Terrain Detection to estimate ground area changes via a U-Net for Semantic Segmentation

Abstract

Applications of deep-learning to process and use remote-sensing data i.e. Satellite Images for Geoclimatic studies are prominent. Neural Networks have proved highly efficient for classification, annotation and even image generation. This is highly applicable in every field. In this paper, we implement a U-Net to perform semantic segmentation to detect and localize landslides. This is the first step towards building a pipeline where landslides can be identified, their impact zone and area quantified to study changes in terrain and even for rescue efforts in case of emergencies. We trained our model on the landslides dataset from the Landslide4sense 2022 challenge and tested our results on the test set as well as on a waterbodies dataset of satellite images from Sentinel-2. We found that our U-Net which performs per pixel classification and localization predicts landslides with a 98% accuracy score. It falters when the images have high vegetation and fails to produce meaningful masks for the waterbodies dataset. However, our model validates the proof-of-idea of landslide detection as a first step in our chosen case study of the Pakistan Floods 2022.

Introduction

Geoclimatic research industry rely heavily on remote-sensing data from satellites to study every layer of the earths surface, from the terrain to sub-surface properties but specially for extreme events. This is highly applicable and important in regions where the possibility to sense and collect on-ground and surface-level data is not possible or requires investments in terms of human resources or funding.

> The most prominent climate related extreme event were the Pakistan Floods of 2022 that caused about one-third of the country to be inundated. The Gilgit-Baltistan region of Pakistan which is home to the Hindu-Kush Mountain range experienced multiple landslides which resulted in flash-floods and glacial lake outbursts affecting more than 33 million people.1These landslides cut-off certain areas from receiving rescue and reilef efforts.2

> Detection of these landslides will be highly beneficial to direct rescue and relief programs as well as for post event damage analysis. The broad idea is to build a network that can effectively detect changes in the terrain to answer questions like: *Which part of the range experienced landslides?* , *What's the area of damage?*, *Did the circumference of contour of a lake change over a short-period of time?*, *Did the width of riverbank change over a short-period of time?*

> The motivation of this research was to perform landslide detection as the first step towards quantifying the extent of these landslides. To answer the question *Which part of the range experienced landslides?*, we asked a counterfactual questions, *Did this particular block of land experience a landslide?* The blocks of land here correspond to one satellite image which covers the area.

Related Work

> Deep learning has been extensively used to process these satellite images to get meaningful classifications, detections and analysis of the topography of the terrain.

> Terrain Detection is a blanket statement for tasks related to Land Use Classification, land-cover detection, Road mapping and segmentation, etc. One research performs Land-cover classification to classify the type of vegetation in a particular area using a Convolutional Neural Network to classify vegetation into seven classes.3 Similarly, another study assessed the change in Forest cover in the SWAT valley, Pakistan.4 Within the deep-learning frameworks, Fully convolutional neural networks are the prominent network of choice to perform classifications. Chen et al. (2017) proposed a new deep neural network-based feature fusion framework that employs deep CNNs to efficiently extract features from lidar data and their proposed fusion framework can effectively improve the classification performance of the resulting model

> To effectively detect landslides our project focused on semantic image segmentation. It is a form of dense prediction that is pixel-wise classification of distortion in slope of the landscape. For this purpose, I decided to build a U-Net. An alternative was to perform combination of object detection, object localization and creation of mask using Fully Convolutional Neural Network. However, U-net uses a loss function for each pixel of the image. This helps in easy identification of individual cells within the segmentation map.

> A previous study has found U-Nets to have more precision for semantic segmentation when compared to CNNs. 5 Therefore, we've decided to build a U-Net model.

Methodology

Data

We decided to focus on the Gilgit-Baltistan area in this study. However, data collection was more expensive in terms of time for the scope of this project. Hence we decided to train the model on landslide images provided by the Institude of Advanced Research in Artifical Intelligence for their Landslide4Sense Challenge 2022. 6 This dataset consists of 3799 training image patches. Each image patch is a composite of 14 bands that include:

Multispectral data from Sentinel-2: B1, B2, B3, B4, B5, B6, B7, B8, B9, B10, B11, B12.

Slope data from ALOS PALSAR: B13.

Digital elevation model (DEM) from ALOS PALSAR: B14.

All bands are resized to the resolution of ~10m per pixel. The image patches have the size of 128 x 128 pixels and are labeled pixel-wise.

The training dataset was split into train and validation with a 80:20 split.

The validation dataset consists of 245 images.

