International Research Journal of Multidisciplinary Scope (IRJMS), 2024; 5(3):337-348 IRJMS

Original Article | ISSN (O): 2582-631X DOI: 10.47857/irjms.2024.v05i03.0721

Flood Detection and Safe Path Finding Using Dense Neural Network (DNN)

Shailza Thakur, Astha Raghvendra Jha, Dev Ankur Mehta, Lavanya K*

Vellore Institute of Technology, SCOPE Department, Vellore, India. *Corresponding Author's Email: lavanya.k@vit.ac.in

Abstract

In contemporary flood management, integrating advanced remote sensing and machine learning marks a significant advancement. This research presents a framework leveraging such tech for flood management. It utilizes the SEN-12 flood dataset for satellite imagery analysis, enabling precise flood mapping via pre-processing. By refining the dataset with bandwidth images, targeted flood detection becomes feasible. Convolutional Neural Networks (CNNs) are employed for correlating flood-affected areas with labelled training data, enhancing mapping accuracy due to their success in image recognition tasks. However, this research extends beyond detection to flood response strategies. Using labelled data, the framework determines safe routes through geospatial analysis. Real-time evaluation of routes' vulnerability to flooding ensures traveller safety. This dual-purpose approach optimizes data and addresses the critical need for proactive flood mitigation measures. By integrating advanced technologies, the research aims to revolutionize flood management by providing timely detection and enhancing response strategies' safety and efficiency. Its significance lies in potentially serving as a blueprint for future flood management initiatives. By offering real-time flood mapping insights and geospatial safety assessments, this research contributes to safeguarding lives and properties in flood-prone regions. It represents a promising shift towards comprehensive flood management systems, ensuring proactive responses to mitigate flooding's impacts on communities and infrastructure.

Keywords: Convolutional Neural Networks, Flood Management, Geospatial Analysis, Machine Learning, Proactive Mitigation, Remote Sensing.

Introduction

Machine Learning (ML) and advanced technologies play a pivotal role in enhancing our preparedness, response, and recovery efforts during natural calamities. In recent years, ML algorithms have managed to analyse historical and real-time data, such as weather patterns, seismic activities, or oceanic conditions, to predict and provide early warnings for natural disasters like hurricanes, earthquakes, or tsunamis. This enables authorities to evacuate people and deploy resources in advance. Python libraries have majorly contributed and enabled us to analyse real time data performing imagery analysis on them for example vegetation pattern of areas (1), spectral bands of satellite captured images (2, 3) etc. Focusing majorly on flood prediction and monitoring, ML models process data from various sources, including satellite imagery, river gauge readings, and weather forecasts, to predict and monitor floods. Algorithms, such as convolutional neural networks (CNNs) and ensemble methods, have shown promise in distinguishing flooded areas from non-flooded ones with high precision (4). Convolutional Neural Networks (CNNs) play a crucial role in flood mapping by leveraging their ability to automatically learn and extract hierarchical features from satellite imagery. These images typically consist of multiple bands (5), each representing different spectral information (e.g., optical, infrared). The pre-processing involves normalizing the data, handling missing values, and resizing images to a consistent format suitable for the CNN. The CNN is trained using a labelled dataset, where each image is associated with a label indicating whether it contains flooded areas. While various research studies have used neural networks to map flooded area and managed to achieve good accuracy (6), but the dataset used are mostly manually labelled for flood mapping and not real time satellite data. Furthermore, just detecting if the mapped satellite image is flooded or not would be of not much use to the on-ground users who are unaware of the radar and remote sensing data.

(Received 08th March 2024; Accepted 16th July 2024; Published 30th July 2024)

To be able to use the trained models much effectively and in real time for users we would majorly focus on using SEN-12 flood data for flood detection and going a step ahead, integrating it with geographical maps to fetch location coordinates and help users identify if the shortest path to provided destination is flooded or non-flooded and whether it's safe or unsafe to take the route. The study in future can be specifically integrated with real time provided satellite images and geo maps making it handy for the users to tackle a natural calamity situation like flood and fulfil one of the major challenges faced by the people and help reduce the human lives damage caused by accidental drowning and getting stuck in damaged or flooded path. OSMnx provides functionality to download and construct street networks directly from OpenStreetMap. Users can specify a location by providing the name of a place, its coordinates, or a bounding box, and OSMnx retrieves the relevant street network data (7, 8). Upon successful identification of flooded areas using the CNN algorithm applied to Sentinel-2 HIF images, the integration of OSMnx becomes pivotal for optimizing emergency response routes through the affected regions. The following methodology section includes how the SEN-12 Flood data has been processed and labelled, Sentinel-2's HIF spectrum, spanning bands B1 to B12, applying different combinations of neural layers and models and we were able to achieve 81% accuracy with the applied CNN model for flood detection and then eventually using OSMnx graphs to visualize street data and getting the output regarding the safety of the route. The exploration of flood detection methodologies in satellite imagery has been a dynamic field, witnessing a spectrum of strategies, algorithms, and data fusion techniques. Smith *et al.,* (2018) showed that using different types of satellite images, like optical and radar, is important for better flood detection. They used machine learning to make it work well. Even though it helped with cloudy areas, it also showed that combining these different types of data is complex, leading to more research. Chen *et al*. (2020) used advanced computer methods, called Convolutional Neural Networks (CNNs), to find floods in detailed satellite images (9). Their method was good for understanding floods after they happened, but it struggled with detecting floods in real-time. This highlighted the need for better ways to handle the changing nature of floods. While CNNs were powerful for analyzing patterns in satellite images, they needed improvements to keep up with the dynamic aspects of flooding. Chaudhary *et al.,* (2020) combined images from satellites and social media to quickly find floods (10). They used advanced computer programs to understand both images and words from social media. This new way showed that people's posts could help know about floods almost immediately. But, the study did not talk much about problems, like making sure the feelings in social media are understood right or keeping people's information private. It's a step forward, but there's still work to do. Yao *et al*., (2023) innovatively combined Synthetic Aperture Radar (SAR) data with a Convolutional Long Short-Term Memory (ConvLSTM) network for accurate prediction of mining area surface deformation benefiting from SAR's all-weather capabilities (11). The study led to more detailed studies of SAR data, which can also be applied on other weather conditions like flood detection. It also showed that neural networks can widely be used on real time data. Dense Neural networks have been effective in recent years. Hernandez *et al.,* (2022) attempted to detect flood using real-time image segmentation from unmanned aerial vehicles on edge computing platform (12). They used a dataset of UAV images from various floods in Spain and apply an AI-based method that leverages three commonly used deep neural networks (DNNs) for semantic image segmentation to automatically identify the area's most impacted by rainfall (flooded regions).

