London South Bank University

SCHOOL OF ENGINEERING

Project title:

Deep Neural Network for Flood Detection and Impact Assessment: CNN and Autoencoder Integration

Student name: Asaduzzaman
Student id: 4142378
Qualification: MSc in Applied Artificial Intelligence
Supervisor: Francis Babayemi
Submission date: May 2024

ACKNOWLEDGEMENT

In the name of Allah, the Most Gracious, the Most Merciful,

Alhamdulillah, I am grateful to have completed my MSc project, and I offer my sincerest thanks to Allah (SWT) for His endless blessings and guidance.

I extend my heartfelt gratitude to Francis Babayemi for his steadfast support and invaluable guidance during both this project and the entire course. As a course director, his unwavering assistance has been instrumental in my journey. Without his help, completing this task would have been immensely challenging.

I am deeply thankful to Enrico Grisan for his prompt and dedicated support in the initial phases of my dissertation. His timely assistance ensured I received the right guidance, laying a strong foundation for this project.

My profound appreciation goes to my dear family for their unwavering support and understanding. Their sacrifices, encouragement, and unwavering presence during the highs and lows have been the cornerstone of my perseverance. This project wouldn't have been possible without their patience and love.

In the words of an old proverb, "It takes a village to raise a child." I am immensely grateful to my 'village' for nurturing and supporting me throughout this endeavor. Their collective contributions have been invaluable in bringing this project to fruition.

Table of Contents

ABSTRACT

The complication of evaluating damage and assigning resources professionally are major difficulties for emergency responders and support organisations during natural disasters. Conventional damage assessment methods regularly rely on manual examination, which may be laborious and resource intensive. This can delay response efforts and make it more difficult to correctly priorities key locations. This dissertation uses the FloodNet dataset to use urbane artificial intelligence (AI) methods for rapid disaster response, precisely for flood catastrophes. The main objective is to create a deep learning-based system that can use drone and satellite images to quickly analyses flood-related damage, with an importance on wedged assets and key substructure. In order to precisely classify and estimate flood damage, the research uses Convolutional Neural Network (CNN) models intended for disaster management and optimises them for feature extraction from FloodNet photos. The model's capacity to recognize and estimate flood-related damage is momentously increased by utilising CNN to extract features from FloodNet photos. This helps to endorse better situational mindfulness and well-organised disaster management. This study also uses autoencoder algorithms to detect anomalies in flood images systematically. The autoencoders are engineered to distinguish prominent changes suggestive of flood damage in both pre- and post-flood imagery, so augmenting the system's ability to recognize and react to nonconforming patterns linked to disasters.

With regard to flood damage assessment, the suggested model is especially made to perform better than the most advanced models available. It exhibits superior performance measures, such as accuracy (93.50%), precision (91.51%), recall (95.90%), and F1-score (93.65%). The effectiveness of using cutting-edge AI approaches to improve disaster response capabilities is shown by this efficacious demonstration. This will allow for more effective resource allocation and mitigation tactics in areas that are prone to flooding.

With implications for improving real-time flood monitoring and response operations, the use of CNN and autoencoder models in conjunction with FloodNet imagery represents creative ways to leverage AI in disaster response. This research helps to improve disaster management procedures and preparedness by utilising deep learning and anomaly detection, which eventually increases community resilience to natural catastrophes like floods.

List of Figures

List of Tables

Chapter 1 – Introduction

Chapter 1 is organized into eight sub-sections. Sub-section 1.1 investigates the background and scope of the project, while sub-section 1.2 elaborates on the explanation of the investigation problem. Sub-section 1.3 presents the examination questions and their inferences, while sub-section 1.4 defines the research objectives and aims. Then, sub-section 1.5 converses the potential impact of the project, shadowed by sub-section 1.6 if a summary of methodological method. Sub-section 1.7 highlights the research influence, and finally, sub-section 1.8 offers a thesis overview, providing a summary of the subsequent chapters.

Background and Scope of Project

Natural tragedies represent profound threats to human lives and substructure on a global scale. In the aftermath of such events, real disaster management hinges on the rapid and precise valuation of the incurred damage. Conservative methods of damage assessment often entail laborious processes, overriding considerable time, and incomes, and are susceptible to inaccuracies. Though, recent development in deep learning algorithms offer a transformative avenue for automating this overbearing task by harnessing the logical potential of satellite or drone imagery. Cable and drone imagery afford a panoramic viewpoint of disaster-affected regions, enabling meticulous scrutiny of infrastructure, buildings, roads, and other essential assets. Deep learning algorithms, which showcased extraordinary efficacy outside just CNNs, present unmatched capabilities in sharp designs and features within images. By leveraging these algorithms and training them on pre-and post-disaster images, the potential appears to develop sophisticated model’s adept at quickly and precisely classifying damaged structures. Such models can play a pivotal role in accelerating response efforts by reorganization the distribution of resources to the area’s most in need (Linardos *et al.*, 2022).

The expertise of deep learning algorithms, particularly CNNs, in advancing automatic damage assessment is commendable. By acceptance a diverse array of deep learning models and mixing multi-modal data sources, the potential explains for developing robust, adaptive systems talented of addressing the multifaceted tests of disaster management with unparalleled precision and competence.

The scope of this dissertation project is focused on harnessing the power of deep learning algorithms to develop an automated system for damage assessment in the realm of disaster management. Usually disasters, ranging from earthquakes to hurricanes, often leave late widespread obliteration, necessitating swift and accurate assessment of the damage incurred. Usually, this process has relied on manual review methods, which can be slow, resource-intensive, and disposed to errors (Blomeier *et al.*, 2024). Though, with recent progressions in deep learning and computer dream, there is a chance to revolutionize this process by leveraging satellite and drone imagery. Essential to the scope of this project is the gaining and pre-processing of data. This involves sourcing high-resolution satellite images and drone footage taken before and after a tragedy strikes. Pre-processing tasks contain image registration, noise discount, and normalization to ensure constancy and accuracy in the following analysis. The quality and reliability of the data are supreme in training robust deep learning models proficient of accurately detecting and measuring injury.

At the core of the project is the development of custom deep learning models tailored specifically for injury estimation. CNNs adept during image recognition tasks, will be utilized to analyse before-and-after images, categorizing damaged infrastructure, buildings, roads, and other vulnerable assets. Ended iterative training and optimization, these models will learn to differentiate between numerous types of damage and measure its harshness, laying the basis for an automatic damage assessment system. Adding with existing tragedy management systems is another vital feature of the project's scope. The progressive system must seamlessly contribute with GIS software, decision provision systems, and statement networks commonly used by substitute responders and relief formations. This integration will ease real-time data analysis, allowing conversant decision-making and real resource delivery in the aftermath of a disaster.

Throughout the project, difficult assessment and validation will be showed to measure the performance and constancy of the progressive system. Metrics such as accuracy, precision, recollection, and F1-score will be used to measure the competence of the deep learning models in detecting and measuring damage. Real-world testing conditions and ground reality data will be utilized to authorize the system's presentation under various situations, confirming its applied utility in tragedy management processes.

Justification of Research Problem

Disaster response, primarily in the setting of flood events, poses important tests for emergency responders and aid formations due to the difficulties of assessing damage and assigning resources jobwise. Traditional methods of damage estimate often rely on manual review, which can be time- consuming and resource-intensive, foremost to delays in reply to efforts and hindering the aptitude to order critical areas efficiently. The cumulative frequency and severity of floods impair these tasks, necessitating advanced solutions to expedite tragedy retort and minimize the effect on affected communities.

The justification of the research problem lies in the unwavering need for progressive solutions to improve disaster reply, chiefly in the context of flood events. Overflows present sole challenges due to their random fauna and the extensive damage they impose on substructure and groups. Traditional methods of damage assessment, dependent on physical review, are integrally slow and resource- intensive, impeding the aptitude of spare responders and aid governments to instruction and assign resources effectively. The bulge frequency and harshness of floods exacerbate these tasks, underlining the urgency for novel approaches to accelerate disaster response and lessen the impact on pretentious societies. Deep learning algorithms, joined with cable and drone images, offer a talented avenue for addressing these topics by mechanizing the damage valuation process. Though, to realize the full potential of this method, some dangerous tasks must be overcome (Linardos *et al.*, 2022).

Firstly, there is a need to professionally train deep learning models, chiefly CNN on flood-specific datasets to recognize patterns and features skimpy of hurt accurately. The development and optimization of such models are vital for allowing accurate and consistent damage assessment in flood- affected areas. Also, the integration of trained models into a consistent system skilled of providing real- time or near-real-time analysis presents technical and logistical challenges that require careful reflection. This contains confirming seamless interoperability with current disaster management systems and substructure, as well as addressing issues connected to data processing, storage, and communication.

Research Questions and their implications.

1. How effectively can CNN trained on the FloodNet dataset classify and evaluate flood-related damage in satellite and drone images?

The research question explores the effectiveness of CNN trained on the FloodNet dataset in accurately assessing flood-related damage within satellite and drone imagery. By measuring the performance of these CNNs, researchers aim to device their accuracy and reliability in classifying various types of damage caused by floods, counting structural damage, debris accumulation, and infrastructure disturbances. This examination investigates into the potential of leveraging progressive machine learning methods to mechanise the process of damage assessment, which typically relies on manual evaluation and is often time-consuming and resource focused. Understanding the capabilities of CNNs in this setting could pave the way for more effective disaster rejoinder policies, enabling quicker interferences and improved allocation of resources to alleviate the impression of floods on pretentious societies (Ramasubramanian, 2021).

The implications of this research question are far-reaching. If CNNs trained on the FloodNet dataset show to be effective in identifying and assessing flood-related damage in cable and drone imagery, it could revolutionise tragedy management practices. Powering damage valuation through machine learning algorithms could meaningfully expedite response pains, allowing spare responders and aid organizations to promptly address dangerous areas and assign resources where they are most needed. This could finally lead to improved outcomes in terms of minimising the loss of life and property during flood events, increasing community resilience, and justifying the overall impact of natural disasters on society and substructure.

2. What is the performance of modified autoencoder models for irregularity detection in flood imagery, and how accurately can they highlight important changes revealing of damage in pre- and post-flood imageries?

The research question investigates into the performance of modified autoencoder models for anomaly detection in flood imagery, aiming to ascertain their efficiency in accurately identifying important changes indicative of damage in pre- and post-flood imageries. By assessing the performance of these autoencoder models, researchers seek to appreciate their ability to distinguish subtle alterations in flood-affected areas, such as physical damage, water levels, and changes in scenery features. This investigation is vital for developing advanced anomaly detection techniques that can complement outdated methods of damage assessment, offering a more comprehensive impact of floods on substructure and groups. By focusing on modified autoencoder models tailored precisely for flood imagery, researchers aim to oppress the full potential of machine learning in disaster response efforts and justifying the consequences of flood movements.

