Dissertation on

"Natural Disaster damage assessment using Image Segmentation by Deep Learning"

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology
In
Computer Science & Engineering

UE17CS490B – Capstone Project Phase - 2

Submitted by:

Bhoomika S	**PES1201700172**
Pushpender Singh	**PES1201700243**
Keshav Agarwal	**PES1201700289**
Aditya Shankaran	**PES1201700710**

Under the guidance of

Prof. A. Vinay
PES University

January - May 2021

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

‘Natural Disaster damage assessment using Image Segmentation by Deep Learning’

is a bonafide work carried out by

Bhoomika S	**PES1201700172**
Pushpender Singh	**PES1201700243**
Keshav Agarwal	**PES1201700289**
Aditya Shankaran	**PES1201700710**

in partial fulfilment for the completion of eighth semester Capstone Project Phase - 2 (UE17CS490B) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period Jan. 2021 – May. 2021. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 8th semester academic requirements in respect of project work.

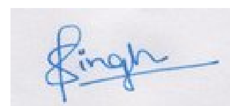
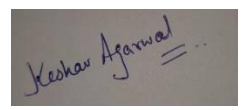
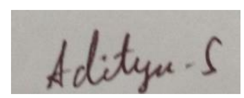
Signature	Signature	Signature
Mr. A Vinay	**Dr. Shylaja S S**	**Dr. B K Keshavan**
Professor	Chairperson	Dean of Faculty

External Viva

Name of the Examiners	**Signature with Date**
1. ____________________________	____________________________
2. ____________________________	____________________________

DECLARATION

We hereby declare that the Capstone Project Phase - 2 entitled **“Natural Disaster damage assessment using Image Segmentation by Deep Learning”** has been carried out by us under the guidance of Prof. A Vinay and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester January – May 2021. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

PES1201700172	**Bhoomika S**	
PES1201700243	**Pushpender Singh**	
PES1201700289	**Keshav Agarwal**	
PES1201700710	**Aditya Shankaran**	

ACKNOWLEDGEMENT

I would like to express my gratitude to Prof. A Vinay, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE17CS490B - Capstone Project Phase – 2.

I am grateful to the project coordinators, Prof. Silviya Nancy J, Prof. Sunitha R, for organizing, managing, and helping with the entire process.

I take this opportunity to thank Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to me various opportunities and enlightenment every step of the way. Finally, this project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

The satellite remote-sensing-based damage-mapping technique has played an indispensable role in rapid disaster response practice, whereas the current disaster response practice remains subject to the low damage assessment accuracy and lag in timeliness, which dramatically reduces the significance and feasibility of extending the present method to practical operational applications. Therefore, a highly efficient and intelligent remote-sensing image-processing framework is urgently required to mitigate these challenges. In this article, a deep learning algorithm for the semantic segmentation of high-resolution remote-sensing images using the U-net convolutional network was proposed to map the damage rapidly. The algorithm was implemented using the Keras library in the python programming language. The study takes the xView 2 Challenge Dataset as a case study, for which the pre- and post-disaster high-resolution image is used.

Our proposed damage-mapping framework has significantly improved the application value in operational disaster response practice by substantially reducing the manual operation steps required in the actual disaster response. Besides, the proposed framework is highly flexible to extend to other scenarios and various disaster types, which can accelerate operational disaster response practice.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER-1

INTRODUCTION

When a disaster strikes, quick and accurate situational information is critical to an Effective response. Before responders can act in the affected area, they need to know the Location, cause and severity of damage. But disasters can strike anywhere, disrupting local communication and transportation infrastructure, making the process of assessing specific local damage difficult, dangerous, and slow.

New building construction into floodplains extreme weather events. A building's location, shape, and construction materials may make it more vulnerable to earthquake or wind damage than nearby buildings. Managing disaster risk is most effective with an accurate, detailed, and current understanding of the building stock of a city. With a new set of constantly updated and better data we aim to accelerate the development of more accurate, relevant, and usable open-source AI models to support mapping for disaster risk management in cities. Better performing and responsibly used AI systems can provide more accurate, faster, and lower-cost approaches to assessing risk and protecting lives and property.

In recent years, mega natural disasters such as the 2011 Tohoku Earthquake-Tsunami and 2004 Sumatra Earthquake Tsunami have frequently hit the world and are considered some of the primary tremendous and tragic threats to the safety of human life and property. The increased awareness of the role of rapid damage assessment in post-disaster response to reduce the damage losses and casualties has raised much attention in implementing satellite-based methods to monitor the disaster damage information. Considering the high timeliness requirements of disaster emergency response, high-precision and efficient damage estimation methods at a fine scale to support the response are urgently required.

The implementation of streamlined, efficient damage assessment is critical in operational disaster response. There is an ongoing lag with existing damaged assessment methods considering the timeliness and accuracy. To grasp the damage situation, the current practice largely relies on the field survey and social media report. Since late 2017, Digital Globe's open data program has provided a dedicated stream of accurate high-resolution satellite imagery to support large-scale disaster response activities worldwide, which provides a good opportunity for developing a satellite-based method to map the damage. Visual interpretation of the damage from the satellite imagery has been widely used in practice for damage assessment for a long time because of its high precision. However, this method is time-consuming, particularly if the affected areas are notably large.

CHAPTER 2

PROBLEM STATEMENT

As urban populations grow, more people are exposed to the benefits and hazards of city life. One challenge for cities is managing the risk of disasters in a constantly changing built environment. Buildings, roads, and critical infrastructure need to be mapped frequently, accurately, and in enough detail to represent assets important to every community. Knowing where and how assets are vulnerable to damage or disruption by natural hazards is key to disaster risk management.

To accelerate the development, we propose a deep learning based semantic image segmentation algorithm called U-Net, which is at pixel space with very high accuracy and efficiency. This U-Net architecture is built upon the Fully Convolutional Network and modified in a way that it yields better segmentation in imaging.

CHAPTER 3

LITERATURE SURVEY

3.1 SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation by Vijay Badrinarayanan, Alex Kendall, Roberto Cipolla, Senior Member, IEEE

3.1.1 Introduction

In the literature survey, we studied across various image segmentation architectures, such as SegNet, ResNet, PSPNet and Unet. The first architecture that we covered was SegNet which stands for Semantic Segmentation. It uses semantic pixel-wise image segmentation and is a novel and practical deep fully convolutional neural network architecture. This segmentation architecture is inclusive of an encoder, a decoder and a pixel-wise layer of classification. The encoder gives low resolution feature maps, which then goes to the decoder which is responsible for mapping it into full resolution feature maps.

3.1.2 Characteristics and Implementation

The originality of this architecture lies in the way that the decoder upsamples the low level input feature maps given out by the encoder. Pooling indices computed in max-pooling is done after the encoder performs non-linear upsampling. Hence there is no need for separate upsampling. The already upsampled maps are sparse, which are then convolved with the help of trainable filters to produce dense feature maps. Segnet is designed to be efficient both in the terms of memory usage and computational time during inference. There are certain similarities and differences between SegNet and Unet.

Both use a similar upsampling approach called unpooling. However, there are fully-connected layers that make the model larger when it comes to space complexity.

3.1.3 Conclusion

U-Net is also majorly used for biomedical image segmentation. Instead of using pooling indices, the entire feature maps are transferred from encoder to decoder, then concatenation to perform convolution. This makes the model larger and needs more memory.

SegNet is slower than FCN and DeepLabv1 because SegNet contains the decoder architecture. And it is faster than DeconvNet because it does not have fully connected layers. And SegNet has low memory requirements during both training and testing. And the model size is much smaller than FCN and DeconvNet.

