Jyothi Hills, Panjal Road,
Vettikattiri PO, Cheruthuruthy , Thrissur,
Kerala 679531

Jyothi Engineering College
NAAC Accredited College with NBA Accredited Programmes*
Approved by AICTE & affiliated to APJ Abdul Kalam Technological University
A CENTRE OF EXCELLENCE IN SCIENCE & TECHNOLOGY BY THE CATHOLIC ARCHDIOCESE OF TRICHUR
JYOTHI HILLS, VETTIKATTIRI P.O, CHERUTHURUTHY, THRISSUR. PIN-679531 PH : +91- 4884-259000, 274423 FAX : 04884-274777
NBA accredited B.Tech Programmes in Computer Science & Engineering, Electronics & Communication Engineering, Electrical & Electronics Engineering and Mechanical Engineering valid for the academic years 2016-2022. NBA accredited B.Tech Programme in Civil Engineering valid for the academic years 2019-2022.

DETECTING WATER PRESENCE USING SAR IMAGES

MAIN PROJECT REPORT

NAIR ANJALI VALSALAN (JEC17CS072)

SHIBANA (JEC17CS092)

SAN JOSE (JEC17CS089)

NAJEEB VK (JEC16CS068)

in partial fulfilment for the award of the degree of

BACHELOR OF TECHNOLOGY (B.Tech)

in

COMPUTER SCIENCE & ENGINEERING

of

A P J ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Under the guidance of

Mr. ANEESH CHANDRAN

CREATING TECHNOLOGY LEADERS OF TOMORROW

JUNE 2021

Department of Computer Science & Engineering

Jyothi Hills, Panjal Road,
Vettikattiri PO, Cheruthuruthy , Thrissur,
Kerala 679531

Jyothi Engineering College
NAAC Accredited College with NBA Accredited Programmes*
Approved by AICTE & affiliated to APJ Abdul Kalam Technological University
A CENTRE OF EXCELLENCE IN SCIENCE & TECHNOLOGY BY THE CATHOLIC ARCHDIOCESE OF TRICHUR
JYOTHI HILLS, VETTIKATTIRI P.O, CHERUTHURUTHY, THRISSUR. PIN-679531 PH : +91- 4884-259000, 274423 FAX : 04884-274777
NBA accredited B.Tech Programmes in Computer Science & Engineering, Electronics & Communication Engineering, Electrical & Electronics Engineering and Mechanical Engineering valid for the academic years 2016-2022. NBA accredited B.Tech Programme in Civil Engineering valid for the academic years 2019-2022.

DETECTING WATER PRESENCE USING SAR IMAGES

MAIN PROJECT REPORT

NAIR ANJALI VALSALAN (JEC17CS072)

SHIBANA (JEC17CS092)

SAN JOSE (JEC17CS089)

NAJEEB VK (JEC17CS068)

in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY (B.Tech)

in

COMPUTER SCIENCE & ENGINEERING

of

A P J ABDUL KALAM TECHNOLOGICAL UNIVERSITY

Under the guidance of

Mr. ANEESH CHANDRAN

CREATING TECHNOLOGY LEADERS OF TOMORROW

JUNE 2021

Department of Computer Science & Engineering

Department of Computer Science and Engineering

JYOTHI ENGINEERING COLLEGE, CHERUTHURUTHY

THRISSUR 679 531

JUNE 2021

BONAFIDE CERTIFICATE

This is to certify that the main project report entitled **Detecting Water Presence using SAR images** submitted by **Nair Anjali Valsan (JEC17CS072), Shibana (JEC17CS092), San Jose (JEC17CS089) and Najeeb V K (JEC17CS068)** in partial fulfilment of the requirements for the award of **Bachelor of Technology** degree in **Computer Science and Engineering** of **A P J Abdul Kalam Technological University** is the bonafide work carried out by them under our supervision and guidance.

Mr. Aneesh Chandran	**Dr. Swapna B Sasi**	**Dr Saju P John**
Project Guide	Project Coordinator	Associate Professor &
Assistant Professor	Associate Professor	Head of The Dept
Dept. of CSE	Dept. of CSE	Dept. of CSE

DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

COLLEGE VISION

Creating eminent and ethical leaders through quality professional education with emphasis on holistic excellence.

COLLEGE MISSION

- To emerge as an institution par excellence of global standards by imparting quality engineering and other professional programmes with state-of- the-art facilities.
- To equip the students with appropriate skills for a meaningful career in the global scenario.
- To inculcate ethical values among students and ignite their passion for holistic excellence through social initiatives.
- To participate in the development of society through technology incubation, entrepreneurship and industry interaction.

DEPARTMENT OF

COMPUTER SCIENCE & ENGINEERING

DEPARTMENT VISION

Creating eminent and ethical leaders in the domain of computational sciences through quality professional education with a focus on holistic learning and excellence.

DEPARTMENT MISSION

- To create technically competent and ethically conscious graduates in the field of Computer Science & Engineering by encouraging holistic learning and excellence.
- To prepare students for careers in Industry, Academia and the Government.
- To instill Entrepreneurial Orientation and research motivation among the students of the department.
- To emerge as a leader in education in the region by encouraging teaching, learning, industry and societal connect

PROGRAMME OUTCOMES (POs)

1. **Engineering Knowledge**: Apply the knowledge of mathematics, science, engineering fundamentals, and an engineering specialization to the solution of complex engineering problems.
2. **Problem Analysis**: Identify, formulate, review research literature, and analyze complex engineering problems reaching substantiated conclusions using first principles of mathematics, natural sciences, and engineering sciences.
3. **Design/Development of Solutions**: Design solutions for complex engineering problems and design system components or processes that meet the specified needs with appropriate consideration for the public health and safety, and the cultural, societal, and environmental considerations.
4. **Conduct Investigations of Complex Problems**: Use research-based knowledge and research methods including design of experiments, analysis and interpretation of data, and synthesis of the information to provide valid conclusions.
5. **Modern Tool Usage**: Create, select, and apply appropriate techniques, resources, and modern engineering and IT tools including prediction and modeling to complex engineering activities with an understanding of the limitations.
6. **The Engineer and Society**: Apply reasoning informed by the contextual knowledge to assess societal, health, safety, legal and cultural issues and the consequent responsibilities relevant to the professional engineering practice.
7. **Environment and Sustainability**: Understand the impact of the professional engineering solutions in societal and environmental contexts, and demonstrate the knowledge of, and need for sustainable development.
8. **Ethics**: Apply ethical principles and commit to professional ethics and responsibilities and norms of the engineering practice.
9. **Individual and Team Work**: Function effectively as an individual, and as a member or leader in diverse teams, and in multidisciplinary settings.
10. **Communication**: Communicate effectively on complex engineering activities with the engineering community and with society at large, such as, being able to comprehend and write effective reports and design documentation, make effective presentations, and give and receive clear instructions.
11. **Project Management and Finance**: Demonstrate knowledge and understanding of the engineering and management principles and apply these to one's own work, as a member and leader in a team, to manage projects and in multidisciplinary environments.
12. **Life-Long Learning**: Recognize the need for, and have the preparation and ability to engage in independent and life-long learning in the broadest context of technological change.

PROGRAMME EDUCATIONAL OBJECTIVES (PEOs)

1. The graduates shall have sound knowledge of Mathematics, Science, Engineering and Management to be able to offer practical software and hardware solutions for the problems of industry and society at large.
2. The graduates shall be able to establish themselves as practising professionals, researchers or Entrepreneurs in computer science or allied areas and shall also be able to pursue higher education in reputed institutes.
3. The graduates shall be able to communicate effectively and work in multidisciplinary teams with team spirit demonstrating value driven and ethical leadership.

PROGRAMME SPECIFIC OUTCOMES (PSOs)

1. An ability to apply knowledge of data structures and algorithms appropriate to computational problems.
2. An ability to apply knowledge of operating systems, programming languages, data management, or networking principles to computational assignments.
3. An ability to apply design, development, maintenance or evaluation of software engineering principles in the construction of computer and software systems of varying complexity and quality.
4. An ability to understand concepts involved in modeling and design of computer science applications in a way that demonstrates comprehension of the fundamentals and trade-offs involved in design choices.

COURSE OUTCOMES (COs)

C410.1 The students will be able to analyse a current topic of professional interest and present it before an audience.

C410.2 Students will be able to identify an engineering problem, analyse it and propose a work plan to solve it.

C410.3 Students will have gained thorough knowledge in design, implementations and execution of Computer science related projects.

C410.4 Students will have attained the practical knowledge of what they learned in theory subjects.

C410.5 Students will become familiar with usage of modern tools.

C410.6 Students will have ability to plan and work in a team.

ACKNOWLEDGEMENT

We take this opportunity to express our heartfelt gratitude to all respected personalities who had guided, inspired and helped us in the successful completion of this interim project. First and foremost, we express my thanks to **The Lord Almighty** for guiding us in this endeavour and making it a success.

We take immense pleasure in thanking the **Management** of Jyothi Engineering College and **Fr. Dr. Sunny Joseph Kalayathankal**, principal, Jyothi Engineering College for having permitted me to carry out this seminar. Our sincere thanks to **Dr Saju P John**, Head of the Department of Computer Science and Engineering for permitting us to make use of the facilities available in the department to carry out the interim project successfully.

We express our sincere gratitude to **Mr. Shaiju Paul & Dr. Swapna B Sasi**, project coordinators for their invaluable supervision and timely suggestions. We are very happy to express our deepest gratitude to our mentor **Mr. Aneesh Chandran**, Associate Professor,Department of Computer Science and Engineering, Jyothi Engineering College for his able guidance and continuous encouragement.

Last but not least, we extend our gratefulness to all teaching and non-teaching staff who directly or indirectly involved in the successful completion of this interim project work and to all friends who have patiently extended all sorts of help for accomplishing this undertaking.

ABSTRACT

Satellites images became a prevalent modern technology which is being used in numerous applications. We can get data and information about any phenomenon or object without making physical contact with it. It is used in various fields including, land surveying, various Earth Science disciplines such as ecology, hydrology, etc. One of the interesting areas of research is the use of Satellite images in water detection and management. This paper presents an approach to detect Presence of water using satellite data. For this purpose, data from radar satellite is used. The use of radar data advantage is that it is independent of any kind of weather conditions. optical sensors are also easily obscured by clouds and vegetation. This limitation can be reduced by integrating optical data with synthetic aperture radar data. Moreover, radar data can be useful for flood mapping because water bodies appear as dark areas due to the back scattering effect of water. Detecting Water in a fast and precise way is crucial as it helps in improving crisis management and consequently reducing damages of the natural disaster phenomenon. Deep neural networks help to handle big data to implement a variety of tasks such as object detection, change detection, and object classification. In this paper, a supervised deep learning network is used, which is considered as one of the latest trending methods to map Water regions using radar satellite data.
Here we are presenting, a Novel method for detecting presence of water using Sentinel-1 radar Satellite data, for which supervised deep learning is applied to map water regions.

Keywords: Convolutional Neural Network, SAR ,U-Net, Deep learning.

CONTENTS

List of Figures

List of Abbreviations

CNN	: *Convolutional Neural Network*
SAR	: *Synthetic Aperture Radar*
DNN	: *Deep Neural Network*
SNN	: *Siamese Neural Network*
NN	: *Neural Network*
PIR	: *Passive Infrared*
UWB	: *Ultra White Band*
HAND	: *Height Above Nearest Drainage*
UAV	: *Unmanned Aerial Vehicle*
FFT	: *fastr fourier Transformation*

CHAPTER 1
INTRODUCTION

1.1 Overview

Water is one of the most important natural resources on the earth. It plays a significant role in our day-to-day life. There are numerous resources of the consumable water which include the rainfall, groundwater and the various surface water bodies such as ponds, rivers, lakes, etc. Many approaches and techniques have been used to conserve the water. But none of them proved to be a complete way to get the sustainable growth. Thus, there is a high requirement of some strategy in which resources can be utilized in the best way without degrading the quality of the water. Because of the lack of accurate data and information, evaluating the performance of the implemented water conserving approaches is not possible.Apart from this, there are various water related issues which are being faced by many people in the World. Floods are one of those major issues that hinder the progress of happy life. Flood cause major damage to agriculture, people, and infrastructure. Thus there is a high necessity to look for some strategies through which the flood affected areas can be managed easily and quickly.

Synthetic Aperture Radar (SAR) data have many advantages over optical images as they are independent of cloud cover, the sensors are able to operate day and night and are not subject to sun glint. Observation of surface water is a functional requirement for studying ecological and hydrological processes Surface water bodies are dynamic in nature as they shrink, expand, or change their appearance or course of flow with time, owing to different natural and human-induced factors. Variations in water bodies impact other natural resources and human assets and further influence the environment. Change in surface water volume usually causes serious consequences. In extreme cases, rapid increase of surface water can result in flooding. Therefore, it is crucial to efficiently detect the existence of surface water, to extract its extent, to quantify its volume, and to monitor its dynamics.

Satellite imaging technology offers effective ways to observe surface water dynamics. Compared to traditional in situ measurements, remote sensing is much more efficient, because of its ability to continuously monitor Earth's surface at multiple scales.[6]

Mapping of natural and man-made water bodies is a critical component of environmental studies, including climate change, agriculture, flood forecasting and groundwater recharge estimation

Currently, satellite data can be considered as an effective tool to estimate disaster damages and enhance catastrophe risks because of their sensor's diverse modalities, and their huge volume with various space, time, and spectral resolutions. Neural networks became the new trend in remote sensing because of their success in many computer vision tasks. The contributions of this thesis are the following:[1]

- **S**tudying and highlighting the problems encountered in detecting water using Synthetic Aperture Radar (SAR) and optical imagery acquired over urban and non-urban areas.
- **T**wo end-to-end trainable neural networks for water detection are proposed using optical and SAR satellite data.
- **T**he proposed methods also attempt to map only water bodies resulting from permanent water surfaces.

