National College of Ireland

Rockfall Detection on Mars using Deep Learning Algorithm

MSc Research Project
Data Analytics

Soumi Dutta Bhowmik
Student ID: x20189460

School of Computing
National College of Ireland

Supervisor: Christian Horn

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……. …Soumi Dutta Bhowmik……

Student ID: ……x20189460……

Programme:……Msc Data Analytics… **Year:** …2021-2022..

Module: ……Research Project……..……

Supervisor: ……Christian Horn……..……

Submission Due Date: ……15/08/2022……..……

Project Title: ……Rockfall Detection on Mars Using Deep Learning……..……

Word Count: ……5305…… **Page Count**…17……..……..

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……Soumi Dutta Bhowmik……

Date: ……15/08/2022……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Rockfall Detection on Mars by using Deep learning

Soumi Dutta Bhowmik
X20189460

Abstract

The aim of this project is to detect the rockfall on Mars which helps to learn about the surface of Mars and its formation and frequency of it. The trails of this rockfall can help to do the surface analysis for the scientist. Previously all these used happen just by doing manual mappings. The manual technique has its own drawbacks like time consumption and providing less information. Recently there are few research happened which showed good success for rockfall detection by data analysing. This research provides Deep learning approach for the same as there are studies in crater detection, rockfall detection on Moon. In this research three models have been chosen and compared like CNN, VGG16 and VGG19. CNN has given accuracy of 67% and VGG16 is at 94% and VGG19 is at 48%.

1. Introduction

In this recent period, Mars is a planet with lots of complex geographical situation which gives interest to study about this planet and its living existence on its surface level. This geographical environment keeps getting changed day by day and studying the rockfall or its trail gives more input about it like few recent studies showed that it has a herringbone pattern of rockfall. Similar to polar ice or dust avalanches, rockfall is also a mass wasting process. Not only in Mars, there are other places in solar system like Moon where rockfall has happened previously and now. Rockfall leaves a trace which actually helps to understand the shape, size, compound and bonding of the rockfall. Studying this would give more impactful information like the strength of the surface, knowing the properties of surface, the frequency and the timing of the rockfall. Future prediction gets lot better and easier by the study. Because of being huge size of the rockfall, it is very difficult to map them using just the satellite images and additional obstacle is the geographical location. Still there are places in Mars, which is unknown because even satellite could not reach those area and could not cover by fetching images. Even analysing crater gives the similar insights which helps to do the surface study and etc. There is a HiRISE camera which helps to capture the picture of Mars being on satellite. Only a HiRISE images can give the hi pixel images which can be processed further. In the traditional way, manual operator used to go through all the images and try to map the rockfall which clearly takes huge amount of time and it is a tiring process too. These images are having complicated objects in it and detecting rockfall is very difficult task to do, obviously manual process accuracy cannot be predicted. So, reliability can also be compromised by this manual technique. Even though researches are available, the count is very less in comparison of

others. So, to make proper prediction and conclusion on Mars, a greater number of researches should happen in coming days.

In this research CNN has been implemented as a model for image processing to detect the rockfall on Mars by passing HiRISE images. Then two pre trained model VGG16 and VGG19 were implemented to for the comparison. Image segmentation would help to go pixel by pixel and locate the object. Then creating a boundary line over that object from the whole image. The main focus of segmentation is to simplify the image, in this scenario one image file consists of multiple rockfall, so to simplify that small rockfall portion can be placed and bounded by rectangle. Then comes to classification, whenever data is having one or more than one class that data can be classified in different category. In this scenario it could be rockfall or non-rockfall. Classification happens in two phases- training and testing or validation. This approach helps to extract information from the image object. Steps goes like image segmentation which gives training data and testing data. By training data model training happens where model learns to classify the data into the different classes and then validation takes place by using the validation model. The research focus is to build a model and process the images to get a good prediction for future which might help to fetch a new angle in Mars study. Therefore, the research question is: to what extend rockfalls on mars could be detected through image classification using deep learning algorithm with satisfactory scores. By using AI, this research might give new direction to spacecraft anatomy.

2. Literature Review

2.1 Rockfall Detection

(Bickel et al., 2019, 2020)they have used CNN to detect rockfall. For their research, they have processed grey scale images and two steps of CNN were applied in their architecture. First happens is regional detection: it scans the greyscale images then try to find out the location of it. Once region has been found, the next step is starting the classification. The main challenges can be the detection of region because degradation can happen because of region detection. As a data pre-processing technique, image diversity can be part of it. Image transformation, scaling up the images, flipping the images vertically, horizontally, or brightness management helps to build up the data if there is requirement of large dataset.