3.2 Resnet: Deep Residual Learning for Image Recognition by Kaiming He Xiangyu Zhang Shaoqing Ren Jian Sun

3.2.1 Introduction

Resnet is an architecture that uses deep residual learning for Image recognition. But the deeper neural networks are comparatively more difficult to train. A residual learning framework is presented to ease the training of networks that are substantially deeper than those used before. After the first CNN-based architecture (AlexNet) that won the ImageNet 2012xcompetition, every subsequent winning architecture uses more layers in a deep neural network to reduce the error rate. This works for less number of layers, but when we increase the number of layers, there is a common problem in deep learning associated with that called Vanishing/Exploding gradient. This causes the gradient to become 0 or too large. Thus when we increase the number of layers, the training and test error rate also increases.

3.2.2 Characteristics and Implementation

Using the Tensorflow and Keras API, we can design ResNet architecture (including Residual Blocks) from scratch. Below is the implementation of different ResNet architecture. For this implementation we use the CIFAR-10 dataset. This dataset contains 60, 000 32×32 colour images in 10 different classes (airplanes, cars, birds, cats, deer, dogs, frogs, horses, ships, and trucks) etc. These datasets can be assessed from the keras.datasets API function.

3.2.3 Conclusion

One of the biggest advantages of the ResNet is while increasing network depth, it avoids negative outcomes. So we can increase the depth but we have fast training and higher accuracy. It accelerates the speed of training of the deep networks. Instead of widening the network, increasing depth of the network results in less extra parameters. Reduces the effect of the Vanishing Gradient Problem. Obtaining higher accuracy in network performance especially in Image Classification.

3.3 Towards Operational Satellite-Based Damage-Mapping Using U-Net Convolutional Network: A Case Study of 2011 Tohoku Earthquake-Tsunami

3.3.1 Introduction

In this study, a deep learning algorithm using U-Net CNN was used to map damage caused by the Tsunami-Earthquake.

3.3.2 Characteristics and Implementation

The method used is image enhancement based on the change detection technology. The images collected before and after a disaster event are precisely co-registered and subtracted to generate a difference image, which represents the change between the two temporal datasets. The comparison of tsunami damage mapping achieves the overall accuracy of 70.9% in classifying the damage into "washed away," "collapsed," and "survived" at the pixel level.

3.3.3 Conclusion

The development of an intelligent image processing technique with U-Net is capable of accurate damage mapping of satellite based images. The drawback however is that the damage mapping from a single aerial optical sensor does not reveal damage from the sides.

3.4 Building Detection on Aerial Images Using U-NET Neural Networks

3.4.1 Introduction

This article presents developed convolutional neural networks (CNNs) which are implemented on GPU and can be effectively used for building detection on aerial photos. CNN's have a special architecture, aimed at fast and high quality detection and classification of aerial images.

3.4.2 Characteristics and Implementation

This article presents developed CNNs related to algorithms of deep machine learning, which are used to solve some modern problems of computer vision, in particular satellite image segmentation. To evaluate the quality of developed models, Sorensen-Dice coefficient (DSC) was used, which compares expert markup with predicted masks. CNNs are implemented on GPU to effectively detect buildings on aerial photos.

3.4.3 Conclusion

Using the special metrics of similarity between expert markup and predicted masks there was shown that U-Net got good results. The U-Net value of Sorensen-Dice coefficient (DSC) is equal to 0.77. The limitation of the U-Net architecture used is that it requires significantly longer time to train although it provides better results.

3.5 Road Extraction by Deep Residual U-Net

3.5.1 Introduction and Implementation

In this paper, a semantic segmentation neural network which combines the strengths of residual learning and U-Net is proposed for road area extraction. Road extraction is one of the fundamental tasks in the field of remote sensing. Methods based on deep neural networks have achieved state-of-the-art performance on a variety of computer vision tasks, such as scene recognition and object detection.

3.5.2 Characteristics

A semantic segmentation neural network which combines the strengths of residual learning and U-Net is proposed for road area extraction. The network is built with residual units and has similar architecture to that of U-Net. The architecture of U-Net contributes to relieving the training problem.

3.5.3 Conclusion

The benefits of this model are two-fold: first, residual units ease training of deep networks. Second, the rich skip connections within the network could facilitate information propagation, enabling networks to be designed with fewer parameters and better performance. The proposed method outperformed the other methods with which it was compared.

CHAPTER 4

PROJECT REQUIREMENTS SPECIFICATION

4.1 Hardware:

1. Window@10
2. Hard disk 256 GB
3. RAM 8GM
4. Graphic Card 2 GB

4.2 Software:

1. Python 3.5
2. OpenCV Library
3. TensorFlow
4. NumPy
5. Keras

4.3 Python BASICS:

Python is a scripting language, which means it can be run without the need to compile, unlike languages such as C and C ++. This still makes it easier to develop programs with Python. Many things you need to write a program in Python, data structures, functions are already available to you. In this way, as in other languages to solve a problem without having to design to the finest details, you can write presentations with the infrastructure in a much more rapid way. Python has a simple syntax. This makes it easier and more enjoyable to write programs, as well as more easily understand the programs written by others. Python allows you to do much with little code.

The Python language is the center of many world-famous attractions with its advantages. Organizations such as Google, YouTube, and Yahoo always need Python programmers. Russom worked on Google until 2012 and then transferred to Dropbox. This is an indication of the currentity and popularity of the Python language.

We are going to use Python Programming Language while implementing our project. The main reason to use Python is that it provides us with the conveniences that it provides in the implementation process of the software development as well as its ability to work in harmony with every environment. Python programming language is a language that does not need completion like in C or C++. This makes Python faster than others.

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding of the Python programming language.

4.3.1 Why to Learn Python?

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English words frequently whereas other languages use punctuation, and it has fewer syntactic constructions than other languages.

Python is a MUST for students and working professionals to become a great Software Engineer especially when they are working in the Web Development Domain. I will list down some of the key advantages of learning Python:

- **Python is Interpreted** − Python is processed at runtime by the interpreter. You do not need to compile your program before executing it. This is similar to PERL and PHP.
- **Python is Interactive** − you can actually sit at a Python prompt and interact with the interpreter directly to write your programs.
- **Python is Object-Oriented** − Python supports Object-Oriented style or technique of programming that encapsulates code within objects.
- **Python is a Beginner's Language** − Python is a great language for the beginner-level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

4.3.2 Characteristics of Python

Following are important characteristics of **Python Programming** −

- It supports functional and structured programming methods as well as OOP.
- It can be used as a scripting language or can be compiled to byte-code for building large applications.
- It provides very high-level dynamic data types and supports dynamic type checking.
- It supports automatic garbage collection.
- It can be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

4.3.3 Hello World using Python.

Just to give you a little excitement about Python, I'm going to give you a small conventional Python Hello World program. You can try it using the Demo link.

Live Demo

```
print("Hello, Python!");
```

4.3.4 Applications of Python

As mentioned before, Python is one of the most widely used languages over the web. I'm going to list few of them here:

- **Easy-to-learn** − Python has few keywords, simple structure, and a clearly defined syntax. This allows the student to pick up the language quickly.
- **Easy-to-read** − Python code is more clearly defined and visible to the eyes.
- **Easy-to-maintain** − Python's source code is fairly easy-to-maintain.
- **A broad standard library** − Python's bulk of the library is very portable and cross-platform compatible on UNIX, Windows, and Macintosh.
- **Interactive Mode** − Python has support for an interactive mode which allows interactive testing and debugging of snippets of code.
- **Portable** − Python can run on a wide variety of hardware platforms and has the same interface on all platforms.
- **Extendable** − you can add low-level modules to the Python interpreter. These modules enable programmers to add to or customize their tools to be more efficient.
- **Databases** − Python provides interfaces to all major commercial databases.
- **GUI Programming** − Python supports GUI applications that can be created and ported to many system calls, libraries and windows systems, such as Windows MFC, Macintosh, and the X Window system of Unix.

- **Scalable** − Python provides a better structure and support for large programs than shell scripting.