The potential for modified autoencoder models to effectively detect anomalies indicative of damage in pre- and post-flood images could significantly enhance disaster management capabilities. These models could deliver valuable visions into the degree and severity of flood-related damage, facilitation more informed decision-making for another responders and aid governments. By allowing rapid and accurate ID of injured areas, modified autoencoder models have the potential to accelerate response efforts, arrange reserve allocation, and ultimately decrease the influence of floods on exaggerated groups (Damaševičius *et al.*, 2023). This research thus holds potential for ornamental resilience to flood events and purifying overall disaster preparedness and response plans.

Research Objectives and Aims

The aim of this project is to leverage the FloodNet dataset to produce a deep learning-based system for accelerated disaster reply, focusing exactly on flood events. This system will analyse satellite and drone imagery to allow rapid damage assessment of dangerous infrastructure, structures, and other essential assets pretentious by floods.

The objectives of thesis project include:

a) Adjust and fine-tune CNN using the FloodNet dataset to accurately classify and assess flood- related damage to substructure, buildings, and other valid features visible in satellite and drone imagery apprehended before and after flood actions.

The research objective is to familiarize and fine-tune CNN using the FloodNet dataset to achieve accurate ID and assessment of flood-related damage to critical structures noticeable in satellite and drone imagery took both before and after flood events. By leveraging the competences of CNNs, the aim is to develop models proficient of discerning subtle changes in imagery revealing of flood damage, easing efficient and precise disaster response labours. This entails training the CNNs to recognize designs and features specific to flood-related damage, thereby allowing them to provide actionable visions for emergency responders and aid governments to effectively prioritise resources and alleviate the impact of flood events on distressed communities.

b) Employ autoencoder models tailored to the characteristics of flood images within the FloodNet dataset for real anomaly detection, enabling the ID of significant changes revealing of flood damage in pre- and post-flood images.

The research objective is to employment autoencoder models exactly customized to the unique characteristics of flood images within the FloodNet dataset to achieve real anomaly detection. This involves leveraging the capabilities of autoencoders to distinguish subtle changes revealing of flood damage in pre- and post-flood images, easing rapid and accurate ID of important alterations in flood-affected areas. By adjustment these autoencoder models to recognize anomalies exact to flood imagery, the aim is to deliver valuable insights into the extent and harshness of flood-related damage, thereby attracting disaster response efforts and aiding in the ordering of resources for mitigating the impact of flood events on substructure and groups.

c) Develop a comprehensive system that participates the trained deep learning models with real- time or near-real-time flood monitoring competences, if timely and actionable visions to emergency responders and decision-makers for effective resource allocation and flood justification strategies.

The research objective is to grow a comprehensive system that seamlessly mixes trained deep learning models with real-time or near-real-time flood nursing capabilities. By uniting the logical power of deep learning with live flood monitoring data, the system aims to deliver timely and actionable visions to emergency responders and decision-makers. This addition facilitates effectual resource allocation and enables the formulation of actual flood mitigation strategies by rapidly classifying flood-related damage and ordering response efforts in critical areas. Finally, the goal is to improve the resilience of communities and minimize the adverse impact on lives and substructure through knowledgeable decision-making and active intervention.

Potential Impact of Project

From a technical standpoint, the project holds important potential to revolutionise tragedy response efforts, mainly in the context of flood actions. By leveraging deep learning techniques and the FloodNet dataset, the growth of a comprehensive system for accelerated disaster response could lead to several key progressions. Firstly, the integration of modified and fine-tuned CNN could improve the accuracy and competence of flood-related damage assessment in satellite and drone images. This could enable spare responders to fast identify and assess damage to dangerous infrastructure easing more effective resource distribution and response strategies. The operation of modified autoencoder models for anomaly detection in flood imagery could enable the identification of subtle notions indicative of flood damage, further attracting the precision and consistency of damage assessment efforts. General, from a technical standpoint, the project has the potential to meaningfully improve the speed, accuracy, and efficiency of disaster retort efforts in flood- affected areas (Shakibaei *et al.*, 2023).

In university, the project could make several notable donations to the field of disaster organisation and deep learning research. Firstly, the growth of novel methodologies for adapting and fine-tuning CNNs using the FloodNet dataset could loan the state-of-the-art in deep learning techniques for disaster retort applications. This could lead to the magazine of research papers specifying the methodology and findings, donating to the academic discourse on deep learning-based tactics to disaster management. The exploration of modified autoencoder models for anomaly detection in flood imagery could lead to further research into innovative techniques for detecting and measuring damage in disaster-affected parts. This could generate attention among researchers and academics in the growth of more advanced deep learning models for disaster reply to applications (Varsha *et al.*, 2024). Overall, from an academic position, the project has the potential to generate new information, insights, and practices that could notify future research in the fields of disaster management and deep learning.

Summary of Methodological Approach

The methodology of this research involves several key steps to develop a deep learning- based system for advanced disaster reply, with a specific focus on flood events. Firstly, the research familiarizes and fine-tunes CNN using the FloodNet dataset, which includes satellite and drone imagery captured before and after flood events. Techniques such as transmission learning and fine-tuning are working to ensure the CNNs accurately classify and assess flood-related damage to dangerous infrastructure, buildings, and other essential features noticeable in the imagery. Exhibition measures such as precision, recall, F1 score, and accuracy will be used to assess the efficiency of the trained CNNs in wound assessment.

In the following phase, customized autoencoder models are working for real anomaly detection in deluge images. These models are handmade to the unique features of flood images within the FloodNet dataset and trained to classify important changes revealing of flood wound in both pre- and post-flood images (Rumapea *et al.*, 2024). Performance metrics such as precision, memory, F1 score, and accuracy will be used to measure the competence of the autoencoder models in detecting anomalies connected with flood damage. Lastly, the research aims to develop a comprehensive system that participates the trained deep learning models with real-time or near-real-time flood nursing capabilities. This system will deliver timely and actionable insights to emergency responders and decision-makers, allowing efficient resource distribution and flood mitigation strategies. The presentation of the integrated system will be assessed using measures such as precision, recall, F1 score, and accuracy to confirm its effectiveness in disaster response efficiency and community resilience.

Research Contribution

The research contributes to the field of tragedy management by introducing a novel method that leverages deep learning techniques, exactly CNN and modified autoencoder models, to expedite disaster retort, chiefly in the context of flood events. By adapting and fine-tuning CNNs using the FloodNet dataset, the study improves the accuracy and efficiency of flood-related damage assessment in cable and drone imagery. This donates to the development of a deep learning-based system talented of rapidly identifying and measuring damage to dangerous infrastructure, buildings, and other vital assets affected by floods. The research loans the field by introducing customized autoencoder models handmade to the characteristics of flood images within the FloodNet dataset (Deborah *et al.*, 2024).

These models allow effective anomaly detection, easing the identification of significant changes revealing of flood damage in both pre- and post-flood images. By concentrating on anomaly detection, the study improves the capability of the future system to detect subtle alterations in flood-affected areas, thereby refining the overall accuracy and dependability of damage assessment (Blomeier *et al.*, 2024).

Dissertation Overview

The rest of the dissertation is structured is as follows:

Chapter 2 is structured into five subsections, each addressing different surfaces of disaster management and technology integration. Subsection 2.1 research into the potential donations of deep learning procedures to disaster management. Subsection 2.2 specifically focuses on the operation of CNN and autoencoders in disaster management settings. In Subsection 2.3, current studies pertaining to disaster management are judgmentally reviewed, providing valued insights into continuing research efforts. Subsection 2.4 provides an in-depth inspection of a specific application: a CNN model applying the FloodNet dataset for damage assessment, deliberating methodologies, and findings from relevant studies. Finally, Subsection 2.5 bids a concise summary of the literature review, stress key insights and identifying potential streets for future research and development.

Chapter 3 is dedicated to data pre-processing, comprising nine subsections depicting specific aspects of the methodology employed in the study. It commences with an exploration of dataset collection, followed by a discussion on the data analysis approach. The subsequent subsections delve into various stages of data pre-processing, feature extraction techniques, model training methodologies, and a detailed examination of the integrated CNN + Autoencoder model. Additionally, this chapter addresses criteria for assessing model performance, outlines hardware and software configurations, and lists the tools and software utilised throughout the research project.

Chapter 4 unfolds into five subsections, each focusing on different components of the proposed CNN + Autoencoder model. An analysis of the model's performance and efficiency is presented, based on conducted experiments and analysis. Subsequent subsections discuss performance measures, error measures, hyperparameter optimization, and a comparative assessment of the proposed model with the latest models in the field.

Chapter 5 presents the results and discussion derived from the implemented methodology. It covers aspects such as the loss and accuracy of the CNN model, confusion matrix, performance measures, ROC curve, learning curve, training, and validation loss of the autoencoder, effect of different training data on the CNN model, proposed model weights and frequency, feature distribution, histogram of CNN, and comparison of the proposed model with the latest models.

Chapter 6 serves as the conclusion and outlines future work. It summarizes the achievements of the project, identifies areas for future research, and proposes avenues for further exploration and development within the field of disaster management using CNN and autoencoder methodologies.

Chapter 7 provides a space for student reflections, offering insights into personal experiences, challenges faced, and lessons learned throughout the research process.

Finally, the dissertation concludes with a reference list, providing citations for the sources referenced throughout the project.

Chapter 2 – Literature Review

Chapter 2 is segmented into five subsections, each addressing distinct aspects of disaster management and technology integration. Subsection 2.1 discusses deep learning algorithms for disaster management, exploring their potential contributions. Subsection 2.2 focuses on the application of CNN and Autoencoders in disaster management. In Subsection 2.3, current studies related to disaster management are reviewed, offering insights into recent research endeavors. Subsection 2.4 examines a specific application: a CNN model utilizing the FloodNet dataset for damage assessment. It elaborates on methodologies and findings from studies employing this model. Finally, Subsection 2.5 summarizes the literature review, highlighting key insights and potential areas for future research and development.

Deep Learning Algorithms for Disaster Management

In regions susceptible to flooding, effective disaster management and relief delivery are essential to minimize the impact on affected populations. However, conventional methods face challenges such as limited technology and accessibility, prompting the need for alternative approaches. His paper introduces a machine-learning framework tailored for autonomous drones to identify flood-affected areas through image classification. Evaluation of CNN architectures, including Inception v3 and DenseNet, demonstrates their efficacy in detecting flood severity. Inception v3 outclasses DenseNet with 83% and 81% accuracy, separately, on a self-generated flood level dataset. Overflows often disrupt outmoded relief efforts due to logistical tasks and limited access in affected areas. Independent drones offer a promising solution by if a means to prioritize assistance and bring relief goods professionally. His paper suggests leveraging machine learning techniques to improve the capabilities of independent drones in post-flood tragedy management. The methodology includes training CNN, exactly Inception v3 and DenseNet, to categorize flood-affected areas based on airborne images. Investigational results indicate the advantage of Inception v3 over DenseNet in accurately classifying flood severity, achieving 83% and 81% accuracy, correspondingly (Michail *et al.*, 2024).