In (Bickel et al., 2020)paper, has the suggestion of carrying out post processing techniques which gives more reliable predictions and to improve the overall performance of it. There are quite a few post processing techniques which are to be considered such as Adams technique by increasing the number of epochs. Among all the available optimizer such as SGD, RMSprop, ADAM, Adadelta, Adagrad, Adamax, Nadam, Ftrl, ADAM is used as it has less computational time with lesser input parameters for tuning.(Kingma & Ba, 2014), according to this paper the gradient decent method requires less memory and suited based wherever there is huge data and parameters. The required arguments can be learning rate, beta1, beta2, epsilon,

amsgrad, name, kwarg. It is an extension to stochastic and seen newly adaption in deep learning for computer vision. Adam is actually combination of two other extension of gradient like Adaptive Gradient Algorithm (AdaGrad) and Root Mean Square Propagation (RMSProp). AdaGrad helps to learning in per parameter which improves the performance and RMSProp is also per parameter learning and RMSProp is good when problem is online and non-stationary. Adam has both benefits of Adagrad and RMSProp. Adam also makes usage of second moment of uncentred variance, rather using of parameter learning on only first moment. The algorithm considers exponential moving average as well as squared gradient. Especially, Adam has become replacement algorithm for training any deep learning model and it is a comparatively easy to configure. The applied method and technique are described in details in the methodology area.

2.2 Pothole Detection

(An et al., 2018),these researchers have used mobile camera for fetching images and the camera was implemented in a vehicle to capture the images. Those images were used to detect the potholes. The CNN to be specific Deep CNN was implemented for this research. Feed forward artificial neural network used for both coloured images and greyscale images. In the comparison of different models, the CNN has given good results with better accuracy in this experiment. In this research, they found out couple of regions to be considered by using threshold algorithm. So, the region detection helped them to achieve better results and finally examination has been done for those chosen regions.

2.3 Crater Detection

On the other hand, to detect crater, there are automatic method who has given more success rate in this study and talking about those, CNN U-Net has given better output on their research. There are projects with random forest, SVM and CNN. But for any research with segmentation in it, CNN proved to be had better outcome from it. (Silburt et al., 2018), this crater detection has the similar steps of detecting the region of crater by using template algorithm then processing them in the model. The template algorithm works like it creates a ring around the object which needs to be cropped out from the original image.

2.4 Bio-medical Research

There are some medical studies like processing biomedical images, and CNN was beneficial for those researches as well. (Woo & Lee, 2021)this paper has done comparison of two images 2D and 3D of MR brain and processed them in CNN model. Basic CNN model was used in this research. The data processing time was less in this environment and even training the model

took lesser time. (Kora et al., 2021),this biomedical paper has done research on colonoscopy images and data augmentation was vast part of this research such as rotation, flipping, zooming etc which made the model for perfect by training the data which is scaled up. After that, CNN U-Net has taken part for modelling the research. Because of data augmentation, it has definitely started giving better accuracy and performance. Hence it was a good transfer study for this research as well. The techniques of data augmentation used for this paper has been explained in methodology part. (Harsh et al., 2021)(Bakir & Yalim Keles, 2021),these two dental research papers have given details on image segmentation and processing of it. These papers have clearly shown how CNN U-Net made complication image segmentation task easier than any other model and gave more accuracy and better fit. Not only that with the combination of CNN, Adams optimizer were better suited for these researches.

2.5 VGG16

(Huang et al., 2021), This research talks about the VGG16 with U-Net. They have built automatic VGG16 U-Net model with u-Net for Median Nerve segmentation. Their research takes place by following the steps of Data Augmentation (Data flipping horizontally, image rotation clockwise, zoomed in and mirror reflection), Implementation. NVIDIA TITAN X graphics unit was used and Adam optimizer. Experiment has been done in two models one basic U-Net another one VGG16 with U-Net and both models were trained with 20 epochs. VGG16 U-Net gave results 0.895±0.036, 0.812±0.057 and 0.899±0.062 where U-Net had 0.818±0.081, 0.699±0.112 and 0.728±0.118 for the same which gave leads to the VGG16 u-net. (Aung et al., 2021), this paper did face detection by comparing three models of YOLO with VGG16 and YOLOv2 with Alex network model and YOLOv2 with Google network model and results are 93%, 85% and 87% respectively. They represented hybrid method for face detection and their proposed model which is YOLO with VGG16 has given better output for detecting a face in real time and it has reduced false negative and true negative rates in their project. (Prasad et al., 2022) This research has done Covid-19 detection by applying vgg16 in CT scan images. They have shown good accuracy in their paper with images having covid-19 and without covid-19. To predict the covid cases they have implemented 2D conventional VGG model.

2.6 VGG19

(Junaidi et al., 2021), has a comparison experiment of two pretrained models of VGG16 and VGG19 based on image classification of incubator. They implemented VGG16 and VGG19 with trainable param of 75,267. VGG16 is at accuracy of .90 and VGG19 is at .92. Not only that researcher has done test validation with custom CNN model which has given accuracy of .87.