4.4 OpenCv:

- Open CV was started at Intel in 1999 by Gary Bradsky and the first release came out in 2000.
- Open CV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real-time computer vision. Originally developed by Intel, it was later supported by Willow Garage then Itseez (which was later acquired by Intel). The library is cross-platform and free for use under the open-source Apache 2 License. Starting with 2011, OpenCV features GPU acceleration for real-time operations.
- For checking which version of OpenCV is installed in your system.

```
>>> import cv2

>>> print(cv2.__version__)
```

Here are the file formats that are currently supported:

- Windows bitmaps - *.bmp, *.dib
- JPEG files - *.jpeg, *.jpg, *.jpe
- JPEG 2000 files - *.jp2
- Portable Network Graphics - *.png
- Portable image format - *.pbm, *.pgm, *.ppm
- Sun rasters - *.sr, *.ras

- TIFF files - *.tiff, *.tif

4.6 NumPy

It is a Python fundamental package used for efficient manipulations and operations on High-level mathematical functions, Multi-dimensional arrays, Linear algebra, Fourier Transformations, Random Number Capabilities, etc. It provides tools for integrating C, C++, and FORTRAN code in Python. NumPy is mostly used in Python for scientific computing.

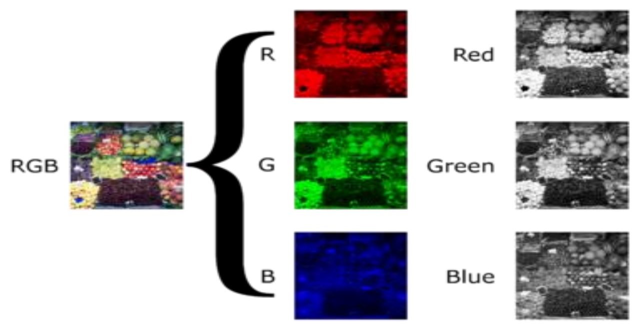
NumPy Arrays execute very much faster than the Lists in Python. There is a big difference between the execution time of arrays and lists.

4.7 Image processing:

- In the context of signal processing, an image is a distributed amplitude of color
- A digital image is an image composed of picture elements, also known as pixels, each with finite, discrete quantities of numeric representation for its intensity or gray level that is an output from its two-dimensional functions fed as input by its spatial coordinates denoted with x, y on the x-axis and y-axis, respectively.
- A grayscale (or gray level) image is simply one in which the only colors are shades of gray. The reason for differentiating such images from any other sort of color image is that less information needs to be provided for each pixel. In fact a `gray' color is one

in which the red, green and blue components all have equal intensity in RGB space, and so it is only necessary to specify a single intensity value for each pixel, as opposed to the three intensities needed to specify each pixel in a full color image.

- Often, the grayscale intensity is stored as an 8-bit integer giving 256 possible different shades of gray from black to white. If the levels are evenly spaced then the difference between successive gray levels is significantly better than the gray-level resolving power of the human eye



4.7.a Composition of RGB from 3 Grayscale Images

```
[137 137 137 136 138 129 138 134 140 136]
[137 137 137 136 138 129 138 134 140 136]
[137 137 137 136 138 129 138 134 140 136]
[137 137 137 136 138 129 138 134 140 136]
[137 137 137 136 138 129 138 134 140 136]
[140 140 131 130 136 133 132 133 136 134]
[134 134 141 133 134 137 132 128 134 137]
[133 133 129 132 131 133 129 131 131 137]
[129 129 133 133 134 134 130 132 139 131]
[130 130 133 134 128 127 129 130 135 128]
```

4.7.b 10 X 10 pixel metric of Grayscale Image

4.7.1 Images and its basic operations using openCV

Image processing is the technique to convert an image into digital format and perform operations on it to get an enhanced image or extract some useful information from it. Changes that take place in images are usually performed automatically and rely on carefully designed algorithms.

In This Module we are going to cover

1) Read & Writing an Image
2) Access pixel properties, values & modifying
3) Splitting & Merging of image channels
4) Arithmetic Operation

5) Bitwise Operation

A digital image may be defined as a two-dimensional function f(x,y), where ‘x’ and ‘y’ are spatial coordinates and the amplitude of ‘f’ at any pair of coordinates is called the intensity of the image at that point. When ‘x,’ ‘y’ and amplitude values of ‘f’ are all finite discrete quantities, the image is referred to as a digital image.

```
cv2.waitKey(0) # wait until any key is pressed
cv2.destroyAllWindows() # destroy all windows
```

4.7.1.1 Reading an Image:

```
img = cv2.imread('Imagename.jpg') # Reading Image

# get dimensions of image
dimensions = img.shape
# height, width, number of channels in image
height = img.shape[0]
width = img.shape[1]
channels = img.shape[2]

print('Image Dimension    : ',dimensions)
print('Image Height       : ',height)
print('Image Width        : ',width)
print('Number of Channels : ',channels)


filename = “New Image.png”
cv2.imwrite(filename, img) # Write a image to same directory
```

4.7.1.2 Load all Images present in particular folder:

```
import cv2

import os


def load_images_from_folder(folder):
    images = []
    for filename in os.listdir(folder):
        img = cv2.imread(os.path.join(folder,filename))
        if img is not None:
            images.append(img)
    return images
```

4.7.1.3 Selection one pixel and range of pixel from image

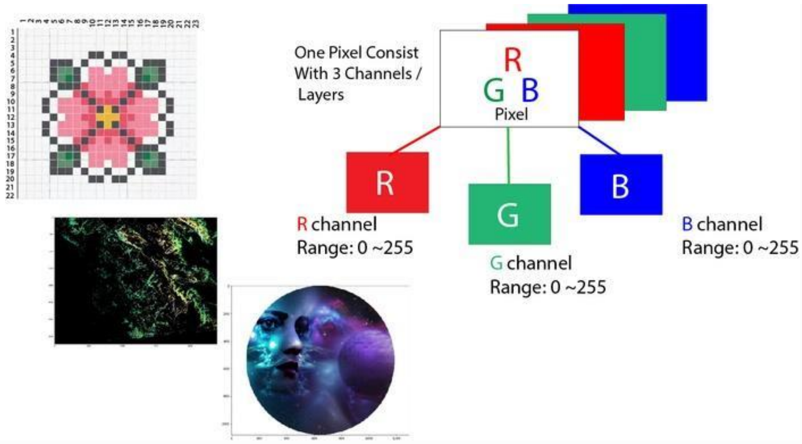
```
box = img[280:340, 330:390]

img[273:333, 100:160] = ball
```

4.7.1.4 Splitting & Merging of image channels

```
b,g,r = cv2.split(img)

img = cv2.merge((b,g,r))
```



4.7.1.4.a Splitting & Merging of image channels

4.7.1.5 Several arithmetic operations on images like addition, subtraction, bitwise operations.

We can add two images by OpenCV function, cv2.add() or simply by numpy operation, res= img1+img2 . Both images should be of same depth and type, or second image can just be a scalar value.

```
Mixedimage  = cv2.addWeighted(img1,0.7,img2,0.3,0)
cv2.imshow(Mixed_image  , Mixedimage  )
```

4.7.1.5.1 Bitwise operators

This includes bitwise AND, OR, NOT and XOR operations. They will be highly useful while extracting any part of the image, defining and working with non-rectangular ROI etc. Below we will see an example on how to change a particular region of an image.

```
# Load two images
img1 = cv2.imread('Image1.jpg')
img2 = cv2.imread('opencv_logo.png')
# I want to put logo on top-left corner, So I create a ROI
rows,cols,channels = img2.shape
roi = img1[0:rows, 0:cols ]
# Now create a mask of logo and create its inverse mask also
img2gray = cv2.cvtColor(img2,cv2.COLOR_BGR2GRAY)
ret, mask = cv2.threshold(img2gray, 10, 255, cv2.THRESH_BINARY)
mask_inv = cv2.bitwise_not(mask)
# Now black-out the area of logo in ROI
img1_bg = cv2.bitwise_and(roi,roi,mask = mask_inv)
# Take only the region of the logo from the logo image.
img2_fg = cv2.bitwise_and(img2,img2,mask = mask)
# Put logo in ROI and modify the main image
dst = cv2.add(img1_bg,img2_fg)
img1[0:rows, 0:cols ] = dst
cv2.imshow('res',img1)
cv2.waitKey(0)
cv2.destroyAllWindows()
```

CHAPTER 5

SYSTEM DESIGN

UNET Architecture and Training

The UNET was developed by Olaf Ronneberger et al. for Bio Medical Image Segmentation. The architecture contains two paths. First path is the contraction path (also called as the encoder) which is used to capture the context in the image. The encoder is just a traditional stack of convolutional and max pooling layers. The second path is the symmetric expanding path (also called as the decoder) which is used to enable precise localization using transposed convolutions. Thus it is an end-to-end fully convolutional network (FCN), i.e. it only contains Convolutional layers and does not contain any dense layer because of which it can accept images of any size.