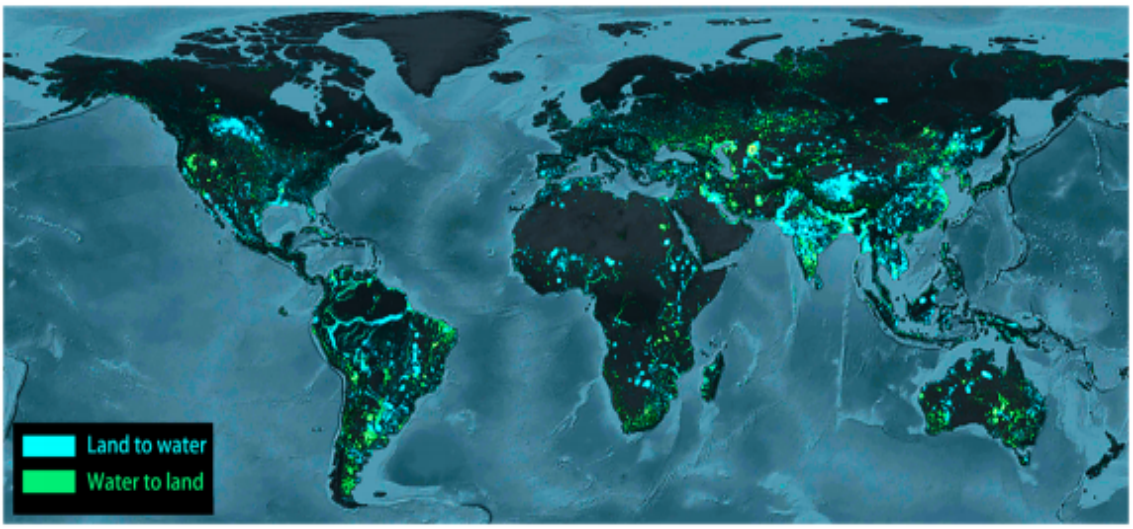

Figure 1.1: Heat map of global surface water

1.2 Objectives

The main objective is to identify the water regions with their corresponding mask image using SAR data. Here we introduce an autonomous water detection and identification system using deep learning.

1.3 Motivation

Implementing the technology of deep learning to efficiently detect the existence of surface water, to extract its extent, to quantify its volume, and to monitor its dynamics.

1.4 Data Description

We are using radar data obtained from Sentinel-1 A/B satellite. The data required for this project is taken from copernicus open acess hub . Further for the training pre-flood images of Sentinel-1 and Terrasar-x are obtained from European Space Agency and Sentinel Scientific Hub. Also we get satellite image of permanent water bodies from European Space Commission Joint research cent global dataset. The data-set as aole is generally divided into three categories: training,test and validation. As is the case with a usual deep learning problem, we would be training the model using training dataset and evaluating the performance with the test dataset.

1.5 Organization of the project

The report is organised as follow:

- **Chapter 1:Introduction** Gives an introduction to "Detecting water presence Using SAR images".
- **Chapter 2:Literature Survey** Summarizes the various existing techniques that helps in achieving the desired result.
- **Chapter 3: Problem Statement** Discuses about the need for the proposed system
- **Chapter 4:Project Management** Contains the effective project management model to be used for the project.
- **Chapter 5:Proposed System** Describes the various steps involved to produce this project.
- **Chapter 5:System Requirements & Specification**Describes the various technologies needed for implementation.
- **Chapter 6:Conclusion** Concludes with the future scope of implementation.
- **References** Includes the references for the project.

CHAPTER 2
LITERATURE SURVEY

2.1 Water Detection using Satellite Images Obtained through Remote Sensing

Water is one of the most important natural resources on the earth. It plays a significant role in our day-to-day life. There are numerous resources of the consumable water which include the rainfall, groundwater and the various surface water bodies such as ponds, rivers, lakes, etc. platforms. But there are numerous competitive sectors including the agriculture, infrastructure, domestic and industrial sectors. The major consumption of water is dedicated to the agriculture sectors.Many approaches and techniques have been used to conserve the water. But none of them proved to be a complete way to get the sustainable growth. Thus, there is a high requirement of some strategy in which resources can be utilized in the best way without degrading the quality of the water.Apart from this, there are various water related issues which are being faced by many people in the World. Floods are one of those major issues that hinder the progress of happy life. Thus there is a high necessity to look for some strategies through which the flood affected areas can be managed easily and quickly. [13]

1. **Remote Sensing:** Remote sensing is the science and craft of getting data about, phenomenon, objects or areas through the recording instruments without coming into physical contact with the items phenomenon or area under investigation. Remote sensors are used for the remote sensing applications. Remote sensors detect the energy which is reflected from the Earth. The remote sensors can be mounted on aircraft or on satellite.[17]

2. **Water Detection using Satellite Images:** One of the significant application of remote sensingis the water detection on the Earth using the satellite images retrieved through remote sensing.It is apparent that the reliable information about the spatial distribution of the surface water is necessary for numerous scientific disciplines.The capability to monitor the global water supply through the satellite images is a great effort in the remote sensing community. The remote sensing of water can be described as observing water from a long distance and checking its availability, properties, and gradual development without taking any water samples.

3. **Satellite Image processing:** There are single-band and multi-band methods used for the image processing. Through the single-band method, only one threshold value is used to extract the features from the water. The error is common in the single-band method because of the mixing of water pixels with the different cover types. But it has been said that classification techniques provide a better result if used to extract the surface water than the single-band methods .The main dataset includes the information retrieved through the satellite images. It is required to produce raw data from the previous images produced through several satellites with some specific resolution. After that these images need to rectify and corrected to measure the surface reflectance.

2.1.1 Techniqes

Various techniques have been adopted to get the high quality of pictures in order to get better information about the presence of water.

3.1 **Standard supervised maximum likelihood classification:** After obtaining the MODIS images from the NASA website, the images covering the Sindh province were classified. The standard supervised maximum likelihood classification was used for the images. In this way desired inundated class was translated into a shape file. Next, the visual interpretation was utilized for the inundated areas. To exert a different kind of info-layers, the shape file was intersected with the topographic maps.

3.2 **Image Pre-Processing :** The image pre-processing technique was used to prepare the images as an input for the satellite. The main pre-processing steps used in the techniques include atmospheric correction, co-registration, mosaicking, radiometric calibration, and resampling.The images were translated to the at-satellite radiance for the atmospheric correction and radiometric calibration.Each image was transformed to at satellite reflectance. The images were converted to surface reflectance from the at-satellite reflectance using the DOS (Dark Object Subtraction) method.The new images were then obtained by mosaicking.One image was used as a reference for the co-registration of multi-temporal images. Root Mean Square Error (RMSE) was used to co-register the images.

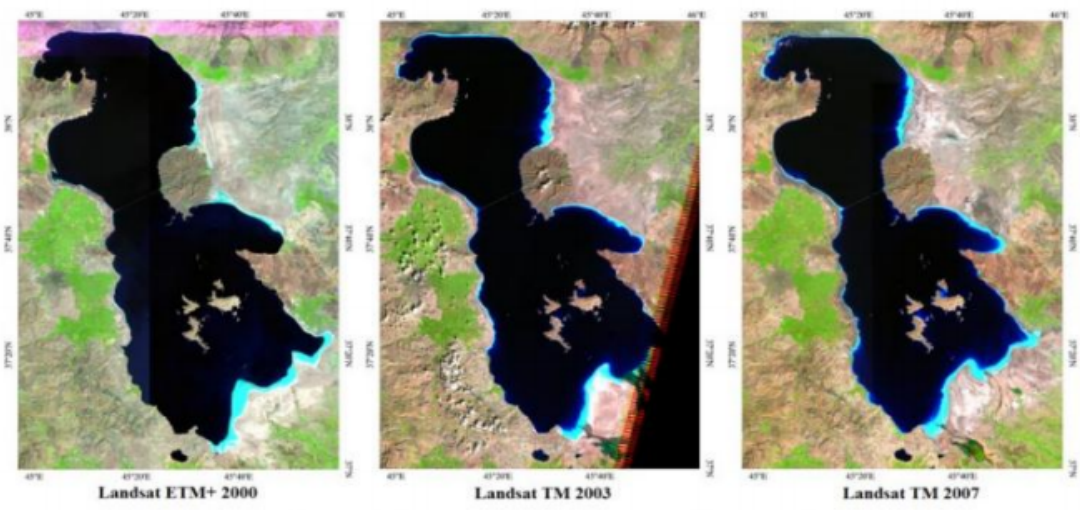

Figure 2.1: Data after pre-processing

3.3 **Water Classification:** The water classification technique was utilized to analyze the areas of water. In this technique, N observations for each tile which is stored in AGDC were taken. The code 1 was used to define the pixel as water and code 0 to define the pixel as not water. The regression tree classification was used to obtain the water classification. The regression tree classification that uses normalized difference rations (NDI) and Landsat Spectral bands.

3.4 **Groundwater Potential Model (GPM) :** Weighted overlay technique was used for implementing the GIS (Geographical Information System). The technique was used to determine the potential groundwater value. Each scale value of reclassified layer was multiplied by its weight to determine the potential groundwater value for a given area.[10]The presence of groundwater was classified into three categories including the low, moderate and high potential for the groundwater

2.2 Automatic detection of surface-water bodies from Sentinel-1 images for effective mosquito larvae control

Surface-water bodies are important breeding habitats of mosquitoes, thus, surface-water bodymaps together with other parameters are often used as predictors of mosquito larvae presence.1 Regular monitoring of surface-water bodies in near real time (NRT) is necessary for the timely prevention of outbreaks.The mapping of surface-water areas can be achieved through observation and recording in the field.Optical data have been used to identify surface-water areas, exploiting the fact that clear water absorbs almost all near-infrared irradiance, in contrast to the highly reflecting nearby soil or vegetation.Remote sensing methods based on the use of radar sensors seem to be more suitable for monitoring water areas, offering the advantage of uninterrupted data supply under any weather conditions, and the ability—under certain circumstances—to detect inundated areas covered by vegetation.The result is low backscatter at water surfaces, which contrasts to higher backscatter of soil and vegetation morphology.The aim is to develop an automated methodology for detecting surface-water bodies in wetlands and agricultural land using Sentinel-1 data. [12]

1. **Sentinel-1 Data Acquisition and Pre-Processing:** The Sentinel-1 images used for the testing and comparison of the water detection algorithms were acquired already pre-processed through the Google Earth Engine (GEE) platform. Images of two dates during the summer period were used for the comparison between all water-presence detection methods tested and images of four additional dates were used to further test the method found to provide the most accurate results and to compare the accuracy achieved between the two thresholding methods developed. All Sentinel-1 images were GRD products acquired in descending orbit direction with the Interferometric Wide swath instrument mode.

2. **Field Data:** Field data were collected on dates concurrent with the Sentinel-1 images' acquisition and consist of various wetland and rice polygons, where the status of water and vegetation was recorded.The minimum required size of the sampling polygons was set to approximately 1000 m2 considering the pixel size of the Sentinel-1 data.The location of the polygons was recorded using a GPS and digital photos were taken in-situ. The polygons were characterized as homogeneous and heterogeneous. The additional information collected in the field for each polygon is the percentage of vegetation cover and its height in each polygon, the percentage of the polygon covered by water and its depth.[15]

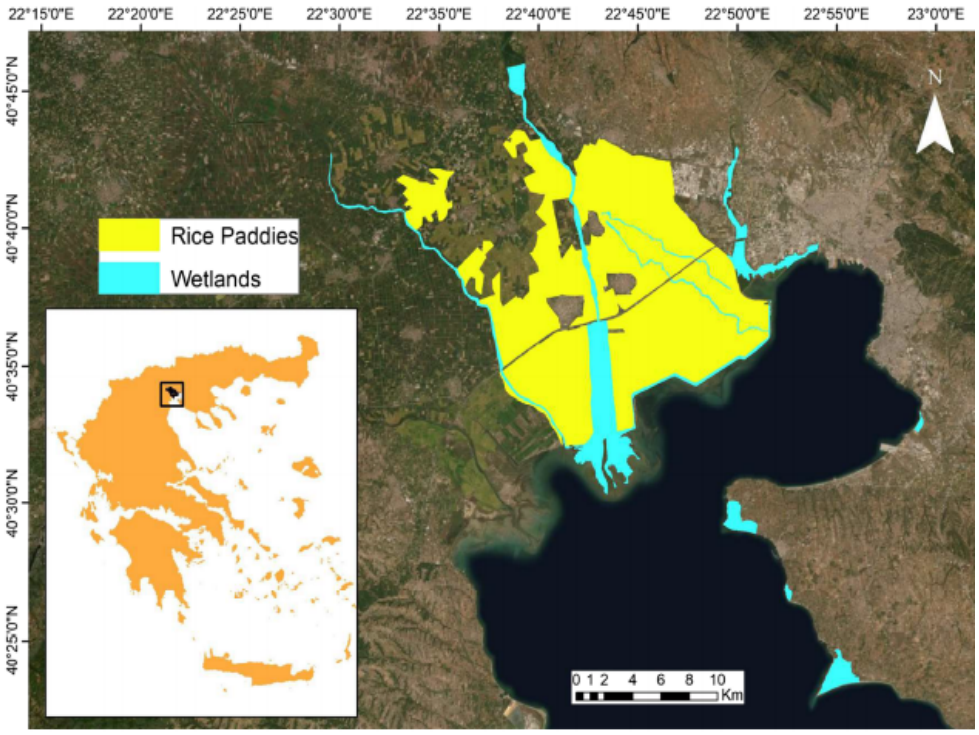

Figure 2.2: Location of rice paddies and wetlands

2.2.1 Water Mapping Methods

2.1 **Automated Thresholding Methods:** Various studies apply a threshold value on SAR backscatter data to identify inundated areas. The threshold value can be set manually by a human operator after examining the distribution of backscatter values on the SAR image's histogram or can be set automatically.Two methods were tested for the automatic detection of a threshold value for water presence: (a) the Otsu method and (b) the Otsu Valley-emphasis method.

- **Otsu method:** The image binarization method developed by Otsu automatically calculates a threshold value from an image's pixels value histogram, minimizing the weighted within class variance.
- **Otsu Valley-Emphasis Method:** The modification of Otsu algorithm proposed by Ng uses the same basic technique as the original Otsu method but gives more weight to the poin between the two peaks of bimodal distribution histograms (valley) or to the bottom of the histogram (low-value histogram area) in cases of unimodal histograms.

