Internship Title	:	RSIP Career Basic AI 043
Project ID	:	SPS_PRO_172
Project Title	:	Rock identification using deep convolution neural network
TEAM		T

ROCK IDENTIFICATION USING DEEP CONVOLUTIONAL NEURAL NETWORKS

1 INTRODUCTION

1.1 OVERVIEW

Rock identification using deep convolutional neural network project is about to classify and identify different types of images .

TOOLS USED

python
flask algorithm
convolutional neural networks.

1.2 PURPOSE

This is used to find the automatic rock type and solves various problems.

This project is used in the filed of geology very extensively to identify types of rocks in an effectively and easily.

2. LITERATURE SURVEY

2.1 EXISTING PROBLEM

Identification of different types of rocks is very hard and often working conditions in the field are generally limited to different visual methods .

2.2PROPESED SOLUTIONS

The proposal solution includes a deep learning based approch to identify different rocks which solves difficulties faced while working in the field.The solutions is to use convolution neural network model to solve the problem of indentification of rocks in an effective manner.

3.THEORITICAL ANALYSIS

3.1 BLOCK DIAGRAM

ROCK
IMAGES.....>CNN.....>IGNEOUS/METAMORPHIC/SEDIMENTARY

3.2 HARDWARE/SOFTWARE DESIGNING

SOFTWARE USED:

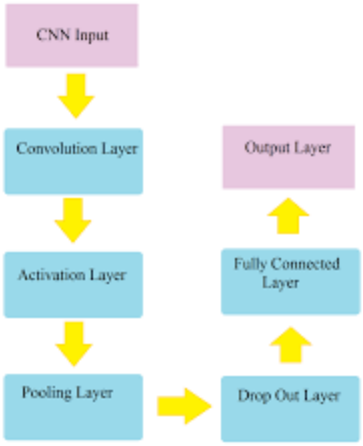
PYTHON
FLASK WEB FRAMEWORK

4.EXPERIMENTAL INVESTIGATION

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. This project can be used in the identification of rocks in an easy manner .The dataset consist of three types of rocks and the CNN used here are identified the rocks.

5.FLOWCHART:



6.RESULT:

The deep cnn learning model can be used in different types of rock images and their structure in a very easy manner. Here class of rocks Igneous,metamorphic,sedimentary
can be identified . The model can identify images using

Flask framework upon uploading
the images.

7.ADVANTAGES AND DISADVANTAGES:

1. Effectively identified rock types from images captured in the field.
2. Improvement in intelligent rock-type identification and solves several difficulties facing the automated identification of rock types in the field
3. Who are experienced in the field of geological they can identify the rocks easily. But who are new to the field, it can help to identify the type of rock.
4. Machine should be trained with a larger data set to get accurate output.

8.APPLICATIONS

Identifying rock types in the fileds
Automated identification of rock tyypes

9.CONCLUSION:

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.

10.FUTURE SCOPE:

Deep learning is receiving significant research attention for pattern recognition and machine learning. Its application here has effectively identified all types from images captured in any field.

BIBILOGRAPHY AND APPENDIX:

https://thesmartbridge.com/documents/spsaimldocs/CNNcollection.pdf
https://thesmartbridge.com/documents/spsaimldocs/CNNprep.pdf
https://thesmartbridge.com/documents/spsaimldocs/CNNflow.pdf
https://www.w3schools.com/bootstrap/bootstrap_forms_inputs.asp

PROJECT DESCRIPTION:

STEP1 : CREATING A TRAINING AND TESTING FOLDER

Follow the below steps to create your data set for image classification Of Rocks.

Create a folder and name it as dataset anywhere in your desktop

Go to the dataset folder and create another two subfolders and name them as trainset and testset.

Go to the trainset folder and create folders with respect to the categories to classify.

Go to testset folder and create three sub folders and name them as Igneous ,sedimentary and metamorphic.

Now collect the images of rocks and 70% of the images collected in the subfolder which lies in dataset/trainset.

Paste remaining 30% in the subfolder lies in dataset/test set.

STEP2:PREPROCESSING THE DATA

Loading and pre-processing the data:

Data is gold as far as deep learning models are concerned. Your image classification model has a far better chance of performing well if you have a good amount of images in the training set. Also, the shape of the data varies according to the architecture/framework that we use. Hence, the critical data pre-processing step (the eternally important step in any project). I highly recommend going through the “basics of image processing using Python we use Keras’ ImageDataGenerator class to perform data augmentation. i.e, we are using some kind of parameters to process our collected data.

The word “augment” means to make something “greater” or “increase” something (in this case, data), the Keras ImageDataGenerator class actually works by:

Accepting a batch of images used for training.

Taking this batch and applying a series of random transformations to each image in the batch (including random rotation, resizing, shearing, etc.).

Replacing the original batch with the new, randomly transformed batch.

