pywsgi import WSGIServer
app = Flask(__name__)
MODEL_PATH = 'models/rock_classification.h5'
model = load_model(MODEL_PATH)
@app.route('/',methods = ["GET"] )
def index():
    return render_template("base.html")
@app.route('/predict',methods = ["GET","POST"])
def upload():
    if request.method == "POST":
        f = request.files['image']
        basepath = os.path.dirname(__file__)
        print("current path: ",basepath)
        file_path = os.path.join(basepath,"uploads",secure_filename(f.filename))
        f.save(file_path)
        print("joined path: ",file_path)
        img = image.load_img(file_path,target_size = (64,64))
        x = image.img_to_array(img)
        x = np.expand_dims(x,axis = 0)

        with graph.as_default():
            prediction_class = model.predict_classes(x)
            print("prediction",prediction_class)

        i=prediction_class.flatten()
        index = ['Igneous rock','Metamorpic rock','Sedimentary rock']
        if(str(index[i[0]])=="Igneous rock"):
            text="The predicted class is : " + str(index[i[0]])
        elif (str(index[i[0]])=="Metamorpic rock"):
            text ="The predicted class is : " + str(index[i[0]])
        elif(str(index[i[0]])=="Sedimentary rock"):
            text="The predicted class is: " + str(index[i[0]])
    return text
if __name__ == "__main__":
    app.run(debug = False ,threaded= False)
```



```
from keras.applications.imagenet_utils import preprocess_input, decode_predictions
from keras.models import load_model
from keras import backend
from tensorflow.keras import backend
import tensorflow as tf
global graph
graph = tf.get_default_graph()
from skimage.transform import resize
from flask import Flask,request,render_template,redirect,url_for
from werkzeug.utils import secure_filename
from gevent.pywsgi import WSGIServer
app = Flask(__name__)
MODEL_PATH = 'models/rock_classification.h5'
model = load_model(MODEL_PATH)
@app.route('/',methods = ["GET"] )
def index():
    return render_template("base.html")
@app.route('/predict',methods = ["GET","POST"])
def upload():
    if request.method == "POST":
        f = request.files['image']
        basepath = os.path.dirname(__file__)
        print("current path: ",basepath)
        file_path = os.path.join(basepath,"uploads",secure_filename(f.filename))
        f.save(file_path)
        print("joined path: ",file_path)
        img = image.load_img(file_path,target_size = (64,64))
        x = image.img_to_array(img)
        x = np.expand_dims(x,axis = 0)
       with graph.as_default():
            prediction_class = model.predict_classes(x)
            print("prediction",prediction_class)
        i=prediction_class.flatten()
        index = ['Igneous rock','Metamorpic rock','Sedimentary rock']
        if(str(index[i[0]])=="Igneous rock"):
            text="The predicted class is : " + str(index[i[0]])
        elif (str(index[i[0]])=="Metamorpic rock"):
            text ="The predicted class is : " + str(index[i[0]])
        elif(str(index[i[0]])=="Sedimentary rock"):
            text="The predicted class is: " + str(index[i[0]])
    return text
if __name__ == "__main__":
    app.run(debug = False ,threaded= False)
```

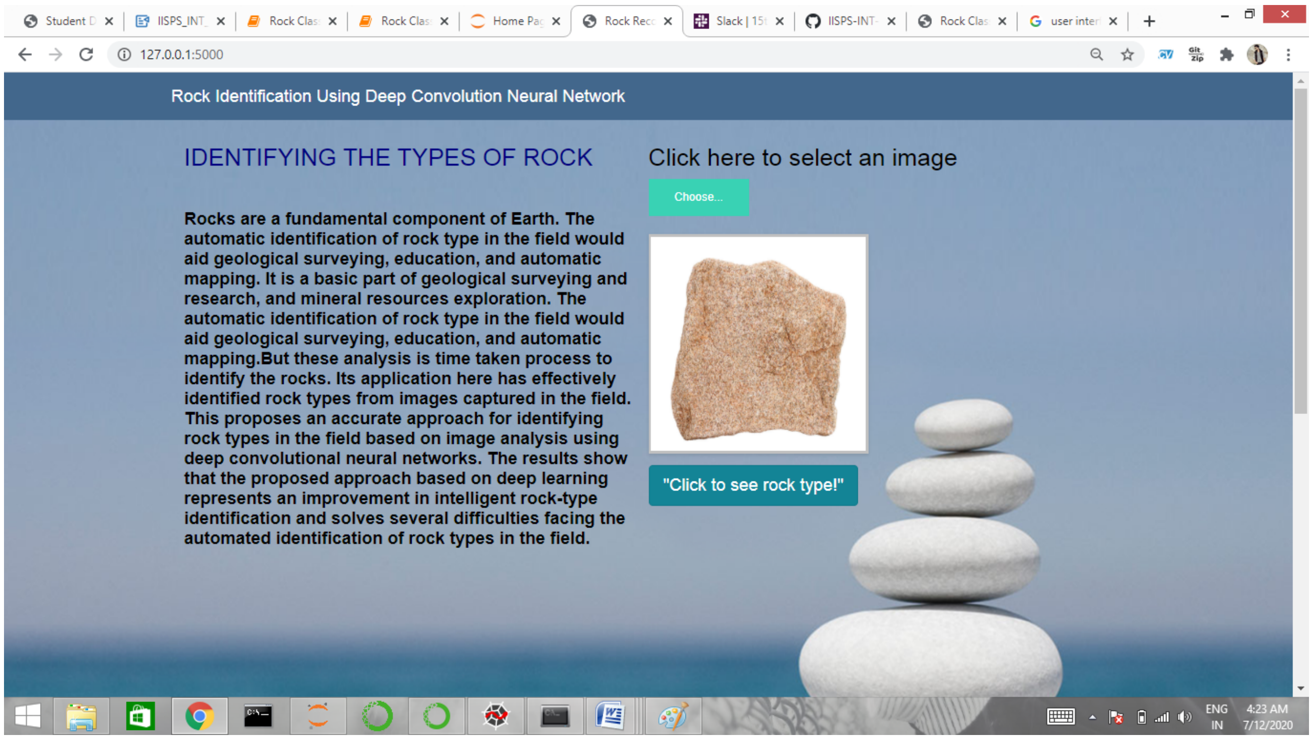
```
Python 3.7.6 (default, Jan  8 2020, 20:23:39) [MSC
v.1916 64 bit (AMD64)]
Type "copyright", "credits" or "license" for more
information.