2.3 Radar Satellite Imagery and Automatic Detection of Water Bodies

The first satellite in the series, Sentinel-1A, is a radar satellite whose data we used.Radar data has many advantages. A combination of the radar signal wavelength and active satellite sensor allows for a continuous data acquisition, regardless of the time of day and weather conditions. [3]

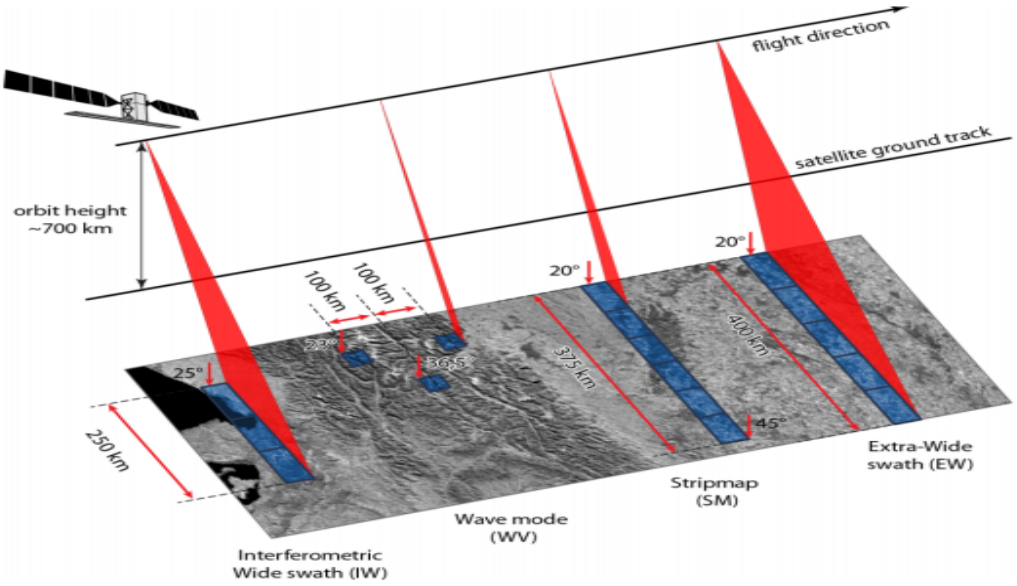

Figure 2.3: Different image acquisition modes of Sentinel-1 satellites

2.3.1 Preprocessing

The first step in the algorithm is preprocessing of downloaded data and its transformation into a format, suitable for further analysis and multitemporal comparison with other radar images of the same type and coverage.Most of the datasets have a narrow transitional band of very weakly detected signal along the left and right swath edges. These values would introduce additional errors into the detection procedure, therefore they are removed from further analysis. The next step is radiometric calibration of the remaining values, detected by the sensor. We consider a variety of factors, such as acquisition incidence angles and radiation pattern of the antenna, and transform the values into physical quantities that express the backscatter intensity of the transmitted radar signal. Because a dynamic range of these values is usually a few orders of magnitudes, they are logarithmically transformed into decibels. This also increases the contrast between water and land.

Coarse Determination of Water Bodies

Algorithm for detection of water bodies relies on auxiliary data about permanent water bodies.The procedure of separation of water bodies from land is based upon a computation of threshold values. Pixels with higher grey vales represent land, while the ones with lower values represent water. Thresholds are determined by a computation and mutual comparison of normalized grey values histograms.Sentinel-1 collects data in two different polarizations. This is roughly similar to two different spectral bands in optical imagery. The described procedure for determination of thresholds is performed independently for both polarizations and their product which yields six different thresholds for water.This first step of water bodies detection does not give the exact extent of water bodies, but only small "seed" areas, which can be classified as water with a high probability. Seed areas are selected from pixels for which their grey values in both polarizations are lower than the computed upper thresholds and the product of both polarizations have a higher value that the upper threshold for water of multiplied polarizations.

Removing Errors

Removing incorrect water detections consists of several steps.First consider each pixel individually, and then merge the neighbouring pixels and consider them as a region.Assume that water bodies do not lie on steep terrain and high in the mountains, where snow covers the ground well into spring. Therefore disregard all the pixels classified as water on slopes steeper than 8 degrees, in areas higher than 1400 m, and in radar shadows. This is called geometric error removal. It is followed by radiometric error removal, which disregards all the pixels where the ratio of polarizations VV/VH is lower than an empirically determined value of 0.75.The first step of considering regions removes the ones with a very small number of elements.The final step of miss detection removal deals with large and flat asphalt surfaces that were frequently identified as water because of their surface characteristics. [5]

Region Growing

In the last step of water bodies detection, determine the exact border between water and land.Using the region growing process that starts with initial seed areas, determined in the previous steps, and expands these areas to the neighbouring pixels with similar grey values. Neighbouring pixels with values lower than the higher water threshold are joined with the seed areas. The process is executed twice, independently for each polarization, with the same seed areas. The region growing process is more reliable and accurate for VV polarization, because the contrast between land and water is higher. The final result therefore consists of waters detected on

VV polarization merged with detections from VH polarization in the areas of permanent water bodies and their vicinity. This fills in the gaps that may be a result of water roughness.

2.4 Automatic near real-time flood detection using Suomi-NPP/VIIRS data

Near real-time satellite-derived flood maps are invaluable to river forecasters and decision-makers for disaster monitoring and relief efforts. With support from the JPSS (Joint Polar Satellite System) Proving Ground and Risk Reduction (PGRR) Program, flood detection software has been developed using Suomi-NPP/VIIRS (Suomi National Polar-orbiting Partnership/Visible Infrared Imaging Radiometer Suite) imagery to automatically generate near real-time flood maps for National Weather Service (NWS) River Forecast Centers (RFC) in the USA. The software, which is called VIIRS NOAA GMU Flood Version 1.0 (hereafter referred to as VNG Flood V1.0), consists of a series of algorithms that include water detection, cloud shadow removal, terrain shadow removal, minor flood detection, water fraction retrieval, and floodwater determination. The software is designed for flood detection in any land region between 80°S and 80°N, and it has been running routinely with direct broadcast SNPP/VIIRS data. Initial feedback from operational forecasters on product accuracy and performance has been largely positive. Offline evaluation efforts include the visual inspection of over 10,000 VIIRS false-color composite images and inter-comparison with MODIS automatic flood products and a quantitative validation using Landsat imagery. The steady performance from the routine process and the promising evaluation results indicate that VNG Flood V1.0 has high feasibility for flood detection at the product level. [9]

2.4.1 Water detection

Based on the underlying conditions, floods can be divided into two types: supra-vegetation/bare soil floods are the most common, and supra-snow/ice floods are generally limited to rivers flowing from low latitudes to high latitudes during spring snowmelt/break-up periods. These two flood types show different spectral features, and therefore, water detection is divided into two types: supra-veg/bare soil water detection and supra-snow/ice water detection.

In SNPP/VIIRS imagery, pixels are first classified into three types: cloud cover, snow/ice cover, and land/water. Cloud cover is masked using the SNPP/VIIRS 750-m cloud mask intermediate product (IICMO). Snow/ice cover is flagged with the 375-m snow cover product or by running a snow detection module and is then subjected to supra-snow/ice water detection tests. The remaining pixels are classified as clear-sky land or water, which are subjected to supra-veg/bare soil water detection tests with a decision-tree approach using the following vari-

ables: RVs, RNIR, RSWIR, NDVI, NDSI, and NDWI. Pre-trained decision trees are used to separate supra-veg/ bare soil water pixels from clear-sky land pixels. Supra-snow/ice water detection constitutes an additional step because most supra-snow/ice water locations are counted as covered by snow/ice using snow/ice detection algorithms before water detection.

2.4.2 Cloud shadow removal

Because cloud shadows are not spectrally different from floodwater, a geometry-based method can be used to remove cloud shadows from water maps. In this method, a spherical geometry model is established between cloud shadows and clouds and is then iteratively applied to construct a one-to-one relationship based on the assumption that one cloud pixel casts, at most, one cloud shadow pixel. If the position of a cloud pixel in the VIIRS imagery is known, the position of that cloud pixel can be located by calculating an arc using the sensor azimuth angle.

Based on the geometric model over a spherical surface, an iteration method is further applied to the cloud height, to construct a one-to-one relationship between the cloud and cloud shadow using a group of adjacent cloud and cloud shadow pixels. Here, the cloud height is coarsely estimated using cloud top temperatures and nearby clear-sky land surface temperatures under average atmospheric temperature profiles.

2.4.3 Terrain shadow removal

Similar to cloud shadows, most terrain shadows are classified as floodwater during water detection. To remove terrain shadows, an object-based method is applied using 375-m DEM data resampled from SRTM-2 and ASTER data based on the surface roughness. Because terrain shadows generally appear in mountainous topography, the surface roughness is usually much larger than floodwater, which mainly accumulates in low-lying areas. The method is object-based, and thus, a surface roughness analysis is performed on a group of adjacent pixels instead of on single pixels. Water pixels are clustered into a group and viewed as one object for calculating the surface roughness parameters. A validation analysis has shown that this method removes > 95 percent of the terrain shadows from VIIRS flood maps, and it also helps to remove other false water detection results, such as some residual cloud shadows, dark lava land, and burn scars.

2.4.4 Minor flood detection

At a 375-m spatial resolution, water signals from many minor floods are too weak to be detected in VIIRS imagery, especially when floodwaters are veiled by vegetation cover or urban development. Change detection is used as the main approach to detect minor floods around water pixels and existing rivers, lakes, and reservoirs in ancillary water reference maps. The method determines a minor water pixel either by comparing water signals from before and after flooding or by comparing water signals with surrounding confirmed clear-sky land pixels that have similar land cover types to the minor water pixel.

2.4.5 Water fraction retrieval

Due to the moderate spatial resolution of the VIIRS data, most detected flood pixels are a mixture of water and other land types, such as vegetation, bare soils, or snow/ice. The water fraction, which is defined as the percentage of the water surface in a satellite pixel, represents the flood status more accurately than a simple water/no water mask classification. For flood detection using VNG Flood V1.0, only supra-veg/bare soil floodwaters are retrieved for the water fractions. A dynamic nearest neighbor search (DNNS) method based on a linear combination model is applied by considering the varying subpixel land portion in a land-water mixed pixel and counting the adjacent land pixels with similar mixture ratios to estimate the reflectance of the land for the retrieval.[4]

2.4.6 Flood determination

The retrieved supra-veg/bare soil water fractions are compared against the water reference map. The MODIS 250-m global water mask is resampled to a 375-m water/no water mask using the nearest neighbor interpolation method to spatially match it with the VIIRS imagery, while the 30-m National Land Cover Dataset is resampled to a 375-m water mask by calculating the water fractions in 375-m grids. If the water fraction of a pixel in the water reference map is < 1 percent, then the floodwater is determined directly and represented with the retrieved fraction. If the water fraction of a pixel is > 1 percent in the water reference map, then the floodwater is only determined if the retrieved water fraction is at least 40 percent larger than that in the water reference map.

To differentiate ice from water using supra-veg/bare soil in VIIRS flood maps, supra-snow/ice water is classified as one type and represented in a simple water/no water mask without any fraction retrieval data. Supra-snow/ice floodwater can also be determined by comparing it against the water reference map; however, supra-snow/ice water within river channels and

lakes is retained to reflect information on the river/lake ice status.

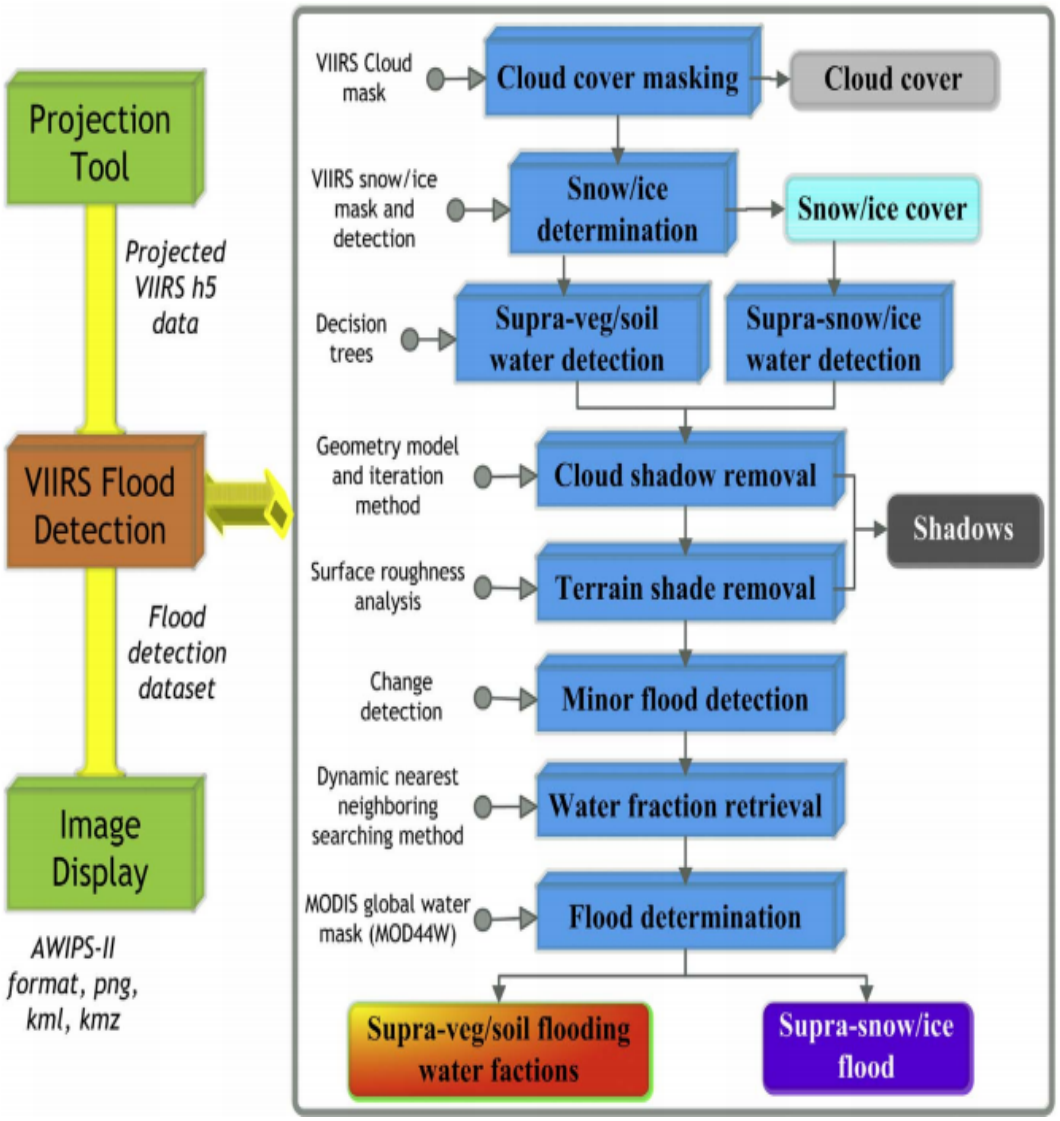

Figure 2.4: Flow chart of VNG Flood V1.0.

2.5 Detecting, Extracting, and Monitoring Surface Water From Space Using Optical Sensors

Surface water refers to water on the surface of the Earth, such as a river, lake, wetland, and the ocean. They are of paramount importance in sustaining all forms of live. Water helps preserve the biodiversity in riparian or wetland ecosystems by providing habitats to a plethora of flora and fauna.Surface water bodies are dynamic in nature as they shrink, expand, or change their appearance or course of flow with time, owing to different natural and human-induced factors.Variations in water bodies impact other natural resources and human assets and further influence the environment. Change in surface water volume usually causes serious consequences. In extreme cases, rapid increase of surface water can result in flooding. Therefore, it is crucial to efficiently detect the existence of surface water,to extract its extent, to quantify its volume, and to monitor its dynamics. [16]

Remote sensing technology offers effective ways to observe surface water dynamics. Compared to traditional in situ measurements, remote sensing is much more efficient, because of its ability to continuously monitor Earth's surface at multiple scales. Remote sensing data sets provide spatially explicit and temporally frequent observational data of a number of physical attributes about the Earth's surface that can be appropriately leveraged to map the extent of water bodies at regional or even global scale, and to monitor their dynamics at regular and frequent time intervals.