2.7 Others work

(Wang et al., 2022),paper gives the insight of power residual block in a U-Net model. Once residual block start getting built, skip connection gives more effective and efficient results in the model. The advantage is like when residual unit comes to the picture in a model, many layers might get skipped in that manner and eventually computational time and cost get reduced for that. That's the reason for introducing skip connection and the shape of U-net is popular to be implemented. For more details (*Research_on_U-Net_Improvement_by_Attention_and_Residual_Block*, n.d.)this paper has done amazing work for comparing all the various type of Unet by their performance, costs effectiveness. Through this comparison study also, it has shown that the residual block has importance to increase the performance of it in image processing and image segmentation. (Mustafa et al., 2020), has got 92% of pixel accuracy and IOU is 86% which 11% increase in comparison of PUN (plain U-net). In this study, they have combined residual blocks to get pixel wise accuracy for high resolution image data. This architecture is not only based on U-net they have implemented batch normalization and then ReLU and max pooling function.

From all the above review and discussion, the key notes to be taken is a) CNN needs to implemented for image classification to set a basic benchmark for an experiment, b) traditional CNN(Ronneberger et al., 2015) can be tweaked as per the requirement, c) There are few pre-trained models like VGG16 and VGG19 available to execute a test validation which can be done for the better comparison.

3. Methodology

As an environment for this research project, Google Colab has been chosen which is simply known as "Colab". It is a Cloud platform having free jupyter notebook. The purpose of using it are a) providing more computational power than local machine because it provides GPU and TPU, b) Data and programme syncing feature, c) need not to do any setup preparation. The methodology followed in this project is KDD.

Below is the illustration about the different steps of KDD:

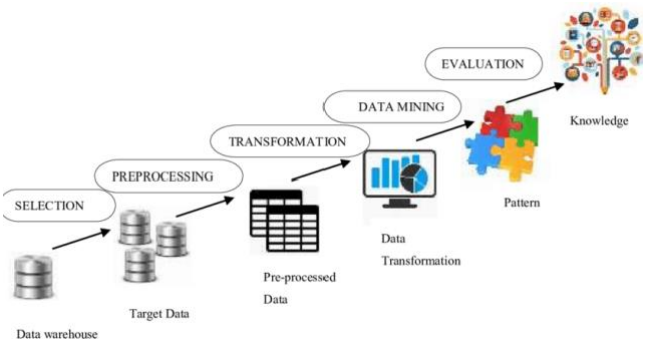

Fig1. KDD steps

Explanation and details of KDD model has been published here (Fayyad et al., 1996)

3.1 Data

HiRISE images are the best resolution images found. There is huge portion in Mars which is purely uncovered by the camera, that's why HiRISE could only capture pictures from limited areas.

Data has been fetched from an online source called Max Planck society https://edmond.mpdl.mpg.de/imeji/collection/DowTY91csU3jv9S2 . It consists of Mars directory with four more than sub directories like test_images, test_labels, train_images, train_labels. There are total of 1000 positives and 300 negative images to train the model and 91 positive images and 30 negative images to test the model. There is a lables.csv file on each label folder which talks about six columns includes name of the images, co-ordinates of detected rockfalls, and class name of it. The class.csv having two columns like name and id.

Below are the few ways to do data augmentations:

1) Sorting the data and creating a label file for the classification tags. Few in built python libraries can be used to import and fetch the data.

2) Next is if there is any requirement of data cleaning or data scaling. Few more libraries can be installed to scale up the data. Dataset should be divided into training data and test data.

3) Once the region is detected or images are cropped out from the original image file, the cropped images turned out to be different shapes and sizes, so image resizing is required to have a same image size file.

4) Once the dataset id finalized with all the count of data, denoising can be done if required to smoothen out the images. Gaussian blur gives a blur effect which helps to remove any noise from the images.

5) Next can be segmentation, which is actually finding out the region of rockfall in an image and then creating a bounding box around it. Then try to crop the images surrounded by the bounding box. These will provide all the positive rockfall images with small cut out.

3.2 Data Pre-processing

Firstly, for accessing the data for python notebook in Colab, data was imported into the driver. The hierarchy of the dataset remain the same. As mentioned previously data was downloaded from the authentic source to the local machine and then the .zip file was unzipped in local and then uploaded in google drive. To eliminate cumbersome, this process has been taken. But it took several amounts of time to upload an unzip time which made it unfeasible. Therefore, decision taken to upload the zip file which took down the importing time of the dataset.

To process the images from dataset for transformation or conversion, Numpy was required.

3.3 Data Transformation

Before starting with real data transformation, one change has been done one the label.csv file that is adding a row of headings of each column such as “Name”,” a”,” b”,”c”,” d”,”LabelName”. This particular change would help to fetch the data from csv file by using the column name.