In the original paper, the UNET is described as follows:

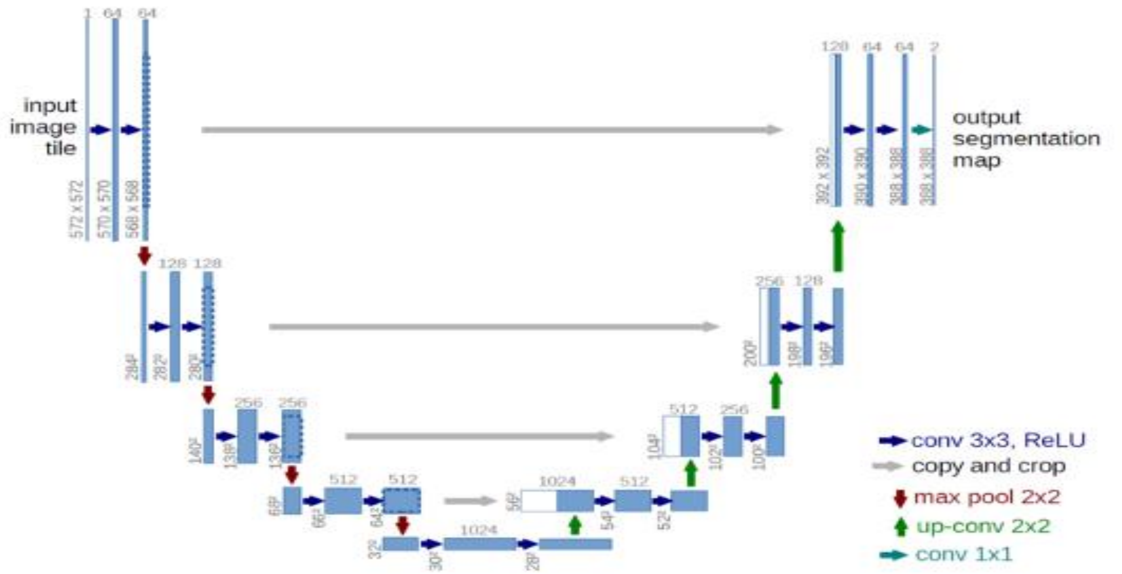
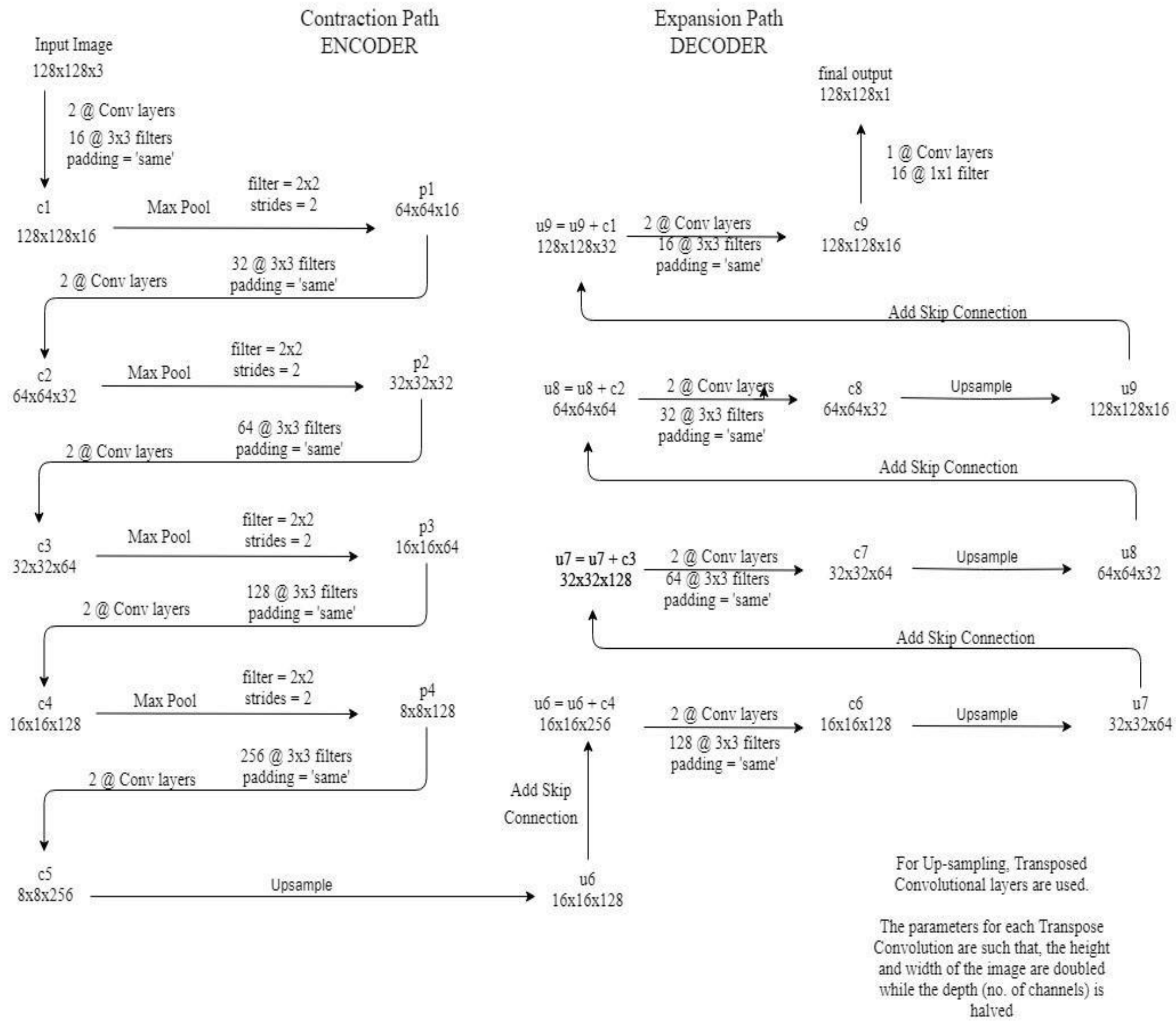


Fig. 1. U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

5.a U-Net Architecture

If you did not understand, it's okay. I will try to describe this architecture much more intuitively. Note that in the original paper, the size of the input image is 572x572x3, however, we will use input image of size 128x128x3. Hence the size at various locations will differ from that in the original paper but the core components remain the same.

