A Survey on SAR Imagery Utilizing Different Image Procsessing Techniques and Proposed System on Flood Water Mapping

Atharva Zagade[1], Dipam Patle[2], Priyanka Salunke[3], Vaibhav Bhapkar[4], Nuzhat. F. Shaikh[5]

Department of Computer Department, Modern Education Society's College of Engineering, Pune, India. 19, Late Prin. V.K. Joag Path, Wadia College Campus, Off, Bund Garden Rd, Pune, Maharashtra 411001

Abstract:

In a time of need and uncertainty for people, the government and humanitarian organizations must be able to accurately measure the flood extent in the near real-time to strengthen early warning systems and assess the risk. Ground measures are spatially limited, costly to maintain, and accurate only in special circumstances. The government and humanitarian organizations must be able to accurately measure the flood extent in the near real-time without expensive remote sensing systems. This is where high-precision satellite imagery can complement ground measurements by supplementing imagery with hydrological information. Rain and stream gauging stations, however, only measure the water height and have limitations by geographic placement and cost of maintenance. High- resolution satellite imagery has strengthened monitoring by providing data in otherwise inaccessible areas at frequent time intervals. In particular, sensors operating in the microwave portion of the spectrum can operate at low signal-to-noise ratios over a greater range of frequencies than sensors operating in other portions of the electromagnetic spectrum. This allows a single sensor array to provide greater bandwidth and spectral resolution, which is important for distinguishing between water and other objects in images containing both water and other objects, particularly where subtle differences, in contrast, exist between water and other objects. High-resolution synthetic-aperture radar (SAR) imaging has strengthened monitoring systems by providing data also in inaccessible areas at frequent time intervals. It operates in the microwave band of electromagnetic spectrum, and is able to capture images through clouds, precipitation, vegetation, and smoke, making it beneficial for flood detection. A high-resolution synthetic-aperture radar (SAR) imaging system uses microwave frequencies to penetrate clouds and other interference that may hinder traditional communication systems. It creates extremely accurate images of the ground just below the surface, allowing for detailed monitoring at any time and from nearly any location. The Sentinel-1 mission comprises two satellites, Sentinel-1A and Sentinel-1B, performing C-band synthetic-aperture radar (SAR) imaging. The primary difference

between SAR systems and optical sensors is that SAR systems operate in the microwave band, where long waves can penetrate through clouds, vegetation, fog, rain showers, and snow.
Keywords: SAR Imagery, Machine Learning, Image Processing, Flood water mapping.
DOI: 10.24297/j.cims.2023.6.2

1. Introduction

Synthetic-aperture radars (SARs) are extensively employed to produce high-fidelity 2D and 3D replicas of scenes and objects. By leveraging advanced radar receiving wireless communication, SARs achieve superior spatial resolution compared to traditional pillar examination radars. Synthetic Aperture Radars (SARs) are a type of radar imaging technology that utilizes a synthetic aperture to achieve high resolution images of distant objects. This is achieved by combining radar pulses from a moving platform, such as an aircraft or satellite, to create a synthetic aperture that is much larger than the physical aperture of the radar antenna. This allows SARs to achieve high resolution images even at long ranges, making them ideal for applications such as remote sensing, surveillance, and mapping. Additionally, SARs have the ability to penetrate through clouds, smoke, and other atmospheric conditions that can impede optical imaging, making them useful in a wide range of environments. Other applications of SARs include monitoring of environmental changes, oil spills, and natural disasters, as well as providing valuable data for scientific research in fields such as oceanography and glaciology. With the advent of low-cost SARs, the potential for new applications in areas such as agriculture, forestry, and urban planning are being explored. Synthetic Aperture Radars (SARs) have the unique ability to have a large aperture for distant objects while maintaining consistent spatial resolutions at varying viewing distances. They have photographic and other imaging capabilities that are not affected by daytime or weather conditions, as well as the unique way that different terrains and structures react to radar frequencies. Despite this, there are many potential uses for this technology that have yet to be fully explored, especially for smaller scale applications, as the cost of SARs is becoming more affordable.