Secondly, the csv file as containing data with label name as either “rockfall” or blank space. To classify the other images, it was named as “not-rockfall” through python code as shown below.

```
train_labels['LabelName'] = train_labels['LabelName'].fillna('not-rockfall')
train_labels.head()
print(train_labels)
```

```
            Name    a    b    c    d     LabelName
0      val_1.jp2  370  220  394  243      rockfall
1      val_1.jp2  311  391  342  423      rockfall
2     val_10.jp2  117  151  144  182      rockfall
3     val_11.jp2  236  112  270  178      rockfall
4     val_11.jp2  259  439  279  477      rockfall
...          ...  ...  ...  ...  ...           ...
1295   neg43.JP2    0    0    0    0  not-rockfall
1296   neg44.JP2    0    0    0    0  not-rockfall
1297   neg45.JP2    0    0    0    0  not-rockfall
```

Fig2. Classification as “rockfall” and “not-rockfall”

Both the above operations have been done for test and train label.csv files.

By understanding the data, one Mars positive image could consists of one or multiple co-ordinates which means it could have one or more rockfall in one single image. So, by reading co-ordinates from the csv file, small section of images was taken out from the real big image. Similar procedure has been done for negative images as well by taking random co-ordinates. All these cropped images are saved in a new folder and it increase the total count of our

dataset irrespective of both positive and negative. JP2 images have been converted into jpg for ease of execution.

The newly created images are all different shapes which are not ideal way of training the model. To rectify the same, average has been taken out of all the small cut outs and resized them to a square image. This gives all pictures same length and height. Once these are achieved for both training and testing data, one graph has been plotted to observe the data balance.

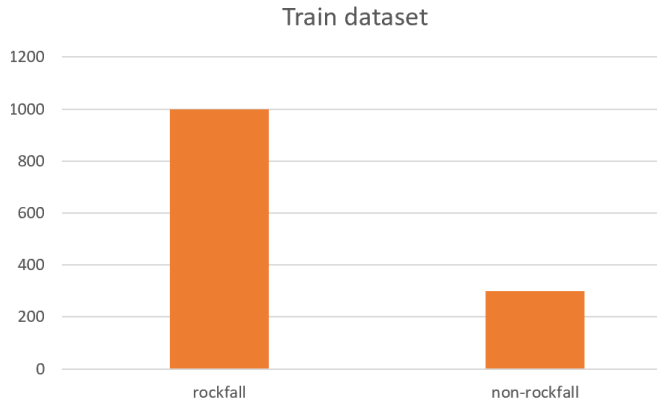

Fig3. Data imbalance graph

This gives the insight of the dataset proportion. In this case, dataset is imbalanced and the solution is to increasing the number of negative images and making it 1:1 ration.

To achieve that, albumentations was installed which helped in data augmentation. This process included vertical flip, Horizontal Flip and Random Brightness Contrast. Equal number of images has been created to balance the dataset and resized them in the similar manner. As new images were added in the dataset, label needs to be created for those extra images. New csv file got introduced by maintaining all above criteria such as all the columns, column names and “rockfall”/ “not-rockfall” classifications.

```
3   train_labels_transformedimages_neg['LabelName'] = train_labels_transformedimages
4   train_labels_transformedimages_neg.head()
5   print(train_labels_transformedimages_neg)
```

```
      Name  a  b  c  d     LabelName
1300.jpg  0  0  0  0  not-rockfall
1301.jpg  0  0  0  0  not-rockfall
1302.jpg  0  0  0  0  not-rockfall
1303.jpg  0  0  0  0  not-rockfall
1304.jpg  0  0  0  0  not-rockfall
     ... .. .. .. ..           ...
1995.jpg  0  0  0  0  not-rockfall
1996.jpg  0  0  0  0  not-rockfall
1997.jpg  0  0  0  0  not-rockfall
1998.jpg  0  0  0  0  not-rockfall
1999.jpg  0  0  0  0  not-rockfall

00 rows x 6 columns]
```

Fig4. Data augmentation

3.4 Data Mining techniques

To start with the experiment, first chosen model is CNN. So traditional CNN implemented and used. Once that is trained it creates a comparison number to match with other two models. Two other selected models are VGG19 and VGG16. These are pretrained CNN model with high number of parameters.

3.4.1 Convolutional neural networks (CNNs)

How human brain works by using neurons, Neural Network is a brain for machine learning algorithm. A group of multiple layers is a neural network. These layers are sequentially stacked one after another, so one-layer output becomes the input of next layer. So final layer of a neural network gives the output of whole algorithm. Here each pixel of an image becomes the input for neural networks layer. CNN works in a bit different way than the old traditional neural networking. If we focus on CNN layers, there are majorly three such as convolution layer, pooling layer and then merge layer. Convolution layer can be considered as the first layer of CNN. How it works is like convolution layer takes the first input and then passes through the filters. In one CNN model, multiple convolution layer can be there and each generated output from those layers are called as featured map. Next layer is pooling layer: all the down sampling happens in this layer. This layer goes through the image and try to reduce each dimension of an image, then it tries to create the required weight of the same image so that it can be acceptable by next layer. Next layer is Merge layer and the name says it merges all the created feature maps from the convolution layers and tries to join with CNN. In this model only convolution layer has the capacity of training, other two layers does not have any training parameters. CNN is very popular when it comes to image classification projects which mean whenever research require image pixel classification or image segmentation CNN is always a choice of model. Mainly because of U-Net has shown more success for pixel wise classification of an image. it has mainly used vastly in biomedical image processing, its merger all the layers and gives output to the next convolution layer.