There are generally two categories of sensors that can serve the purpose of measuring surface water—the optical sensor and the microwave sensor. Microwave sensors, due to their usage of long wavelength radiation, have the ability to penetrate cloud coverage and certain vegetation coverage. Independent of solar radiation, they can work day and night under any weather condition.To demonstrate how the use of remote sensing has advanced surface water and flood inundation studies, this review tracks the latest progress in measuring surface water using satellite-based optical sensors, especially those works that have been conducted in the last decade.[7]

2.5.1 Detection of Surface Water With Different Sensors

As a significant land cover change phenomenon, surface water dynamics have always been an important topic of Earth observation. Spatial resolution refers to the area of ground observed within a pixel and determines the level of details captured by the sensors.

Coarse Spatial Resolution Sensors

Coarse-resolution remote sensors have the inherent defect of low accuracy due to their high information generalization level, but they usually have the characteristics of a high temporal resolution and a broad coverage.This sensor was originally designed to monitor the ocean and atmosphere but was later found to also be effective in detecting large-scale flood events.

Medium Spatial Resolution Sensors

Landsat is one of the most successful satellite series in history. Since the first mission was launched in 1972, it has been continuously supplying medium-resolution images for over 40 years. The sensors that were onboard early Landsat missions are Multispectral Scanner (MSS), and later upgraded to Thematic Mapper (TM) on Landsat-4 and Landsat-5, and Enhanced Thematic Mapper Plus (ETM+) on Landsat-7. Operational Land Imager (OLI) is the latest optical sensor onboard Landsat-8. The spatial resolutions of these sensors have been gradually improved from 80 to 30 m (with a 15 m resolution panchromatic band). Resolutions at this level are ideal for detecting dynamics of almost all kinds of surface water bodies.

High Spatial Resolution Sensors

Increasing the spatial resolution of sensors has been a key goal of remote sensing research, and significant advances have been made in the last decade. Many of the new sensors, for example, IKONOS, RapidEye, Worldview, ZY-3, Quickbird, and GF-1/2, are able to provide images with spatial resolutions at meter or even submeter level. At this level of resolutions, small water bodies can be detected successfully.Applications using high-resolution images to detect surface water are quite extensive.

2.5.2 Extraction of Surface Water Using Different Methods

The principle of extracting surface water from optical images is the obviously lower reflectance of water, compared to that of other land cover types, in infrared channels. Based on this, many methods have been developed for extracting water areas from optical remote sensing imagery. A simple method is to density slice a single infrared band to derive a water map. Either supervised or unsupervised classification methods were used to generate land cover maps, from which water maps could be extracted. Decision trees were also built using multispectral bands to delineate water coverage from others.An easy and effective way to extract water is to use water indices, which are calculated from two or more bands, to identify the differences between

water and nonwater areas. Many indices have been developed to extract surface water areas or flood inundation extent.[2]

Thresholding is one of the most critical issues in using water indices to extract water bodies. Base on the reflectance characteristics of water, NDWI and mNDWI values for water are usually greater than 0. Therefore, a threshold of 0 is often applied to extract water from index images. However, it has been suggested that adjustment of the threshold value could usually achieve better extraction results. This is especially tricky when thresholding either a time series of images that cover the same water body or a single image that covers a group of water bodies, because automation would be impossible if manual adjustments on threshold value for each image were required.

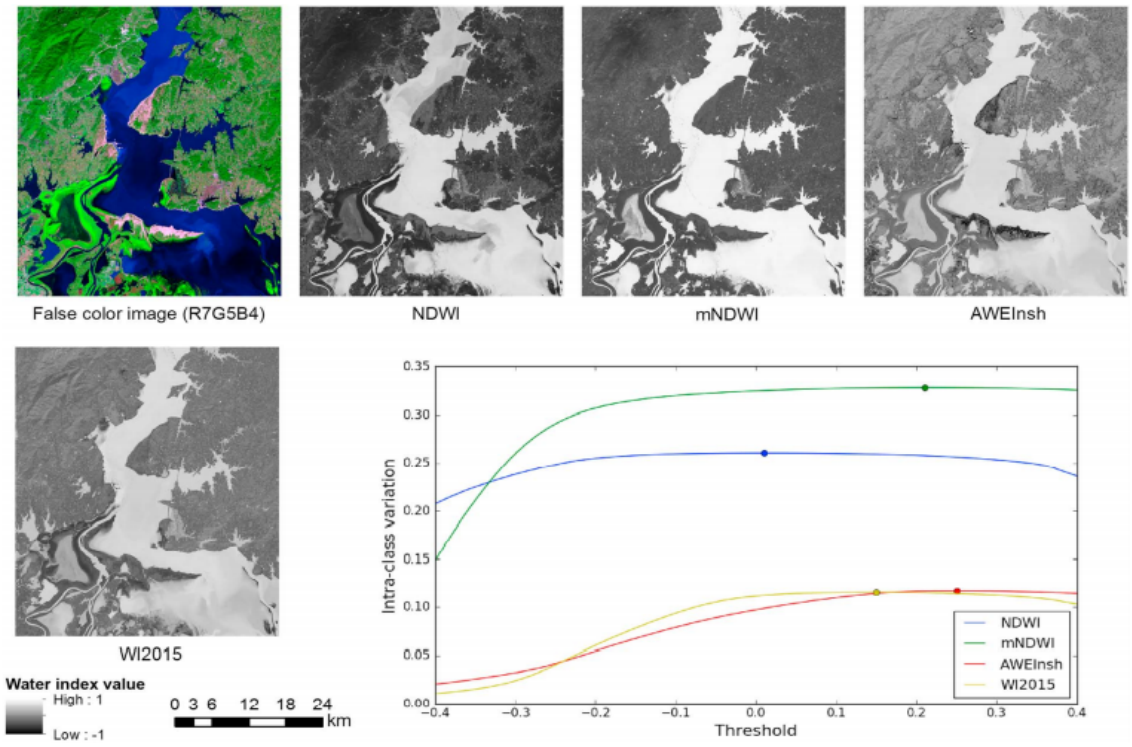

Figure 2.5: Landsat OLI image for Poyang Lake and line graph showing the interclass variation between water and land with different thresholds

2.6 Water Across Synthetic Aperture Radar Data (WASARD): SAR Water Body Classification for the Open Data Cube

Water classification is an important function of Earth imaging satellites, as accurate remote classification of land and water can assist in land use analysis, flood prediction, climate change research, as well as a variety of agricultural applications .The ability to identify bodies of water remotely via satellite is immensely cheaper than contracting surveys of the areas in question, meaning that an application that can accurately use satellite data towards this function can make valuable information available to nations which would not be able to afford it otherwise.Highly reliable applications for the remote detection of water currently exist for use with optical satellite data such as that provided by LANDSAT. The alternative solution for water detection is using SAR data, which images the Earth using cloud penetrating microwaves. [18]

2.6.1 Methadology

Using supervised classification, WASARD compares SAR data to a dataset pre-classified by WOFS in order to train an SVM classifier. This classifier is then used to detect water in other SAR scenes outside the training set. .

Algorithm Selection

The machine learning model used by WASARD is the Linear Support Vector Machine (SVM). This model uses a supervised learning algorithm to develop a classifier, meaning it creates a vector which can be multiplied by the vector formed by the relevant data bands to determine whether a pixel in a SAR scene contains water. This classifier is trained by comparing data points from selected bands in a SAR scene to their respective labels, which in this case are “water” or “not water” as given by the WOFS algorithm. The SVM was selected over the Random Forest model, which outperformed the SVM in training speed, but had a greater classification time and lower accuracy, and the Multilayer Perceptron Artificial Neural Network, which had a slightly higher average accuracy than the SVM, but much greater training and classification times.

Feature Selection

Sentinel-1 collects data from two bands: the Vertical/Vertical polarization (VV) and the Vertical/Horizontal polarization (VH). When 100 SVM classifiers were created for each polarization individually, and for the combination of the two. Both the VV and VH bands trades slightly lower recall for significantly greater precision when compared with the VH band alone, and that using the VV band alone is inferior in both metrics.WASARD therefore defaults to using both the VV and VH bands, and includes the option to use solely the VH band.

Training a Classifier

The steps in training a classifier with WASARD are:

1. Selecting two scenes (one SAR, one optical) with the same spatial extents, and acquired close to each other in time, with a preference that the scenes are taken on the same day.
2. Using the WOFS algorithm to produce an array of the detected water in the scene of optical data, to be used as the labels during supervised learning
3. Data points from the selected bands from the SAR acquisition are bundled together into an array with the corresponding labels gathered from WOFS. A random sample with an equal number of points labeled “Water” and “Not Water” is selected to be partitioned into a training and a testing dataset
4. Using Scikit-Learn’s LinearSVC object, the training dataset is used to produce a classifier, which is then tested against the testing dataset to determine its precision and recall

Classifying a Dataset

Once the classifier has been created, it can be used to detect water in an xarray of SAR data using wasard classify(). By taking the dot product of the classifier’s coefficients and the vector formed by the selected bands of SAR data, an array of predictions is constructed. A classifier can effectively be used on the same spatial extents as the ones where it was trained, or on any area with a similar landscape. While testing WASARD, it was observed that a classifier trained on one lake in Vietnam detected water accurately across the entire nation. [8]

Noise Reduction

One major drawback of SAR data is that the intensity of the returned signals from a SAR satellite's radar pulses will vary in frequency from pixel to pixel due to the waves falling out of phase after hitting the Earth's surface. This results in an image which has pixels that vary in intensity across a homogenous area, referred to as speckle. This speckle noise can potentially cause the classifier to mislabel points, and so reducing speckle is necessary to ensuring WASARD makes the most accurate classifications possible. Since speckle shows up as points normally too small to be separate bodies of water, WASARD reduces noise by scanning over the classified dataset with a moving window and removing isolated pixels labeled as water.

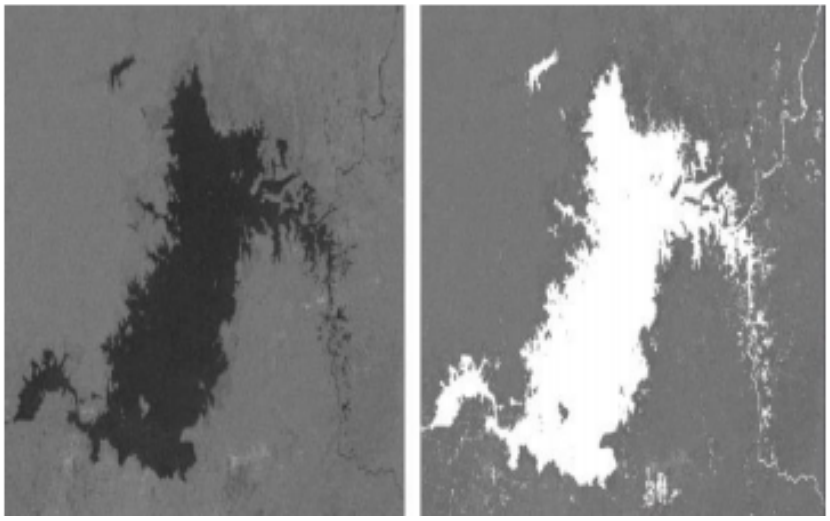

Figure 2.6: : Water Detection by WASARD

CHAPTER 3

PROBLEM STATEMENT

The project "Detecting Presence of Water Using SAR Images" aims at providing an efficient and novel solution to detect permanent water. In contrast to previous, our motivation is to overcome the limitations of the existing methods done by optical satellite images by bringing up a more efficient and accurate detecter by using the Convolutional Neural Networks,u-Net architecture etc.Surface water bodies are dynamic in nature as they shrink, expand,course of flow with time, owing to different natural and human-induced factors. Variations in this impact other natural resources, human assets and environment. Change in surface water volume usually causes serious consequences. In extreme cases, rapid increase of surface water can result in flooding. data obtained from here can be used for detecting floods also and can help in managing crisis during disaster.

CHAPTER 4
PROJECT MANAGEMENT

4.1 Introduction

Project management is the discipline of planning, organizing, securing, managing, leading, and controlling resources to achieve specific goals. A project is a temporary endeavor with a defined beginning and end (usually time-constrained, and often constrained by funding or deliverables), undertaken to meet unique goals and objectives, typically to bring about beneficial change or added value. The temporary nature of projects stands in contrast with business as usual (or operations), which are repetitive, permanent, or semi-permanent functional activities to produce products or services. In practice, the management of these two systems is often quite different, and as such requires the development of distinct technical skills and management strategies.

In our project we are following the typical development phases of an engineering project

1. Initiation
2. Planning and Design
3. Execution and Construction
4. Monitoring and Controlling Systems
5. Completion

4.1.1 Initiation

The initiating processes determine the nature and scope of the project. The initiating stage should include a plan that encompasses the following areas :

1. Analysing the business needs/requirements in measurable goals
2. Reviewing of the current operations
3. Financial analysis of the costs and benefits including a budget
4. Stakeholder analysis, including users, and support personal for the project

5. Project charter including costs, tasks, deliverables, and schedule

4.1.2 Planing and design

After the initiation stage, the project is planned to an appropriate level of detail (see example of a flow-chart). The main purpose is to plan time, cost and resources adequately to estimate the work needed and to effectively manage risk during project execution. As with the initiation process, a failure to adequately plan greatly reduces the project's chances of successfully accomplishing its goals.

- Determining how to plan
- Developing the scope statement
- Selecting the planning team
- Identifying deliverables and creating the work breakdown structure
- Identifying the activities needed to complete those deliverables
- Developing the schedule
- Risk planning

4.1.3 Execution

Executing consists of the processes used to complete the work defined in the project plan to accomplish the project's requirements. The execution process involves coordinating people and resources, as well as integrating and performing the activities of the project in accordance with the project management plan. The deliverables are produced as outputs from the processes performed as defined in the project management plan and other frameworks that might be applicable to the type of project at hand.

4.1.4 Monitoring & controlling

Monitoring and controlling consists of those processes performed to observe project execution so that potential problems can be identified in a timely manner and corrective action can be taken, when necessary, to control the execution of the project. The key benefit is that project performance is observed and measured regularly to identify variances from the project management plan.

4.2 System Development Life Cycle

The Systems development life cycle (SDLC), or Software development process in systems engineering, information systems, and software engineering, is a process of creating or altering information systems, and the models and methodologies that people use to develop these systems. In software engineering, the SDLC concept underpins many kinds of software development methodologies. These methodologies form the framework for planning and controlling the creation of an information system.