Dense Nueral Network

$$number\_of\_parameters = (D\_in * D\_out) + D\_out$$

$$loss\_computation = (1/N) * sum((y\_i - y\_hat\_i) * * 2 \; for \; i \; in \; range(N))$$

$$computational\_efficiency\_comparison = DNN\_time < CNN\_time$$

In a Dense Neural Network (DNN), the Dense Layer constitutes a crucial component where each neuron is connected to every neuron in the preceding layer. The number of parameters in a Dense Layer is determined by the number of input and output neurons, reflecting the network's capacity for learning. During the forward pass, the

output size remains the same as the input size, preserving spatial information. In the training process, the loss function measures the disparity between predicted and actual labels, guiding parameter updates to minimize this difference.

In the forward pass, input data is processed by multiplying it with weights, adding biases, and passing through an activation function. DNNs consist of multiple dense layers stacked together, working like a team of cells. The input layer receives data, hidden layers process it through weighted connections and activation functions, and the final layer produces the network's output. Training and learning involve presenting the network with labelled examples, computing the loss, and updating parameters to minimize the loss. This iterative process helps the team of cells recognize patterns and relationships in the data, continually improving their performance.

When deciding to integrate DNNs alongside Convolutional Neural Networks (CNNs) in our framework, computational efficiency was a key consideration. If the training time of the DNN model is significantly lower than that of the CNN model, it indicates potential computational advantages. Additionally, other factors such as loss computation and output size preservation are important in evaluating the effectiveness of the network architecture.

Existing flood detection technologies consist of threshold-based methods, hydrological models, ground-based sensor networks and many more. Threshold methods involve setting parameters like river water levels, rainfall intensity or soil moisture content. While this is one of the most used conventional methods they come along with deficiencies. High likelihood of false positives and false negatives are encountered due to the simple nature of thresholds which cannot capture complex interactions. These are not completely dependable as performance varies with different environmental conditions. They lack the ability to provide timely warnings in real time operation. Hydrological models simulate water flow and behaviour using mathematical equations based on physical principles. Common models include HEC-RAS (Hydrologic Engineering Centre's River Analysis System) and SWMM (Storm Water Management Model). But they rely on historical data which can again get challenging in diverse and changing environments. In real-time operations these models are often computationally extensive. Networks of sensors placed in rivers, rain gauges, and other critical locations to monitor water levels and weather conditions. Examples include the USGS (United States Geological Survey) stream gauges and NOAA (National Oceanic and Atmospheric Administration) weather stations. These provide localized data which may not capture the full extent of a flood event.

The DNN model is trained using the Adam optimizer, which adjusts the learning rates dynamically based on the first and second moments of the gradients, leading to efficient and effective training. The loss function used is sparse categorical cross entropy, suitable for classification tasks with integer labels. It can learn from large datasets and identify intricate patterns in satellite imagery that traditional methods miss. The ReLU activation in the hidden layer enables the network to model complex, non-linear relationships, significantly enhancing precision. This reduces the likelihood of false positives and negatives, providing more accurate flood predictions. These are highly adaptable and can generalize well to new data, making it dependable. Dense neural networks, particularly when optimized with algorithms like Adam, can process large datasets efficiently in real-time. The ability to handle high-dimensional satellite data and provide timely predictions is crucial for early warning systems

DNNs offer flexibility in handling variable input shapes, which can be advantageous when dealing with diverse data structures or scenarios where spatial hierarchies are less pronounced. This adaptability allows the model to accommodate different types of input data effectively. Secondly, DNNs may exhibit computational efficiency in certain contexts, potentially leading to faster training times compared to more complex CNN architectures. This efficiency can be crucial in scenarios where rapid model training and deployment are prioritized. Additionally, the decision to incorporate a DNN was influenced by its performance in terms of accuracy and loss metrics during validation. The DNN model demonstrated competitive performance in accuracy and loss metrics, indicating its effectiveness in learning complex patterns within the dataset. Overall, the inclusion of a DNN

alongside CNNs enriches the framework by providing a complementary approach that balances computational efficiency with performance, ultimately enhancing the robustness and versatility of our flood detection and safe path finding system.

Data Availability

The SEN12-FLOOD dataset used in this study is an integral component of the broader SEN12 (Satellite Environment for Flood Monitoring) project, which serves as a valuable resource for monitoring and tracking flood events through the utilization of satellite imagery (13). This project brings together data sourced from an array of Earth-observing satellites, with a particular emphasis on data from the Sentinel-1 and Sentinel-2 satellites, integral components of the European Space Agency's Copernicus program. The combination of optical and synthetic aperture radar (SAR) data obtained from these satellite missions proves invaluable for monitoring environmental changes, particularly those arising from flooding events. The Sen12 data has specifically focused on analysing flood prone areas in its open-source datasets prior as well (14). The dataset consists of 336 distinct sequences, each corresponding to geographic regions in West and South-East Africa, the Middle East, and Australia. These sequences are uniquely identified by a sequence ID, ranging from 0001 to 0336, and each sequence encapsulates a specific geographic area with associated temporal data. The dataset is meticulously organized and enriched through the provision of two essential JSON files, namely S1list.json and S2list.json, which offer comprehensive insights into the dataset's contents. These JSON files furnish a wealth of details, including but not limited to:

Number of Images: Each sequence's JSON entry specifies the total number of individual images it encompasses, providing an understanding of the data volume associated with each region.