Below is the detailed explanation of the architecture:



5.b Detailed U-NET Architecture

Points to note:

- 2@Conv layers means that two consecutive Convolution Layers are applied
- c1, c2, …. c9 are the output tensors of Convolutional Layers

- p1, p2, p3 and p4 are the output tensors of Max Pooling Layers
- u6, u7, u8 and u9 are the output tensors of up-sampling (transposed convolutional) layers
- The left hand side is the contraction path (Encoder) where we apply regular convolutions and max pooling layers.
- In the Encoder, the size of the image gradually reduces while the depth gradually increases. Starting from 128x128x3 to 8x8x256
- This basically means the network learns the “WHAT” information in the image, however it has lost the “WHERE” information
- The right hand side is the expansion path (Decoder) where we apply transposed convolutions along with regular convolutions
- In the decoder, the size of the image gradually increases and the depth gradually decreases. Starting from 8x8x256 to 128x128x1
- Intuitively, the Decoder recovers the “WHERE” information (precise localization) by gradually applying up-sampling
- To get better precise locations, at every step of the decoder we use skip connections by concatenating the output of the transposed convolution layers with the feature maps from the Encoder at the
- same level:
  u6 = u6 + c4
  u7 = u7 + c3
  u8 = u8 + c2
  u9 = u9 + c1

After every concatenation we again apply two consecutive regular convolutions so that the model can learn to assemble a more precise output

- This is what gives the architecture a symmetric U-shape, hence the name UNET
- On a high level, we have the following relationship: Input (128x128x1) => Encoder =>(8x8x256) => Decoder =>Output (128x128x1)

1. Using the residual network in the encoder part. If using large skip connections can help enhance the results, one can try using the residual network in the encoder part.
2. If your hardware allows you to train on big images. If you have good memory bandwidth, I would always recommend you to use images with dimensions 512X512X3, 572x572x3.
3. My data also lacked quality. The images had very low contrast. But that is how the natural scene would be. I did not do any pre-processing on the images but it would be really interesting to see improving or decreasing results after pre-processing the images.
4. I would recommend using batch normalization after activation function which will help your model to regularize and learn faster.
5. I would also want to see how using relu helps the model and what effects can we see in the convergence rate.
6. Padding is important as we are missing lots of information from the border area. I have not padded my images. So if your aim is also to predict accurate information from the image border, you must consider padding the images while training.

CHAPTER 6

PROPOSED METHODOLOGY

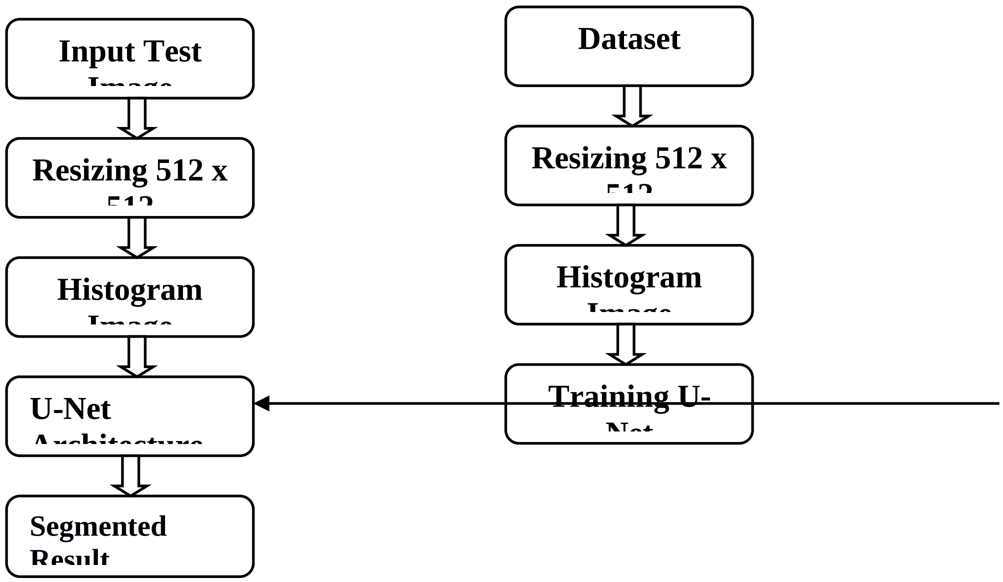
EXISTING SYSTEM:

Complexity and uncertainty in many practical problems require new methods and tools. Image processing is one of these tools that can be used as an efficient process for greater rewards. Further attention towards post disaster emergency response is well justified to make urban areas more disaster-safe. Hence, getting real-time information of damaged urban surfaces rapidly and accurately after a natural disaster is extremely useful for response processes. Li et al. studied the network environment by looking at trading volume and asset price risk when sentimental data was available using both sentiment analysis and popular machine learning approaches in the disaster assessment. Anderson and Davison considered the applications of an innovative spot price model to risk management in electricity markets. Choi and Lee discuss using regional innovation systems for sustainable economic development. Wei et al. propose a framework to assess disasters' social impacts based on network information resources.

PROPOSED METHOD:

We propose a U-Net-Architecture model for damage classification and building segmentation. One element of this architecture is a U-Net model that analyses a single input image and produces a segmentation mask showing building locations in the input image. The U-Net model is a fully convolutional network that was proposed for image segmentation. Besides its encoder-decoder structure for local information extraction, it also utilizes skip connections to retain global information. A single U-Net model analyses input frames IA and IB, which depict the same scene pre-disaster and post-disaster, respectively. Since the U-Net focuses on the building segmentation objective, it is agnostic to the disaster. In other words, we can use the same model for both pre-disaster and post-disaster images to produce binary masks IMA and IMB, corresponding to their respective input frames. The two green regions in Figure 2 indicate the shared U-Net model for IA and IB.

The features extracted from the encoder regions of the U-Net model also assist in the damage scale classification task. The two-stream features produced by the U-Net encoder and a new, separate decoder constitute the Siamese network, shown as the blue region in Figure 2. In the Siamese network, we compare features from the two input frames to detect the damage levels of buildings. Simple differencing and channel-wise concatenation are two methods to compare the two-stream features. By comparing features from the two frames, the model evaluates the differences between the features in order to assess the damage levels. Figure 2 shows the architecture of the U-Net-Architecture in difference mode (, Siam-U-Net-Attn-diff). The Siam-U-Net-Attn in concatenation mode (, Siam-U-Net-Attn-conc) can be obtained by replacing the difference operations with channel-wise concatenation operations. In Section 5, we will compare the performance of the proposed model in difference and concatenation modes.



6.a Block Diagram

IMAGE PROCESSING:

Image processing is an important and rapidly developing area which takes place in signal processing scope. Image processing is a process that tries to create a new picture as a result of changing the view and attributes of the real-life images that become numerical pictures.

Grayscale image:

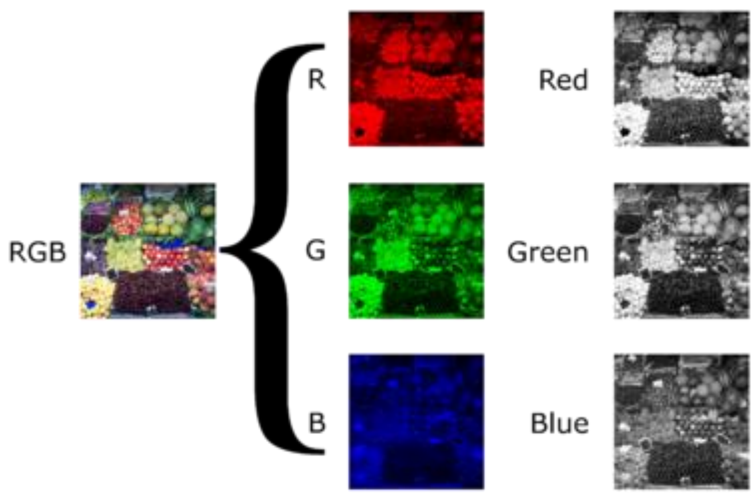
Grayscale images are distinct from one-bit bi-tonal black-and-white images, which, in the context of computer imaging, are images with only two colours: black and white (also called bi-level or binary images). Grayscale images have many shades of grey in between.