The test dataset was a mix of the test dataset from Landslide4sense as well as a waterbodies dataset of Satellite images also from Sentinel-2 to see if the two indexes hold true for lakes or GLOF lake detection in the future for this extreme event. This waterbodies dataset was requested from Institute of Mountain Hazards and Environment, CAS. They have generated this dataset on 89 Glacial Lakes in the China-Pakistan Economic Border which lies at the Hindu-Kush mountains.7

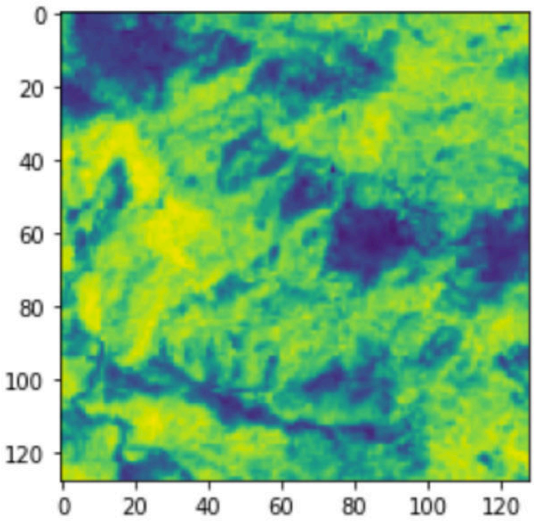
Feature Calculation

For each image patch, the NDVI i.e. Normalized Difference Vegetation Index and the NDWI i.e., Normalized Difference Water Index was computed from the Near Infrared Wavelength (NIR) and the Red and Mid-Infrared Wavelength (MIR) bands respectively.

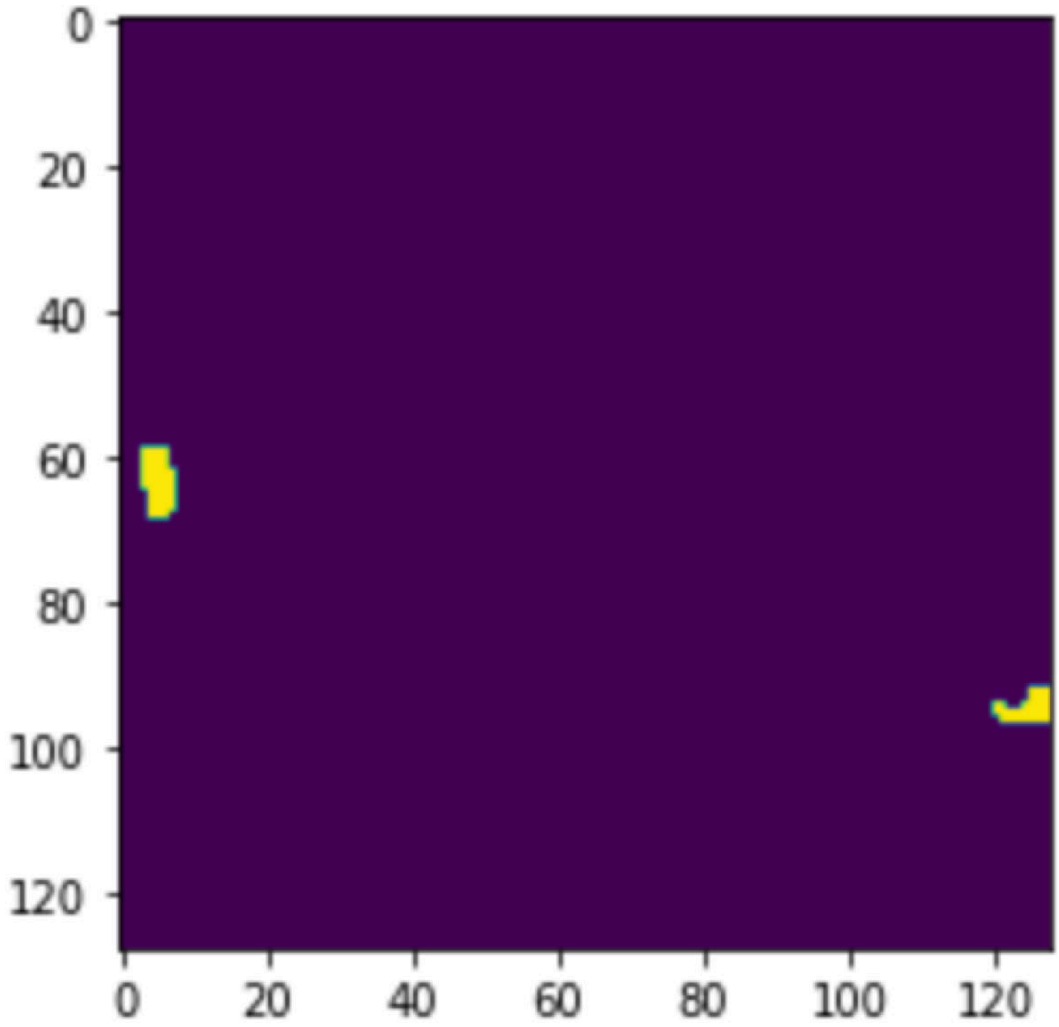
```
NDVI = (NIR – RED)/(NIR + RED)
NDWI = (NIR – MIR)/(NIR + MIR)
```

Slope, Elevation and RGB values were normalized before being fed into the network, along with the red, green and blue bands.