The SDLC phases serve as a programmatic guide to project activity and provide a flexible but consistent way to conduct projects to a depth matching the scope of the project. Each of the SDLC phase objectives is described in this section with key deliverables, a description of recommended tasks, and a summary of related control objectives for effective management. The project manager must establish and monitor control objectives during each SDLC phase while executing projects. Control objectives help to provide a clear statement of the desired result or purpose and should be used throughout the entire SDLC process.

4.2.1 Spiral Model

We have used the Spiral model in our project. The Spiral model incorporates the best characteristics of both- waterfall and prototyping model. In addition, the Spiral model also contains a new component called Risk Analysis, which is not there in the waterfall and prototype model. In the Spiral model, the basic structure of the software product is developed first. After the basic structure is developed, new features such as user interface and data administration are added to the existing software product. This functionality of the Spiral model is similar to a spiral where the circles of the spiral increase in diameter. Each circle represents a more complete version of the software product. The spiral is a risk-reduction oriented model that breaks a software project up into main projects, each addressing one or major risks. After major risks have been addressed the spiral model terminates as a waterfall model. Spiral iteration involves six steps:

1. Determine objectives, alternatives and constraints.
2. Identify and resolve risks.
3. Evaluate alternatives.
4. Develop the deliverables for the iteration and verify that they are correct.
5. Plan the next iteration.

6. Commit to an approach for the next iteration.

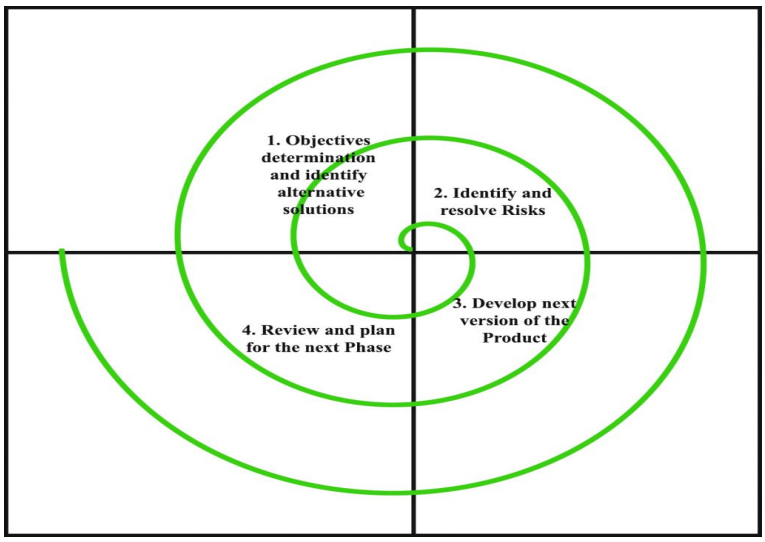

Figure 4.1: Spiral Model

CHAPTER 5
METHODOLOGY

5.1 System Requirements & Specifications

5.1.1 Spyder

Spyder is an open-source cross-platform integrated development environment (IDE) for scientific programming in the Python language. Spyder integrates with several prominent packages in the scientific Python stack, including NumPy, SciPy, Matplotlib, pandas, IPython, SymPy, and Cython, as well as other open-source software. It is released under the MIT license.

5.1.2 Windows 10

Windows 10 is a series of personal computer operating systems produced by Microsoft as part of its Windows NT family of operating systems. It is the successor to Windows 8.1 and was released to manufacturing on July 15, 2015, and to retail on July 29, 2015. Windows 10 receives new builds on an ongoing basis, which are available at no additional cost to users. Mainstream builds of Windows 10 are labeled version YYMM with YY representing the year and MM representing the month of release. For example, the latest mainstream build of Windows 10 is Version 1809. There are additional test builds of Windows 10 available to Windows Insiders. Devices in enterprise environments can receive these updates at a slower pace, or use long-term support milestones that only receive critical updates, such as security patches, over their ten-year lifespan of extended support.

5.1.3 Python 3.6.2

Python is a dynamic object-oriented programming language that can be used for many kinds of software development. It offers strong support for integration with other languages and tools, comes with extensive standard libraries, and can be learned in a few days. Many Python programmers report substantial productivity gains and feel the language encourages the development of higher quality, more maintainable code.

Python runs on Windows, Linux/Unix, Mac OS X, OS/2, Amiga, Palm Handhelds, and

Nokia mobile phones. Python has also been ported to the Java and .NET virtual machines. Python is distributed under an OSI-approved open source license that makes it free to use, even for commercial products.

5.1.4 SCIKIT-learn

Scikit-learn provides a range of supervised and unsupervised learning algorithms via a consistent interface in Python. It is licensed under a permissive simplified BSD license and is distributed under many Linux distributions, encouraging academic and commercial use. The library is built upon the SciPy (Scientific Python) that must be installed before you can use scikit-learn. This stack that includes:

1. NumPy: Base n-dimensional array package
2. SciPy: Fundamental library for scientific computing
3. Matplotlib: Comprehensive 2D/3D plotting
4. IPython: Enhanced interactive console
5. Sympy: Symbolic mathematics
6. Pandas: Data structures and analysis

Extensions or modules for SciPy are conventionally named SciKits. As such, the module provides learning algorithms and is named scikit-learn.

5.1.5 Pandas

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. The name is derived from the term "panel data", in econometrics term for data sets that include observations over multiple periods for the same individuals. Pandas is an open-source, BSD-licensed Python library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language. Python with Pandas is used in a wide range of fields including academic and commercial domains including finance, economics, statistics, analytics, etc

5.1.6 Microsoft Visual Studio

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps.Visual Studio includes a code editor supporting IntelliSense as well as code refactoring. The integrated debugger works both as a source-level debugger and a machine-level debugger. Other built-in tools include a code profiler, designer for building GUI applications, web designer, class designer, and database schema designer.

5.1.7 Jupyter Environment

JupyterLab is a web-based interactive development environment for Jupyter notebooks, code, and data. JupyterLab is flexible: configure and arrange the user interface to support a wide range of workflows in data science, scientific computing, and machine learning. JupyterLab is extensible and modular: write plugins that add new components and integrate with existing ones.

5.2 Deep Learning

The use of machine learning techniques, in particular deep learning, is becoming increasingly important. This tool has proven to be both powerful and efficient in multiple application fields including remote sensing. Deep learning uses neural networks (NN) to get high-level features from raw input. The neural network is formed of multiple layers. To be called a deep neural network, a neural network should involve more than two hidden layers between the input layer and the output layer.[19]

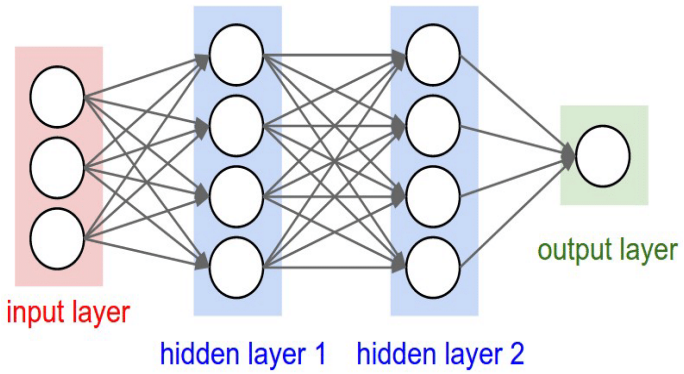

Figure 5.1: Neural Network

The core unit of a neural network is the node or neuron.

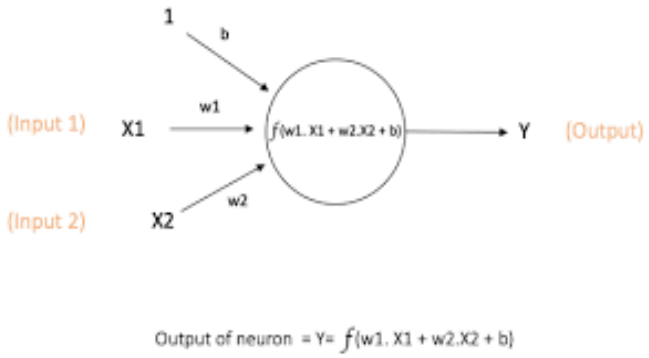

Figure 5.2: A single node

The node uses a function f that takes the inputs given by other nodes connected to it and calculates with a mathematical function the output to eventually transfer it to another node. The nodes are connected with edges. Each edge has a weight w, that is used by function f to do its computation. Biases, like the bias b, can also be added depending on the application of NN. Multiple nodes form a layer and two or more connected layers form a deep NN (DNN).In this

thesis, the interest is focused on one type of neural network which is deep feedforward neural networks. These NNs do not include loops or cycles, and their layers are arranged as an input layer, two or more hidden layers, and an output layer.

The input layer nodes are given the data the user feeds to the network, and they subsequently pass it to the next layer, without performing any computations. Next, the hidden layer takes as input what is forwarded to it from the input layer and performs computations on it before forwarding it to the next layer. The performed calculations are mathematical functions, or activation functions, that introduce non-linearity into the NN. The hidden layers are not connected with the outside world, and the output of a hidden layer is passed only to the next hidden layer or to the output layer. The final layer, or output layer, gets as input the output from the previous layer and forwards it to the user.[11]

The task of the DNN is to classify the pixel of the input image into “flood” pixel or “non-flood” pixel. The model is trained with data containing image pairs and their corresponding flood masks, so the learning of the DNN is supervised. The goal is to DNN is to arrive at the point of least error as soon as possible. A key concept in the learning process of the network is the calculation of the error between the predicted output of the network and the ground-truth labels, or masks. The error mathematical expression is called the loss function. The edges have weights and biases can also be added to the activation function. Therefore, these parameters are part of the model and need to be adjusted accordingly with respect to the error of the loss function. The process of optimizing the weights and biases enables the NN to make better guesses or outputs as the parameters are adjusted according to their contribution to the overall error. The process of scoring the input, calculating the loss, and updating the parameters of the model is repeated until the error can no longer be reduced. For cases where the output should have two classes, as it is the case for flood mapping with classes [flood, non-flood], one of the most used loss functions is the binary cross-entropy or log loss. Its mathematical expression is:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i.log(p(y_i)) + (1-y_i).log(1-p(y_i))$$

where y is the label or true value and p(y) is the predicted probability of the label being flood or non-flood for all N pixels or input.

5.3 CNN

The feedforward NN considered in this study is CNN. CNN's are heavily used in computer vision and image analysis applications, as they were proven to be very efficient in classification problems. They have also been used in comparing images in different contexts. The four essential operations that define CNNs are convolution, non-linearity, and pooling.[14]

The goal of a CNN is to recognize or classify that object. The convolution operation enables CNN to extract features from the input layer, which represents the image. It is done by simultaneously sliding a matrix called "filter" over the input matrix and computing the element-wise multiplication. The outputs of multiplications are added up and the result is a "feature map". The calculations of the convolution layer involve the calculated neuron's value by summing up all neurons' input values (xi), the weighted inputs (wij) of the neuron plus the bias parameter (bj), and the application of an activation function:

$$y = \sum_{i=1}^{n} w_{ij} * x_i + b_j$$

The activation function, or the second characteristic CNN operation, adds the nonlinearity to the model. The mathematical functions used as activation functions can vary depending on the application purpose, and some examples that are widely used are the Sigmoid function, Tanh function, and Rectified Linear Unit function. Some parameters affect the size of the feature map and the main ones are the padding, the depth of filters, and the stride. The padding is a parameter that helps avoid outputs shrinking and information loss on the corners of an image or feature map. The second parameter is depth, and it is the number of filters applied. The stride parameter sets how many pixels or input elements are skipped while sliding the applied filter. Thus, the future map size decreases when the number of skipped pixels increases.

The third operation that defines CNNs is pooling. This operation enables the subsampling of the output of the lower layer to reach translational invariance. The most significant information is kept while reducing the dimension of the lower layer. Pooling also reduces the number of computations done on the network, which regulates overfitting. The CNN architecture can include at the end of it a fully connected layer used as a classifier.

5.4 segmentation

Semantic segmentation is a natural step in the progression from coarse to fine inference:The origin could be located at classification, which consists of making a prediction for a whole input.The next step is localization / detection, which provide not only the classes but also additional information regarding the spatial location of those classes.Finally, semantic segmentation achieves fine-grained inference by making dense predictions inferring labels for every pixel, so that each pixel is labeled with the class of its enclosing object ore region.

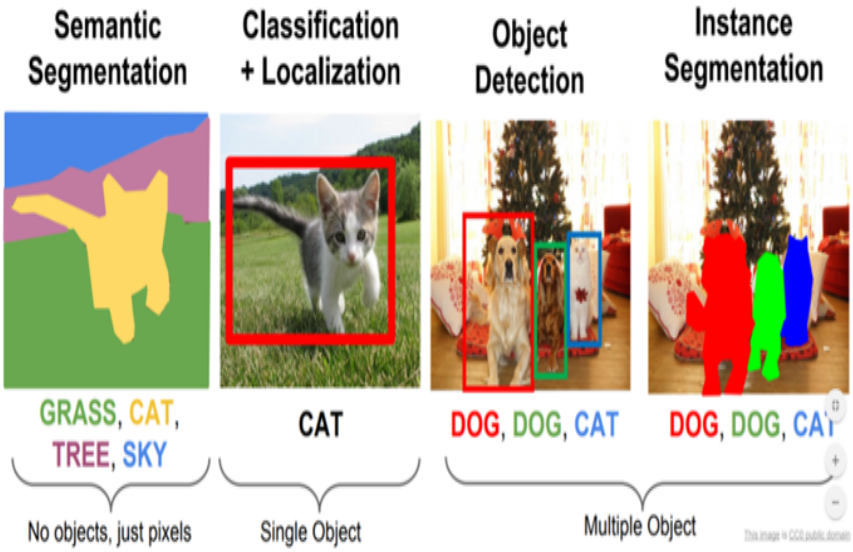

Figure 5.3: Architecture diagram

A general semantic segmentation architecture can be broadly thought of as an encoder network followed by a decoder network:

The encoder is usually is a pre-trained classification network like VGG/ResNet followed by a decoder network. The task of the decoder is to semantically project the discriminative features (lower resolution) learnt by the encoder onto the pixel space (higher resolution) to get a dense classification. Unlike classification where the end result of the very deep network is the only important thing, semantic segmentation not only requires discrimination at pixel level but also a mechanism to project the discriminative features learnt at different stages of the encoder onto the pixel space. Different approaches employ different mechanisms as a part of the decoding mechanism

5.5 Proposed System

5.5.1 Modules

Data Acquisition Module

One of the major challenges with using DNNs for remote sensing applications is the availability of data. labeled Sentinel 1 data is hard to find because attempts to use SAR data in deep learning only started recently. Only some selected flood maps are made available to users through the Copernicus open Acess hub . The service was not available for use by the public, and only authorized users were able to trigger the process of producing flood maps for the desired region. Thus, only the available flood maps accessible to the public could be checked. However, when we verified Sentinel 1 products on the corresponding dates used by the inundation mapping service for different locations, the found images were mainly covered. In total we got 70 images from various location and various time period.