2. Related Work

Radar images can be enhanced using digital image processing techniques to improve their interpretability. A survey paper discusses these techniques, their advantages, and disadvantages for analyzing Synthetic Aperture Radar (SAR) images. Wiebke Aldenhoff proposed the ARTIST sea-ice algorithm, which utilizes multi-frequency data (L-band and C-band) to classify ice and water. The algorithm employs Artificial Neural Networks (ANN) for classifying unknown

information based on backscatter intensities. While outperforming traditional ice charts, the proposed method has lower accuracy compared to Radiometer data [1].

Satellite images provide efficient and consistent real-time mapping of flood extents over large areas. Synthetic Aperture Radar (SAR) data is particularly valuable due to its all-weather and day/night acquisition capabilities. However, mapping water bodies in SAR images is challenging due to various factors such as built environments, vegetation, and radar limitations. This paper introduces a Bayesian approach for creating probabilistic flood maps from SAR images, considering backscatter values and prior flood probabilities. The approach was evaluated using validation maps and showed promising results for flood mapping applications [2].

Creating a forecasting model for detecting oil, gas, and water layers involves several steps. The Support Vector Machine (SVM) algorithm, rooted in statistical learning theory, is commonly employed to build models using sample datasets. It is then evaluated using a separate testing dataset for reliability. The SVM model is also compared to a neural network model using the same datasets. However, SVM has limitations for flood mapping using Synthetic Aperture Radar (SAR) imagery. It requires a large amount of labeled data, which may be challenging to obtain for flood mapping. SVM is sensitive to the choice of kernel function and parameters and does not scale well with high-dimensional data, such as SAR imagery. Other models like CNNs and Random Forest may be more suitable for SAR- based flood mapping, despite their own limitations [3].

The paper focuses on flood area detection using Sentinel-1 SAR images and Res-U-net-based methods. Four models, including U-net and Res-U-net with Experiment and Experiment2, were trained and tested on different images. Pre-processing involved log10 transformation and normalization. Results indicate successful flood area detection, with Res-U-net showing better visual results. However, Res-U-net requires significant computational resources and may be challenging to interpret. Acquiring labeled data for training is time-consuming, and overfitting and vulnerability to adversarial attacks are potential concerns [4].

Detecting changes in land cover after a disaster, such as cyclonic floods, involves using the K-means clustering algorithm on pre- and post-event images. The algorithm groups similar pixels into clusters, enabling identification of land cover changes. A study applied this methodology using Google Earth Engine and Sentinel-1 imagery for the Amphan disaster in West Bengal,

India. The K-means algorithm is used for clustering and change detection. However, determining the number of clusters is challenging, and the algorithm's sensitivity to initial centroids can lead to varying results. It also assumes uniform variance and shape, which may not align with real floodwater mapping scenarios. Spatial and temporal relationships between data points are not considered, potentially resulting in misclassification [5].

Hyperspectral and SAR imagery utilize multiple spectral bands to gather detailed information about an area. SAR imagery, despite being grayscale, can be used for flood mapping by

analyzing multiple spectral bands and validating with topography attributes. SAR offers high resolution, all-weather capability, and repeat coverage for monitoring changes over time. It is cost-effective and provides accurate results. However, limitations include limited wavelength bands, radar shadows, complex interpretation, and computational intensity for processing [6].

Deep Learning (DL) models like MLPs, RNNs, and CNNs have been widely used for flood modelling. Studies have compared their performance, showing that RNNs, 1D CNNs, and 3D CNNs generally outperform MLPs. CNNs are favoured due to their inductive bias capabilities. Factors to consider when comparing DL models include accuracy, number of parameters, and data requirements. Higher parameter counts can improve performance but may lead to over-fitting. Model performance can also be influenced by data availability, making fair comparisons between models with different data budgets challenging.

Supervised learning encompasses two primary problem types: regression and classification. Regression tasks involve predicting continuous target values, such as water depth. On the other hand, classification tasks involve predicting discrete target values, for example, distinguishing between flooded and non- flooded areas. Different metrics are used to evaluate the performance of a model depending on the task. When dealing with regression problems, several commonly used metrics are employed, including the Root Mean Squared Error (RMSE), Coefficient of Determination (R2), and Mean Absolute Error (MAE). These metrics strive to minimize their values, approaching zero, as they indicate better performance. However, in certain cases, MAE is favored over RMSE due to the latter being sensitive to extreme outliers, which can disproportionately affect its value. However, when comparing the performance of different models, it is important to consider the domain and any outliers that may be present. When working with classification problems, commonly utilized metrics include accuracy, recall,