CNN Architecture

In CNN U-Net architecture, it creates a shape like U because it has two arms one is used for encoding which captures the information from an image and the next arm acts as a decoder which helps to localize. In this model there is no separate convolution layer and pooling layer. The encoder is the combination of both convolution and poooling layer. This model does not have a dense layer too. Because of encoder and decoder arm it creates that shape of U.

3.4.2 VGG16

VGG16 is one pretrained model which used for object classification. The proposal of this model was given by K. Simonyan and A. Zisserman from the University of Oxford in year of 2014. To increase the depth of this model, the researcher implemented 3*3 convolutional filters. The layer which is having 2*2 filters and stride 2 is padded with max pooling.

3.4.3 VGG19

VGG19 is another pretrained model of CNN type which is having 19 deep layers. The main difference of VGG19 to VGG16 is having three connecting layers. The matrix shape of 224x224x3 image is the input. This also used 3*3filters with stride 1. Again, max pooling happened on 2*2 layers with stride 2.

4. Discussion

Object of having multiple experiment is to evaluate and compare what is best suited and why. All the models have their own confusion matrix which basically precision, recall and F1 score and by those parameters it is been judged and finalized. Because at first when we see the model training result or accuracy it can have good percentage which is always does not lead to a good accuracy when comes to real time validation. That's why after doing test data validation, it is important to draw a confusion matrix which has multiple parameters to evaluate a model whether it is a well-trained reliable model or not. As it is described in methodology, there are three models Traditional CNN, VGG16 and VGG19 has been implemented in this research and compared.

4.1 Experiment 1

First experiment takes place by using traditional CNN. Below is the CNN model summary shows all the layers, trainable parameters (388,290) and non-trainable parameters:

```
Model: "sequential_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_3 (Conv2D)            (None, 49, 49, 32)        896

max_pooling2d_2 (MaxPooling  (None, 24, 24, 32)        0
2D)

dropout_2 (Dropout)          (None, 24, 24, 32)        0

conv2d_4 (Conv2D)            (None, 22, 22, 64)        18496

max_pooling2d_3 (MaxPooling  (None, 11, 11, 64)        0
2D)

dropout_3 (Dropout)          (None, 11, 11, 64)        0

conv2d_5 (Conv2D)            (None, 9, 9, 64)          36928

flatten_1 (Flatten)          (None, 5184)              0

dense_2 (Dense)              (None, 64)                331840

dense_3 (Dense)              (None, 2)                 130

=================================================================
Total params: 388,290
Trainable params: 388,290
Non-trainable params: 0
_________________________________________________________________
```

Fig5. CNN model

The number of epochs chosen for this model was 30 with the bath size of 60. As we can see the accuracy has 50% after training the data.

```
Epoch 26/30
27/27 [==============================] - 0s 8ms/step - loss: 0.6932 - accuracy: 0.5006 - val_loss: 0.6931 - val_accuracy: 0.4975
Epoch 27/30
27/27 [==============================] - 0s 8ms/step - loss: 0.6932 - accuracy: 0.5031 - val_loss: 0.6931 - val_accuracy: 0.5025
Epoch 28/30
27/27 [==============================] - 0s 7ms/step - loss: 0.6932 - accuracy: 0.4994 - val_loss: 0.6931 - val_accuracy: 0.5025
Epoch 29/30
27/27 [==============================] - 0s 8ms/step - loss: 0.6932 - accuracy: 0.4994 - val_loss: 0.6931 - val_accuracy: 0.4975
Epoch 30/30
27/27 [==============================] - 0s 8ms/step - loss: 0.6932 - accuracy: 0.4906 - val_loss: 0.6931 - val_accuracy: 0.5025
```

Fig6. Training epochs of CNN

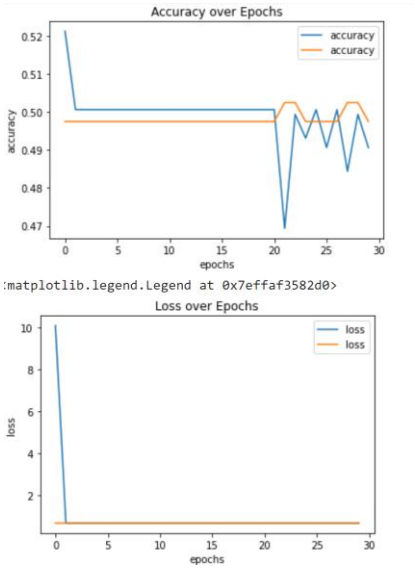

Fig7. accuracy and loss after end of training

Earlier data was split in two dataset train and test and train was split into train and validation. After training the model with train and validation data, verification done by test data and gave accuracy of 68%.