Folder Names: These files reveal the names of the folders that house the image data for each sequence, facilitating systematic data retrieval and management.

Geographic Locations: The dataset's geographic coverage is clearly articulated, enabling precise localization of the observed areas.

Image Descriptions: The JSON entries contain detailed image descriptions, offering context and metadata for each image, including its acquisition date.

Attributes: For SAR images, specific attributes such as acquisition orbit type (ascending or descending) are provided, and each image is labelled as either fully or partially imaged, denoted by the FULL-DATA-COVERAGE attribute.

The core of the dataset lies in the inclusion of Level 2A atmospheric-corrected Sentinel-2 images. For each acquisition within a sequence, the dataset offers an impressive array of 12 single-channel raster images, each corresponding to distinct spectral bands. These spectral bands enable a comprehensive examination of the Earth's surface and are instrumental in various remote sensing applications. Notably, the dataset is segmented into two fundamental parts, "Label" and "Source," each serving a unique purpose:

Label: The "Label" component of the dataset contains JSON files that indicate specific dates were flooded areas are labelled. This information is pivotal for mapping and facilitates the identification of flooded and non-flooded areas within the dataset.

Source: The "Source" component encompasses real-time satellite images that can be directly compared to the labelled flooded areas. This allows for a detailed analysis of the dataset and its relevance to flood monitoring. Spectral bands, B1 to B12 (Figure 1) are crucial for flood detection as they capture unique signatures of water and its interaction with surrounding features in satellite imagery. Water absorption bands in the infrared spectrum, contrast with land features, vegetation responses in multispectral bands, and the use of thermal bands contribute to effective flood mapping (2). Analysing changes over time through a series of spectral images enhances detection accuracy. These bands serve as key input features for machine learning algorithms, allowing for automated flood identification based on the distinctive spectral characteristics associated with flooded areas.

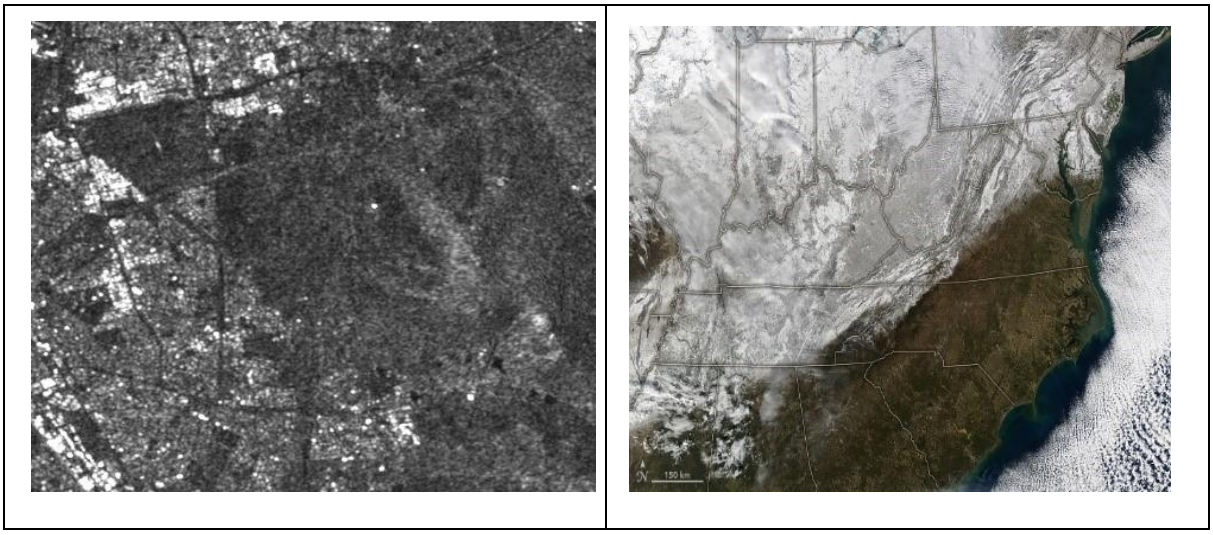

Figure 1: Satellite Imagery from Sen-12 Showcasing Diverse Land Features

The dataset provides high-resolution images capturing various land types and features, contributing valuable insights for earth observation and environmental monitoring. The SEN12-FLOOD dataset, with its rich and diverse content, serves as a crucial asset for not only flood monitoring but also broader geospatial and mapping applications. Its extensive coverage and comprehensive attributes empower researchers and analysts to derive valuable insights into environmental changes and support various decision-making processes. The dataset is divided into two parts: Label and Source. The "Label" folder contains JSON files indicating dates with labelled flooded areas, facilitating mapping with real-time satellite images in the "Source" folder for identifying flooded and non-flooded areas. In the second part of the research, the study builds upon the flood detection methodology's output to enhance path assessment capabilities. The project leverages the Python library OSMnx (15) and open-source mapping resources to create a sophisticated street mapping system, enabling users to select preferred locations and modes of transportation. This system efficiently computes and presents the shortest distance between chosen points (16).

Methodology

The methodology employed in this research involves a rigorous process of data collection and pre-processing, centring on the SEN12-FLOOD dataset derived from Sentinel-2 satellite images. Leveraging Level 2A atmospheric-corrected imagery, the dataset encompasses 336 sequences across various regions, providing a diverse and extensive range of flood scenarios (14). Essential metadata, including acquisition dates, geographical locations, and image descriptions, is extracted from the S2list.json file for each sequence. Specifically, the 12 single-channel raster images representing distinct spectral bands are utilized for comprehensive examination of the Earth's surface (Figure 1).