and precision. These metrics serve as means to assess the performance of a binary classification model, where one class is designated as positive (e.g., flooded areas) and the other class as negative (e.g., non-flooded areas). However, these metrics can be problematic when dealing with imbalanced datasets, where some classes are more represented than others. In such cases, it may be better to use metrics that account for false negatives and false positives, such as the F1 score or the Kappa score. In assessing model performance, the Receiver Operating Characteristic (ROC) curve is a valuable tool. It enables the evaluation of a model's performance across various classification thresholds. Additionally, the Area Under the ROC Curve (AUC) is commonly employed as a summary metric, providing a single value to represent the overall performance of the ROC curve.

3. Proposed System

The U-Net architecture is a convolutional neural network widely employed for image segmentation in computer vision. Introduced in 2015 by Ronneberger. it comprises a contracting path that encodes the input image into a smaller representation and an expansive path that decodes it back to the original size. The contracting path employs convolutional and max pooling layers to reduce spatial resolution while increasing depth. Conversely, the expansive path employs transposed convolutional layers for up-sampling. A key innovation of U- Net is the integration of skip connections, enabling the expansive

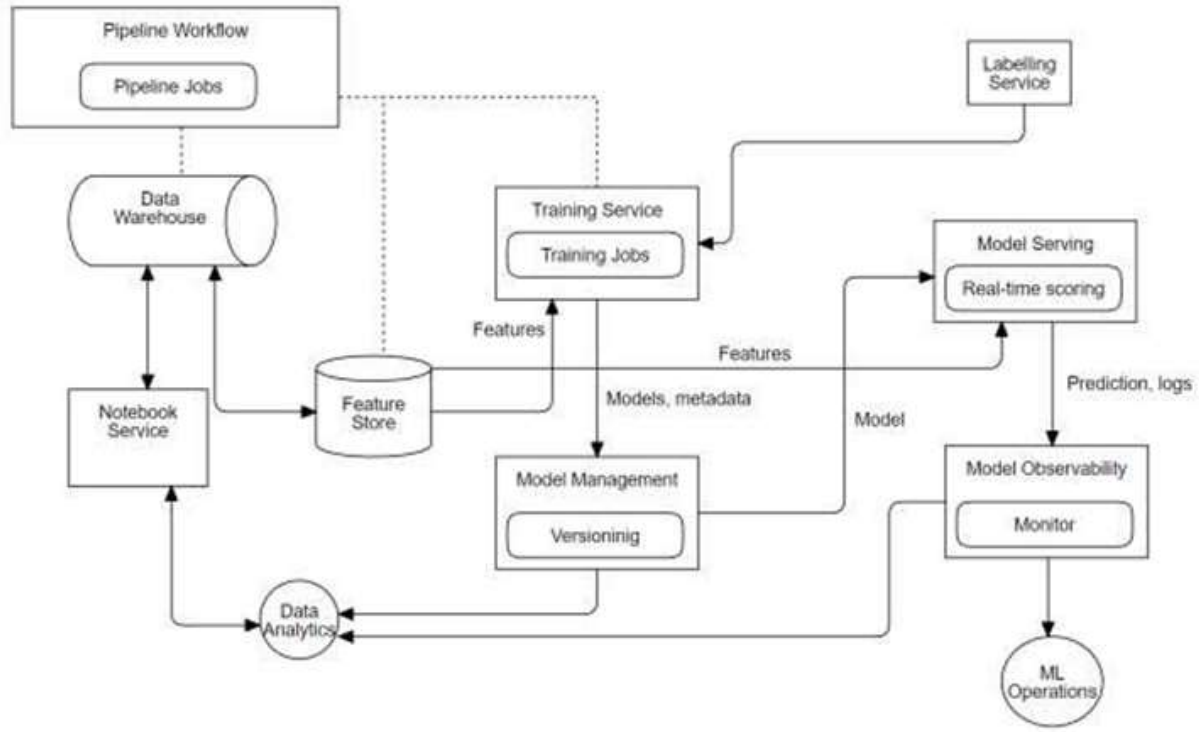

Figure 1.1 System Architecture

path to leverage information from earlier contracting path layers, thereby preserving fine-grained details in the segmentation output. U-Net finds extensive application in various biomedical image segmentation tasks, including cell, nucleus, and tissue segmentation, but has also been utilized beyond the biomedical domain for general image segmentation purposes.