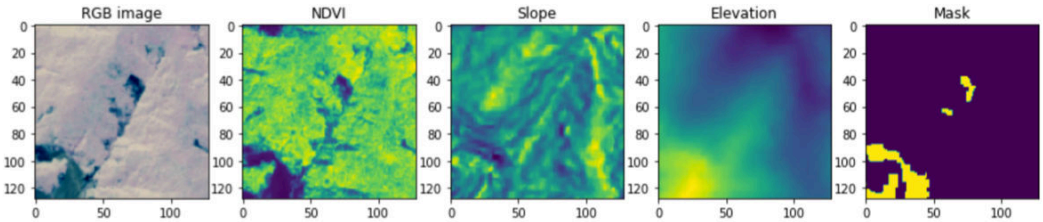
```
Shape of input image:  (128, 128, 14)
Image Vegetation shape:  (128, 128) Image features shape: (1, 128, 128, 3)
```



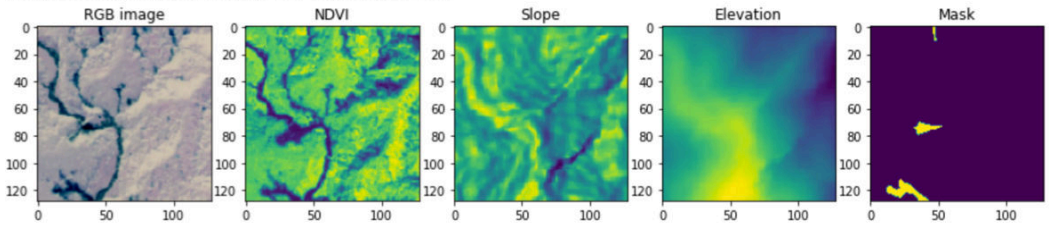
```
Key of image ['mask']
input mask shape: (128, 128)
```




```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
<matplotlib.image.AxesImage at 0x7f12b30f0460>
```



```
WARNING:matplotlib.image:Clipping input data to the valid range for imshow with RGB data ([0..1] for floats or [0..255] for integers).
<matplotlib.image.AxesImage at 0x7f12b5676670>
```



Model

Our U-Net model consists of five layers in the 'Contraction Path' which performs the Convolutions. Each layer has two Convolutional layers followed by a Dropout layer and 2x2 Maxpooling.

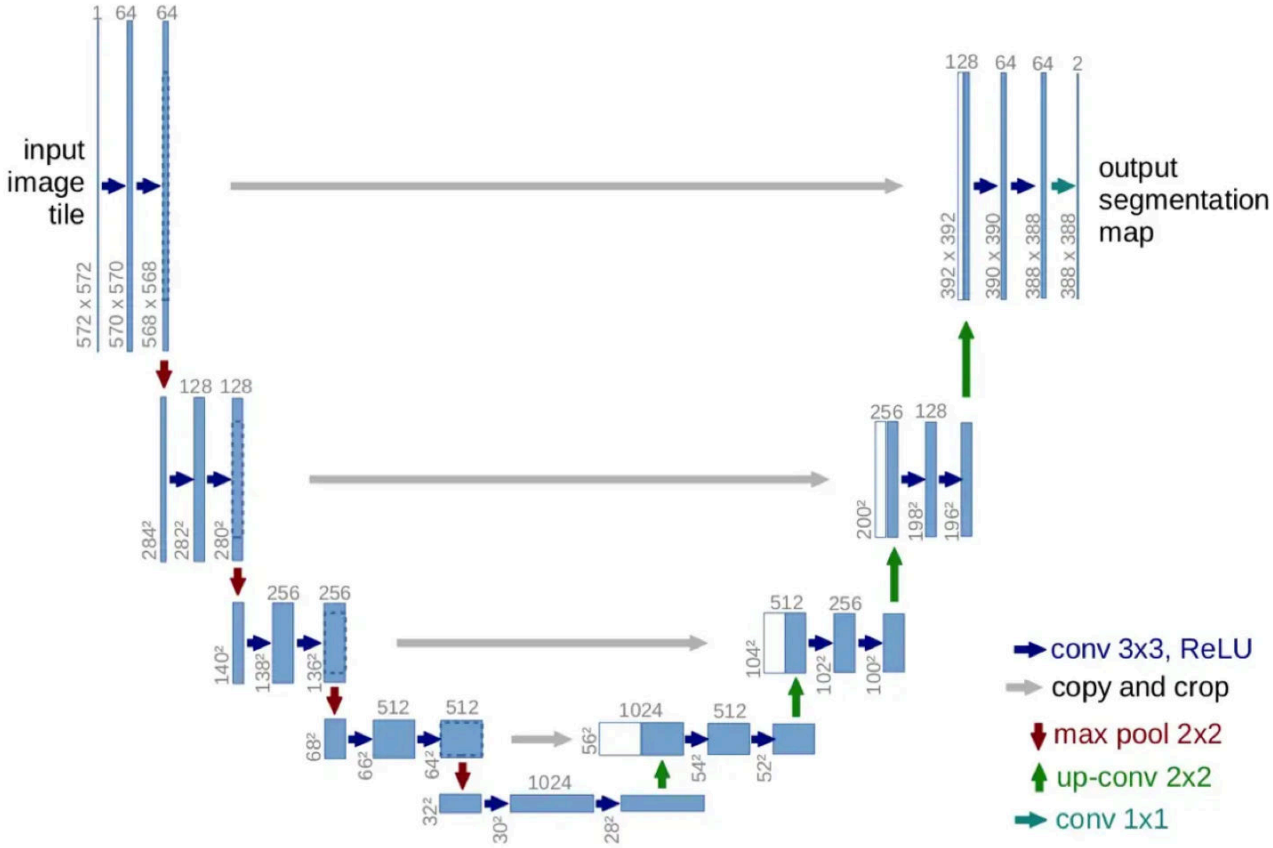
The output of the fifth layers convolutions are then fed into the 'Expansive Path' that performs upsampling or deconvolution to get the image back to its original size. Our model has four layers in this path. In each layer after the deconvolution step using Coonv2DTranspose, the image is concatenated with the corresponding image from the contracting path. This is done to combine the information from the previous layers in order to get a more precise prediction.