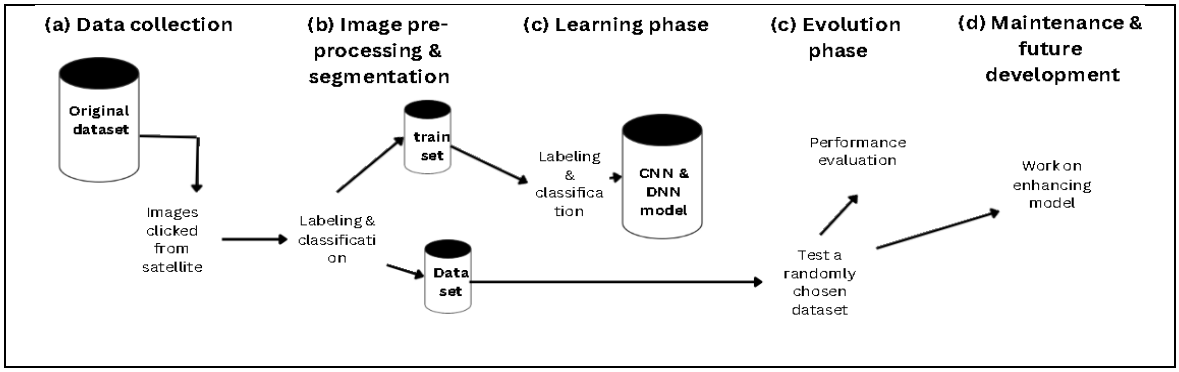

Figure 2: Safest Path Finding Framework

To facilitate flood mapping, Convolutional Neural Network (CNN) architecture is employed. The adoption of CNNs is driven by their exceptional capability to automatically learn hierarchical features from input data, making them particularly effective for image-related tasks such as flood mapping. This neural network is structured with an input layer accommodating the spectral band images, hidden layers comprising convolutional and pooling layers for spatial feature extraction, and an output layer for binary classification indicating flooded or non-flooded areas (15). The CNN is trained on labelled flooded areas from the "Label" component of the dataset, with a meticulous dataset split into training and validation sets. Data augmentation techniques, such as rotation and flipping, are applied to enhance dataset diversity. Binary cross-entropy serves as the loss function, optimized using the Adam optimization algorithm, and model performance is evaluated using accuracy, precision, recall, and F1 score metrics on the validation set (Figure 2).

Phase1- Pre-processing

The pre-processing step in this code aims to prepare satellite image data for further analysis, particularly for flood detection.

1. Identifying Empty Images: The first part of the pre-processing involves checking each folder in the specified directory (`rootdir`) to see if it contains empty images. An empty image is identified based on its filename; specifically, it checks if the filename contains 'B01', which suggests the absence of valid image data. If an empty image is found within a folder, that folder is removed from further processing.

2. Stacking Bands: After ensuring that the images are not empty, the code proceeds to stack bands from the remaining images. Bands are individual layers of satellite image data corresponding to specific wavelengths. The code collects bands named 'B02.tif', 'B03.tif', 'B04.tif', and 'B08.tif', which are commonly used for various remote sensing applications. These bands are stacked together to form a single multi-band image named 'stack.tif'. Stacking bands in this way consolidates information from different wavelengths, enhancing the richness of the data for subsequent analysis.

3. Clean-up and Validation: In case of any errors during the processing or if a folder contains no valid data, the code removes those folders entirely. This clean up step ensures that only relevant data is retained for further analysis. At the end of the processing and cleaning up of data, we are left with 1189 valid folders on which we will perform feature extraction modelling in the upcoming section.

Feature Extraction

The selection of specific bands within the hyper spectral infrared (HIF) spectrum, namely B4 (Red) and B5 (Near-Infrared), for the application of Convolutional Neural Network (CNN) algorithms in flood mapping is driven by their unique spectral characteristics and their relevance to the discrimination of flooded areas. The selection of specific bands within the hyper spectral infrared (HIF) spectrum, namely B4 (Red) and B5 (Near-Infrared), for the application of Convolutional Neural Network (CNN) algorithms in flood mapping is driven by their unique spectral characteristics and their relevance to the discrimination of flooded areas. In the Sentinel-2 HIF spectrum, B4 captures red light, while B5 spans the near-infrared region. The red and near-infrared bands have proven to be particularly informative in remote sensing applications due to their sensitivity to variations in vegetation health and water content. Specifically, water absorbs light in the red band, resulting in lower reflectance, while it exhibits higher reflectance in the near-infrared band. This inherent difference allows for the identification of water bodies and facilitates the discrimination of flooded areas from other land cover types.