Calculating only accuracy is not enough for a model. Sometime with high accuracy can give fail prediction in real time. Moreover, in real time data will not be balanced all the time so only accuracy calculation is not enough and that's why evaluation matrix is required which talks about Recall, Precision, F1 score. For this evaluation, "classification_report" needs to be imported from sklearn.metrics. Now if we try to get the precision and recall of the baseline CNN, below is the outcome of it.

```
[43] from sklearn.metrics import classification_report

     print(classification_report(test_labels, y_pred))
```

	precision	recall	f1-score	support
0	0.10	0.03	0.05	30
1	0.74	0.90	0.81	91
accuracy			0.69	121
macro avg	0.42	0.47	0.43	121
weighted avg	0.58	0.69	0.62	121

Fig8. Confusion matrix

All the above evaluation is based on Baseline model of CNN. There are few pre trained model available which are profitable for image classification. This dataset has evaluated by processing through those models as well.

4.2 Experiment 2

Experiment 2 happens by using VGG16 model. By doing the transfer learning, VGG16 and VGG19 has implemented. Below is the model summary for VGG16 with the training parameter as 527,362:

```
block3_conv3 (Conv2D)       (None, 12, 12, 256)       590080

block3_pool (MaxPooling2D)  (None, 6, 6, 256)         0

block4_conv1 (Conv2D)       (None, 6, 6, 512)         1180160

block4_conv2 (Conv2D)       (None, 6, 6, 512)         2359808

block4_conv3 (Conv2D)       (None, 6, 6, 512)         2359808

block4_pool (MaxPooling2D)  (None, 3, 3, 512)         0

block5_conv1 (Conv2D)       (None, 3, 3, 512)         2359808

block5_conv2 (Conv2D)       (None, 3, 3, 512)         2359808

block5_conv3 (Conv2D)       (None, 3, 3, 512)         2359808

block5_pool (MaxPooling2D)  (None, 1, 1, 512)         0

sequential_2 (Sequential)   (None, 2)                 527362

=================================================================
Total params: 15,242,050
Trainable params: 527,362
Non-trainable params: 14,714,688
_________________________________________________________________
```

Fig9. VGG16 model summary

The configured epochs for VGG16 are 30 with the batch size of 60.

```
Epoch 25/30
27/27 [==============================] - 1s 27ms/step - loss: 0.1196 - accuracy: 0.9550 - val_loss: 0.7201 - val_accuracy: 0.8100
Epoch 26/30
27/27 [==============================] - 1s 26ms/step - loss: 0.1405 - accuracy: 0.9544 - val_loss: 0.8628 - val_accuracy: 0.7750
Epoch 27/30
27/27 [==============================] - 1s 26ms/step - loss: 0.1427 - accuracy: 0.9488 - val_loss: 0.8616 - val_accuracy: 0.8050
Epoch 28/30
27/27 [==============================] - 1s 26ms/step - loss: 0.1383 - accuracy: 0.9519 - val_loss: 0.8481 - val_accuracy: 0.8125
Epoch 29/30
27/27 [==============================] - 1s 26ms/step - loss: 0.1172 - accuracy: 0.9538 - val_loss: 0.7826 - val_accuracy: 0.8050
Epoch 30/30
27/27 [==============================] - 1s 26ms/step - loss: 0.1397 - accuracy: 0.9556 - val_loss: 0.7681 - val_accuracy: 0.7900
```

Fig10. Training epochs for VGG16

Once we plot the same

```
pd.DataFrame(history.history).plot(figsize=(8, 5))
plt.grid(True)
plt.gca().set_ylim(0, 1) # setting limits for y-axis
plt.show()
```

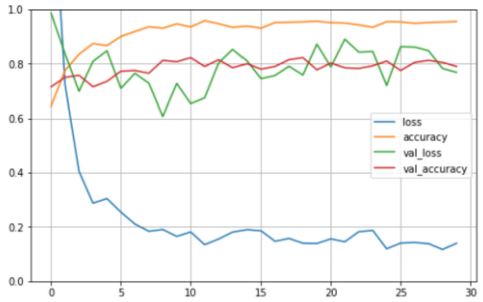

Fig11. VGG16 graph

```
[48] vgg16_test_pred=np.argmax(model.predict(X_test), axis=-1)
     print(classification_report(test_labels, vgg16_test_pred))
```

	precision	recall	f1-score	support
0	0.25	0.10	0.14	30
1	0.75	0.90	0.82	91
accuracy			0.70	121
macro avg	0.50	0.50	0.48	121
weighted avg	0.63	0.70	0.65	121