Training the CNN on this randomly transformed batch (i.e., the original data itself is not used for training).

Note: The ImageDataGenerator accepts the original data, randomly transforms it, and returns only the new, transformed data.

Import the library

Define the parameters /arguments for ImageDataGenerator class

Note: The ImageDataGenerator transforms each image in the batch by a series of random translations, these translations are based on the arguments

Applying ImageDataGenerator functionality to trainset and testset

STEP 3: MODEL BUILDING

Building A CNN model Steps to Build a Deep Learning Model

1. Defining the model architecture
2. Configure the learning process
3. Train The Model
4. Save the Model
5. Predictions

Step1: Defining Model architecture: For image

classification we use Convolution neural network This is a very crucial step in our deep learning model building process. We have to define how our model will look and that requires

Importing the libraries

Initializing the model

Adding CNN (Convolution Neural Network) Layers

Adding Dense layers

Importing the Libraries:

Initializing the model: Keras has 2 ways to define a neural network:

Sequential

Function API

The Sequential class is used to define a linear initializations of network layers which then, collectively, constitute a model.

In our example below, we will use the Sequential constructor to create a model, which will then have layers added to it using the add() method. Adding a CNN Layers: We will be adding three layers for CNN

Convolution layer

Pooling layer

Flattening layer

Adding Dense Layers:

Once you are done with initializing all the layers you have to pre-process the images.

Step 2: Configuring the learning process: With both the training data defined and model defined, it's time configure the learning process. This is accomplished with a call to the compile() method of the Sequential model class.

Compilation requires 3 arguments: an optimizer, a loss function, and a list of metrics. In our example, set up as a multi-class classification problem, we will use the Adam optimizer, the categorical cross entropy loss function, and include solely the accuracy metric.

Step 3: Train the model At this point we have training data and a fully configured neural network to train with said data. All that is left is to pass the data to the model for the training process to commence, a process which is completed by iterating on the training data. Training begins by calling the fit() method. Note: This steps takes few minutes based on the epochs (no : of time you would like to train the machine with the given data set) you give .

Step 4: Save The Model: Your model is to be saved for the future purpose. This saved model ac also be integrated with android application or web application in order to predict something

Step 5: Prediction: The last and final step is to make use of Saved model to do predictions. We use load model class to load the model. We use imread() class from opencv library to read an image and give it to the model to predict the result. Before giving the original image to predict the class, we have to pre-process that image and apply predictions to get accurate results

STEP 4: APPLICATION BUILDING

Flask Frame Work with Deep Learning Model In this section we will be building a web application which is integrated to the model we built. An UI is provided for the uses where he has to upload a picture/photo. The uploaded photo is given to the model built, and prediction is showcased on the UI. We are using a Deep Learning Model which is built for ROCK recognition and saved this file as “model1.h5”.

We have built our model using 5 classes To build this you should know basics of “HTML, CSS, Bootstrap, flask framework and python, Opencv” Create a project folder which should contains

An python fill called app.y

Your Deep learning alogorithm file

Models folder with Model file

Templates folder which contains sample.HTML file

c folder which contains css folder which contains styles.css and js foder which contains js file

Steps 1: building an Index. Html file This is the basic HTML page for our Project. Here we are creating two buttons one used to browse pictures from local drive and other button is used to send this picture to the model file which analyses the picture and show cases the prediction in the result. The browsed picture is displayed on the html page using on image preview using main.js script. You can see the scripted file is called in html page using src tag

Step 2: Build python code We will be using python for server side scripting. Let's see step by step process for writing backend code.

Importing Libraries We are importing the libraries for processing the uploaded picture Loading the built model, saving the browsed pictures in uploads folder. And dependent library default graph for model prediction Importing flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of current module (__name__) as argument We are giving the path where our model file is located and from that path we are loading our model file

Routing to the html Page Here we will be using declared constructor to route to the html page which we have created earlier. In the above example, '/' URL is bound with index.html function. Hence, when the home page of web server is opened in browser, the html page will be rendered. Whenever you browse an image from the html page this photo can be accesses through POST or GET Method.

Showcasing prediction on UI Here we are defining a function which request the browsed file from html page using post method. The requested picture file is then saved to the uploads folder in this same directory using OS library. Using load image class from Keras library we are retrieving the saved picture from the path declared. We are applying some image processing techniques and then sending that preprocessed image to the model for predicting the class. This returns the numerical value of a class (like 0,1 ,2 etc.) which lies in the 0th index of the variable preds. This numerical value is passed to the index variable declared. This returns the name of the class. This name is rendered to the predict variable used in html page Main Function This is used to run the application in local host Total python code:

Step 3: How to Run The app

Open anaconda prompt from start menu

Navigate to the folder where your app.py resides

Now type “python app.py” command

It will show the local host where your app is running.

Navigate to the localhost where you can view your web page

Browse the image and click on predict function to see prediction.