The proposed methodology encompasses the following sequential steps:

Step 1: Data Collection and Pre-processing

The first step in flood water mapping using SAR imagery is to collect the relevant data. This includes acquiring SAR images of the area of interest before and after the flood event. These images are then pre-processed to remove noise and other artifacts using standard pre-processing techniques such as speckle filtering, radiometric calibration, and geometric correction. The pre- processed images are then stacked to create a multi-temporal dataset that will be used for analysis.

Step 2: Ground Truth Generation

The second step is to generate ground truth data that will be used to train and validate the deep learning model. This involves manually delineating the flood extent on the SAR images using an established methodology such as visual interpretation or thresholding. The ground truth data should be representative of the range of conditions present in the study area, including different water depths and types of flood events.

Step 3: Model Training

The third step is to train the deep learning model using the multi- temporal SAR dataset and the ground truth data. In this study, the U-Net architecture is used due to its effectiveness in semantic segmentation tasks. The model is trained using a combination of cross-entropy loss and dice loss to optimize the network for accurate flood water mapping.

Step 4: Model Validation

The fourth step is to validate the trained model using an independent dataset. This dataset should contain SAR images and ground truth data from a different flood event or a different

location than the training dataset. The model's performance is evaluated using standard metrics such as overall accuracy, kappa coefficient, and F1 score.

Step 5: Flood Water Mapping

The final step is to apply the trained model to map flood water extent using SAR images of the area of interest. This involves inputting the pre-processed SAR images into the trained U-Net model to generate a flood water map. The output map can be visualized and analysed to gain insights into the extent and severity of the flood event.

4. Implementation

Dataset Handler Module: The data pre-processing module plays a crucial role in the paper, as it handles the transformation of raw data into a clean and usable format that can be readily fed into the deep learning model. The dataset obtained through web scraping or other means may contain inconsistencies, noise, or missing values, making it unsuitable for direct interpretation by the deep learning system. To address this, the data pre- processing module aims to apply various operations and feature extraction techniques to convert the raw data into a proper format. These operations are identified based on a thorough literature review, which helps determine the most effective approaches for processing the specific type of data used in the paper. By implementing the identified operations and feature extraction techniques, the data pre-processing module performs tasks such as data cleaning, data normalization or scaling, handling missing values, removing outliers, and transforming the data into a suitable representation for the deep learning model. The module ensures that the data is properly formatted, standardized, and prepared to maximize the deep learning model's ability to interpret and learn from the information it contains. By performing these pre-processing steps, the module enhances the quality and usability of the dataset, making it more compatible with the requirements of the deep learning model. It is important to note that the specific operations and feature extraction techniques applied in the data pre-processing module will depend on the nature of the raw data and the specific requirements of the deep learning model. The module's implementation is based on careful consideration of the literature and selecting the most appropriate methods to optimize the model's performance.

Tech Stack for the module: NumPy, Pandas, Rasterio, pre- processing Neural Network Module: After the dataset has been processed in the data pre-processing module, it can be passed on to the model development module. In this module, the deep learning system is developed based

on the proposed architecture, which is derived from a thorough literature survey. Through the literature survey, it has been determined that using SAR images can lead to better results in the context of the paper. SAR images provide unique advantages such as all-weather capability and cloud-penetration, making them suitable for accurate flood water mapping. The research conducted suggests that the U-Net architecture is the most accurate deep learning system for image processing in this specific paper. Therefore, the U-Net architecture will be employed in the development of the deep learning system. The tech stack selected for this module is U- Net, which is a widely recognized and utilized architecture for tasks involving image segmentation and semantic segmentation. Its effectiveness in accurately capturing intricate patterns and details in images makes it a suitable choice for the deep learning system in this paper. It is important to note that the specific implementation details of the U-Net architecture, such as the number of layers, filters, and activation functions, will be determined based on the requirements of the paper and the characteristics of the SAR images used. The goal is to develop a robust and accurate deep learning system that can effectively generate captions or perform other relevant tasks in the context of flood water mapping.