IPython 7.12.0 -- An enhanced Interactive Python.

In [1]:
```



Anaconda Prompt (anaconda3) - python approck.py

```
:517: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprec
ated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,
)type'.
  _np_quint8 = np.dtype([("quint8", np.uint8, 1)])
C:\Users\sahil\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py
:518: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprec
ated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,
)type'.
  _np_qint16 = np.dtype([("qint16", np.int16, 1)])
C:\Users\sahil\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py
:519: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprec
ated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,
)type'.
  _np_quint16 = np.dtype([("quint16", np.uint16, 1)])
C:\Users\sahil\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py
:520: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprec
ated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,
)type'.
  _np_qint32 = np.dtype([("qint32", np.int32, 1)])
C:\Users\sahil\anaconda3\lib\site-packages\tensorflow\python\framework\dtypes.py
:525: FutureWarning: Passing (type, 1) or '1type' as a synonym of type is deprec
ated; in a future version of numpy, it will be understood as (type, (1,)) / '(1,
)type'.
  np_resource = np.dtype([("resource", np.ubyte, 1)])
WARNING:tensorflow:From approck.py:18: The name tf.get_default_graph is deprecat
ed. Please use tf.compat.v1.get_default_graph instead.

WARNING:tensorflow:From C:\Users\sahil\anaconda3\lib\site-packages\keras\backend
\tensorflow_backend.py:4070: The name tf.nn.max_pool is deprecated. Please use t
f.nn.max_pool2d instead.

2020-07-11 03:24:13.764578: I tensorflow/core/platform/cpu_feature_guard.cc:142]
 Your CPU supports instructions that this TensorFlow binary was not compiled to
use: AVX AVX2
WARNING:tensorflow:From C:\Users\sahil\anaconda3\lib\site-packages\keras\backend
\tensorflow_backend.py:422: The name tf.global_variables is deprecated. Please u
se tf.compat.v1.global_variables instead.