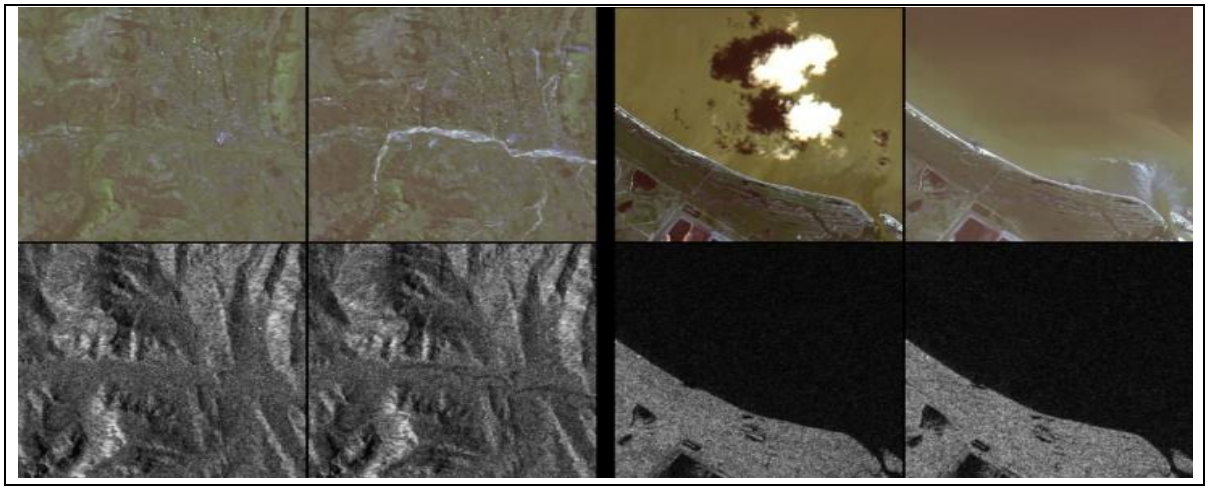

Figure 3: Pre-processed Satellite Images for Flood Mapping Using CNN with Hyperspectral Infrared Bands B4 (Red) and B5 (Near-Infrared)

The choice of B4 and B5 in the CNN algorithm aligns with their utility in capturing the spectral signatures associated with flooded regions. By leveraging the distinctive responses of these bands to water, the CNN can learn to recognize patterns indicative of inundation. The incorporation of these bands in the training data empowers the model to discern subtle variations in reflectance that signify the presence of water, contributing to the precision and accuracy of flood mapping. Additionally, focusing on these key bands streamlines the computational complexity of the CNN, making the model more efficient while retaining the essential spectral information needed for robust flood detection (Figure 3).

```
model = tf.keras.Sequential([
tf.keras.layers.Conv2D(32, (3, 3), activation =
'relu', input_shape = (512, 512, 3)),
tf.keras.layers.MaxPooling2D(2,2),
tf.keras.layers.Conv2D(32, (3, 3), activation =
'relu'),
tf.keras.layers.MaxPooling2D(2,2),
tf.keras.layers.Flatten(),
tf.keras.layers.Dense(128, activation=tf.nn.relu),
tf.keras.layers.Dense(2,
activation=tf.nn.softmax)])
```

The CNN model begins with a convolutional layer containing 32 filters, each of size [3, 3], and uses the rectified linear unit (ReLU) activation function. This layer is tailored for processing input images with dimensions [512, 512, 3], where 3 represents the colour channels (RGB).

Subsequently, a max-pooling layer with a pool size of [2, 2] is applied, reducing the spatial dimensions of the data. Following this, a second convolutional layer and another max-pooling layer are added to further enhance feature extraction capabilities. The model then incorporates a flattening layer, which transforms the output from the previous layers into a one-dimensional array. This is a necessary step before passing the data to fully connected layers. Two fully connected layers follow the flattening layer: the first dense layer consists of 128 neurons with the ReLU activation function, capturing complex patterns learned from the convolutional layers. The final layer, also dense, has two neurons representing the binary classes (flooded or non-flooded) and utilizes the SoftMax activation function to produce a probability distribution over the classes.

Phase 2- Safest Path Finding in Flooded Regions

In our project, we've harnessed the power of the Python library, OSMnx (7), along with open-source mapping resources to create a sophisticated system for street mapping. With this technology, users have the flexibility to select their desired location and mode of transport. Our system then swiftly computes and displays the shortest distance between the chosen points (16). Integrating our initial model for flood detection, we take it a step further. Before suggesting a path, our system assesses the safety of the route by evaluating if the path is flooded or not.

This dynamic fusion of street mapping and flood detection ensures that users can make informed decisions about their travel routes, especially during flood-related situations, thus enhancing safety and efficiency.

OSMnx, short for OpenStreetMap (OSM) networks in Python, is a Python library that simplifies the process of downloading, modeling, and analyzing OpenStreetMap data. Developed by Geoff Boeing, OSMnx provides a user-friendly interface for working with spatial network data, particularly for urban environments. It enables users to retrieve and analyze complex networks based on OpenStreetMap data, including streets, buildings, parks, and other urban features. OSMnx allows users to download street network data for specific locations by providing either place names, addresses, or geographical coordinates. It interfaces with the Overpass API, retrieving data from OpenStreetMap to construct detailed and accurate representations of urban infrastructure.