Tech Stack for the module: U-Net

Frontend Module: The deep learning model developed in the previous module cannot be directly interacted with by users. While it is theoretically possible, such direct interaction is not user-friendly and may present challenges. Therefore, the implementation of a user interface (UI) module becomes necessary to provide users with a seamless and hassle-free experience. The purpose of the UI module is to enable users to easily utilize the developed model without having to navigate complex technical processes. To achieve this, a user-friendly interface is built using ReactJS, a popular JavaScript library for building user interfaces. The UI, developed using ReactJS, allows users to interact with the deep learning system effortlessly. Users can input SAR images through the UI and receive the predicted output in a clear and intuitive manner. By implementing the UI module with ReactJS, users can conveniently provide the necessary input data and view the corresponding predictions generated by the deep learning model. ReactJS offers a robust and flexible framework for building interactive and responsive web interfaces, making it an ideal choice for creating a user-friendly experience in this paper. It' s important to note that the specific design and functionality of the UI module will be tailored to the requirements of the paper and the desired user experience. The goal is to develop an intuitive and efficient user interface that facilitates the interaction between users and the deep learning system, enhancing accessibility and usability.

Tech Stack for the module: ReactJS

Integration Model: Modules 1 and 2, which involve data pre- processing and model development, can be considered as the backend components of the system. On the other hand, Module 4, which focuses on the user interface, serves as the frontend. However, direct interaction between the frontend and the ML model backend is not feasible. To bridge the gap and enable communication between the frontend and backend, a separate module is required. This module acts as an intermediary responsible for facilitating the exchange of information between the frontend and the ML model backend. The primary function of this module is to receive inputs from the frontend, pass them to the deep learning model backend for processing, and then relay the processed output back to the frontend. Since the ML model backend is primarily implemented in Python, Flask emerges as a suitable choice for this intermediary module. Flask is a popular web framework in Python that enables the creation of web applications and APIs. It provides the necessary tools and functionalities to handle HTTP requests and responses, making it well-suited for building the communication layer between the frontend and the ML model backend. By utilizing Flask, this module establishes a seamless connection between the frontend and backend components of the system. It receives user inputs through HTTP requests from the frontend, forwards them to the ML model backend for processing, retrieves the processed results, and sends them back to the frontend for display or further actions. It is important to note that the Flask module acts as a crucial interface facilitating communication and data flow between the frontend and the ML model backend. Its implementation ensures effective coordination and integration of the various system components, providing a smooth user experience and enabling the utilization of the ML model's capabilities in a user-friendly manner.

Tech Stack for the module: Flask

Tools and Technologies Used:

Jupyter Notebook: Jupyter Notebook is an interactive web- based development environment that allows users to write and execute Python code in real-time. It provides a powerful platform for data analysis, scientific research, and education. With Jupyter Notebook, you can easily write and execute Python code in cells, which can be edited and run independently or together as a complete script. Jupyter Notebook also allows you to create visualizations, write documentation, and share your work with others. Overall, Jupyter Notebook is a versatile and essential tool for Python programming and data analysis.

NumPy: NumPy is a Python library used for numerical computing. It provides fast and efficient mathematical operations on arrays, matrices, and vectors. It is extensively used in scientific computing, data analysis, machine learning, and other areas where numerical computations are required. NumPy is designed to work efficiently with large multi-dimensional arrays and provides a comprehensive set of mathematical functions for these arrays. Its key feature is vectorized operations, making it much faster than equivalent Python code. NumPy is often used in conjunction with other Python libraries such as Pandas, SciPy, and Matplotlib, and is an essential tool for scientific computing and data analysis in Python.

Pandas: Pandas is a widely adopted Python library renowned for its ability to handle data manipulation and analysis tasks. It offers intuitive data structures that facilitate efficient storage and manipulation of structured data, encompassing various types such as tabular, time-series, and heterogeneous data. Pandas is widely used in data analysis, scientific research, and other areas where data processing and analysis are required. It provides powerful tools for data cleaning, filtering, and transformation, as well as data aggregation and grouping. Pandas seamlessly integrates with other popular Python libraries like NumPy, Matplotlib, and Scikit-learn. This seamless integration enhances its capabilities and makes it an indispensable tool for any data analysis paper in Python. With its powerful features and versatility, Pandas is a fundamental asset in the field of data analysis.