Fig12. Confusion Matrix of VGG16

4.3 Experiment 3

Similarly, experiment has been carried out for VGG19 and below are the results for the same. Model has been built for 20,551,746 as trainable parameter.

```
block3_conv4 (Conv2D)       (None, 12, 12, 256)       590080

block3_pool (MaxPooling2D)  (None, 6, 6, 256)         0

block4_conv1 (Conv2D)       (None, 6, 6, 512)         1180160

block4_conv2 (Conv2D)       (None, 6, 6, 512)         2359808

block4_conv3 (Conv2D)       (None, 6, 6, 512)         2359808

block4_conv4 (Conv2D)       (None, 6, 6, 512)         2359808

block4_pool (MaxPooling2D)  (None, 3, 3, 512)         0

block5_conv1 (Conv2D)       (None, 3, 3, 512)         2359808

block5_conv2 (Conv2D)       (None, 3, 3, 512)         2359808

block5_conv3 (Conv2D)       (None, 3, 3, 512)         2359808

block5_conv4 (Conv2D)       (None, 3, 3, 512)         2359808

block5_pool (MaxPooling2D)  (None, 1, 1, 512)         0

sequential_3 (Sequential)   (None, 2)                 527362

=================================================================
Total params: 20,551,746
Trainable params: 20,551,746
Non-trainable params: 0
_________________________________________________________________
```

Fig13. VGG19 model summary

Number of epochs is 30 with the batch size of 60.

```
Epoch 25/30
27/27 [==============================] - 2s 79ms/step - loss: 0.7000 - accuracy: 0.5063 - val_loss: 0.6963 - val_accuracy: 0.4975
Epoch 26/30
27/27 [==============================] - 2s 79ms/step - loss: 0.6987 - accuracy: 0.4956 - val_loss: 0.6933 - val_accuracy: 0.4975
Epoch 27/30
27/27 [==============================] - 2s 78ms/step - loss: 0.6936 - accuracy: 0.5019 - val_loss: 0.6933 - val_accuracy: 0.4975
Epoch 28/30
27/27 [==============================] - 2s 78ms/step - loss: 0.6935 - accuracy: 0.4881 - val_loss: 0.6931 - val_accuracy: 0.5025
Epoch 29/30
27/27 [==============================] - 2s 78ms/step - loss: 0.6934 - accuracy: 0.4938 - val_loss: 0.6932 - val_accuracy: 0.4975
Epoch 30/30
27/27 [==============================] - 2s 78ms/step - loss: 0.6932 - accuracy: 0.4800 - val_loss: 0.6931 - val_accuracy: 0.4975
```

Fig14. Training epochs of VGG19

Once we plot the graph

```
pd.DataFrame(history.history).plot(figsize=(8, 5))
plt.grid(True)
plt.gca().set_ylim(0, 1) # setting limits for y-axis
plt.show()
```

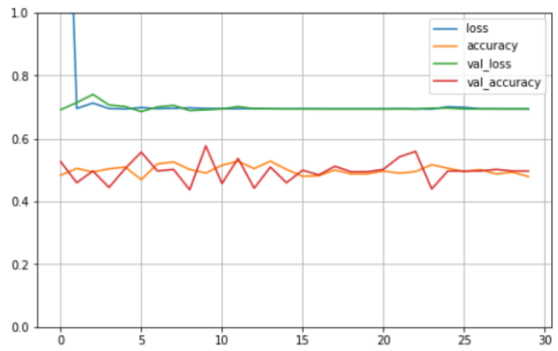

Fig15. VGG19 graph

```
print(classification_report(test_labels, vgg19_test_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      0.03      0.06        30
           1       0.76      1.00      0.86        91

    accuracy                           0.76       121
   macro avg       0.88      0.52      0.46       121
weighted avg       0.82      0.76      0.66       121
```

Fig16. Confusion Matrix VGG19

Deep Learning models for image classification	**Accuracy**	**Precision**	**Recall**	**F1-Score**
Classic CNN	.68	.74	.90	.81
VGG 16	.94	.75	.90	.82
VGG 19	.48	.76	1	.86

Table1: Comparison of all the models.

5. Conclusion & Future Work:

By running three models parallelly and comparing them afterword's, it is been seen that the VGG16 has given better result by having above 90% of accuracy compared to VGG19 and traditional CNN model.

There are lots of scope to improve and potential steps to carry out such as

a) Would like to implement and apply few more other models which are latest or custom model like Res-U-Net where it has showed that combining U-net with Residual blocks can give pixelwise accuracy for image classification. This can achieve more success on image classification projects.
b) Checking few more layers of CNN architecture which can be beneficial for the project. Adding more layers can have two profits: more pooling layers with same amount of information will be applicable more to the ground truth, then starting layers trained easily by lines where as deeper layer can learn more complex features. By finding the perfect trade-off with trial and error, more filters can help to achieve more accuracy.
c) Adapting different techniques of pre-processing the images. In the above research train time data augmentation has been performed, But not in test. In Test data augmentation can be done by being in a limit.

d) Another very important action would be training the model with a greater number of positive images in the dataset and negative images. In this case data was manipulated by flipping vertically, Horizontally and with image brightness. Few other techniques can be applied for the synthetic data preparation.