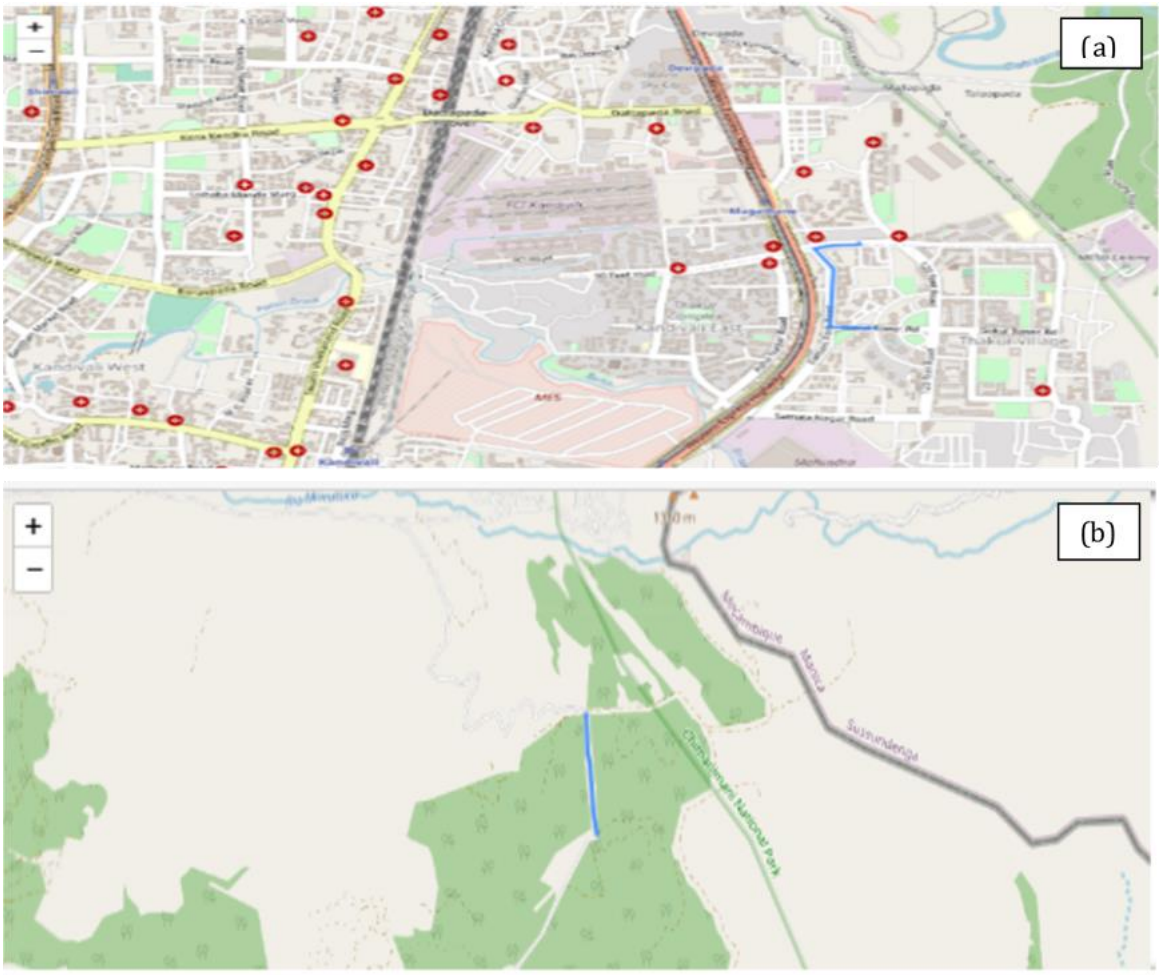

Figure 4: (a) Flooded Path, (b) Safe Path Using OSMnx

Existing flood detection technologies have several constraints, such as limited precision, dependability, and real-time operation. To address these challenges, we are using the OSMnx Python library to determine the shortest path during flood events. This integration is crucial as it allows for efficient and accurate route planning in real-time, ensuring that emergency responders and affected individuals can navigate safely and quickly. By combining advanced flood detection methods with robust pathfinding capabilities, we can significantly enhance disaster response times and improve the safety and efficiency of evacuation and resource distribution efforts. The library creates a network model from the downloaded OpenStreetMap data, representing the connectivity and spatial relationships between different elements such as nodes and edges. Nodes typically represent intersections or points of interest, while edges denote the streets or pathways connecting them. OSMnx provides a suite of functions for analyzing the network structure. Users can calculate various network metrics, such as centrality, connectivity, and shortest path distances. This analysis facilitates a deeper understanding of the urban layout, connectivity patterns, and accessibility (Figure 4).

Function to find the shortest route given start and end locations
Function to check if the route is safe from flood Get true or false
Define route information for each route
Find shortest routes
Check for flood along the routes
Print safety status of routes

Incorporating Convolutional Neural Network (CNN) results with OpenStreetMap data for flood detection involves a sophisticated methodology aimed at assessing the safety of specific areas in the context of potential flooding. Incorporating with flood mapped areas.

The process begins with the acquisition of satellite imagery, particularly bands B1 to B12, capturing diverse spectral information. These images are then processed using a CNN model trained to distinguish flooded and non-flooded areas. The adoption of CNNs is instrumental in extracting intricate spatial features and relationships from multi-band satellite imagery, providing a robust foundation for accurate flood predictions. The CNN leverages its ability to learn complex patterns and relationships within multi-band satellite imagery to generate predictions about the presence of floods.

Simultaneously, OpenStreetMap data is utilized to obtain detailed information about the street network, land use, and geographical features (Figure 5 and Figure 6). This information is particularly valuable for understanding the infrastructure and layout of the areas under consideration. The integration of CNN results with OpenStreetMap involves overlaying the flood predictions onto the geographic context provided by OpenStreetMap. The CNN predictions, indicating areas susceptible to flooding, are spatially mapped onto the OpenStreetMap representation. This mapping enables a comprehensive visualization of potentially flooded regions within the context of the local infrastructure. By aligning CNN results with the street network and relevant geographic features, it becomes possible to evaluate the safety of specific routes and areas in real-world terms. To assess the safety of a given path, the methodology involves calculating the intersection between the predicted flooded areas and the street network from OpenStreetMap. Areas where the CNN predicts flooding that coincide with the mapped street network suggest potential danger zones. Conversely, areas where the CNN indicates safety aligning with the street network indicate paths less likely to be affected by floods.

Results and Discussion

Model 1 employs Convolutional Neural Network (CNN) architecture, a well-established choice for tasks involving image data. The model begins with convolutional layers, capturing spatial hierarchies in the input images. The subsequent max-pooling layers help down sample the learned features. The architecture is then flattened before passing through dense layers, culminating in a SoftMax activation layer for classification. The input shape is fixed at [512, 512, 3], making it suitable for image data with consistent dimensions.

On the other hand, Model 2 follows a simpler architecture, relying solely on dense layers. It begins with a flattening layer, accommodating variable input shapes denoted as [x, y, z]. This flexibility allows the model to adapt to diverse data structures. The subsequent dense layers capture intricate patterns in a flattened representation, and the final layer uses softmax activation for classification. The architecture is computationally less complex compared to Model 1, potentially facilitating faster training (Table 1).