Flask: Flask is a lightweight web framework developed in Python. It falls under the category of micro frameworks as it operates without imposing any specific tools or libraries. Unlike some frameworks, Flask does not come bundled with a database abstraction layer, form validation, or other components that are commonly provided by third-party libraries. Nevertheless, Flask offers extensibility through various extensions, allowing developers to seamlessly integrate additional features into their applications. These extensions encompass object-relational mappers, form validation, upload handling, popular authentication technologies, and various tools related to framework functionality.

Algorithms Details:

U-net: The U-Net neural network architecture was introduced by Olaf Ronneberger and his team in 2015 specifically for biomedical image segmentation tasks. The traditional sliding window approach for segmentation tasks involves creating a local patch for each pixel and generating

separate class labels for each patch. However, this approach has two main drawbacks the creation of redundant information due to overlapping patches and a slow training procedure that requires significant resources. U-Net overcomes these limitations by combining classification and localization tasks in its architecture. It comprises convolutional layers and two networks, namely the encoder and decoder, arranged in a U shape. U-Net is effective for segmentation tasks as it provides solutions for both the" what" and" where" questions of segmentation. The encoder network, which is also known as the contracting network, is responsible for learning the features of the input image and answering the first question of segmentation- identifying" what" is in the

Figure 1.2 U-net Architecture

image. Unlike typical convolutional neural net- work classification tasks, there are no fully connected layers in a U- Net encoder network. The encoder network comprises four encoder blocks, each containing two convolutional layers with a kernel size of 33 and valid padding. A Relu activation function follows the convolutional layers, and then a max pooling layer with a kernel size of 22 reduces the spatial dimensions learned, which helps in reducing the computation cost of training the model. The bottleneck layer, located between the encoder and decoder networks, contains two convolutional layers followed by Relu and represents the final feature map output. The expansive network, also known as the decoder network, aims to upsample the feature maps to the size of the input image and generate a segmentation mask with the help of skip connections. The decoder network is responsible for answering the question" where" the object is in the image. It is composed of four decoder blocks, each starting with a transpose convolution with a 22- kernel size. The output of the convolutional layer is concatenated with the corresponding skip layer connection originating from the encoder block. The resultant feature map, obtained through this concatenation, is subsequently processed by two convolutional layers using a 3x3 kernel size and a Rectified Linear Unit (ReLU) activation function.

Pixel accuracy: - Pixel accuracy is a commonly used metric for assessing the performance of semantic segmentation models. It quantifies the percentage of correctly classified pixels in an image. Typically, pixel accuracy is reported individually for each class and also as an overall measure across all classes. To determine per-class pixel accuracy, a binary mask is evaluated, with true positives representing correctly predicted pixels belonging to a specific class based on the target mask. Conversely, true negatives indicate pixels correctly identified as not belonging to the class. However, it's important to note that this metric may not provide an accurate evaluation when the class representation is small within the image, as it can be biased towards assessing the identification of negative cases (i.e., when the class is absent).

$$Pixel\ Accuracy = \frac{\#TP + \#TN}{\#TP + \#TN + \#FP + \#FN}$$

Deep Learning Models:

Using the TensorFlow framework, a Keras segmentation neural network model is constructed and trained to perform Semantic Segmentation on given images. The model incorporates dense layers and operates on pre-processed images and labels represented as vectors.

In the Keras neural network model, the input layer is of (2048) and the flood map vector is the output. The model contains 2 major networks, the encoder, and the decoder. The encoder network comprises four encoder blocks, each containing two convolutional layers with a kernel size of 33 and valid padding and Relu activation function follows the convolutional layer.

In the U-net neural network, the number of neurons in the input layer depends on the number of channels in the input image. Assuming that the input image has three channels (e.g., RGB), the input layer of the U-net model would have 512 x 512 x 3 = 786,432 neurons for each image in the batch. Input Shape = 512 and the N Channel = 3 with a batch size 4.

Since the batch size is 4, the total number of neurons in the input layer would be 4 x 786,432 = 3,145,728 neurons. This means that the input layer of the U-net model would have 3,145,728 neurons for a batch of 4 input images, each with a size of 512 x 512 pixels and three channels.

The final layer of the Keras neural network model, also referred to as the output layer, is supplemented with the Adam activation function. The model is trained for a total of 41 epochs to optimize its performance.