6. References:

An, K. E., Lee, S. W., Ryu, S. K., & Seo, D. (2018). Detecting a pothole using deep convolutional neural network models for an adaptive shock observing in a vehicle driving. *2018 IEEE International Conference on Consumer Electronics, ICCE 2018*, *2018-January*, 1–2. https://doi.org/10.1109/ICCE.2018.8326142

Aung, H., Bobkov, A. v., & Tun, N. L. (2021). Face detection in real time live video using yolo algorithm based on VGG16 convolutional neural network. *Proceedings - 2021 International Conference on Industrial Engineering, Applications and Manufacturing, ICIEAM 2021*, 697–702. https://doi.org/10.1109/ICIEAM51226.2021.9446291

Bakir, M. E., & Yalim Keles, H. (2021, June 9). Deep learning based cell segmentation using cascaded U-net models. *SIU 2021 - 29th IEEE Conference on Signal Processing and Communications Applications, Proceedings*. https://doi.org/10.1109/SIU53274.2021.9477937

Bickel, V. T., Conway, S. J., Tesson, P. A., Manconi, A., Loew, S., & Mall, U. (2020). Deep Learning-Driven Detection and Mapping of Rockfalls on Mars. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *13*, 2831–2841. https://doi.org/10.1109/JSTARS.2020.2991588

Bickel, V. T., Lanaras, C., Manconi, A., Loew, S., & Mall, U. (2019). Automated Detection of Lunar Rockfalls Using a Convolutional Neural Network. *IEEE Transactions on Geoscience and Remote Sensing*, *57*(6), 3501–3511. https://doi.org/10.1109/TGRS.2018.2885280

Fayyad, U., Piatetsky-Shapiro, G., & Smyth, P. (1996). *Knowledge Discovery and Data Mining: Towards a Unifying Framework*. www.aaai.org

Harsh, P., Chakraborty, R., Tripathi, S., & Sharma, K. (2021, June 25). Attention U-Net Architecture for Dental Image Segmentation. *2021 International Conference on Intelligent Technologies, CONIT 2021*. https://doi.org/10.1109/CONIT51480.2021.9498422

Huang, A., Wang, Q., Jiang, L., & Zhang, J. (2021). Automatic Segmentation of Median Nerve in Ultrasound Image by a Combined Use of U-Net and VGG16. *IEEE International Ultrasonics Symposium, IUS*. https://doi.org/10.1109/IUS52206.2021.9593861

Junaidi, A., Lasama, J., Adhinata, F. D., & Iskandar, A. R. (2021). Image Classification for Egg Incubator using Transfer Learning of VGG16 and VGG19. *10th IEEE International Conference on Communication, Networks and Satellite, Comnetsat 2021 - Proceedings*, 324–328. https://doi.org/10.1109/COMNETSAT53002.2021.9530826

Kingma, D. P., & Ba, J. (2014). *Adam: A Method for Stochastic Optimization*. http://arxiv.org/abs/1412.6980

Kora, P., Haneesha, B., Sahith, D., Grace, S. P., Benny Jasper, K., Swaraja, K., & Meenakshi, K. (2021). Automatic Segmentation of Polyps using U-Net from Colonoscopy images. *Proceedings of the 3rd International Conference on Inventive Research in Computing Applications, ICIRCA 2021*, 855–859. https://doi.org/10.1109/ICIRCA51532.2021.9544573

Mustafa, N., Zhao, J., Liu, Z., Zhang, Z., & Yu, W. (2020). Iron ORE Region Segmentation Using High-Resolution Remote Sensing Images Based on Res-U-Net. *International Geoscience and Remote Sensing Symposium (IGARSS)*, 2563–2566. https://doi.org/10.1109/IGARSS39084.2020.9324218

Prasad, K. S., Pasupathy, S., Chinnasamy, P., & Kalaiarasi, A. (2022). An Approach to Detect COVID-19 Disease from CT Scan Images using CNN - VGG16 Model. *2022 International Conference on Computer Communication and Informatics, ICCCI 2022*. https://doi.org/10.1109/ICCCI54379.2022.9741050

H. Deng, "Research on U-Net Improvement by Attention and Residual Block," ICMLCA 2021; 2nd International Conference on Machine Learning and Computer Application, 2021, pp. 1-5.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *9351*, 234–241. https://doi.org/10.1007/978-3-319-24574-4_28

Silburt, A., Ali-Dib, M., Zhu, C., Jackson, A., Valencia, D., Kissin, Y., Tamayo, D., & Menou, K. (2018). *Lunar crater identification via deep learning*. https://doi.org/10.5281/zenodo.1133969

Wang, J., Chen, F., Zhang, M., & Yu, B. (2022). NAU-Net: A New Deep Learning Framework in Glacial Lake Detection. *IEEE Geoscience and Remote Sensing Letters*, 1–1. https://doi.org/10.1109/LGRS.2022.3165045

Woo, B., & Lee, M. (2021, January 31). Comparison of tissue segmentation performance between 2D U-Net and 3D U-Net on brain MR Images. *2021 International Conference on Electronics, Information, and Communication, ICEIC 2021*. https://doi.org/10.1109/ICEIC51217.2021.9369797