After the 'Expansive Path', the image is passed through a sigmoid for reshaping.

The Adam optimizer is used to update weights in the training phase and Binary Crossentropy loss function is used.

Validation f1 score is used as a metric to save the best model. Other performance metrics are custom metrics of recall, precision and f1 score.

The network has basic foundation looks like:



UNet architecture

```
Model: "model"
_________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to
=================================================================================================
 input_2 (InputLayer)           [(None, 128, 128, 6  0           []
                                )]

 conv2d_11 (Conv2D)             (None, 128, 128, 16  880         ['input_2[0][0]']
                                )

 dropout_6 (Dropout)            (None, 128, 128, 16  0           ['conv2d_11[0][0]']
                                )

 conv2d_12 (Conv2D)             (None, 128, 128, 16  2320        ['dropout_6[0][0]']
                                )

 max_pooling2d_4 (MaxPooling2D)  (None, 64, 64, 16)  0           ['conv2d_12[0][0]']

 conv2d_13 (Conv2D)             (None, 64, 64, 32)   4640        ['max_pooling2d_4[0][0]']

 dropout_7 (Dropout)            (None, 64, 64, 32)   0           ['conv2d_13[0][0]']

 conv2d_14 (Conv2D)             (None, 64, 64, 32)   9248        ['dropout_7[0][0]']

 max_pooling2d_5 (MaxPooling2D)  (None, 32, 32, 32)  0           ['conv2d_14[0][0]']

 conv2d_15 (Conv2D)             (None, 32, 32, 64)   18496       ['max_pooling2d_5[0][0]']

 dropout_8 (Dropout)            (None, 32, 32, 64)   0           ['conv2d_15[0][0]']

 conv2d_16 (Conv2D)             (None, 32, 32, 64)   36928       ['dropout_8[0][0]']

 max_pooling2d_6 (MaxPooling2D)  (None, 16, 16, 64)  0           ['conv2d_16[0][0]']

 conv2d_17 (Conv2D)             (None, 16, 16, 128)  73856       ['max_pooling2d_6[0][0]']

 dropout_9 (Dropout)            (None, 16, 16, 128)  0           ['conv2d_17[0][0]']

 conv2d_18 (Conv2D)             (None, 16, 16, 128)  147584      ['dropout_9[0][0]']

 max_pooling2d_7 (MaxPooling2D)  (None, 8, 8, 128)   0           ['conv2d_18[0][0]']

 conv2d_19 (Conv2D)             (None, 8, 8, 256)    295168      ['max_pooling2d_7[0][0]']

 dropout_10 (Dropout)           (None, 8, 8, 256)    0           ['conv2d_19[0][0]']
```

```
dropout_10 (Dropout)           (None, 8, 8, 256)    0           ['conv2d_19[0][0]']

conv2d_20 (Conv2D)             (None, 8, 8, 256)    590080      ['dropout_10[0][0]']

conv2d_transpose_1 (Conv2DTran  (None, 16, 16, 128)  131200     ['conv2d_20[0][0]']
spose)

concatenate_1 (Concatenate)    (None, 16, 16, 256)  0           ['conv2d_transpose_1[0][0]',
                                                                 'conv2d_18[0][0]']

conv2d_21 (Conv2D)             (None, 16, 16, 128)  295040      ['concatenate_1[0][0]']

dropout_11 (Dropout)           (None, 16, 16, 128)  0           ['conv2d_21[0][0]']

conv2d_22 (Conv2D)             (None, 16, 16, 128)  147584      ['dropout_11[0][0]']

conv2d_transpose_2 (Conv2DTran  (None, 32, 32, 64)  32832      ['conv2d_22[0][0]']
spose)

concatenate_2 (Concatenate)    (None, 32, 32, 128)  0           ['conv2d_transpose_2[0][0]',
                                                                 'conv2d_16[0][0]']

conv2d_23 (Conv2D)             (None, 32, 32, 64)   73792       ['concatenate_2[0][0]']

dropout_12 (Dropout)           (None, 32, 32, 64)   0           ['conv2d_23[0][0]']

conv2d_24 (Conv2D)             (None, 32, 32, 64)   36928       ['dropout_12[0][0]']

conv2d_transpose_3 (Conv2DTran  (None, 64, 64, 32)  8224        ['conv2d_24[0][0]']
spose)

concatenate_3 (Concatenate)    (None, 64, 64, 64)   0           ['conv2d_transpose_3[0][0]',
                                                                 'conv2d_14[0][0]']

conv2d_25 (Conv2D)             (None, 64, 64, 32)   18464       ['concatenate_3[0][0]']

dropout_13 (Dropout)           (None, 64, 64, 32)   0           ['conv2d_25[0][0]']

conv2d_26 (Conv2D)             (None, 64, 64, 32)   9248        ['dropout_13[0][0]']

conv2d_transpose_4 (Conv2DTran  (None, 128, 128, 16  2064       ['conv2d_26[0][0]']
spose)                         )

concatenate_4 (Concatenate)    (None, 128, 128, 32  0           ['conv2d_transpose_4[0][0]',
                               )                                 'conv2d_12[0][0]']
```

```
conv2d_27 (Conv2D)          (None, 128, 128, 16  4624        ['concatenate_4[0][0]']
                            )

dropout_14 (Dropout)        (None, 128, 128, 16  0           ['conv2d_27[0][0]']
                            )

conv2d_28 (Conv2D)          (None, 128, 128, 16  2320        ['dropout_14[0][0]']
                            )

conv2d_29 (Conv2D)          (None, 128, 128, 1)  17          ['conv2d_28[0][0]']

==================================================================================================
Total params: 1,941,537
Trainable params: 1,941,537
Non-trainable params: 0
__________________________________________________________________________________________________
```