5. Results

The flood water mapped image illustrates the spatial extent of water distribution resulting from a flood event in a specific area. The image displays the regions that are submerged or covered by water, providing a visual representation of the affected areas. It helps to identify the boundaries and patterns of water spread, showcasing the flooded zones and their relationship to nearby land features or infrastructure. This type of image is crucial for assessing the scale of the flood, estimating the impacted population and infrastructure, and aiding in emergency response and relief efforts. It allows stakeholders, such as disaster management authorities and relief organizations, to gain a comprehensive understanding of the affected areas and prioritize rescue and recovery operations accordingly.

Masking:

The image visually represents the process of masking flood water in a flood water mapping image. It showcases the selective removal or isolation of specific areas corresponding to flood water, effectively separating them from the rest of the image.

In the image, the masked regions where flood water is present are typically highlighted or represented in a distinct colour or visual indicator. This allows for clear differentiation between the flooded areas and other elements in the image, such as land, buildings, or infrastructure.

The masking process plays a crucial role in flood water mapping as it helps to accurately delineate and identify the extent of flooded regions. By isolating the flood water areas, it enables further analysis, measurement, and understanding of the impacted areas. Additionally, the masked image can be used as a basis for generating flood water maps or conducting further assessments related to flood management, disaster response, or infrastructure planning.

Overall, the image depicting the masking of flood water mapping images visually showcases the separation of flood water areas from the surrounding environment, enabling a clearer representation and analysis of the flooded regions.

Training Epochs:

The image illustrates the progression of training epochs for a machine learning model or algorithm specifically designed for flood water mapping. Each epoch represents a complete iteration through the training dataset, where the model learns and improves its ability to accurately identify and map flood water regions.