In the kingdom of image processing, de-noising plays a crucial role in attractive the quality of images by eliminating unwanted aberrations. This process finds requests across various domains, counting disaster management, where clear and accurate images is essential for decision-making. His research suggests a novel method, termed the Profound Memory-Affiliated Neural Network (PMANN), which leverages deep learning methods for de-noising aerial images in the context of disaster management scenarios. This optimization strategy aims to recover the effectiveness of the de-noising process, thereby attractive the overall quality of the imagery. Likened to existing methods, the future architecture demonstrates larger presentation in both de-noising and disaster management requests. To evaluate the efficiency of the PMANN framework, a relative analysis is conducted with some state- of-the-art de-noising techniques using the valuation metrics comprise the PSNR, SSI, and MSE. Results indicate that the PMANN model outdoes the baseline methods across all assessment metrics. Exactly, the PSNR value of the future PMANN is observed to increase by 0.24%, 0.086%, and 0.643% associated to the CNN, WNNM and CNN-LSTM models, separately (Islam *et al.*, 2023).

Earthquake calculation stands as a critical aspect of disaster management and community care efforts, given the potentially catastrophic significances of seismic events. The Hybrid CNN-GRU model combines CNN's aptitude to capture complicated spatial features within seismic data with GRU's effectiveness in judicious evolving patterns of seismic movement over time.

In the seismic intention study conducted, the Hybrid CNN-GRU model attained extraordinary results with a tall accuracy rate of 98.67% (Raj *et al.*, 2023). The paper starts by introducing the fundamental notions and structures of CNN and ML methods, along with the steps complicated in LSA. Results indicate that CNN attains the highest accuracy (86.41%) and the uppermost Area Under the Curve (AUC) value of 0.9249 in LSA. The larger accuracy and concentrated opportunity of landslide occurrence achieved by CNN hold important implications for tragedy prevention and justification efforts, easing the efficient allocation of resources. More research could focus on refining CNN models to address present limitations and increase their applicability in landslide exposure assessment (Puthran, 2024).

Detecting, preventing, and extinguishing forest fires pose significant challenges due to their rapid spread and destructive nature. Moreover, forest fires can lead to habitat destruction, resulting in substantial material and moral losses. The objective is to detect forest fires in the dataset using various deep learning algorithms, including InceptionV3, DenseNet121 and ResNet50V2, along with hybrid models proposed with SVM, RF, BiLSTM, and GRU algorithms. In the classification study with the Fire Luminosity Airborne-based Machine Learning Evaluation dataset, the DenseNet121 model achieved the highest accuracy of 97.95% when initialized with random weights. In the transfer learning study applying ImageNet weights, the DenseNet121 model showed satisfactory performance, attaining an accuracy of 99.32% (Jiang *et al.*, 2023).

In literature, the integration of developing knowledges such as deep learning algorithms presents promising avenues for ornamental disaster management strategies crossways various domains. The application of CNN and other progressive techniques facilitates more effective and accurate detection, assessment, and rejoinder to natural disasters, reaching from floods to earthquakes. For example, in flood-prone regions, the operation of machine learning frameworks handmade for autonomous drones enables the rapid documentation of affected areas through image organization, thereby streamlining relief efforts and minimizing the impact on precious populations. Similarly, in the setting of earthquake detection, the fusion of CNNs with recurring neural networks (RNNs) yields excellent results in taking complex spatial and temporal patterns, authorizing pre-emptive measures and ensuring timely interventions to alleviate the consequences of seismic events.

Disaster Management using CNN and Autoencoders

Landslides represent a significant natural disaster globally, particularly in hilly terrains characterized by high slopes and numerous lineaments. Recent years have seen a surge in landslide hazard zonation studies leveraging geospatial technologies. His paper focuses on employing advanced techniques such as AI and GIS to delineate Landslide Susceptibility Zones (LSZ). The study area designated for his research is the alpine region of Sirumalai, a part of the Palani Hills situated in Dindigul taluk. IRS P6 satellite imagery helps as the primary data source for making various thematic layers vital for LSZ mapping. The Weighted Overlay algorithm within GIS is working to integrate these thematic layers and make the LSZ map. To recover the accuracy of landslide detection, advanced AI methods, specifically the multilayer feedforwarded CNN, are used. Training datasets are working to train the CNN algorithm for victory detection. Performance evaluation metrics are applied to assess the effectiveness of CNN networks and associate the results with Recurrent Neural Networks (RNNs). The projected classification performance metrics establish superior presentation within the CNN architecture, elastic more accurate results likened to RNNs. The LSZ map produced using the integrated method is validated using remaining landslide records within the physical area of interest (Reis and Turk, 2023).

The propagation of unmanned aerial vehicles (UAVs) across numerous domains has spurred efforts to prepare these devices with advanced intelligence, allowing them to execute complex tasks autonomously. Chiefly in disaster response scenarios, where substructure deficiencies often incumber real-time data analysis, there is a rising need to integrate artificial intelligence (AI) techniques at the advantage. This allows UAVs to detention and process information independently, thereby easing swift and real decision-making. The selected DNN algorithms are enhanced for implementation on GPU- based edge computing platforms, enabling organization to be carried out onboard UAVs. Later, only the algorithm outputs are communicated to the cloud for real-time monitoring of underwater areas. By implementing organization tasks onboard UAVs, their method reduces reliance on external infrastructure and reduces network resource ingesting. This not only improves the conservation sustainability of the adversity response process but also recovers resilience against connectivity disruptions. Their experimental results, showed across diverse hardware formations and architectures, establish the possibility of performing advanced real-time image processing tasks using sophisticated DNN-based solutions. In summary, their research donates to the progression of UAV autonomy in tragedy scenarios, particularly in flood answer efforts (Subhashini *et al.*, 2022).

Satellite imagery helps as a valuable tool for monitoring global environmental features over space and time, easing various applications across multiple punishments. In his study, writers propose a novel method smearing spatiotemporal Long Short-Term Memory (LSTM) convolutional autoencoders to competently reconstruct inattentive data caused by thick cloud meddling in MODIS AOD data. Traditional methods for gap sufficient often fight with large, continuous blocks of overcast pixels and the high variability of AOD values over time. Their proposed technique aims to address these tasks by leveraging the capabilities of LSTM networks to detention temporal dependencies and the altitudinal information encoded by convolutional autoencoders. By leveraging both spatial and sequential information, their method proposals a promising solution for rebuilding missing AOD values caused by thick mist interference, thereby attractive the usability and reliability of satellite-based ecological monitoring datasets (Hernández *et al.*, 2022).

Smart devices linked to the Internet of Things (IoT) are playing a progressively pivotal role in the evolution of smart cities, which are unceasingly advancing technically. One area where IoT-based monitoring systems, ambitious by artificial intelligence (AI), establish significant potential is in attractive disaster management competences, ultimately leading to the good of lives and the mitigation of stuff damage. The integration of AI-driven monitoring systems into disaster management frameworks enables rapid identification of critical conditions and facilitates swift rejoinder efforts. Visual equipment, now widely accessible and interconnected through IoT substructure, allows for real- time situational analysis and the identification of actionable interferences. The Region-based Convolutional Neural Network (R-CNN) algorithm is working as a foundational tool for object detection and tracking. R-CNN excels in removing relevant structures from visual data, if a holistic view of the monitored environment. In spare scenarios, this method offers a nuanced understanding of the condition by combining multiple visual cues to leader response actions effectively (Daniels *et al.*, 2022).

Landslides represent a significant geological hazard, particularly in highland and mountainous regions, posing serious threats to human life and the environment. The study area chosen for investigation is Icheon City, South Korea, benefiting from the availability of a precise landslide inventory dataset. The integration of deep learning algorithms with metaheuristic optimization algorithms offers several advantages in landslide susceptibility mapping. By leveraging the capabilities of CNNs to extract intricate spatial patterns from landslide-related factors, coupled with the optimization prowess of GWO and ICA in fine-tuning model parameters, more accurate and robust susceptibility models are achieved. The improved accuracy of the proposed models underscores their suitability for supporting effective landslide risk assessment and management strategies in Icheon City and similar regions prone to landslide hazards (Kumar *et al.*, 2022).

Current Studies related to Disaster Management

Natural disasters, stemming from uncontrollable natural processes, can wreak havoc on human, animal, and plant life, often resulting in substantial material and emotional losses. Among these calamities, floods, and tsunamis stand out as particularly devastating events, necessitating swift and effective search and rescue operations to mitigate the impact and save lives. To evaluate its performance, they compare the results obtained by FASegNet with those from common image segmentation models utilized across various domains. The primary attention is on measuring the model's accuracy and competence in flood area detection, crucial for enhancing rescue efforts and minimalizing response time. In their research, FASegNet establishes superior performance metrics compared to existing segmentation models, reaching higher accuracy rates while applying fewer strictures. Accurately, when tested on both the "Flood Area" and "Water Body" datasets without previous pretraining and data growth, FASegNet achieves mean Joining over Union (mIoU) accuracies of 84.3% and 84.5%, distinctly (Sener *et al.*, 2024).

Urban floods pose important threats to both economic and social assets, necessitating actual flood charting tools for disaster reply. Remote sensing, chiefly high-resolution (HR) optical imagery, has established to be instrumental in flood mapping labors. This component focuses on inferring HR DEMs from existing low-resolution (LR) DEMs using a fusion approach. A new deep learning-based upsampling network is developed to enhance the resolution of DEMs, enabling the generation of HR DEMs from LR counterparts (Hakim *et al.*, 2022). In his paper, authors introduce a novel approach called the urban waterlogging risk evaluation network (WaRENet) designed to assess the risk of waterlogging in urban areas. Meanwhile, the MCROD module achieves a mean Average Precision (mAP) of 54.9%, demonstrating robust performance in detecting reference objects within waterlogged scenes. Notably, despite the high accuracy achieved, the processing speed remains high at 70.04 frames per second (fps), ensuring real-time applicability of the system (Tan *et al.*, 2024).

In the aftermath of natural disasters, image data collected serves as crucial evidence for forensic analysis of structural failures. However, the management and curation of large volumes of post-disaster imagery present significant challenges. Oftentimes, data users must expend considerable effort sifting through extensive image archives spanning decades to locate relevant images for studying specific types of disasters. To address this issue, his paper proposes a novel machine learning-based method aimed at automating the labelling and classification of vast quantities of post-natural disaster image data (Yu *et al.*, 2024). Authors present a comprehensive dataset collected during the catastrophic 2019 Central US Flooding events, which lasted for more than two seasons in Mississippi and Missouri River tributaries. The availability of such a dataset will enable researchers to develop and test innovative methodologies for leveraging RS imagery in predicting and mitigating the impacts of future natural hazards. Ultimately, their goal is to advance the field of Earth observation and contribute to more effective decision-making and disaster response efforts (Ro *et al.*, 2024).