After an image has been predicted, a mask is generated.

To calculate the estimated area of impact we decided to calculate the area of the contour in the map. OpenCV library was used to calculate this. The result is given in terms of number of connected pixels in the mask. Ground sampling distance in kilometers or meters would be the ideal metric for area of impact. However, for each image, the exact ground surface distance is different and depends on the sensors properties in terms of altitude, distance and angle.

```
GSD =  (flight altitude x sensor height) / (focal length x image height and/or wid
th)
```

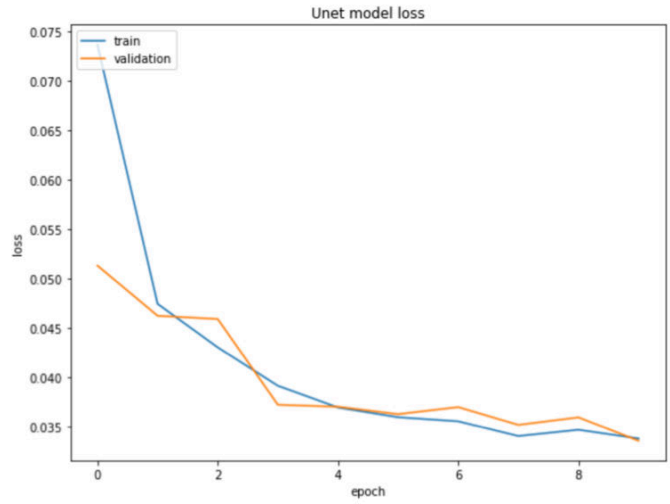
In the future scope, with our curated dataset, GSD value needs to be computed to convert the number of pixels into the exact ground sampling distance.