 * Serving Flask app "approck" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployme
nt.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
127.0.0.1 - - [11/Jul/2020 03:24:19] "←[37mGET / HTTP/1.1←[0m" 200 -
127.0.0.1 - - [11/Jul/2020 03:24:19] "←[37mGET / HTTP/1.1←[0m" 200 -
current path:
joined path:  uploads\sedimentry1.jpg
prediction [2]
127.0.0.1 - - [11/Jul/2020 03:25:39] "←[37mPOST /predict HTTP/1.1←[0m" 200 -
```

B.Ouput Screenshots

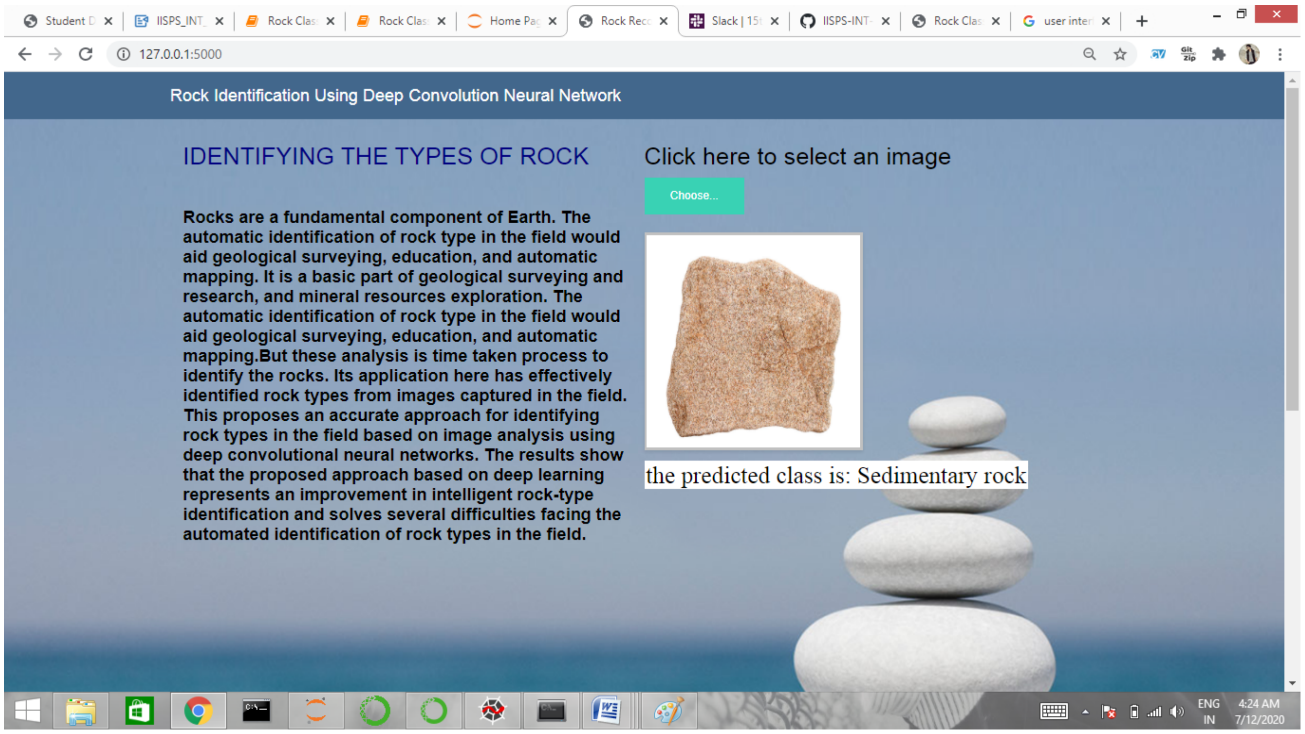


Rock Identification Using Deep Convolution Neural Network

IDENTIFYING THE TYPES OF ROCK

Click here to select an image

Rocks are a fundamental component of Earth. The automatic identification of rock type in the field would aid geological surveying, education, and automatic mapping. It is a basic part of geological surveying and research, and mineral resources exploration. The automatic identification of rock type in the field would aid geological surveying, education, and automatic mapping.But these analysis is time taken process to identify the rocks. Its application here has effectively identified rock types from images captured in the field. This proposes an accurate approach for identifying rock types in the field based on image analysis using deep convolutional neural networks. The results show that the proposed approach based on deep learning represents an improvement in intelligent rock-type identification and solves several difficulties facing the automated identification of rock types in the field.



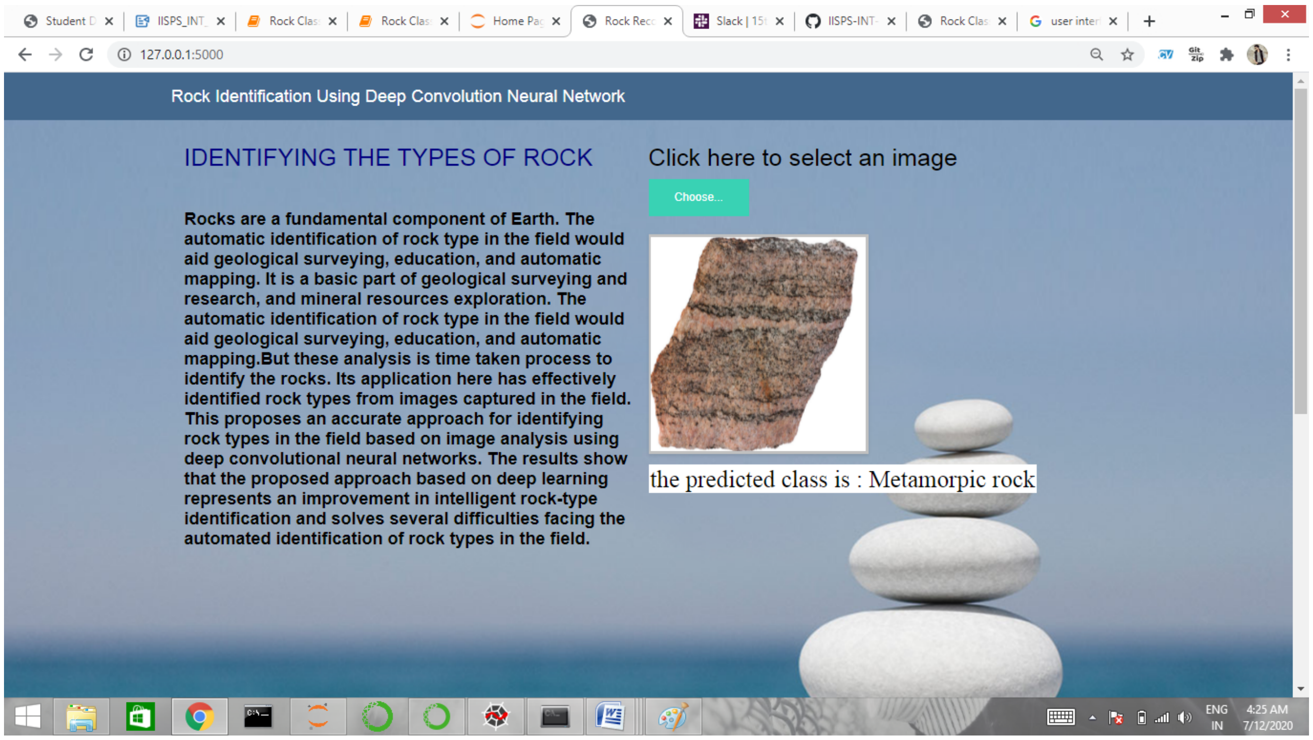
Rock Identification Using Deep Convolution Neural Network

IDENTIFYING THE TYPES OF ROCK

Click here to select an image