The primary distinction lies in their handling of spatial relationships. Model 1, with its CNN layers, excels at capturing spatial features crucial for tasks like image classification. In contrast, Model 2, relying on dense layers, may be more appropriate for datasets where spatial hierarchies are less pronounced or when input shapes vary.

While Model 1 assumes a fixed image size, it might be more suitable for tasks like flood detection in satellite imagery, where spatial details play a significant role. However, the validation performance and potential overfitting should be carefully assessed. Model 2 offers more flexibility in terms of input shapes, making it advantageous for scenarios where data structures vary.

Model 2, Dense Neural Network showed more CPU and GPU utilization and took way more time to get trained than the Model 1, Convolutional Neural Network. The model 1 took over 1 hours to get trained over 1000-image dataset but the model 2 took over 3 hours to complete it training.

The DNN model gave better results with the same number of epochs and steps and gave more

validation and training accuracy of 96.78% and the loss is less in DNN model as compared to the CNN model. Furthermore, we have plotted accuracy on training and validation datasets as well as training and validation loss during the training process (Table 2). We randomly selected a set of images from validation dataset and tested our model on that small dataset and we got an accuracy of 81.00% for CNN Model. Similarly, the training and validation accuracy and loss for DNN Model.

Table 1: Time Comparison Between CNN and DNN

Algorithm	Average CPU Utilization	Average GPU Utilization
Convolutional Neural Network (CNN)	2.54%	29.16%
Dense Neural Network (DNN)	3.08%	35.95%

Table 2: Performance Evaluation Metrics

Algorithm	Epoch	Steps per epoch	Training Accuracy (%)	Training Loss (%)	Validation Accuracy (%)	Validation Loss (%)
CNN	15	28	96.43	0.781	79.76	2.2644
	20		96.78	0.787	80.45	2.2293
	25		96.85	0.782	**81.01**	2.3879
Dense Neural Network	15	28	95.39	0.1096	79.17	1.8687
	20		96.43	0.1044	80.36	1.4510
	25		96.78	0.0922	**82.12**	1.7252

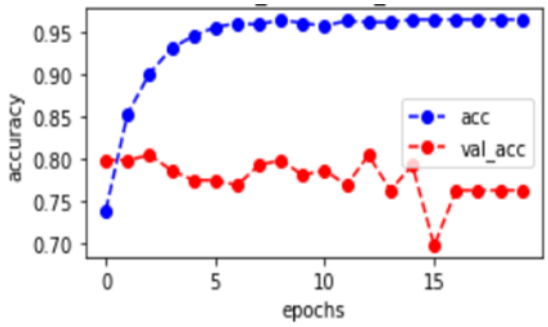

Figure 5: Graph Depicting the Training and Validation Accuracy in CNN Model

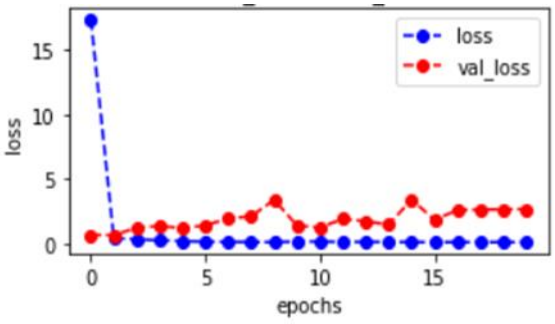

Figure 6: Graph Depicting the Training and Validation Loss Accuracy in CNN Model

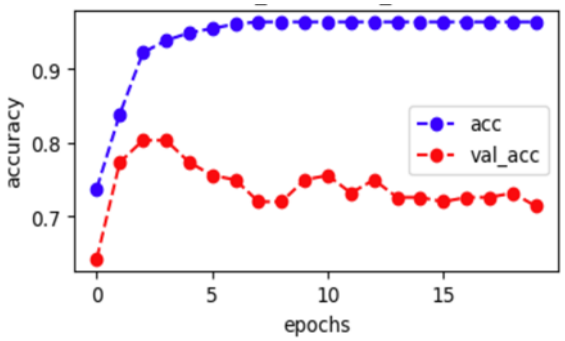

Figure 7: Graph Depicting the Training and Validation Accuracy in DNN Model

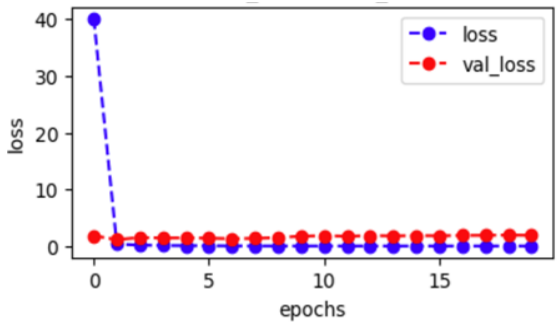

Figure 8: Graph Depicting the Training and Validation Loss in DNN Model

The higher validation accuracy of the DNN suggests it likely has better precision and recall compared to the CNN (Figure 7 and Figure 8). This means the DNN is more effective at correctly identifying flood events (high recall) and minimizing false alarms (high precision). Given the better performance of the DNN in terms of both accuracy and loss, it is likely that the F1

score for the DNN is also higher. The balance between precision and recall in the DNN makes it a more robust model for flood detection, especially in imbalanced datasets where flood events are less common.