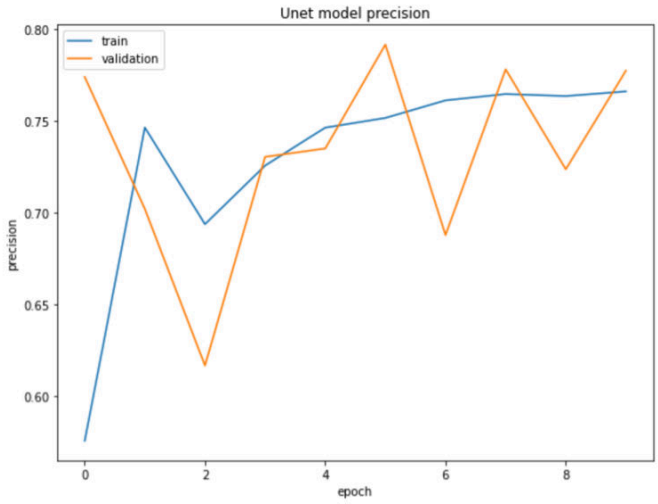
Results

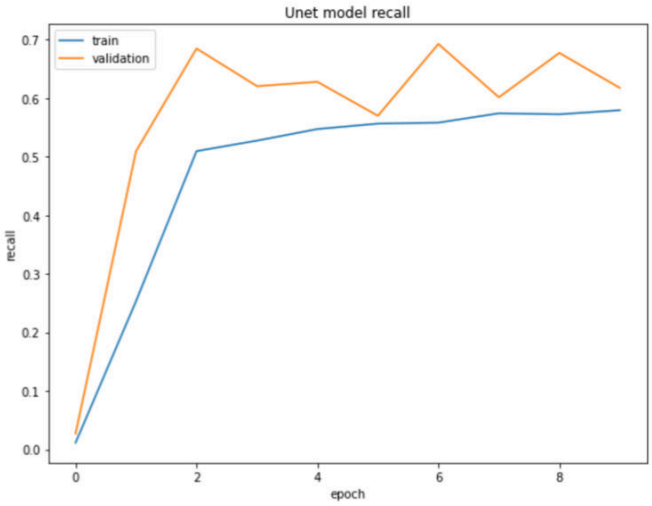
We initially run our model on 50 epochs with early stopping. We found that stopping around 10 epochs had the best model with an

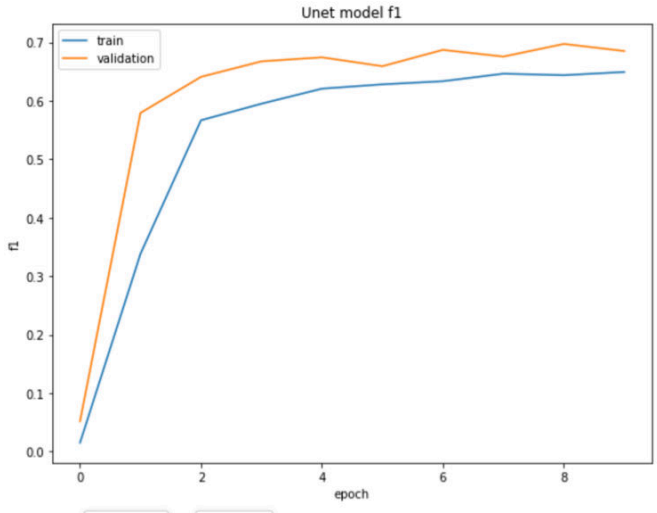
- Accuracy: 0.9866776466369629
- F1 score: 0.6928379535675049
- Precision: 0.7853184342384338
- Recall: 0.6222373843193054

```
<matplotlib.legend.Legend at 0x7f2fdd766a30>
```