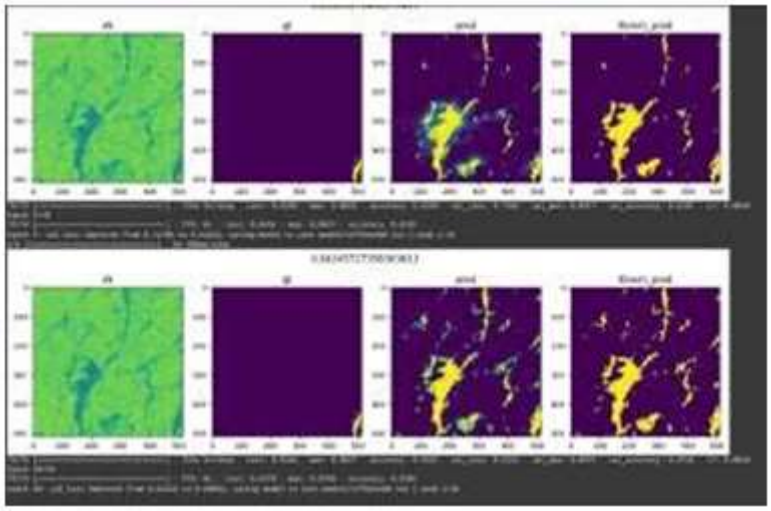

Figure 1.4 Epochs

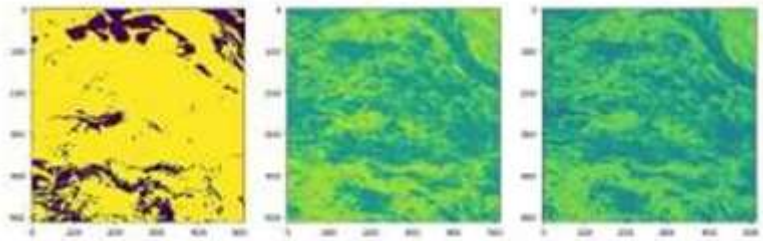

Figure 1.3 Masking

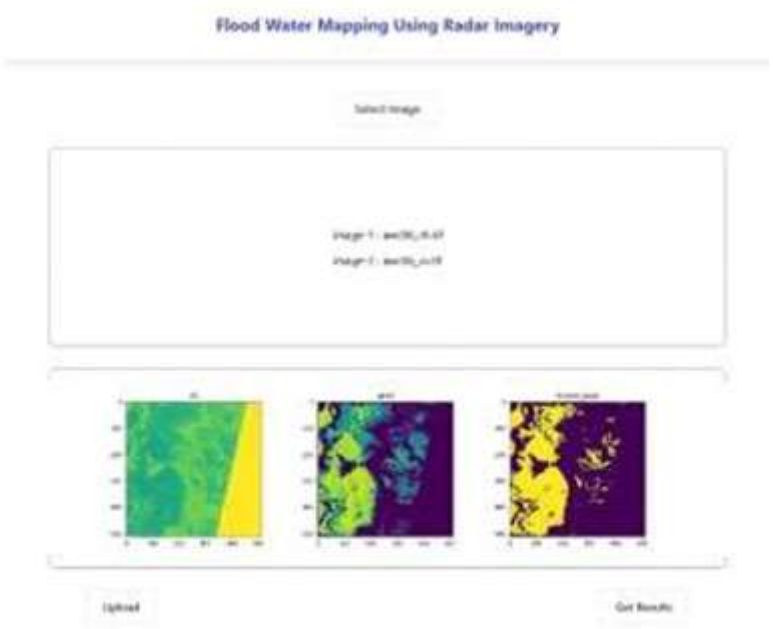

Figure 1.5 Final Project Proof

6. Conclusion

The use of radar technology in imaging can be challenging due to the inherent noise and lack of clarity in the images produced. However, through the application of digital image processing techniques, such as decomposition and SAR interpretation, the potential of radar imaging can be fully realized. This paper explores a range of algorithms that have been developed for the processing of SAR images, providing an in-depth analysis of the strengths and limitations of each method. By studying the various proposed methods and techniques one can gain a comprehensive understanding of the different SAR images that can be obtained through the use of these algorithms. This knowledge can be applied to various fields such as agriculture, geology, and topography. These techniques can be used to extract information from SAR images and to enhance the images, making them more useful for the intended purpose.

The paper offers significant insights into the extensive capabilities of radar imaging and its effective utilization through digital image processing methods, making it a valuable resource for those seeking comprehensive understanding in this field.

In conclusion, Flooding is one of the most severe natural hazards in the world and radar imagery is a valuable tool for identifying areas where such disasters are likely to occur. Radar images can be noisy and often require smoothing. But using pre-processing techniques and a bit of SAR

interpretations, SAR images become a valuable tool. This paper proposes a review of SAR images that uses Neural Networks for satellite image processing techniques. It also focuses on building a relatively simple model that takes radar imagery as an input and provides outputs in the form of binary masks that would indicate which pixels in a scene contain floodwater. It will demonstrate how image processing algorithms using Deep Convolutional Neural Networks (DCNN) provide useful information to mitigate the impacts of flooding.

Reference

1. Ramkumar, G. and Parkavi, P. and Ramya, K. and Priya, M. Swarna," A Survey on SAR Images Using Image Processing Techniques", International Conference on Advanced Computing and Communication Systems (ICACCS), 2020
2. Giustarini, Laura and Hostache, Renaud and Kavetski, Dmitri and Chini, Marco and Corato, Giovanni and Schlaffer, Stefan and Matgen, Patrick," Probabilistic Flood Mapping Using Synthetic Aperture Radar Data",
3. IEEE Transactions on Geoscience and Remote Sensing, 2016
4. Hong Liu, Chunbi Xu, Xiaolu Wang, Tianyou Wang" Identification of OilGas and Water Zones in Geological Logging with Support Vector Machine" International Conference on Cognitive Informatics and Cognitive Computing, 2012
5. R.Jaturapitpornchai, R. Saito, N. Kanemoto and S. Kuzuoka, " Flood Area Detection from a Pair of Sentinel-1 Synthetic Aperture Radar Images Using Res-Unet Based Method", IEEE International Geoscience and Remote Sensing Symposium, 2022
6. Kaushik, Prachi and Jabin, Suraiya," Flood Mapping of Amphan Disaster Using SENTINEL-1 IMAGES", IEEE Global Conference on Computing,
7. Power and Communication Technologies (GlobConPT), 2022
8. Parth Jardosh, Anjali Kanvinde, Anish Dixit and Prof. Surekha Dholay," Detection of Flood Prone Areas by Flood Mapping of SAR Imagery", IEEE Third International Conference on Smart Systems and Inventive Technology (ICSSIT), 2020.
9. Roberto Bentivoglio, Elvin Isufi, Sebastian Nicolaas Jonkman, Riccardo Taormina," Deep learning methods for flood mapping: a review of existing applications and future research directions", EGU Hydrology and Earth System Sciences, 2022.