Grayscale images can be the result of measuring the intensity of light at each pixel according to a particular weighted combination of frequencies (or wavelengths), and in such cases they are monochromatic proper when only a single frequency (in practice, a narrow band of frequencies) is captured. The frequencies can in principle be from anywhere in the electromagnetic spectrum (e.g. infrared, visible light, ultraviolet, etc.).

Grayscale as single channels of multichannel colour images:

Color images are often built of several stacked color channels, each of them representing value levels of the given channel. For example, RGB images are composed of three independent channels for red, green and blue primary color components; CMYK images have four channels for cyan, magenta, yellow and black ink plates, etc.

Here is an example of color channel splitting of a full RGB color image. The column at left shows the isolated color channels in natural colors, while at right there are their grayscale equivalences:



6.b Composition of RGB from 3 Grayscale images

The reverse is also possible: to build a full color image from their separate grayscale channels. By mangling channels, using offsets, rotating and other manipulations, artistic effects can be achieved instead of accurately reproducing the original image.

Satellite images (also Earth observation imagery, space borne photography, or simply satellite photo) are images of Earth collected by imaging satellites operated by governments and businesses around the world. Satellite imaging companies sell images by licensing them to governments and businesses such as Apple Maps and Google Maps. It should not be confused with astronomy images collected by space telescopes.

Satellite Image Processing is an important field in research and development and consists of the images of earth and satellites taken by the means of artificial satellites. Firstly, the photographs are taken in digital form and later are processed by the computers to extract the information. Statistical methods are applied to the digital images and after processing the various discrete surfaces are identified by analyzing the pixel values.

The satellite imagery is widely used to plan the infrastructures or to monitor the environmental conditions or to detect the responses of upcoming disasters.
In broader terms we can say that the Satellite Image Processing is a kind of remote sensing which works on pixel resolutions to collect coherent information about the earth surface.

Majorly there are four kinds of resolutions associated with satellite imagery. These are:

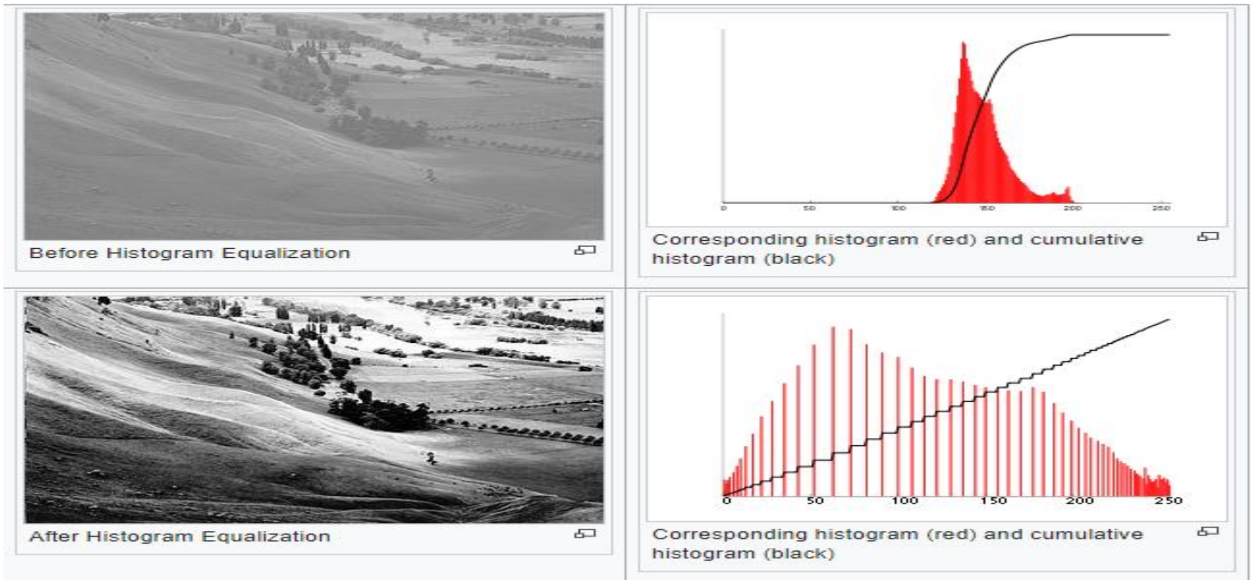
- Spatial resolution–It is determined by the sensor's Instantaneous Field of View (IFoV) and is defined as the pixel size of an image that is visible to the human eye being measured on the ground. Since it has high resolving power or the ability to separate and hence is termed as Spatial Resolution.
- Spectral resolution–This resolution measures the wavelength internal size and determines the number of wavelength intervals that the sensor measures.
- Temporal resolution–The word temporal is associated with time or days and is defined as the time that passes between various imagery cloud periods.

- Radiometric resolution–This resolution provides the actual characteristics of the image and is generally expressed in bits size. It gives the effective bit depth and records the various levels of brightness of the imaging system.

Thus, Satellite Image Processing has a huge amount of applications in research and development fields, in remote sensing, in astronomy and now even in cloud computing on a large scale.

Histogram based Image Enhancement:

Histogram equalization is applied on a grayscale image. However it can also be used on colour images by applying the same method separately to the Red, Green and Blue components of the RGB colour values of the image. However, applying the same method on the Red, Green, and Blue components of an RGB image may yield dramatic changes in the image's colour balance since the relative distributions of the colour channels change as a result of applying the algorithm. However, if the image is first converted to another colour space, Lab colour space, or HSL/HSV colour space in particular, then the algorithm can be applied to the luminance or value channel without resulting in changes to the hue and saturation of the image.[4] There are several histogram equalization methods in 3D space. Trahanias and Venetsanopoulos applied histogram equalization in 3D colour space.



6.c Shows the effect of Histogram Enhancement

Machine Learning

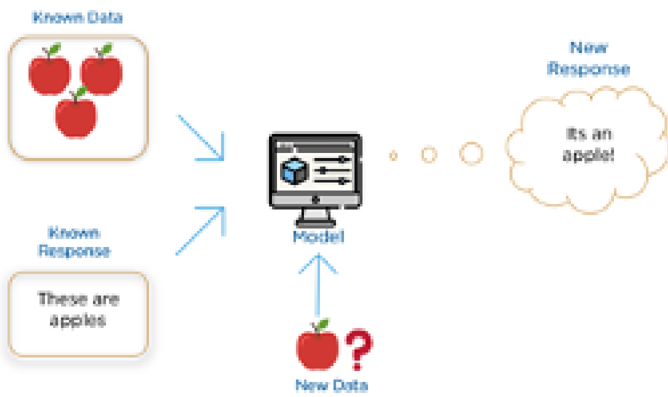
At a high-level, machine learning is simply the study of teaching a computer program or algorithm how to progressively improve upon a set task that it is given. On the research-side of things, machine learning can be viewed through the lens of theoretical and mathematical modeling of how this process works. However, more practically it is the study of how to build applications that exhibit this iterative improvement. There are many ways to frame this idea, but largely there are three major recognized categories: supervised learning, unsupervised learning, and reinforcement learning.

Supervised Learning

Supervised learning is the most popular paradigm for machine learning. It is the easiest to understand and the simplest to implement. It is very similar to teaching a child with the use of flash cards.

6.d Flash Cards

Given data in the form of examples with labels, we can feed a learning algorithm these example-label pairs one by one, allowing the algorithm to predict the label for each example, and giving it feedback as to whether it predicted the right answer or not. Over time, the algorithm will learn to approximate the exact nature of the relationship between examples and their labels. When fully-trained, the supervised learning algorithm will be able to observe a new, never-before-seen example and predict a good label for it.