Floods represent one of the most frequent natural disasters occurring worldwide on an annual basis. Addressing the aftermath of floods requires timely assessment of damage magnitude and efficient execution of rescue operations, security deployments, and resource allocation to affected areas. However, obtaining accurate information during post-flood crises poses significant challenges. To address these challenges, his work proposes a novel deep learning semantic segmentation model aimed at reducing the loss of multi-scale features and enhancing global context awareness. Both quantitative and qualitative analyses were performed to assess model performance (Li *et al.*, 2024). His paper explores UAV-based aerial imagery coupled with CNNs for flood detection, aiming to extract flood-related features from disaster zone images. By facilitating prompt responses to emergent disasters, the model contributes to the smart governance of cities, ensuring effective disaster mitigation and recovery efforts. Overall, leveraging UAV-based imagery and CNNs for flood detection holds great potential in enhancing disaster management capabilities worldwide (Khan and Basalamah, 2023a).

CNN Model using FloodNet dataset for Damage Assessment

An intelligent earthquake signal detector designed to enhance traditional disaster response mechanisms by providing automated assistance. Addressing crises effectively often requires additional sensors and automation. Leveraging the success of deep learning in tasks with low signal-to-noise ratios, authors propose a novel approach based on a 3-dimensional (3D) CNN combined with RNN for earthquake detection, transitioning from a demonstration paradigm to real-time implementation. They utilize data from the STanford EA rthquake D ataset (STEAD) for training the network (Munawar *et al.*, 2021). The proposed technique employs a descriptive method to detect earthquake-induced ground failure areas and damaged structures by analysing satellite photographs obtained from Google Earth using a pretrained Faster R-CNN model. The analyses were conducted at different image scales. His technique offers a valuable tool for identifying and assessing areas at risk of ground failure and structures prone to earthquake damage, facilitating rapid response and mitigation efforts in the aftermath of seismic events (Shakeel *et al.*, 2021). Traditional methods for damage assessment typically involve on-site evaluations and consensus-based techniques. However, with the rise of unmanned aerial vehicles (UAVs), there has been a shift towards utilizing these platforms for disaster response and mitigation due to their affordability, ease of operation, and on-demand deployment capabilities. Through these improvements, the best performing Mask-RCNN model achieves a mean average precision (mAP) of 51.54% for pixel-level segmentations of countable objects (Pi *et al.*, 2021).

During inference, the model takes either real-time video streams or pre-recorded videos as input for disaster detection. Each frame of the video stream is passed through the model sequentially, and the model computes the probability of each class for that frame. The class label with the highest chance is selected as the predicted tragedy type for that frame. The model then displays the foretold disaster label on the top of the frame, along with the consistent probability intended by the model. In cases where no tragedy is detected, the model outputs "normal." However, if a tragedy is predicted, the model shows the specific type of disaster recognized by the highest likelihood class label (Agrawal and Meleet, 2021). In his paper, authors suggest a novel fire disaster discovery method capable of treatment images with varied scales. Their method integrates several key components to address this task. Firstly, they employ dense connections to improve the flow of information between different layers of the network. This simplifies more effective feature extraction and representation, enabling the model to capture intricate details crossways different scales. Experimental results prove the effectiveness of their method, achieving an accuracy of 91.4% when trained on fixed scale images and 92.4% when trained on multiscale images (Feng and Sun, 2024).

The primary drive is to propose an automated system for precise tragedy detection, with a exact focus on flood detection. By utilizing real-time image processing, the system can notice floods promptly. The investigational results demonstrate the efficiency of the proposed approach, with GoogleNet attaining the highest accuracy of 92.50%. Also, the second-highest performance was observed with AlexNet, achieving an accuracy of 92.20%. These answers underscore the potential of deep learning techniques, mainly CNN architectures, in precisely detecting floods from image data (Hussain *et al.*, 2024). In his study, writers focus on the discovery of road damage resulting from the 6 February 2023 Gaziantep- Kahramanmaras earthquakes, which had important and fatal consequences. Upkeep and repair of roads are crucial for transport infrastructure, ensuring safe and effective travel. However, natural disasters or human issues can lead to road damage, requiring timely detection and repair. Their investigational results revealed that the SVM algorithm, in combination with the EL method, achieved the most successful consequences, elastic an accuracy of 98.68% (Reis *et al.*, 2024).

Landslides pose a significant threat in China, causing considerable property damage and loss of life. Rapid and accurate identification of landslide areas is crucial for effective disaster prevention and mitigation efforts. Currently, Faster Region-based Convolutional Neural Networks (Faster R-CNN) have shown effectiveness in landslide detection. However, the deep feature extraction capabilities of Faster R-CNN can lead to the problem of gradient vanishing during training, hindering the adjustment of network layer weights. To address this subject, his paper suggests a novel remote-sensing image object detection procedure based on Faster R-CNN. General, the proposed algorithm offers improved performance in landslide detection, highlighting its potential for real disaster management and justification strategies (Qin *et al.*, 2024). The innovative methodologies and datasets propel disaster response efficacy in flood and landslide scenarios shown in the Table 2.1.

Table 2.1. Innovative methodologies and datasets propel disaster response efficacy in flood and landslide scenarios.

Author	**Year**	**Methodology**	**Dataset**	**Performance Measures**	**Key Findings**
Islam *et al.*	2023	CNN architectures, including Inception v 3 and DenseNet	Self-generated flood leve l dataset	Precision, Recall, F1 Score , Accuracy	Enhancing disaster response efforts in flood-affected regions

Raj *et al.*	2023	Profound Memory-Affiliated Neural Network (PMANN)	Aerial images dataset	AUROC, F1 PSNR, SSI, MSE	Facilitating more accurate decision-making in disaster response scenarios
Jiang *et al.*	2023	CNN, Landslide Susceptibility Maps (LSMs), Decision Tree, Naïve Bayes	LandSlide DatasetNS 2	AUC, Accuracy, Recall, Precision, F1 Score	Advancing landslide risk assessment techniques by enhancing their applicability in landslide susceptibility assessment
Khan *et al.*	2023	Autoencoder	FloodNet dataset	Precision, Recall, F1 Score, Accuracy	Promising solution for accurately assessing flood damage and guiding response operations
Munawar *et al.*	2021	CNN, Autoencoder	FloodNet (pre- and post-disaster images) dataset	Recall, F1 Score, Accuracy	Ensuring effective disaster mitigation and recovery efforts
Shakeel *et al.*	2021	CNN+RNN	STanford EArthquake Dataset (STEAD)	Accuracy, Sensitivity, Specificity	Enhanced capabilities for timely and accurate earthquake detection

Hacıefendioğlu *et al.*	2021	Faster R-CNN model	Ground failure areas and damaged structures images dataset	Recall, F1 Score, Computational Efficiency	Facilitating rapid response and mitigation efforts in the aftermath of seismic events
Pi *et al.*	2021	CNN, Mask-RCNN, Pyramid Scene Parsing Network (PSPNet)	FloodNet dataset	AUROC, F1 Score, mAP, mIoU	Providing a robust framework for automated object detection and segmentation in aerial footage
Daniels *et al.*	2022	Long Short-Term Memory (LSTM) convolutional autoencoders	MCD19A2 Aerosol Optical Depth (AOD) dataset	PSNR, R^2, SSIM	enhancing the usability and reliability of satellite-based environmental monitoring datasets
Kumar *et al.*	2022	Region-based Convolutional Neural Network (R-CNN)	visual disaster dataset	Precision, Recall, F1 Score, Accuracy	Minimizing property damage in diverse disaster scenarios
Şener *et al.*	2024	Flood Area Segmentation Network (FASegNet)	Flood Area and Water Body datasets	Precision, Recall, MSE, MAE, mIoU	Contributes to the ongoing efforts to improve disaster response capabilities in the context of flood and tsunami events
Tan *et al.*	2024	DEM Upscaling, Rapid Flood Segmentation, GIS-Based Floodwater Estimation	FloodNet dataset	Accuracy, Sensitivity, Specificity	Enable accurate and efficient flood mapping, facilitating timely disaster response

Yu *et al.*	2024	Atrous convolution (SAC) into the YOLOv5	UWRDataset	AUROC, F1 Score, mAP, mIoU	Comprehensive solution fo r evaluating urban waterlogging risk
Ro *et al.*	2024	Natural languag e processing (NLP) model	Post-disaster image dataset	Precision, Recall, F1 Score , Accuracy	Efficiency and effectiveness of post-disaster image data analysis
Puthran *et al.*	2024	Hybrid CNN + GRU	FloodNe t dataset	AUROC, F1 Score, Accuracy	Enhancing earthquak e forecastin g accuracy, making it a valuable contribution to early warnin g systems
Our Model	**2024**	**Convolutional Neural Network (CNN)+ Autoencoder**	**FloodN et dataset**	**Precision, Recall, F1 Score and Accuracy**	**Combining Convolutional Neural Network (CNN) with a Autoencoder, our model achieves superior performance in Disaster Management of FloodNet data for Damag e Assessment while maintaining computational efficiency.**

Conclusion of Literature Review

The literature review focused on the dangerous area of disaster management, chiefly on the assessment of damage experienced during such events, with a specific stress on the FloodNet dataset. This examination unearthed a wealth of scholarly discourse nearby the challenges and chances inherent in evaluating the outcome of disasters, particularly those stopping from flooding events. Through a meticulous inspection of current research, it became obvious that the FloodNet dataset helps as a linchpin for detectives seeking to unravel the problems of flood-induced injury. By leveraging progressive performs and technologies like machine learning and distant sensing, academics have made important strides in attractive the accuracy and timeliness of damage assessment, thereby increasing disaster response and recovery efforts.

Chapter 3 – Data Pre-processing

In the chapter on Data Pre-processing, we embark on a critical phase of our methodology, ensuring that the FloodNet dataset is appropriately curated and prepared for subsequent machine learning modelling. This section comprises several subsections, each addressing vital aspects of our approach. We commence with a detailed exploration of the dataset itself (subsection 3.1), followed by an investigation into our chosen data analysis approach (subsection 3.2). Specifically, we delve into the pre-processing steps undertaken to optimize the dataset for model training (subsection 3.2.1), as well as the techniques employed for feature extraction (subsection 3.2.2). Subsequently, we discuss our rationale behind selecting model architectures (section 3.3), followed by the processes of model training (subsection 3.3.1), disaster prediction (subsection 3.3.2), and performance evaluation (subsection 3.3.3). Through these organised steps, we aim to develop accurate and reliable models for disaster prediction, leveraging the capabilities of CNNs and autoencoders on the FloodNet dataset.

Dataset Description

The FloodNet dataset represents an important resource in the field of disaster retort and machine learning. Advanced in the aftermath of Hurricane Harvey, it includes a robust collection of 2343 high- resolution aerial images taken by DJI Mavic Pro quadcopters. These images are accurately categorized into training, justification, and test sets, which are important for the development and evaluation of machine learning models, mostly those focused on image organization tasks. The primary objective of the FloodNet dataset is to provide a full visual account of areas affected by flooding. This enables researchers and practitioners in the field of disaster management to gain a deeper understanding of the influence of floods on various landscapes and organisations. Availability is a key feature of the FloodNet dataset, as it is made existing on widely used platforms like GitHub and arXiv. This certifies that the dataset can be leveraged by a broad spectrum of entities and organizations complicated in computer vision and natural disaster research (GHOSH *et al.*, 2022).