Rocks are a fundamental component of Earth. The automatic identification of rock type in the field would aid geological surveying, education, and automatic mapping. It is a basic part of geological surveying and research, and mineral resources exploration. The automatic identification of rock type in the field would aid geological surveying, education, and automatic mapping.But these analysis is time taken process to identify the rocks. Its application here has effectively identified rock types from images captured in the field. This proposes an accurate approach for identifying rock types in the field based on image analysis using deep convolutional neural networks. The results show that the proposed approach based on deep learning represents an improvement in intelligent rock-type identification and solves several difficulties facing the automated identification of rock types in the field.

the predicted class is: Sedimentary rock



Rock Identification Using Deep Convolution Neural Network

IDENTIFYING THE TYPES OF ROCK

Rocks are a fundamental component of Earth. The automatic identification of rock type in the field would aid geological surveying, education, and automatic mapping. It is a basic part of geological surveying and research, and mineral resources exploration. The automatic identification of rock type in the field would aid geological surveying, education, and automatic mapping.But these analysis is time taken process to identify the rocks. Its application here has effectively identified rock types from images captured in the field. This proposes an accurate approach for identifying rock types in the field based on image analysis using deep convolutional neural networks. The results show that the proposed approach based on deep learning represents an improvement in intelligent rock-type identification and solves several difficulties facing the automated identification of rock types in the field.

Click here to select an image

the predicted class is : Metamorpic rock