In order to evaluate the effect of different methodological choices, we prepared three standard data sets for training and prediction:
D1: The standard training gives the model an insight into different types of scenarios where water can be detected and identified. We have used 90 percent of total dataset for training.
D2:The prediction set or validation set includes the rest 10 % of the training set.
D3: The test set include 5 image, which we set aside for the final evaluation.
The forecasts will be evaluated on future data (D4 - test set).

Neural Network Identified

Neural network used here is Convolutional neural Network. We have used U-Net Architecture for segmentation.

Architecture Used: U-Net

A widely used architecture of CNNs in semantic segmentation is the U-net architecture. Semantic segmentation with U-nets is mainly done by down-sampling of an image for feature extraction, then up-sampling using deconvolutional layers, to construct a pixel-wise classification labeled map. A deconvolution layer is simply a layer that where deconvolution operation is applied, or in other words, the transpose of a convolution operation is applied. The U-net architecture consists of an encoder joined to a symmetrical decoder.[20]

The encoder is a convolutional network and its task is to wrap the spatial dimension of the input image into a lot of meaningful and significant features. For this purpose, the encoder applies convolution operations followed by max-pooling or down-sampling operation to encode the input image into a matrix of feature representations at different levels. The extracted features from the input image become more abstract and subsequently have more semantic information. CNN used for change detection need to preserve the spatial information to predict pixel-wise semantic changes. Thus, a decoder is essential for the change detection architecture.

The decoder is the deconvolutional network and it serves to extract these sets of features to construct a segmented or labeled representation of the input image. The decoder applies up-sampling operation, concatenation, then convolution operations. The concatenation in U-net is done by copying the features from layers in the encoder and concatenating them with the corresponding layer (of the same level) in the decoder part. This method ensures the preservation of pixel positions after convolution and max pooling operations. Consequently, U-nets can be used for change detection between two images from different time periods.

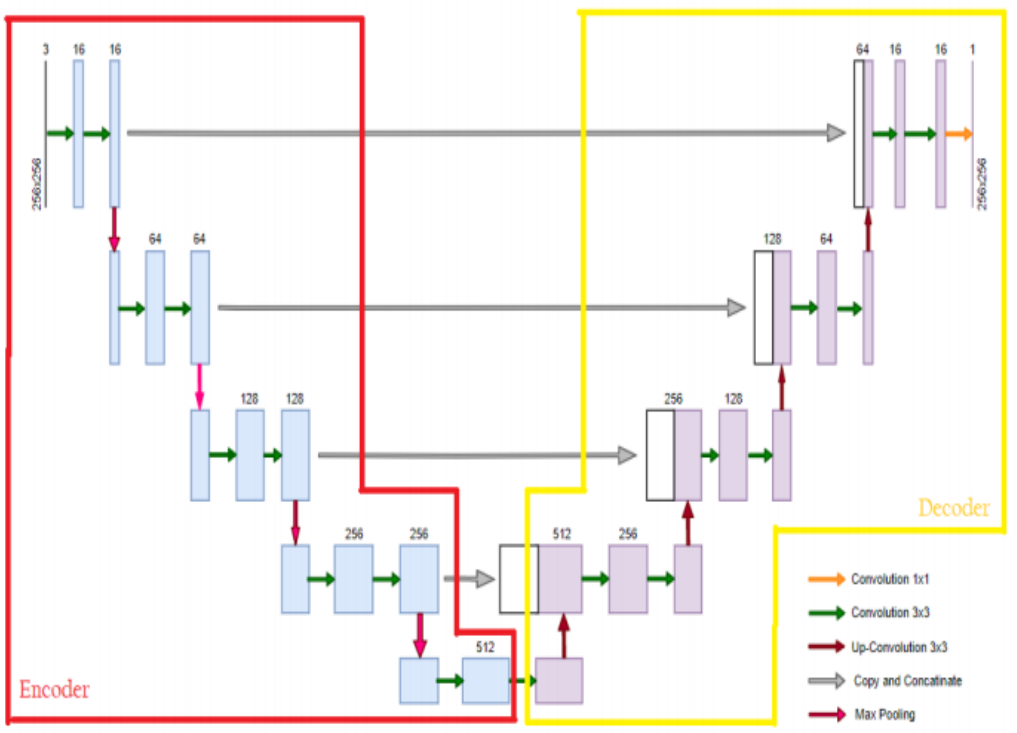

Figure 5.4: A U-net Architecture

The above figure shows the base U-net architecture for flood mapping. The input and output dimensionality is 256x256x3. All convolution operations in the model have a kernel size of 3x3 and zero padding. Each max-pooling layer and concatenation layer was followed by a dropout layer. All deconvolution operations in the model use a kernel size of 3x3, a stride of 2, and padding of 0.

The encoder of the architecture has five levels. The first four levels have two convolution operations applied to the input image, followed by a max-pooling operation with a kernel size of 2. At the bridge, or fifth level, two convolution operations are applied, but no max-pooling operations are used. The decoder has also four levels, each consisting of one deconvolution operation, followed by two convolution operations. The final level of the decoder had a third convolution operation to get a one channel output as the flood map. The number of kernels at the first level of the encode is 16, and the number is doubled at each subsequent level. The number of kernels at the fifth level is 256. The opposite operation was done for the decoder. The number of filters was halved until it reached 16 again before the final convolution was applied. The output is a flood map of size 256x256x1.

5.6 Data Flow Diagrams

5.6.1 Data Flow Diagram- Level 0

Figure 5.5: DFD- Level 0

5.6.2 Data Flow Diagram- Level 1

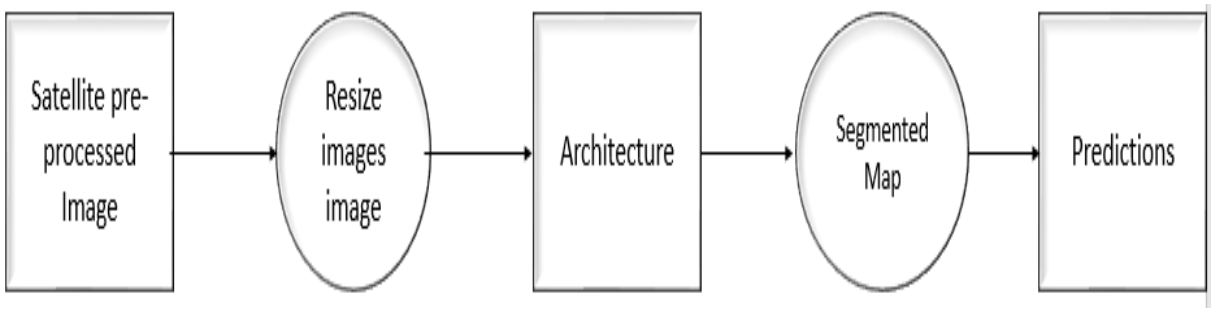

Figure 5.6: DFD- Level 1

5.7 Architecture

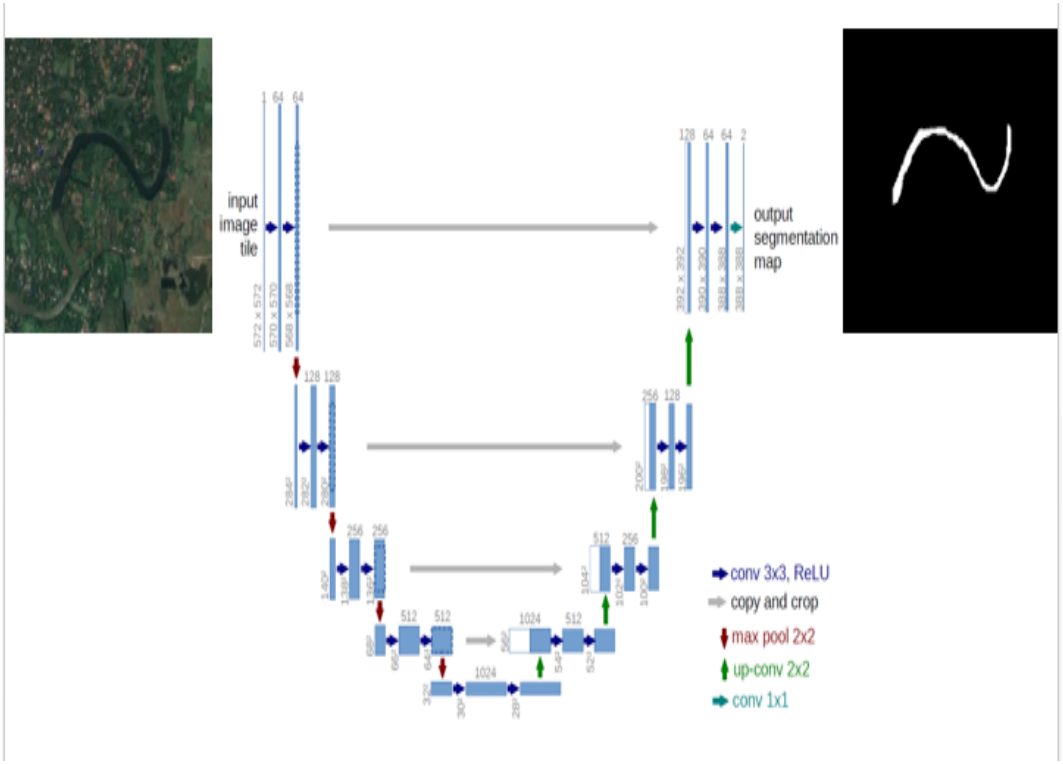

Figure 5.7: Architecture diagram

5.8 IMPLEMENTATION

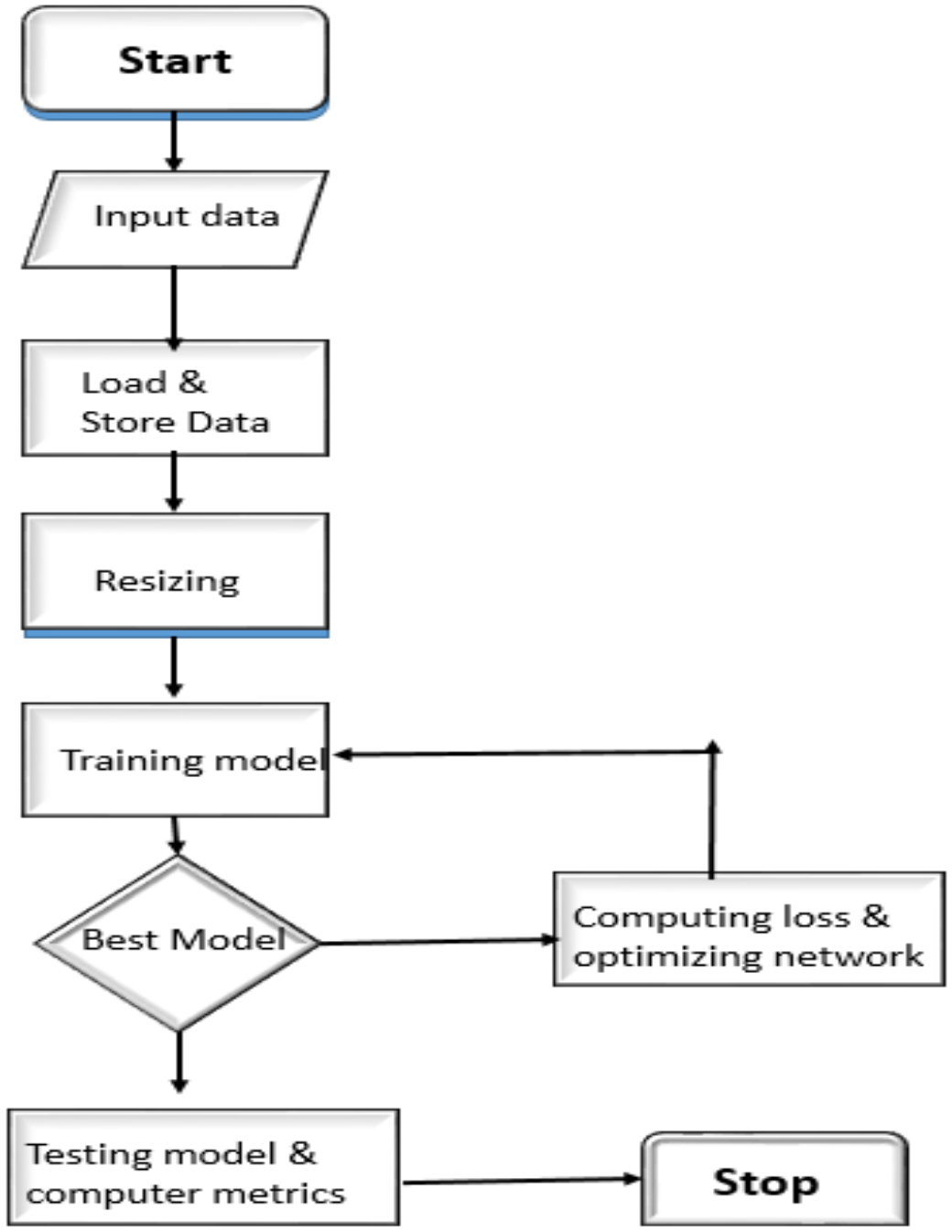

Figure 5.8: Implementation Steps

5.8.1 Data collection

Data has been acquired from Copernicus open acess data Hub.In total we have collected approximately 70 images from varying locations and various time period. We have got sentinel 1A/b product image. The image that we get from site are pre processed images. I.e we get RGB image from the open access hub. so there is no further need to process it. We have stored 5 images for test set. Remaining images 90 percent are used for training and remaining 10 percent for validation.