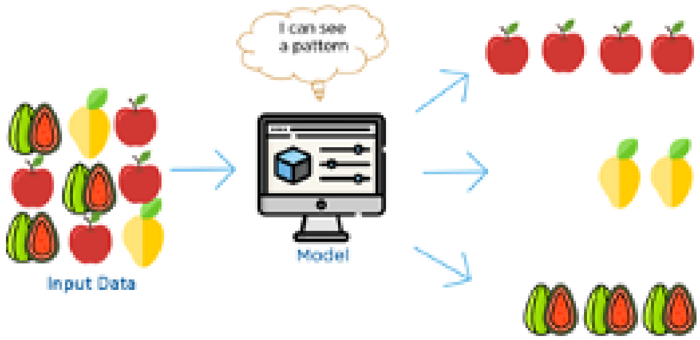


6.e Example of Supervised Learning Diagram

Supervised learning is often described as task-oriented because of this. It is highly focused on a singular task, feeding more and more examples to the algorithm until it can accurately perform on that task. This is the learning type that you will most likely encounter, as it is exhibited in many of the following common applications:

Unsupervised Learning

Unsupervised learning is very much the opposite of supervised learning. It features no labels. Instead, our algorithm would be fed a lot of data and given the tools to understand the properties of the data. From there, it can learn to group, cluster, and/or organize the data in a way such that a human (or other intelligent algorithm) can come in and make sense of the newly organized data.



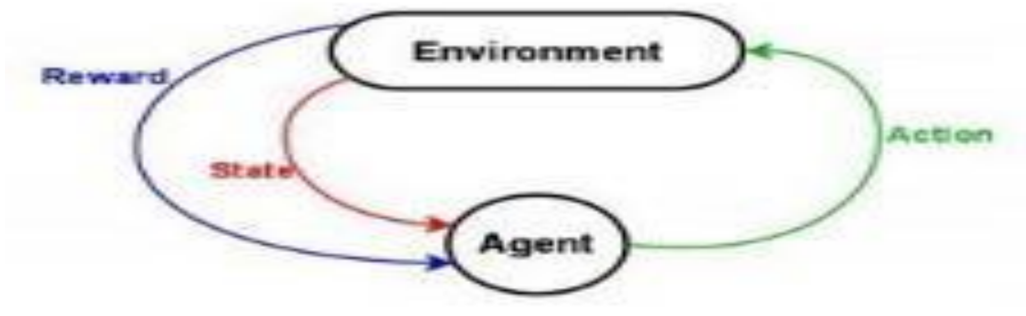
6.f Example of unsupervised Learning

What makes unsupervised learning such an interesting area is that an overwhelming majority of data in this world is unlabelled. Having intelligent algorithms that can take our terabytes and terabytes of unlabelled data and make sense of it is a huge source of potential profit for many industries. That alone could help boost productivity in a number of fields.

Reinforcement Learning

Reinforcement learning is fairly different when compared to supervised and unsupervised learning. Where we can easily see the relationship between supervised and unsupervised (the presence or absence of labels), the relationship to reinforcement learning is a bit murkier. Some people try to tie reinforcement learning closer to the two by describing it as a type of learning that relies on a time-dependent sequence of labels, however, my opinion is that that simply makes things more confusing.

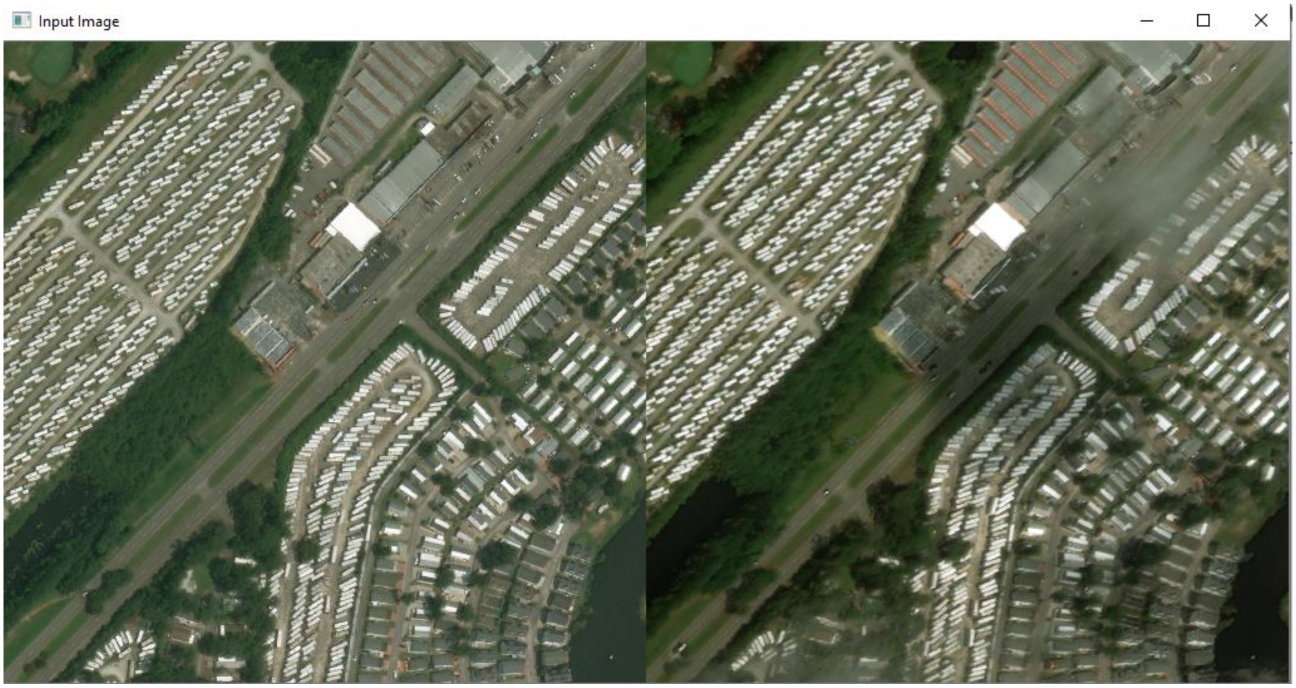
Reinforcement learning algorithm into any environment and it will make a lot of mistakes in the beginning. So long as we provide some sort of signal to the algorithm that associates good behaviours with a positive signal and bad behaviours with a negative one, we can reinforce our algorithm to prefer good behaviours over bad ones. Over time, our learning algorithm learns to make less mistakes than it used to.