In this study, we leverage the FloodNet dataset, a comprehensive gathering comprising 2343 high- resolution UAV images opposite with segmentation masks. The dataset is meticulously partitioned into training, validation, and test sets, encompassing 1445, 450, and 448 images, respectively. Each image is accompanied by a segmentation mask outlining 10 distinct classes, including background, flooded and non-flooded buildings, flooded and non-flooded roads, water bodies, trees, vehicles, pools, and grass. Notably, the inclusion of the "water" class serves to differentiate natural water bodies like lakes or rivers from floodwater, with flooded buildings defined as those in contact with floodwater on any side. Image and mask sizes vary between 3000 x 4000 and 3072 x 4592; however, for compatibility with CNN and autoencoder models, we resize images and masks to 512 x 512. Additionally, pre-trained models necessitate input image normalization within the range [0,1] using specified mean and standard deviation values. The characteristics and specifications of the FloodNet dataset is represented in the Table 3.1.

Table 3.1. Characteristics and specifications of the FloodNet dataset.

Characteristic	Description
Dataset Name	FloodNet
Source	DJI Mavic Pro quadcopters
Number of Images	2343
Resolution	High-resolution
Categories	Training, validation, test
Accessibility	GitHub, arXiv
Objective	Flood area visualization
Benefits	Collaboration, innovation
Purpose	Disaster management enhancement
Image Sizes	Varied (3000 x 4000 or 3072 x 4592), resized to 512 x 512
Classes	Background, flooded/non-flooded buildings, roads, water bodies, trees, vehicles, pools, grass
Mask Sizes	Same as input images (512 x 512)
Class Definitions	Includes "water" class to differentiate natural water bodies from floodwater; flooded buildings are those in contact with floodwater on any side
Pre-processing	Missing value treatment, feature regularization, normalization
Augmentations	Horizontal and vertical flipping, colour jitter

The FloodNet dataset showcases a various collection of images arresting both flood and non-flood scenarios, if a comprehensive visual representation of areas precious by flooding. The flood images depict various landscapes and infrastructures flooded with water, showcasing the extent and influence of flood events on urban and natural environments as shown in the Figure 3.1 and Figure 3.2. Structures, roads, and other structures submerged in water convey the strictness of flooding and its inferences for communities. In distinction, the non-flood images present scenes of unpretentious regions, portraying intact infrastructure, undergrowth, and natural water bodies.

Figure 3.1. Exploring the visual dynamics of flood and non-flood scenarios through the FloodNet dataset.

Figure 3.2. The top image depicts a real scenario, while the subsequent five images showcase synthetic flooding in various locations with varying extents of coverage.

The collection includes flood images taken at varying distances, contribution a spectrum of perspectives extending from close-up shots to wide-angle views, if comprehensive coverage of flood- affected parts as demonstrated in the Figure 3.3.

Figure 3.3. From close-up shots to wide-angle views, a diverse collection of flood images offers comprehensive coverage of flood-affected areas.

Data Analysis Approach

The data analysis methodology dynamic in this research contains the application of CNN and autoencoders to the FloodNet dataset for disaster intention. The approach includes several key steps handmade to the features of the dataset and the purposes of the research (Rahnemoonfar *et al.*, 2021).

Pre-processing

Pre-processing is an energetic phase where the raw dataset knowledges necessary alterations to ensure its suitability for machine learning algorithms.

In the example of the FloodNet dataset, pre-processing contains several tasks:

- Handling missing values: Absent data points can harmfully disturb model presentation. Imputation methods such as mean imputation or shout may be active to address absent values (Jackson *et al.*, 2023).
- Feature regularisation: Standardising or normalising features certifies that each feature donates comparably to the model's learning process, foiling bias towards positive features.
- Data formatting: The dataset may need to be ready into apposite structures advantageous to the CNN and autoencoder architectures. For example, images may need to be resized or normalised before nursing into the models.

Feature Extraction

The image represents the CNN architecture used for feature abstraction from the input images. The input images go through a sequence of convolutional and combining layers, which regularly extract more complex and mental topographies from the raw pixel data (Seydi *et al.*, 2022). The convolutional layers smear a set of learnable filters to the input, taking local spatial relationships, while the combining layers reduce the dimensionality of the feature charts, recollecting the most striking info. The Figure 3.4 illustrates the collective feature map sizes and the discount in spatial dimensions as the image passes through the network, eventually producing a final set of structures that can be used for numerous tasks such as classification or disaster prediction.

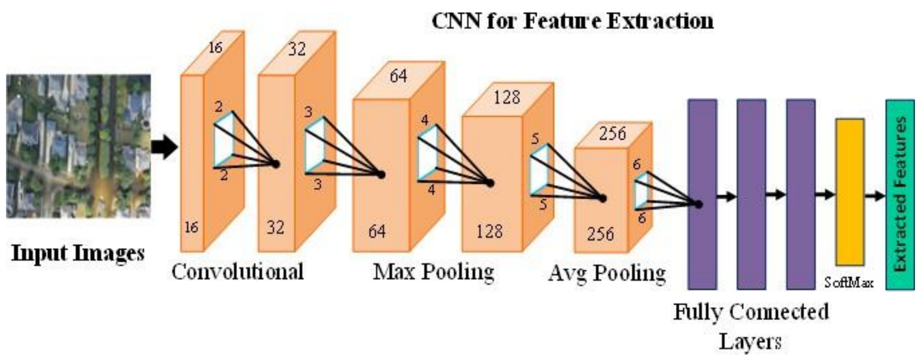

Figure 3.4. Visualization of the CNN feature extraction process, showcasing the transformation of input images through convolutional and pooling operations.

In this equation, $Y[i, j]$ signifies the output feature map at situation (i, j) after applying the convolution procedure. X denotes the contribution feature map, W signifies the learnable convolutional screen, and b is the bias term. The convolution operation includes sliding the filter over the contribution feature chart and calculating the element-wise increase between the filter and the corresponding area of the input, followed by summation.

$$Y[i, j] = \sum_{m} \sum_{n} X[i - m, j - n] \times W[m, n] + b \qquad (1)$$

The repaired linear unit (ReLU) beginning function is applied elementwise to the output of the difficulty operation. It announces non-linearity into the network by location adverse ideals to zero while departure positive standards untouched. This benefits the network learn composite patterns and improves its aptitude to capture non-linear relationships within the data.

$$ReLU(x) = \max(0, x) \qquad (2)$$

In this equation, $Y[i, j]$ signifies the output of the max combining operation at position (i, j). X is the contribution feature map, and s is the stride size. The max combining operation partitions the contribution feature map into non-overlapping areas and productions the extreme price within each region, effectively down sample the feature map and reducing its spatial dimensions.

$$Y[i, j] = max_{m,n} X[i \times s + m, j \times s + n] \qquad (3)$$

$$Z = \sigma(W . X + b) \qquad (4)$$

Here, Z signifies the output of the fully related layer, X denotes the input feature trajectory, W is the bulk matrix, b is the partiality vector, and σ signifies the beginning purpose (e.g., sigmoid or softmax). The fully linked layer attaches each neuron in the input to each neuron in the output, allowing the network to learn compound relations between features crossways the whole input space.

The softmax purpose is practical to the production of the fully connected layer to convert the raw slashes into probabilities. It exponentiates each score and normalises them by separating by the sum of all exponentiated scores. This certifies that the production prospects sum to one, making it appropriate for multi-class organisation responsibilities.

$$Softmax(x_i) = \frac{e^{x_i}}{\sum_j e^{x_j}} \qquad (5)$$

Model Architecture Selection

Choosing suitable neural network architectures is critical for operative disaster prediction.

- CNN: CNNs are suited for image analysis responsibilities due to their aptitude to automatically learn hierarchical constructions from uncooked pixel data. Trendy the situation of the FloodNet dataset, CNNs can citation spatial features from aerial images, allowing the documentation of flood-affected areas (Bashir *et al.*, 2024).
- Autoencoders: Autoencoders are unverified learning models used for dimensionality reduction and feature removal. By reducing and then reconstructing input data, autoencoders can detain meaningful symbols of the dataset's latent features. In the case of FloodNet, autoencoders can help classify patterns suggestive of flood actions within the image data.

Model Training

- Once the architectures are selected, the models need to be competent using the pre-processed FloodNet dataset.
- Training involves enhancing model parameters to minimalize prediction errors on the training data.
- For CNNs, training entails iterative forward and regressive passes through convolutional layers, apprising weights to curtail the difference between predicted and actual flood existences in the images.
- Autoencoder training involves minimising reconstruction errors between input images and their corresponding reconstructions, effectively learning compressed representations of flood- affected areas.

Disaster Prediction

- Trained models are then utilised to predict disaster occurrences in unseen or future data.
- CNNs analyse new aerial images to identify regions exhibiting characteristics indicative of flooding, such as water accumulation, changes in landscape coloration, or structural damage.
- Autoencoders detect anomalies within the images that deviate significantly from learned patterns, signalling potential flood events.

Performance Evaluation

- The presentation of the disaster prediction models is measured using a range of evaluation metrics tailored to the specific task.
- Metrics such as accuracy, precision, recall, F1-score, and area below the bend (AUC) are computed to quantify the models' effectiveness in predicting flood events.
- Additionally, visual inspection of model predictions against ground truth labels can provide qualitative insights into model performance and areas for improvement.

Through a systematic method encompassing pre-processing, model selection, training, forecast, and evaluation, the research goals to mature accurate and reliable models for disaster prediction using CNN and Autoencoder methodologies on the FloodNet dataset.

Chapter 4 – Proposed CNN and Autoencoders

In the chapter on proposed CNN and autoencoders, we present a comprehensive methodology for disaster prediction. Beginning with a detailed workflow from data pre-processing to prediction (subsection 4.1), we emphasize systematic processing. We then discuss data pre-processing and model architecture (subsection 4.2), followed by an integrated CNN and autoencoder design (subsection 4.3). Additionally, we introduce our segmentation process (subsection 4.4) and the Flood Detection Network (subsection 4.5). This approach aims to develop accurate disaster prediction models using the FloodNet dataset.

Comprehensive Workflow: From Data to Disaster Prediction

The workflow diagram presented in the Figure 4.1 frameworks a comprehensive and systematic method to disaster estimate using the FloodNet dataset and a grouping of CNN and Autoencoders. The process initiates with the critical step of Data Collection and Preparation, where the FloodNet dataset is sensibly curated and pre-processed to safeguard its suitability for the following machine learning models. The next stage includes Building the CNN + Autoencoder Architecture, which leverages the controlling abilities of these neural network models to cite meaningful features and patterns from the inflight images. The CNN component is answerable for capturing the spatial characteristics of the data, allowing the identification of flood-affected areas, while the Autoencoder component learns trampled symbols of the input, letting for the detection of anomalies that may indicate potential disaster events. The workflow diagram illustrates the various layers and processes complicated in the CNN and Autoencoder architectures if a clear visual picture of the model's structure.