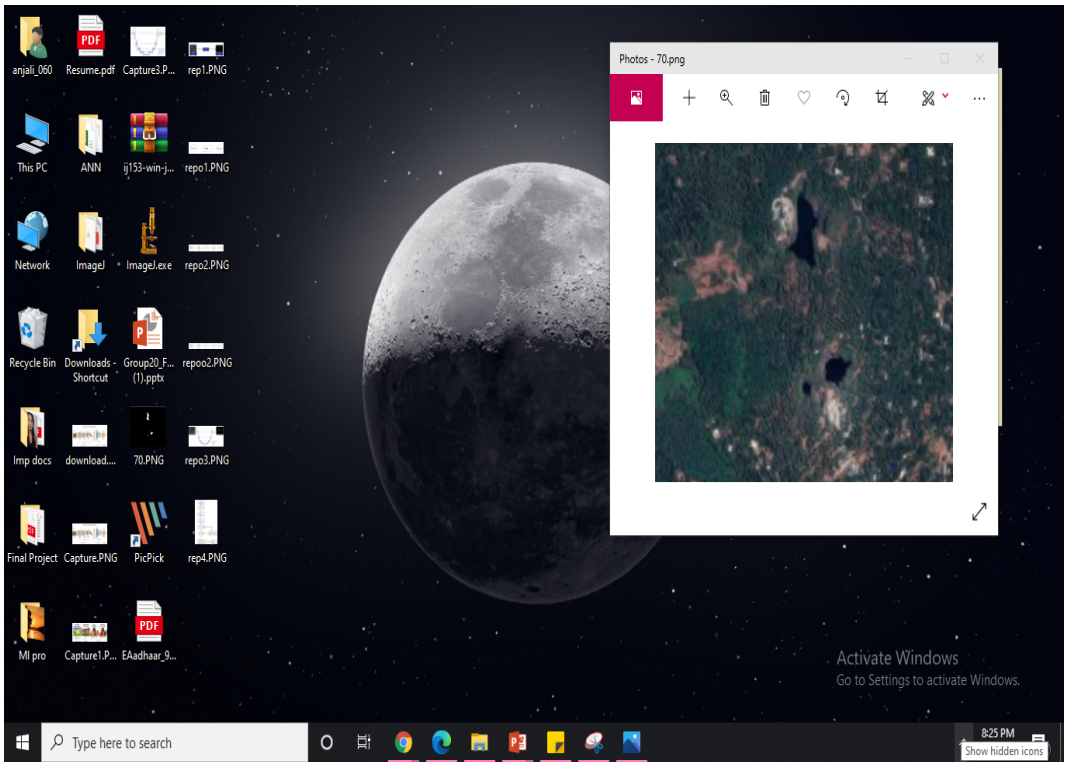

Figure 5.9: Satellite image collected from COPERNICUS OPEN ACCESS HUB

5.8.2 Data Pereperation

Since we are using Supervised neural network we have to train input as well as output with network.so we are expecting segmented map as output. For creation of masked image we are using IMAGEJ software. So in total we have 140 images in dataset including masked images.

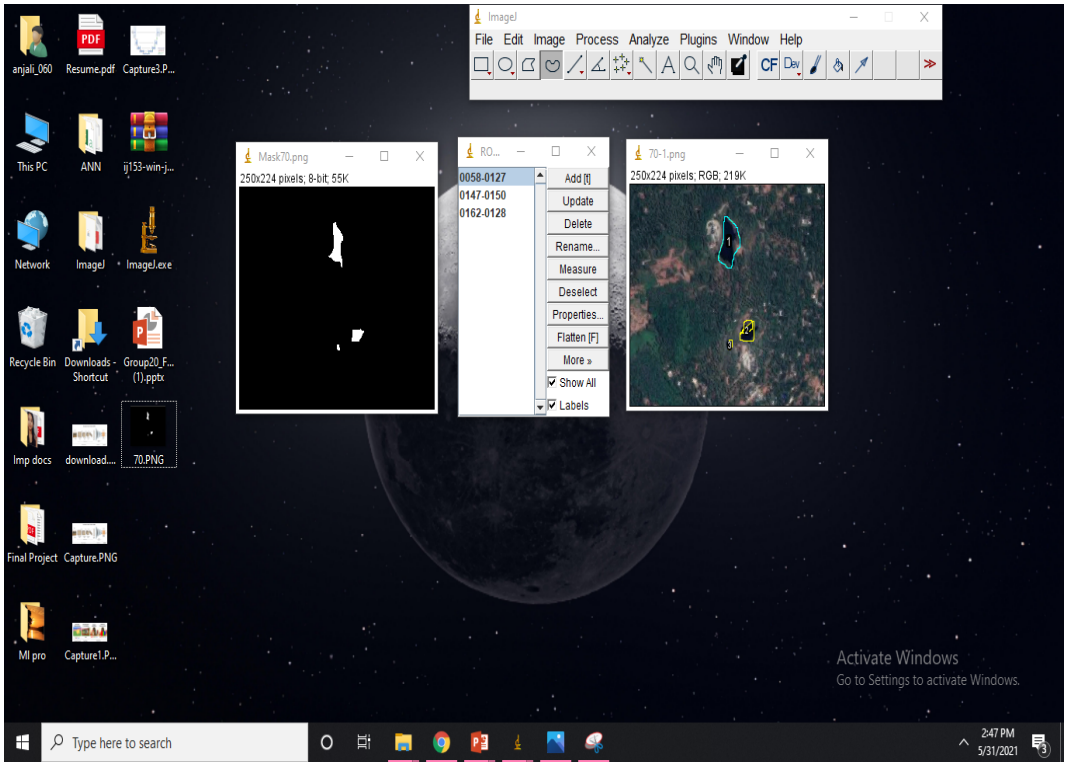

Figure 5.10: masked image creation

5.8.3 Jupyter Environment

Jupyter notebook is ideal for everything from improving your Python coding skills to working with deep learning libraries, like PyTorch, Keras, TensorFlow, and OpenCV. One can create notebooks, upload notebooks, store notebooks, share notebooks, mount your Google Drive. One can import most of their favorite directories, upload personal Jupyter Notebooks, upload notebooks directly from GitHub, upload Kaggle files, download your notebooks, and do just about everything else.

5.8.4 Import Libraries

1. numpy
2. Tensorflow
3. tqdm
4. scikit-learn
5. matplotlib

5.8.5 Steps of Implementation/Code overview

Step 1: Import all the required modules.

Step 2: Declare or Pre-define our dataset dimensions.

IMG WIDTH = 128
IMG HEIGHT = 128
IMG CHANNELS = 3

Step 3: Load the dataset and store the datset in to a matrix

```
In [33]: TRAIN_PATH = 'C:/U-Net/stage1_train/'
         TEST_PATH = 'C:/U-Net/stage1_test/'

In [30]: train_ids = next(os.walk(TRAIN_PATH))[1]
         test_ids = next(os.walk(TEST_PATH))[1]
```

Figure 5.11: load data

```
In [35]: test_ids

Out[35]: ['water_body_1',
          'water_body_2',
          'water_body_3',
          'water_body_4',
          'water_body_5']

In [36]: X_train = np.zeros((len(train_ids), IMG_HEIGHT, IMG_WIDTH, IMG_CHANNELS), dtype=np.uint8)
         Y_train = np.zeros((len(train_ids), IMG_HEIGHT, IMG_WIDTH, 1), dtype=np.bool)
```

Figure 5.12: Store images to matrix

Step 3: Resize the images to pre defined value

```
In [37]: print('Resizing training images and masks')
         for n, id_ in tqdm(enumerate(train_ids), total=len(train_ids)):
             path = TRAIN_PATH + id_
             img = imread(path + '/images/' + id_ + '.png')[:,:,:IMG_CHANNELS]
             img = resize(img, (IMG_HEIGHT, IMG_WIDTH), mode='constant', preserve_range=True)
             X_train[n] = img  #Fill empty X_train with values from img
             mask = np.zeros((IMG_HEIGHT, IMG_WIDTH, 1), dtype=np.bool)
             for mask_file in next(os.walk(path + '/masks/'))[2]:
                 mask_ = imread(path + '/masks/' + mask_file)
                 mask_ = np.expand_dims(resize(mask_, (IMG_HEIGHT, IMG_WIDTH), mode='constant',
                                               preserve_range=True), axis=-1)
                 mask = np.maximum(mask, mask_)

             Y_train[n] = mask
```

Figure 5.13: Resize Images

Step 4: Built Model

```
#Build the model
inputs = tf.keras.layers.Input((IMG_HEIGHT, IMG_WIDTH, IMG_CHANNELS))
s = tf.keras.layers.Lambda(lambda x: x / 255)(inputs)
```

```
Backbones: ['resnet18', 'resnet34', 'resnet50', 'resnet101', 'resnet152', 'seresnet18', 'seresnet34', 'seresnet50', 'seresnet101', 'seresnet152', 'seresnext50',
'seresnext101', 'senet154', 'resnext50', 'resnext101', 'vgg16', 'vgg19', 'densenet121', 'densenet169', 'densenet201', 'inceptionresnetv2', 'inceptionv3',
'mobilenet', 'mobilenetv2', 'efficientnetb0', 'efficientnetb1', 'efficientnetb2', 'efficientnetb3', 'efficientnetb4', 'efficientnetb5', 'efficientnetb6', 'efficientnetb7']
```

Figure 5.14: Built the model

Step 5: Output layer and model check point

```
In [50]: outputs = tf.keras.layers.Conv2D(1, (1, 1), activation='sigmoid')(c9)

In [51]: model = tf.keras.Model(inputs=[inputs], outputs=[outputs])
         model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
         model.summary()
```

Model: "model"

Layer (type)	Output Shape	Param #	Connected to
input_1 (InputLayer)	[(None, 128, 128, 3)	0	
lambda (Lambda)	(None, 128, 128, 3)	0	input_1[0][0]
conv2d_19 (Conv2D)	(None, 128, 128, 16)	448	lambda[0][0]
dropout_9 (Dropout)	(None, 128, 128, 16)	0	conv2d_19[0][0]
conv2d_20 (Conv2D)	(None, 128, 128, 16)	2320	dropout_9[0][0]
max_pooling2d_4 (MaxPooling2D)	(None, 64, 64, 16)	0	conv2d_20[0][0]
conv2d_21 (Conv2D)	(None, 64, 64, 32)	4640	max_pooling2d_4[0][0]
dropout_10 (Dropout)	(None, 64, 64, 32)	0	conv2d_21[0][0]

Figure 5.15: Output Layer and model check point

Step 6: Architecture

In [48]:
```
#Contraction path/Encoding
c1 = tf.keras.layers.Conv2D(16, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(s)
c1 = tf.keras.layers.Dropout(0.1)(c1)
c1 = tf.keras.layers.Conv2D(16, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c1)
p1 = tf.keras.layers.MaxPooling2D((2, 2))(c1)

c2 = tf.keras.layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(p1)
c2 = tf.keras.layers.Dropout(0.1)(c2)
c2 = tf.keras.layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c2)
p2 = tf.keras.layers.MaxPooling2D((2, 2))(c2)

c3 = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(p2)
c3 = tf.keras.layers.Dropout(0.2)(c3)
c3 = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c3)
p3 = tf.keras.layers.MaxPooling2D((2, 2))(c3)

c4 = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(p3)
c4 = tf.keras.layers.Dropout(0.2)(c4)
c4 = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c4)
p4 = tf.keras.layers.MaxPooling2D(pool_size=(2, 2))(c4)

c5 = tf.keras.layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(p4)
c5 = tf.keras.layers.Dropout(0.3)(c5)
c5 = tf.keras.layers.Conv2D(256, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c5)
```

Figure 5.16: U-Net Encoder

In [49]:
```
#Expansive path/Decoder
u6 = tf.keras.layers.Conv2DTranspose(128, (2, 2), strides=(2, 2), padding='same')(c5)
u6 = tf.keras.layers.concatenate([u6, c4])
c6 = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(u6)
c6 = tf.keras.layers.Dropout(0.2)(c6)
c6 = tf.keras.layers.Conv2D(128, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c6)

u7 = tf.keras.layers.Conv2DTranspose(64, (2, 2), strides=(2, 2), padding='same')(c6)
u7 = tf.keras.layers.concatenate([u7, c3])
c7 = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(u7)
c7 = tf.keras.layers.Dropout(0.2)(c7)
c7 = tf.keras.layers.Conv2D(64, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c7)

u8 = tf.keras.layers.Conv2DTranspose(32, (2, 2), strides=(2, 2), padding='same')(c7)
u8 = tf.keras.layers.concatenate([u8, c2])
c8 = tf.keras.layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(u8)
c8 = tf.keras.layers.Dropout(0.1)(c8)
c8 = tf.keras.layers.Conv2D(32, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c8)

u9 = tf.keras.layers.Conv2DTranspose(16, (2, 2), strides=(2, 2), padding='same')(c8)
u9 = tf.keras.layers.concatenate([u9, c1], axis=3)
c9 = tf.keras.layers.Conv2D(16, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(u9)
c9 = tf.keras.layers.Dropout(0.1)(c9)
c9 = tf.keras.layers.Conv2D(16, (3, 3), activation='relu', kernel_initializer='he_normal', padding='same')(c9)
```

Figure 5.17: U-Net deccoder

CHAPTER 6

RESULTS

The effectiveness of the model in detecting water areas using their SAR images are examined. To train and test the networks, upyter botebook platform is used.The input data images have a common size which is 128x128x3. The data was split into training, validation and testing sets. The optimization of the parameters is done by using the binary cross-entropy function to compare the predicted values with the ground truth. The parameters are then optimized using Adam technique. Twe have given 2o epochs. But we got the best model or minimum validation loss during 12th epoch.we have given patience level as 2.we got approx 79 percent validation accuracy

1.Model Summary

```
In [51]: model = tf.keras.Model(inputs=[inputs], outputs=[outputs])
         model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
         model.summary()
```

```
Model: "model"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            [(None, 128, 128, 3) 0
__________________________________________________________________________________________________
lambda (Lambda)                 (None, 128, 128, 3)  0           input_1[0][0]
__________________________________________________________________________________________________
conv2d_19 (Conv2D)              (None, 128, 128, 16) 448         lambda[0][0]
__________________________________________________________________________________________________
dropout_9 (Dropout)             (None, 128, 128, 16) 0           conv2d_19[0][0]
__________________________________________________________________________________________________
conv2d_20 (Conv2D)              (None, 128, 128, 16) 2320        dropout_9[0][0]
__________________________________________________________________________________________________
max_pooling2d_4 (MaxPooling2D)  (None, 64, 64, 16)   0           conv2d_20[0][0]
__________________________________________________________________________________________________
conv2d_21 (Conv2D)              (None, 64, 64, 32)   4640        max_pooling2d_4[0][0]
__________________________________________________________________________________________________
dropout_10 (Dropout)            (None, 64, 64, 32)   0           conv2d_21[0][0]
```

Figure 6.1: model Summary

Here we can see the details regarded to each layer,i.e no of parameters ,bias,weights etc.