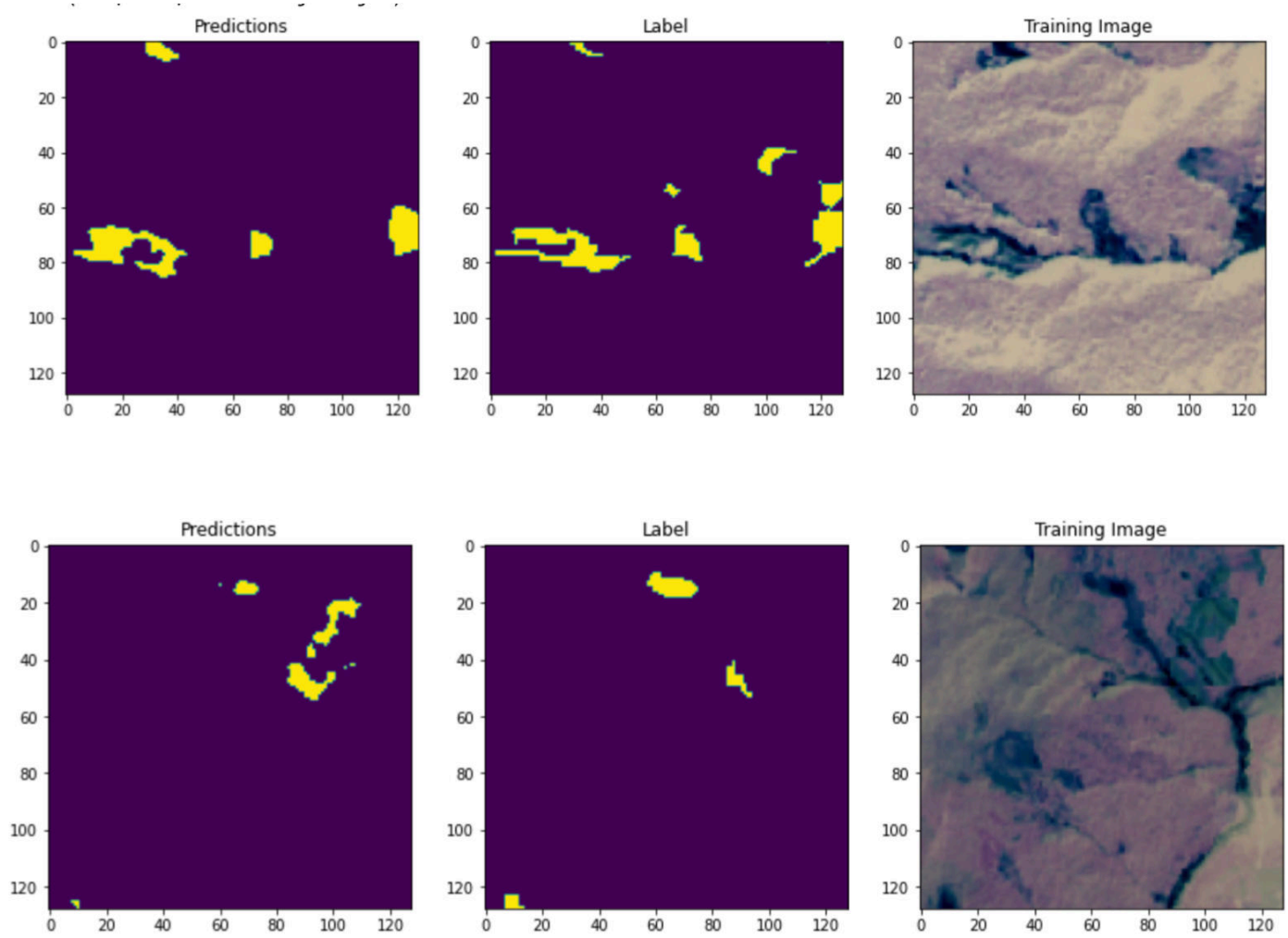






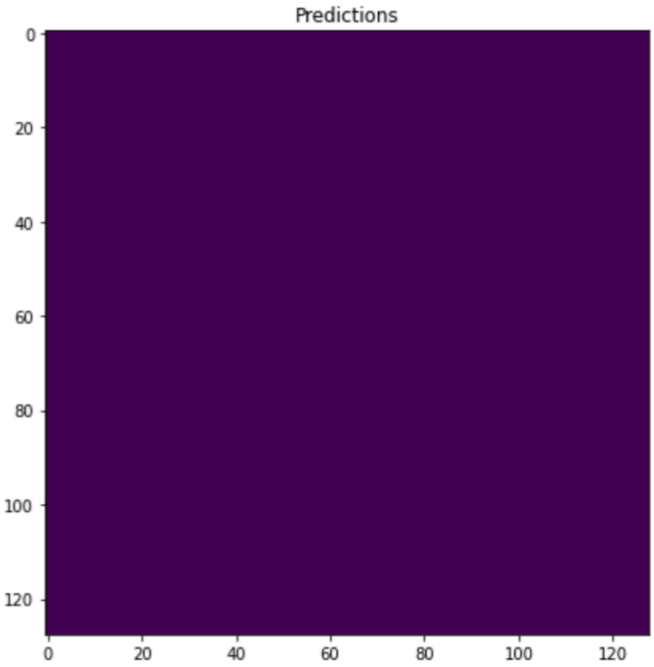


We notice that the model performed very well for images that had significant impact of the landslide. Images that were bright, had less vegetation or rivers through them were predicted better.



For images that had a high vegetation on the mountain range, the model predictions did not detect any landslides which shows a scope for improvement in the model.

We tested the model on a dataset of waterbodies from Satellite images also from [7]
However, the model only generated the mask from the green band and could not be used. This shows that maybe the model needs to be retrained on waterbodies dataset despite using the NDWI feature.