With the architecture in apartment, the research methodology improvements to the Model Training phase, where the models are enhanced using methods such as cross-entropy loss, stochastic incline descent, and hyperparameter tuning. The trained models are then assessed through Validation and Hyperparameter Tuning, certifying their robustness and presentation. Finally, the Disaster Prediction Based on CNN + Autoencoders is directed, and the results are compared with Previous Methods to assess the efficiency of the proposed method. The workflow concludes with brief key findings and suggestions of the research.

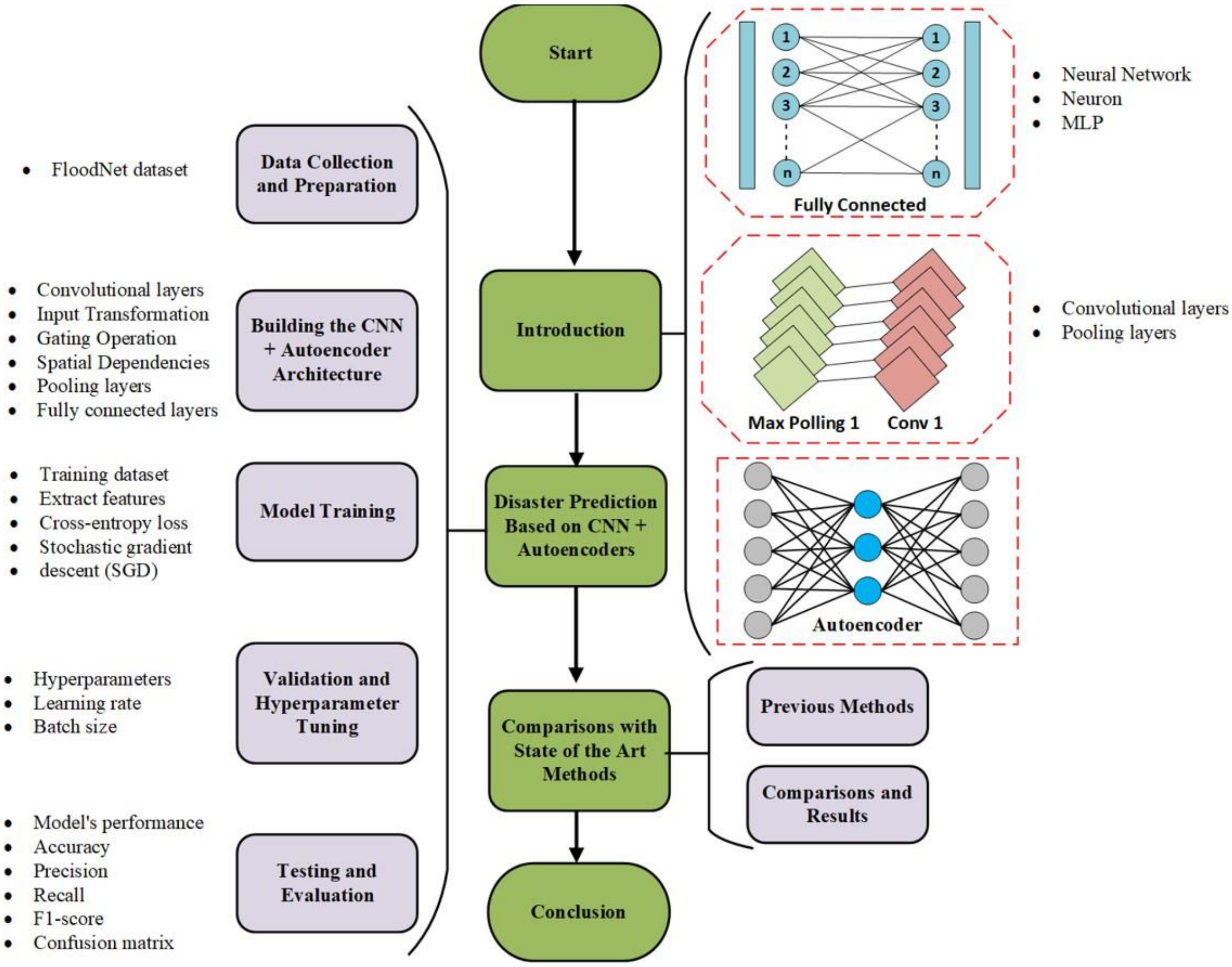

Figure 4.1. The comprehensive workflow diagram outlines the step-by-step process for disaster prediction using the FloodNet dataset and a combination of Convolutional Neural Networks and Autoencoders.

Comprehensive Data Pre-processing and Model Architecture

The data pre-processing phase is a critical step in the research methodology, certifying the FloodNet dataset is enhanced for effective machine learning modelling. This phase includes several key steps, counting image resizing to standardise dimensions, removal of inconsistencies, and class opposite to address any differences in the dataset. Also, noise reduction methods and normalization are applied to recover the quality and suitability of the data for the subsequent CNN and autoencoder models. The pre-processing steps strained in the workflow diagram, such as data growth, further supplement the dataset by generating synthetic trials, eventually improving the models' capacity to streamline and perform accurately on unseen data. The research methodology services a grouping of CNN and Autoencoders to task the disaster estimate using the FloodNet dataset. CNNs, recognised for their excellent performance in image analysis, are leveraged to extract spatial features from the aerial images, permitting the documentation of flood-affected areas. Autoencoders, on the other hand, are practical to learn trampled representations of the input data, letting the detection of anomalies that may indicate potential flood actions. The workflow diagram illustrates in the Figure 4.2 request of combining and convolution operations within both the CNN and autoencoder architectures, rank their complementary parts in the general disaster prediction process.

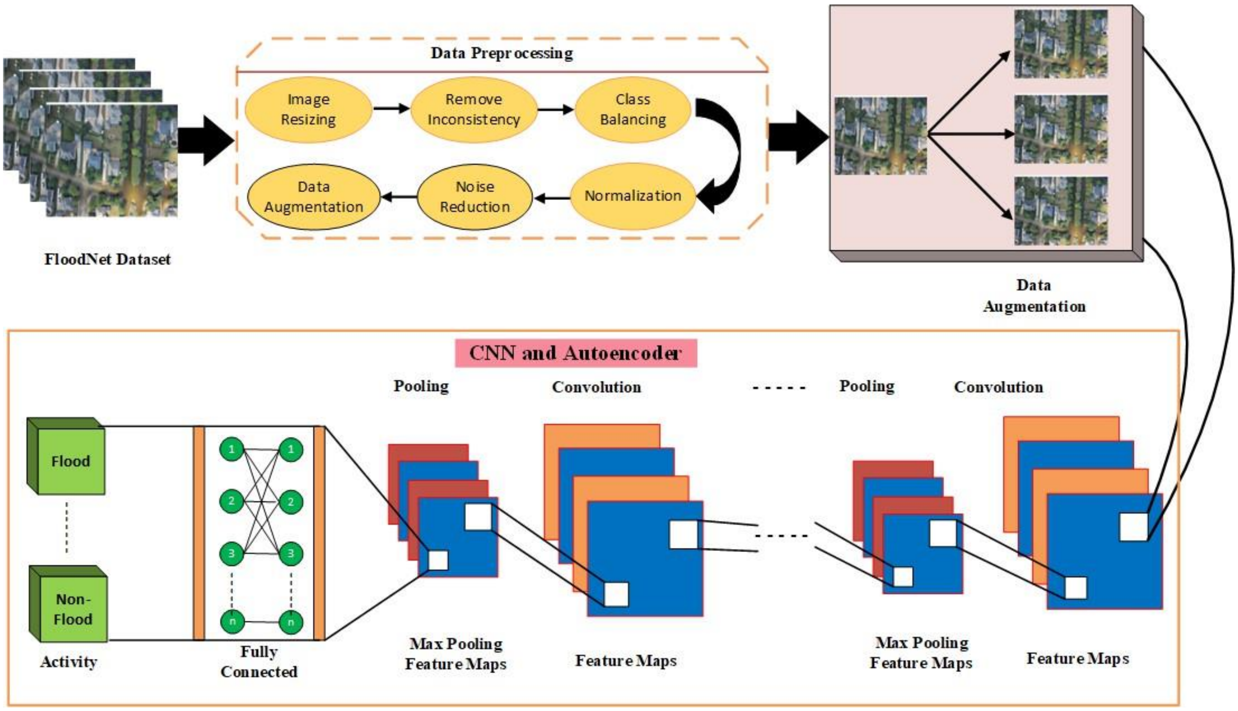

Figure 4.2. The diagram presents the comprehensive workflow of the research methodology, outlining the key steps involved in data pre-processing and the implementation of CNN and autoencoder models for disaster prediction using the FloodNet dataset.

Integrated CNN and Autoencoder Architecture for Disaster Prediction

The future model assurances the metiers of CNN and Autoencoders to produce a powerful framework for disaster prediction using the FloodNet dataset. The CNN component is responsible for extracting sensitive spatial features from the input aerial images, leveraging a classification of convolutional and uniting layers. As shown in the diagram, the input images go through a resizing operation to standardise the dimensions, followed by the convolutional (Conv) and pooling (Pool) layers. These layers slowly reduce the spatial dimensions while united the number of feature maps, permitting the CNN to detain hierarchical depictions of the input data. The Autoencoder section, on the other hand, learns a crushed depiction of the input images, letting the detection of anomalies that may designate potential disaster events. The Autoencoder is constructed with an hourglass-like construction, where the input is first resolute into a low-dimensional hidden space and then recreated back to the original dimensions. This process of compression and rebuilding allows the Autoencoder to classify patterns and characteristics in the data that diverge from the learned representations, which can be critical for identifying flood-affected areas. The diagram illustrates in the Figure 4.3 consistent nature of the CNN and Autoencoder architectures, with the removed features from the CNN portion as input to the Autoencoder, development a synergistic method to disaster prediction.