Constraints and Sources of Inaccuracies

The performance of both models is heavily dependent on the quality and quantity of training data. Limited or poor-quality data can lead to under fitting, where the model fails to capture underlying patterns. Overfitting is another concern, particularly with complex models like DNNs and CNNs, which may perform well on training data but poorly on unseen data due to memorization rather than generalization. Additionally, training deep learning models, especially DNNs and CNNs, requires substantial computational resources. The DNN model (Model 2), being more computationally intensive, took three times longer to train compared to the CNN model (Model 1). Inaccuracies can stem from inadequate spatial resolution of satellite images, leading to missed flood details and inaccurate predictions. Temporal resolution is also a critical factor; delays in acquiring satellite images can hinder the real-time applicability of the models. Environmental variability, such as seasonal changes and unforeseen events, introduces noise and variability in the data, affecting model accuracy. In a real-time context, these constraints are exacerbated by the need for timely data processing and updates. Integrating shortest path finding using OSMnx during floods adds further constraints. The dynamic nature of flood events requires continuous updates to the road network data to reflect real-time conditions accurately. Any delays or inaccuracies in updating the network can result in suboptimal or unsafe route recommendations. The computational overhead of real-time pathfinding, combined with the demand for high precision and recall in flood detection, presents significant challenges in deploying an integrated, real-time flood response system. However, this improvement in prediction accuracy enables efficient allocation of resources, such as emergency services and relief efforts, to the most affected areas. For communities, the benefits are substantial: better preparedness for impending floods can significantly reduce potential loss of life and property damage. Reliable flood predictions also support the formulation of effective evacuation plans, ensuring the safety of residents in vulnerable areas. Additionally, timely and accurate flood information helps protect critical infrastructure, such as roads, bridges, and utility services, from flood damage. These advancements collectively contribute to enhancing catastrophe response times and improving the safety and resilience of affected communities.

Conclusion

In conclusion, this research work presents a pioneering and comprehensive framework for flood detection and safe pathfinding by integrating advanced remote sensing technology, machine learning methodologies, and geographic information systems. The utilization of the SEN12 flood dataset and Convolutional Neural Networks (CNNs) for flood mapping demonstrates the efficacy of data-driven approaches. The integration of OpenStreetMap (OSM) data through OSMnx further enhances the framework by optimizing emergency response routes. The proposed methodology not only excels in real-time flood detection with high accuracy but also addresses the critical need for assessing the safety of travel routes in flood-affected regions.

Abbreviations

Nil.

Acknowledgement

Nil.

Author contributions

All Authors contributed entire manuscript in writing, reviewing, implementing, conceptualization and analysis.

Conflicts of Interest

The authors declare no conflict of interest.

Ethics Approval

Not applicable.

Funding

Nil.

References

1. Calculate Vegetation Indices in Python. Earth Data Science - Earth Lab. 2020. Available from: https://www.earthdatascience.org/courses/use-data-open-source-python/multispectral-remote-sensing/vegetation-indices-in-python/

2. Dutta G. Satellite Imagery Analysis using Python. kaggle.com. 2021. Available from:https://www.kaggle.com/code/gauravduttakiit/satellite-imagery-analysis-using-python
3. Lillesand TM, Kiefer RW, Chipman JW. Remote sensing and image interpretation. 7th ed. Hoboken, NJ, John Wiley & Sons, Inc; 2015.
4. Sarker C, Mejias L, Maire F, Woodley A. Flood Mapping with Convolutional Neural Networks Using Spatio-Contextual Pixel Information. Remote Sensing. 2019 Oct 8;11(19):2331.
5. Nemni E, Bullock J, Belabbes S, Bromley L. Fully Convolutional Neural Network for Rapid Flood Segmentation in Synthetic Aperture Radar Imagery. Remote Sensing. 2020 Aug 6;12(16):2532.
6. Adetunji OJ, Adeyanju IA, Esan AO, Sobowale AA. Flood image classification using convolutional neural networks. ABUAD Journal of Engineering Research and Development. 2023;6(2):113-21.
7. Boeing G. OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. Computers, Environment and Urban Systems. 2017 Sep;65:126–39.
8. Hacar M, Kılıç B, Şahbaz K. Analyzing OpenStreetMap Road Data and Characterizing the Behavior of Contributors in Ankara, Turkey. ISPRS International Journal of Geo-Information. 2018 Oct 6;7(10):400.
9. Chen C, Hui Q, Xie W, Wan S, Zhou Y, Pei Q. Convolutional Neural Networks for forecasting flood process in Internet-of-Things enabled smart city. Computer Networks. 2021 Feb 26;186:107744.
10. Chaudhary P, D'Aronco S, Leitão JP, Schindler K, Wegner JD. Water level prediction from social media images with a multi-task ranking approach. ISPRS Journal of Photogrammetry and Remote Sensing. 2020 Sep 1;167:252-62.
11. Yao S, He Y, Zhang L, Yang W, Chen Y, Sun Q, Zhao ZA, Cao S. A convLSTM neural network model for spatiotemporal prediction of mining area surface deformation based on SBAS-InSAR monitoring data. IEEE Transactions on Geoscience and Remote Sensing. 2023 Jan 12;61:1-22.
12. Hernández D, Cecilia JM, Cano JC, Calafate CT. Flood detection using real-time image segmentation from unmanned aerial vehicles on edge-computing platform. remote Sensing. 2022 Jan 4;14(1):223.
13. Rambour C, Audebert N, Koeniguer E, Le Saux B, Crucianu M, Datcu M. Sen12-flood: a sar and multispectral dataset for flood detection. IEEE: Piscataway, NJ, USA. 2020 Sep. Available from: https://dx.doi.org/10.21227/w6xz-s898.
14. SEN12-FLOOD Dataset. Availablefrom: https://clmrmb.github.io/SEN12-FLOOD/
15. Boeing G. OSMnx 1.1.2 documentation. osmnx.readthedocs.io. Available from: https://osmnx.readthedocs.io/en/stable/
16. Mahmood M. Finding Shortest Distance Between Points on Map Using OSMNX Package. Medium. 2023. Available from: https://blog.devgenius.io/finding-shortest-distance-between-points-on-map-using-osmnx-package-2a92afefe4d1