Predictions
0
20
40
60
80
100
120
0
20
40
60
80
100
120

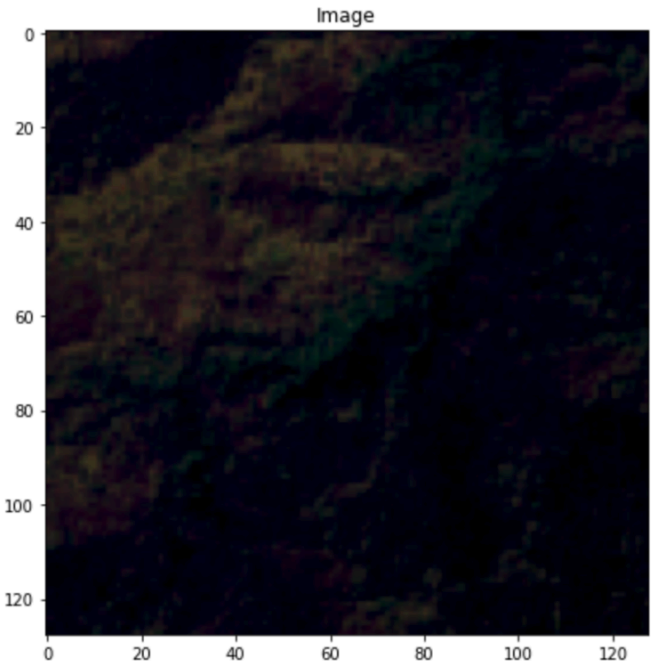
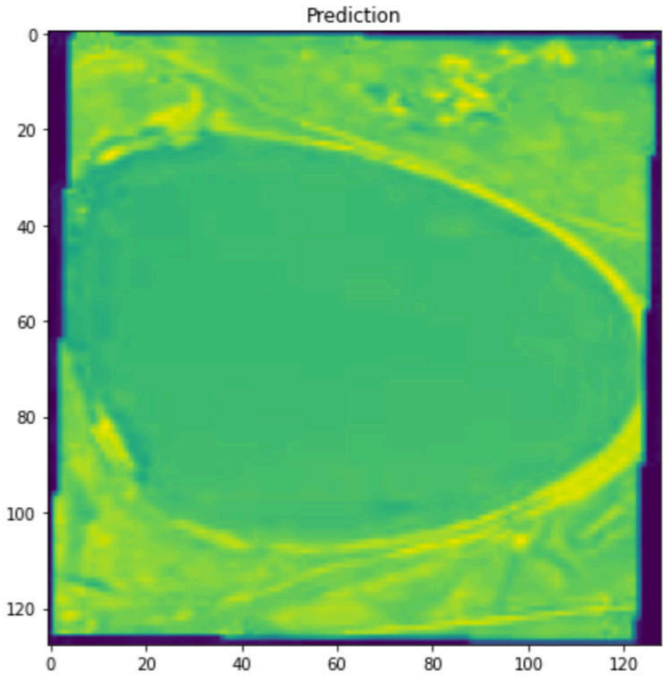


Image
0
20
40
60
80
100
120
0
20
40
60
80
100
120


Prediction
0
20
40
60
80
100
120
0
20
40
60
80
100
120

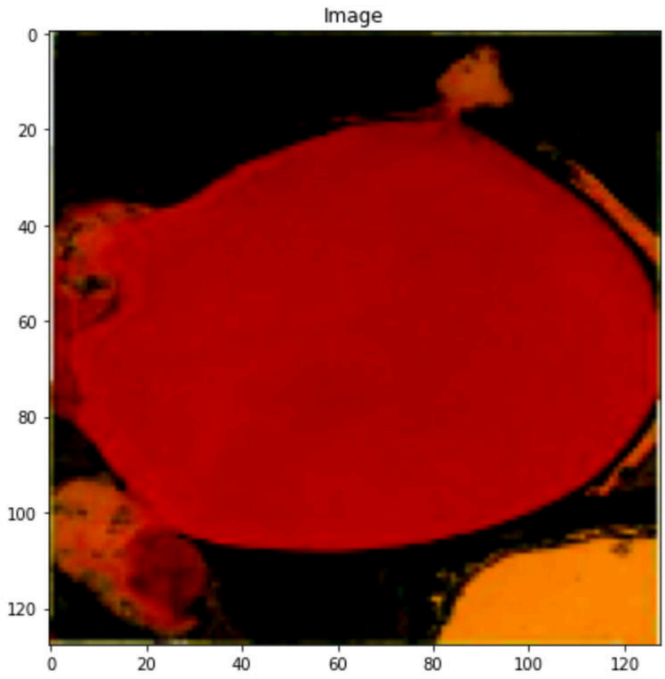


Image
0
20
40
60
80
100
120
0
20
40
60
80
100
120

Acknowledgements

I acknowledge the use of open-source data from Landslides4Sense by Institute of Advanced Research in Artificial Intelligence.

Code for the U-Net was inspired from the following resources:

- https://arxiv.org/abs/1505.04597
- https://github.com/milesial/Pytorch-UNet
- https://github.com/iamtekson/deep-learning-for-earth-observation/tree/main/Notebooks
- https://github.com/usuyama/pytorch-unet

I thank Professor Peter Belhuemur for his support and the Teaching Assistants of the course COMSW4995_006 at School of Engineering and Applied Science at Columbia University, New York.

I declare NO CONFLCIT OF INTEREST

References

1. https://floodlist.com/asia/khyber-pakhtunkhwa-pakistan-landslide-january-2022
2. https://www.ifrc.org/emergency/pakistan-monsoon-floods
3. https://www.researchgate.net/publication/230605159_Qamer_F_Abbas_S_Saleem_R_Shehzad_K_Ali_H_Gilani_H_Forest_cover_change_assessment_in_conflict-affected_areas_of_northwest_Pakistan_The_case_of_Swat_and_Shangla_districts_Journal_of_Mountain_Science
4. https://towardsdatascience.com/land-cover-classification-of-satellite-imagery-using-convolutional-neural-networks-91b5bb7fe808
5. https://www.researchgate.net/publication/347115383_Segmentation_of_Nuclei_in_Histopathology_images_using_Fully_Convolutional_Deep_Neural_Architecture
6. https://www.iarai.ac.at/landslide4sense/
7. http://www.msdc.ac.cn/#/datadetails?id=67

> -----Document Ends Here------