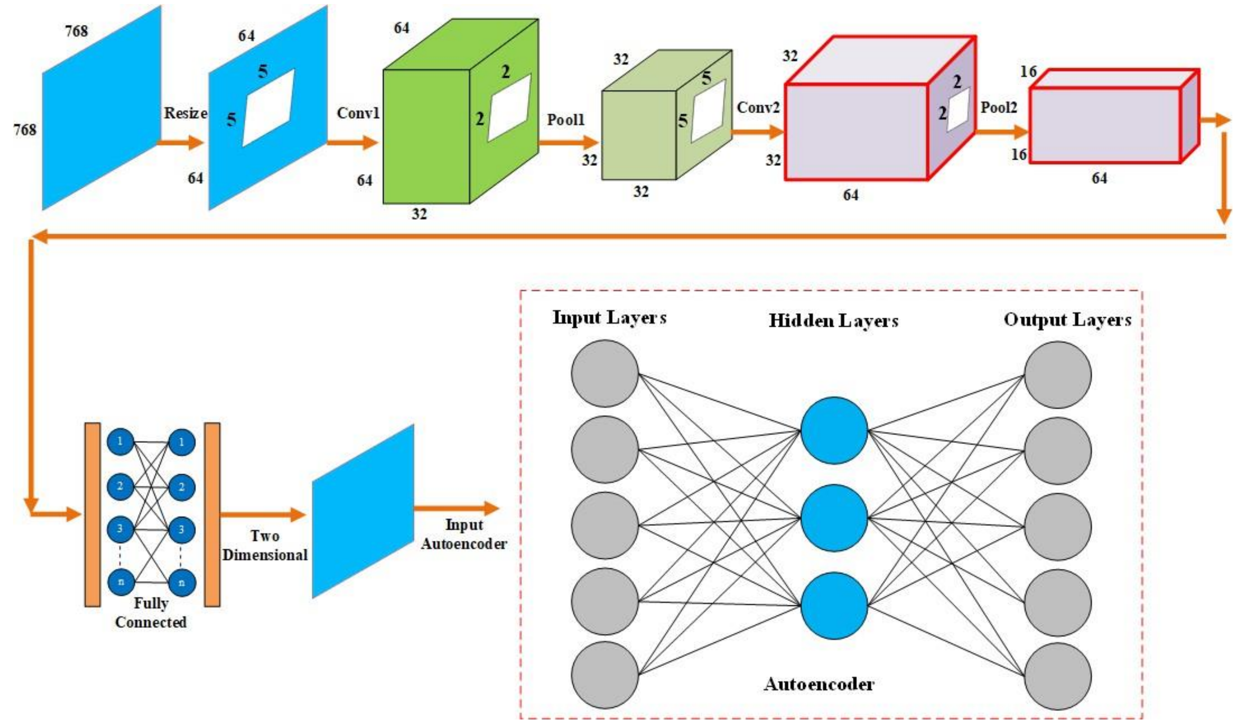

Figure 4.3. Integrated CNN and Autoencoder Internal architecture working for comprehensive disaster prediction using the FloodNet dataset, leveraging the complementary capabilities of these deep learning models.

In this equation, $Y[i, j, k]$ signifies the output feature map at position (i, j) and station k after smearing the convolution operation. X denotes the input feature map, W represents the learnable convolutional filter tensor, and b is the bias trajectory (Khan and Basalamah, 2023b). The convolution process involves gliding the sieve tensor ended the input feature map and calculating the element-wise increase between the sieve and the corresponding region of the contribution, followed by summation across all input channels.

$$Y[i, j, k] = \sum_m \sum_n \sum_l X[i-m, j-n, l] \times W[m, n, l, k] + b[k] \quad (6)$$

$$Y[i, j, k] = max_{m,n} X[i \times s + m, j \times s + n, k] \quad (7)$$

Where, $Y[i, j, k]$ signifies the production of the max pooling process at position (i, j) and channel k. X is the contribution feature map, and s is the stride size. The max pooling process dividers each input channel into non-overlapping areas and outputs the all-out value within each area, effectively down sample the feature map along the spatial dimensions.

$$Z = Encoder(X) = ReLU(W_1 X + b_1) \quad (8)$$

Here, Z represents the encoded representation (latent space) of the input image X obtained from the encoder component of the autoencoder. W_1 is the heaviness matrix, and b_1 is the bias course (ES *et al.*, 2023). The encoder applies a series of linear transformations followed by a non-linear beginning function (ReLU) to map the contribution data into a lower-dimensional latent space.

$$\hat{X} = Decoder(Z) = Sigmoid(W_2 Z + b_2) \quad (9)$$

Where, $\hat{X}$ represents the reconstructed image obtained from the decoder component of the autoencoder using the encoded representation Z. W_2 is the heaviness matrix, and b_2 is the prejudice vector. The decoder applies a series of linear transformations followed by a sigmoid activation function to map the encoded depiction back to the unique input space, reconstructing the image.

$$L = \lambda_1 . CNN_{loss} + \lambda_2 . Autoencoder_{loss} \quad (10)$$

Here, L represents the overall loss function for the integrated CNN and autoencoder model, which is a weighted sum of the losses from the CNN component CNN_{loss} and the autoencoder component $Autoencoder_{loss}$. λ_1 and λ_2 are hyperparameters that switch the relative position of each loss period (Kumbam and Vejre, 2024). This combined loss purpose guides the training process to simultaneously optimize both components of the model for effective disaster prediction.

Segmentation Process

The proposed model associates the feature extraction competences of CNN with the anomaly detection potential of Autoencoders to tackle the task of disaster prediction using the FloodNet dataset. The feature abstraction process, as denoted in the image, comprises a series of convolutional and combining layers that progressively improve the altitudinal features of the input aerial images. This hierarchical feature concept allows the model to detain increasingly complex depictions, allowing the ID of flood-affected areas and other pertinent characteristics. The Autoencoder constituent, on the other hand, plays a critical role in learning compressed representations of the input data. By encoding the features unconcerned by the CNN into a lower-dimensional latent space and then transformation the input, the Autoencoder can classify anomalies that may designate potential disaster events. This synergistic integration of the CNN and Autoencoder architectures leverages the balancing gifts of these deep learning models, letting the overall system to make informed and disaster predictions.

The Object Detection Network and Segmentation portion of the model further progresses the disaster prediction aptitudes by identifying and localising careful objects of interest, such as buildings, roads, and water bodies as shown in the Figure 4.4. This full semantic considerate of the scene, devoted with the feature abstraction and anomaly detection competences, provides a comprehensive outline for comprehensive disaster estimate and response planning.

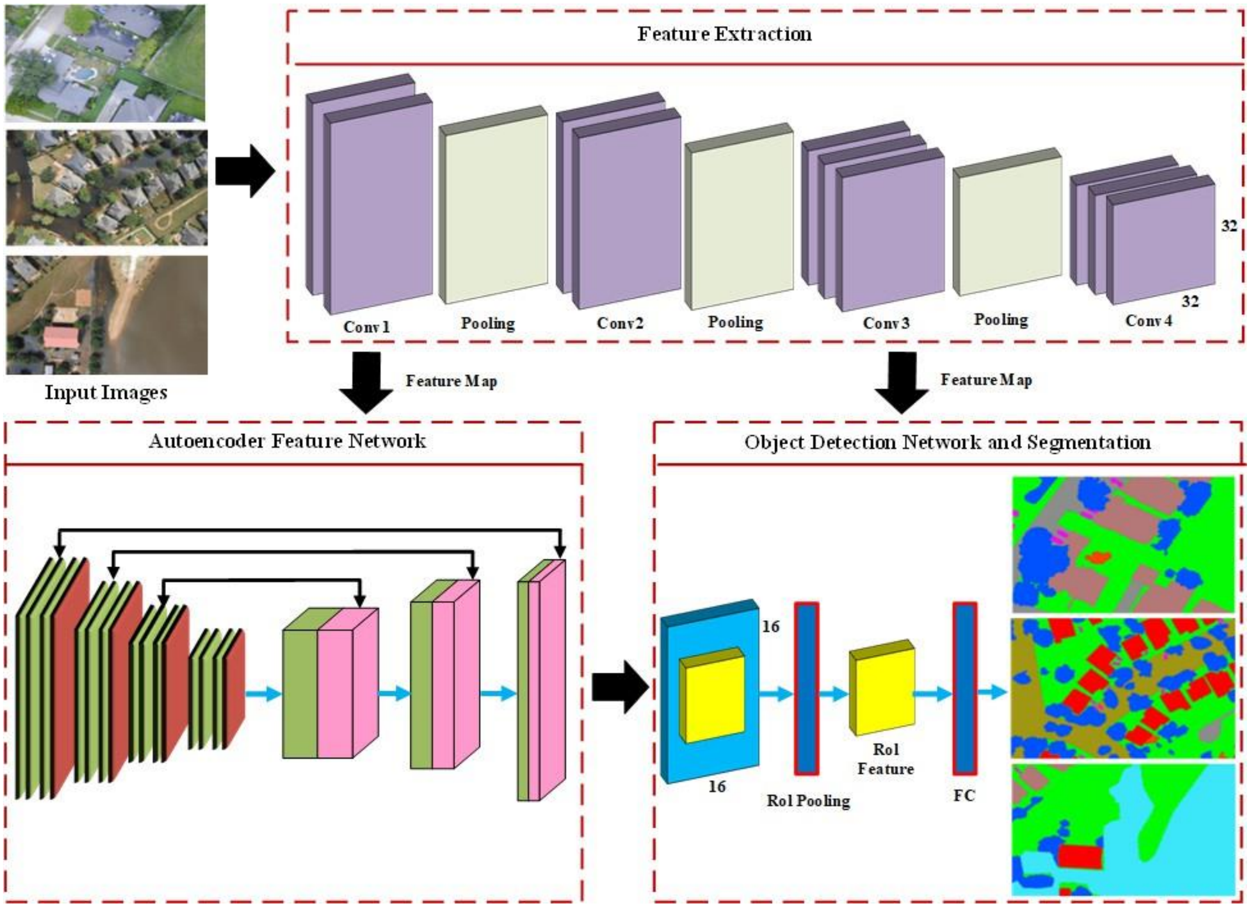

Figure 4.4. Integrated CNN and Autoencoder architecture for disaster prediction, incorporating feature extraction and object-level semantic understanding to enable accurate and comprehensive disaster assessment.

Where, L_{AE} signifies the loss function for the autoencoder component, which is intended as the mean squared error (MSE) among the rebuilt features $\hat{X}$ and the unique input features X. This loss purpose controllers the drill process to diminish the reconstruction error.

$$L_{AE} = MSE(\hat{X}X) \qquad (11)$$

In this equation, img signifies the contribution image, $mean$ is the mean value of pixel strengths, std is the standard aberration of pixel intensities, and $maxPixelValue$ is the maximum pixel strength value (typically 255 for 8-bit images). This equation designates the process of normalisation practical to the input image before feeding it into the model. Normalization certifies that pixel values are climbed to a standardized range, centred around zero mean and with an ordinary deviation of one, which helps improve the constancy and convergence of the training process. The segmentation process for detecting flooded and non-flooded areas is depicted in the Figure 4.5.

$$img = \frac{(img - mean \times maxPixelValue)}{(std \times maxPixelValue)} \qquad (12)$$

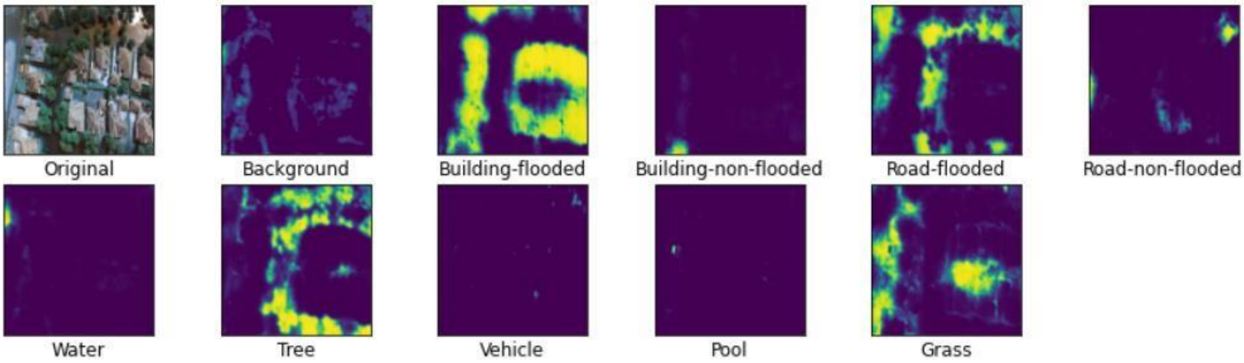

Figure 4.5. Segmentation Process for Detecting Flooded and Non-flooded Areas.