2. Random test on Validation Set

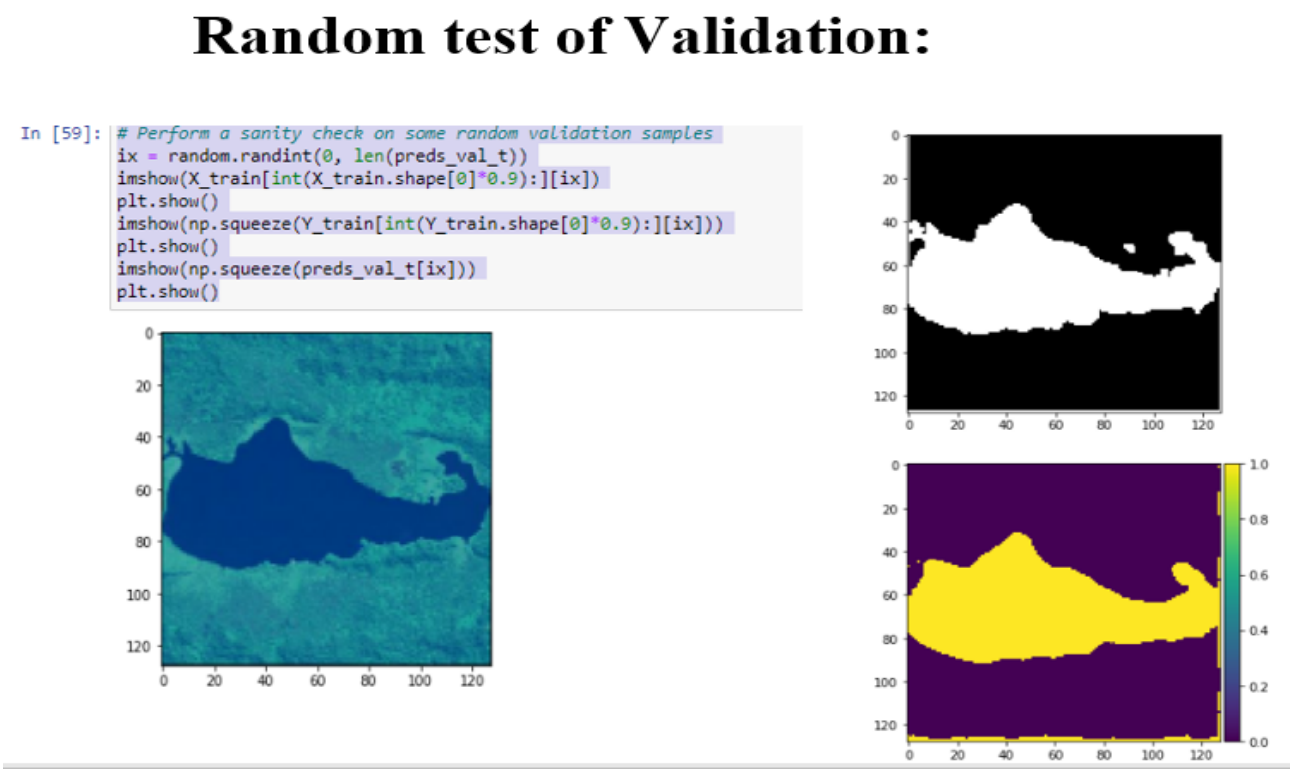

Figure 6.2: Validation set testing

3. Prediction

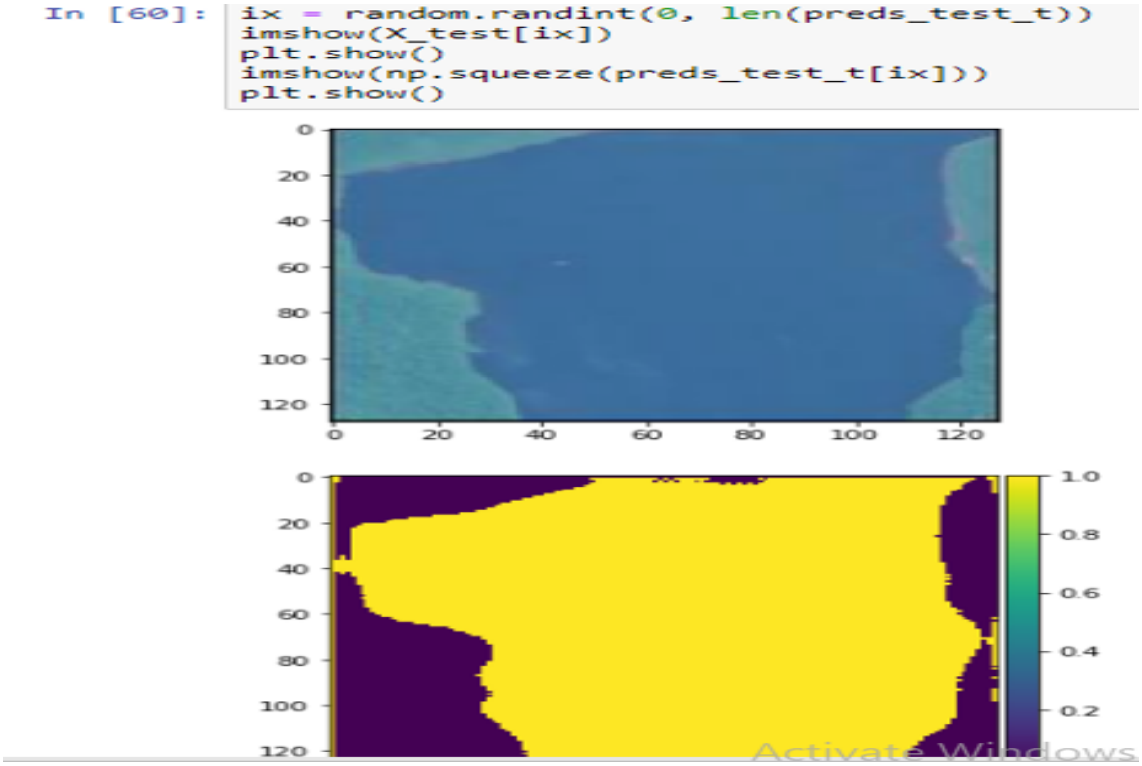

Figure 6.3: prediction

Here we can verify weather our system is working properly or not providing some rant test images from test data.

4.Loss Graph

```
In [64]: loss = history.history['loss']
         val_loss = history.history['val_loss']
         epochs = range(1,len(loss)+1)
         plt.plot(epochs, loss,'y',label='Training loss')
         plt.plot(epochs, val_loss,'r',label='Validation loss')
         plt.title('Training and Validation loss')
         plt.xlabel('Epochs')
         plt.ylabel('Loss')
         plt.legend()
         plt.show()
```

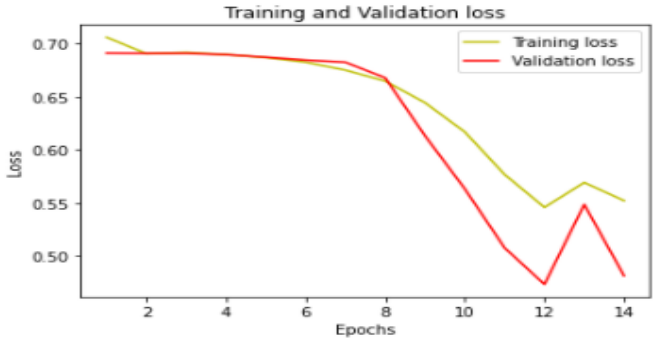

Figure 6.4: Validation set testing

5. Accuracy Graph

```
In [66]: acc = history.history['accuracy']
         val_acc = history.history['val_accuracy']
         epochs = range(1,len(loss)+1)
         plt.plot(epochs, acc,'y',label='Training acc')
         plt.plot(epochs, val_acc,'r',label='Validation acc')
         plt.title('Training and Validation loss')
         plt.xlabel('Epochs')
         plt.ylabel('Accuracy')
         plt.legend()
         plt.show()
```

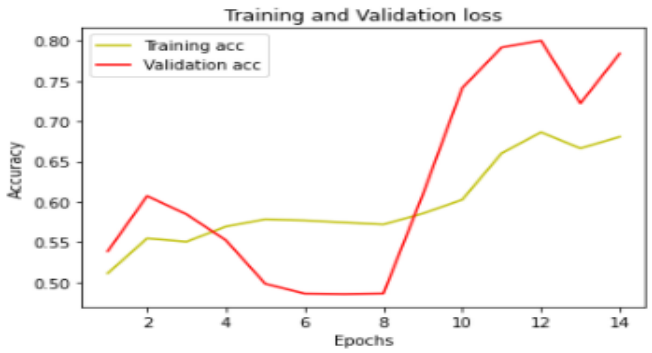

Figure 6.5: Validation set testing

6. Model check point or best model

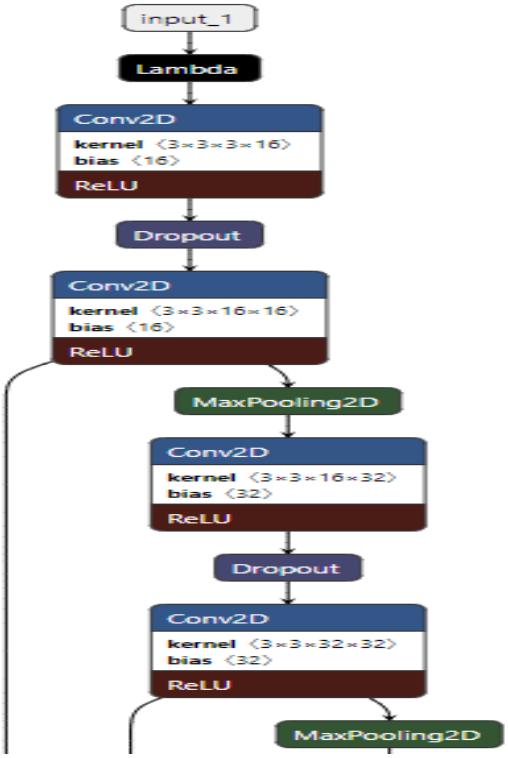

Figure 6.6: Validation set testing

This is the steps or layers until 12th epoch present in our neural network.

CHAPTER 7
CONCLUSION AND FUTURE WORKS

Deep learning techniques have been widely used in many applications such as health-care, big data analysis, etc. We use these deep learning techniques to detect the presence of water , thus to overcome the limitations of the existing system and provide more precise and efficient solution to tackle the issues caused by water like floods and to monitor water dynamics.

We have outlined the design of the proposed project, which aims to identify algorithms and features that can best detect and identify the water areass. U-net architecture is used o detect the presence of water . The purpose is to make costless and fast algorithms for water detection using different satellite sensor modalities. In the future, we are planning do develop a model for flood detection, which Send alert to user if risk of flood detected in location specified by user using real time images, also to identify landslides.

REFERENCES

[1] Sara El Amrani Abou El Assad. Flood detection with a deep learning approach using optical and sar satellite data. 2019.

[2] A Bartsch, AM Trofaier, G Hayman, D Sabel, S Schlaffer, DB Clark, and E Blyth. Detection of open water dynamics with envisat asar in support of land surface modelling at high latitudes. *Biogeosciences*, 9(2):703–714, 2012.

[3] Han Cao, Hong Zhang, Chao Wang, and Bo Zhang. Operational flood detection using sentinel-1 sar data over large areas. *Water*, 11(4):786, 2019.

[4] Francisco Carreño Conde and María De Mata Muñoz. Flood monitoring based on the study of sentinel-1 sar images: The ebro river case study. *Water*, 11(12):2454, 2019.

[5] Klemen Čotar, Krištof Oštir, and Žiga Kokalj. Radar satellite imagery and automatic detection of water bodies. *Geodetski glasnik*, 50(47):5–15, 2016.

[6] Elham Goumehei, V Tolpekin, A Stein, and Wanglin Yan. Surface water body detection in polarimetric sar data using contextual complex wishart classification. *Water resources research*, 55(8):7047–7059, 2019.

[7] Chang Huang, Yun Chen, Shiqiang Zhang, and Jianping Wu. Detecting, extracting, and monitoring surface water from space using optical sensors: A review. *Reviews of Geophysics*, 56(2):333–360, 2018.

[8] Zachary Kreiser, Brian Killough, and Syed R Rizvi. Water across synthetic aperture radar data (wasard): Sar water body classification for the open data cube. In *IGARSS 2018-2018 IEEE International Geoscience and Remote Sensing Symposium*, pages 437–440. IEEE, 2018.

[9] Sanmei Li, Donglian Sun, Mitchell D Goldberg, Bill Sjoberg, David Santek, Jay P Hoffman, Mike DeWeese, Pedro Restrepo, Scott Lindsey, and Eric Holloway. Automatic near real-time flood detection using suomi-npp/viirs data. *Remote sensing of environment*, 204:672–689, 2018.

[10] Alejandra A López-Caloca, Felipe Omar Tapia-Silva, Fernando López, Henao Pilar, Aymara O Ramirez Gonzalez, and Guadalupe Rivera. Analyzing short term spatial and temporal dynamics of water presence at a basin-scale in mexico using sar data. *GIScience & Remote Sensing*, 57(7):985–1004, 2020.

[11] Laura Lopez-Fuentes, Joost van de Weijer, Marc Bolanos, and Harald Skinnemoen. Multimodal deep learning approach for flood detection. *MediaEval*, 17:13–15, 2017.

[12] David C Mason, Rainer Speck, Bernard Devereux, Guy J-P Schumann, Jeffrey C Neal, and Paul D Bates. Flood detection in urban areas using terrasar-x. *IEEE Transactions on Geoscience and Remote Sensing*, 48(2):882–894, 2009.

[13] Mustafa Mousa, Xiangliang Zhang, and Christian Claudel. Flash flood detection in urban cities using ultrasonic and infrared sensors. *IEEE Sensors Journal*, 16(19):7204–7216, 2016.

[14] Edoardo Nemni, Joseph Bullock, Samir Belabbes, and Lars Bromley. Fully convolutional neural network for rapid flood segmentation in synthetic aperture radar imagery. *Remote Sensing*, 12(16):2532, 2020.

[15] Georgios Ovakoglou, Ines Cherif, Thomas K Alexandridis, Xanthoula-Eirini Pantazi, Afroditi-Alexandra Tamouridou, Dimitrios Moshou, Xanthi Tseni, Iason Raptis, Stella Kalaitzopoulou, and Spiros Mourelatos. Automatic detection of surface-water bodies from sentinel-1 images for effective mosquito larvae control. *Journal of Applied Remote Sensing*, 15(1):014507, 2021.

[16] Simon Richard Proud, Rasmus Fensholt, Laura Vang Rasmussen, and Inge Sandholt. Rapid response flood detection using the msg geostationary satellite. *International Journal of Applied Earth Observation and Geoinformation*, 13(4):536–544, 2011.

[17] Himanshu Rana and Nirvair Neeru. Water detection using satellite images obtained through remote sensing. *Adv. Comput. Sci. Technol*, 10:1923–1940, 2017.

[18] Stefan Schlaffer, Patrick Matgen, Markus Hollaus, and Wolfgang Wagner. Flood detection from multi-temporal sar data using harmonic analysis and change detection. *International Journal of Applied Earth Observation and Geoinformation*, 38:15–24, 2015.

[19] Sergii Skakun. A neural network approach to flood mapping using satellite imagery. *Computing and Informatics*, 29(6):1013–1024, 2012.

[20] Congcong Wu, Xuezhi Yang, and Jun Wang. Flood detection in sar images based on multi-depth flood detection convolutional neural network. In *2019 6th Asia-Pacific Conference on Synthetic Aperture Radar (APSAR)*, pages 1–6. IEEE, 2019.