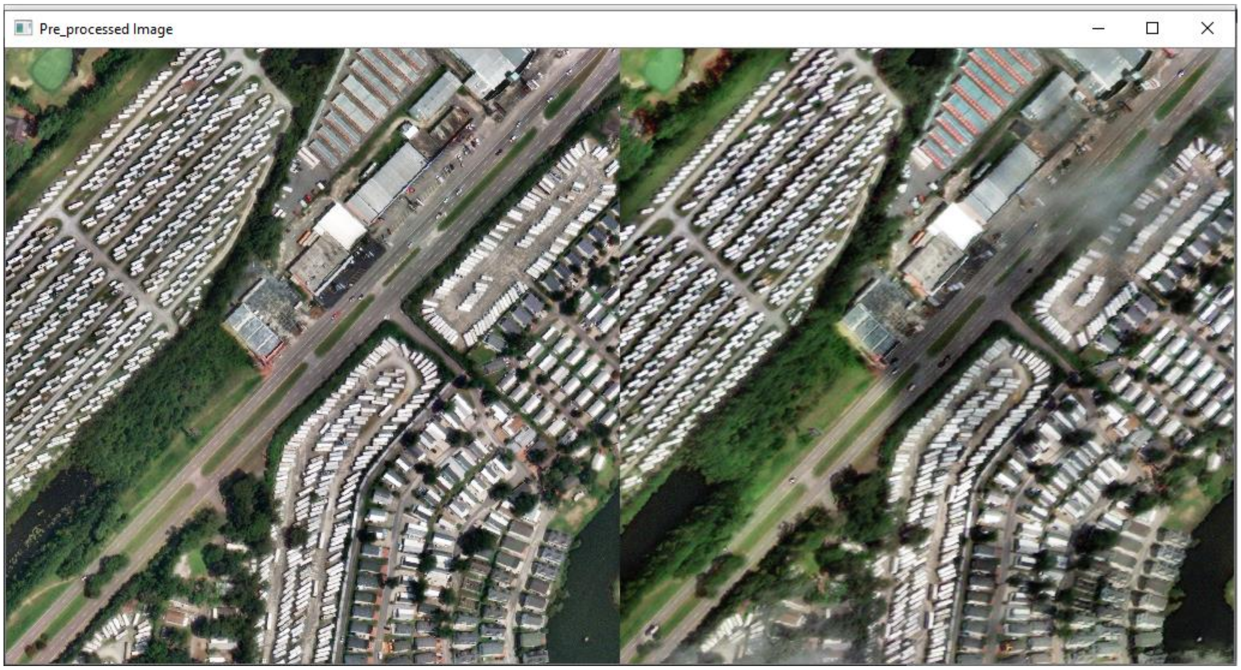
6.g Reinforcement Learning

CHAPTER 7

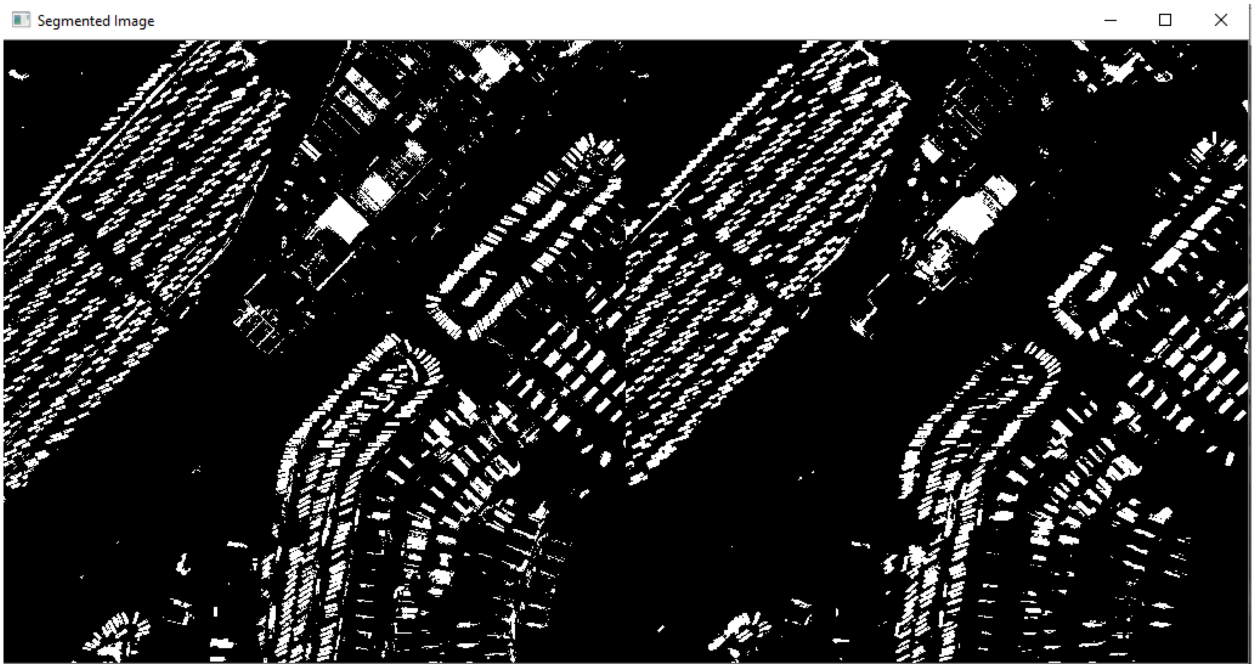
IMPLEMENTATION RESULTS



7.a Input Image: pre disaster and post disaster



7.b Enhanced Image: Pre disaster and Post Disaster



7.c U_Net segmented Output: Pre disaster and Post Disaster

7.d Different between pre and Post disaster segmented image

CHAPTER 8

CONCLUSION

In this paper, we present a Siam-U-Net-Attn model with self-attention for building segmentation and damage scale classification in satellite imagery. The proposed technique compares pairs of images captured before and after disasters to produce segmentation masks that indicate damage scale classifications and building locations. Results show that the proposed model accomplishes both damage classification and building segmentation more accurately than other approaches with the xView2 dataset. We use the self-attention module to enhance damage scale classification by considering information from the entire image.

REFERENCES/BIBLIOGRAPHY

Vijay, B., Alex, K., Roberto, C.,: SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation, IEEE 2016. [**CrossRef**]

He, K., Zhang, X., Ren, S., Sun, J.,: Deep Residual Learning for Image Recognition, Microsoft Research, 2015. [**CrossRef**]

Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger; Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 4700-4708. [**CrossRef**]

C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," June 2015. [**CrossRef**]

Zhao, Hengshuang, et al. *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017.
[**CrossRef**]

Zhao, Hengshuang, et al. "Icnet for real-time semantic segmentation on high-resolution images." *Proceedings of the European Conference on Computer Vision (ECCV)*. 2018.[**CrossRef**]

Bai, Y., Mas, E., Koshimura, S.: Towards operational satellite-based damage-mapping using u-net convolutional network: A case study of 2011 tohoku earthquake-tsunami. Remote Sensing 10(10), 1626 (2018) [**CrossRef**]

Yi, Y., Zhang, Z., Zhang, W., Zhang, C., Li, W., Zhao, T.: Semantic Segmentation of Urban Buildings fromVHR Remote Sensing Imagery Using a Deep Convolutional Neural Network, 2019. [**CrossRef**]

Hu, J., Li, L., Li, L., Wu, W., Zhao, J.: A Comparison and Strategy of Semantic Segmentation on Remote Sensing Images, Beijing, China, 2019. [**CrossRef**]

Wu, G.; Shao, X.; Guo, Z.; Chen, Q.; Yuan, W.; Shi, X.; Xu, Y.; Shibasaki, R. Automatic Building Segmentation of Aerial Imagery Using Multi-Constraint Fully Convolutional Networks. *Remote Sens.* **2018**, *10*, 407. [**CrossRef**]

Kampffmeyer, M.; Salberg, A.B.; Jenssen, R. Semantic Segmentation of Small Objects and

Modeling of Uncertainty in Urban Remote Sensing Images Using Deep Convolutional Neural Networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Las Vegas, NV, USA, 26 June–1 July 2016; pp. 680–688. [**Google Scholar**]

Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation.In Proceedings of the 18th International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), Munich, Germany, 5–9 October 2015; pp. 234–241. [**CrossRef**]

Badrinarayanan, V.; Kendall, A.; Cipolla, R. SegNet: A Deep Convolutional Encoder-Decoder Architecturefor Image Segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 2017,39, 2481–2495. [**CrossRef**]

Ma, L.; Li, M.; Ma, X.; Cheng, L.; Du, P.; Liu, Y. A review of supervised object-based land-cover image classification. ISPRS J. Photogramm. Remote Sens. 2017, 130, 277–293. [**CrossRef**]

LeCun, Y.; Bengio, Y. Convolutional networks for images, speech, and time series. In The Handbook of Brain Theory and Neural Networks; MIT Press: Cambridge, MA, USA, 1995; Volume 3361, p. 1995.[**CrossRef**]

Li, R.; Liu, W.; Yang, L.; Sun, S.; Hu, W.; Zhang, F.; Li, W. DeepUNet: A Deep Fully Convolutional Network For Pixel-Level Sealand Segmentation. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens.2018,11, 3954–3962. [**CrossRef**]