The Intersection over Union (IoU) metric, represented by IoU assesses the accuracy of object segmentation by calculating the ratio of the intersection of predicted and pounded truth segmentation covers to their union. Meanwhile, the Mean Intersection over Union ($MIoU$), spoken as $MIoU$ provides an average IoU across all classes, contribution a comprehensive assessment of segmentation accuracy (Rahnemoonfar *et al.*, 2023). Here, tp specifies true confident pixels, fn designates false negative pixels, and fp signifies false positive pixels. A higher IoU or $MIoU$ value implies better segmentation presentation, crucial for accurate disaster prediction using the FloodNet dataset.

$$IoU = \frac{tp}{\sum_{i=1}^{n} fn + \sum_{i=1}^{n} fp - fn} \tag{13}$$

$$MIoU = \frac{1}{l+1} \sum_{i=1}^{n} \frac{tp}{\sum_{j=0}^{k} fn + \sum_{j=0}^{k} fp - fn} \tag{14}$$

In the providing notation, $Loss$ represents the cumulative loss value, calculated across all samples N, using double cross-entropy. The negative normal of the cross-entropy loss for each model i and class j is calculated, where $y_{i,j}$ indicates the ground truth label and $\hat{y}_{i,j}$ denotes the predicted probability by the model. This formulation aims to calculate the difference between predicted probabilities and actual labels, crucial for enhancing the model's presentation in binary classification tasks.

$$Loss = -\frac{1}{N} \sum_{i=1}^{\hat{y}i,j} [\sum_{j=1}^{C} [yi,j \log(\hat{y}i,j) + (1 - yi,j) \log(1 - \hat{y}i,j)]] \tag{15}$$

The diagram presented in the Figure 4.6 signifies the procedure of flood image segmentation through three separate methodologies: educing CNN alone, smearing CNN with image resizing to 256x256 dimensions, and integrating CNN with an encoder. In the first technique, the flood image undergoes segmentation traditional through CNN, leveraging its characteristic feature removal capabilities to delineate flood-affected areas. The second method comprises image resizing to 256x256 dimensions before separation, aiming to recover computational effectiveness while preservative the image's vital structures. Lastly, the third strategy integrates CNN with an encoder, compulsory the encoder's ability to packaging input data into a lower-dimensional resting space before segmentation, potentially attractive separation accuracy. These methodologies denote diverse methods to flood image segmentation, each influence single rewards in terms of accuracy, computational capability, and feature protection.

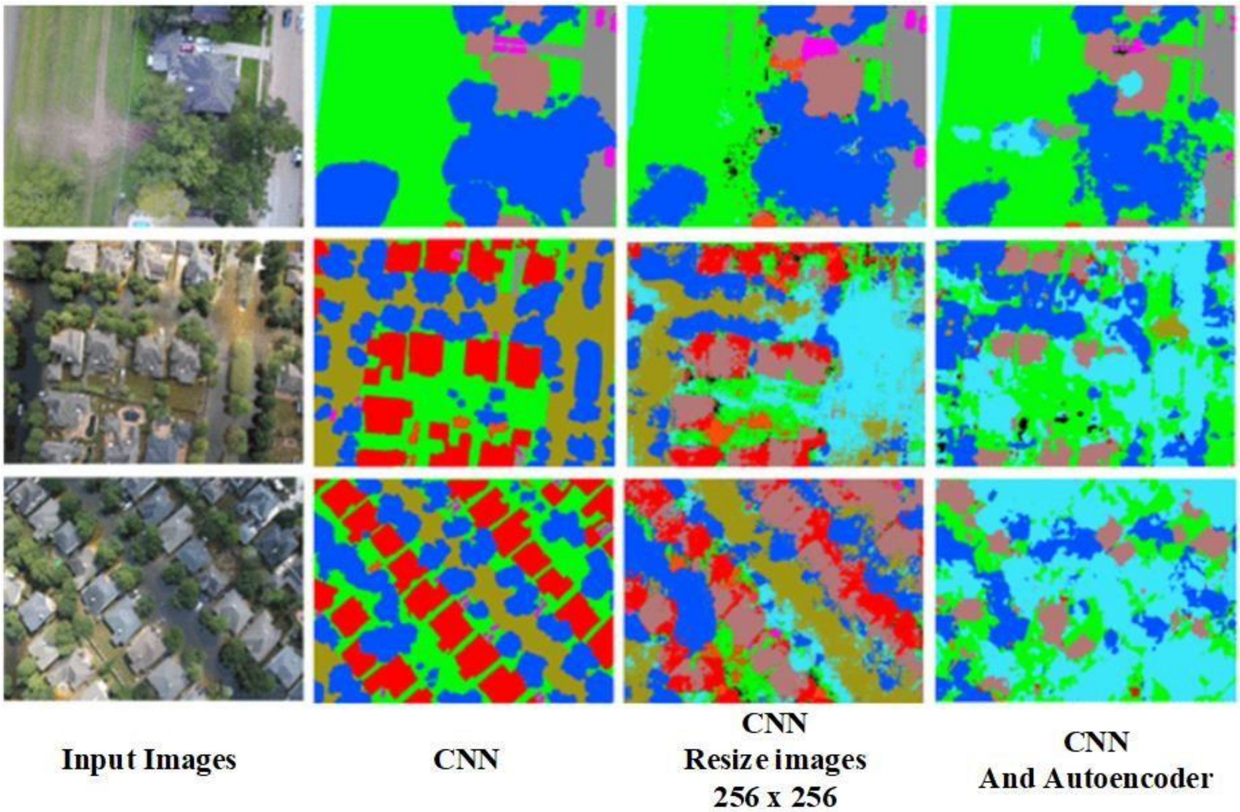

Figure 4.6. Exploring segmentation strategies for flood images: CNN, CNN with resizing, and CNN with encoder integration.

Flood Detection Network

The image shown in the Figure 4.7 represents an inclusive workflow for disaster forecast using a group of CNN and Autoencoders. The process starts with an input image of size 3072 x 4590, which knowledges a sequence of convolutional (Conv) and combining processes to unkind expressive spatial features. The CNN component slowly recovers these features, dipping the spatial dimensions while increasing the number of feature maps. This permits the model to detention classified symbols of the input data, allowing the documentation of flood-affected areas. Parallel, the Autoencoder component engrosses a beaten representation of the input images, training them into a lower-dimensional dormant space and then alteration them. This process of encrypting and decoding helps the Autoencoder classify anomalies that may indicate potential disaster events, completing the flood detection competences of the CNN. The synergistic integration of the CNN and Autoencoder architectures leverages their strengths if a comprehensive framework for accurate and consistent disaster prediction using the FloodNet dataset.

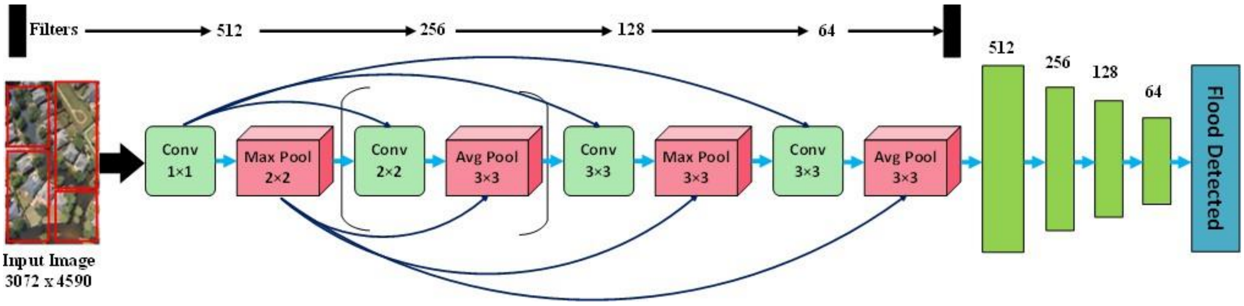

Figure 4.7. CNN Architecture for Comprehensive Object Detection (Flood) Network using the FloodNet Dataset.

These equations play an important role in assessing the performance of the machine learning models active for disaster prediction using the FloodNet dataset. Equation (16) computes the accuracy, which events the overall accuracy of predictions by seeing both true positive (TP) and true negative (TN) organizations relative to all instances. Equation (17) computes precision, counting the proportion of correctly recognized positive instances (TP) among all instances secret as positive, thereby evaluating the model's ability to avoid false positive (FP) errors. Equation (18) controls recall, measuring the proportion of actual positive instances (TP) properly identified by the model relative to all definite positive instances, indicating the model's sensitivity to taking true positive gears and minimizing false negative (FN) errors (Jamali *et al.*, 2024). Finally, Equation (19) computes the F1 score, which corresponds precision and recall by computing their harmonic mean, contribution a balanced measure of the model's performance, chiefly when dealing with unfair datasets.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{16}$$

$$Precision = \frac{TP + TN}{TP + FP} \tag{17}$$

$$Recall = \frac{TP}{TP + FN} \tag{18}$$

$$F1_{score} = 2 \times \frac{precision \times recall}{precision + recall} = \frac{2TP}{2TP + FN + FP} \qquad (19)$$

Chapter 5 – Results and Discussion

Chapter 4 comprises 9 subsections that delve into various aspects of the experimental results and discussions. Subsection 4.1 evaluates the loss and accuracy of the CNN Model, providing performance metrics. Subsection 4.2 presents the confusion matrix of the proposed model for detection insights. Subsection 4.3 discusses performance measures of the proposed CNN model. Subsection 4.4 analyses the ROC and learning curves for detection behaviour. Subsection 4.5 examines training and validation loss of the Autoencoder. Subsection 4.6 explores the effect of different training data on the CNN Model's robustness. Subsection 4.7 investigates proposed model weights and frequency. Subsection 4.8 examines feature distribution and histogram of CNN. Finally, subsection 4.9 compares the proposed model with latest models, highlighting performance benchmarks and advancements. Each subsection contributes to a comprehensive analysis of the experimental outcomes.

Loss and Accuracy of CNN Model

The graph depicts the training and validation losses of a CNN over 25 epochs as shown in the Figure 5.1. The x-axis represents the number of epochs, which are iterations over the entire dataset used for training. The y-axis shows the loss, a measure of how well the CNN is performing. The blue line shows the training loss, indicating how the model's performance improves on the training data as it learns. The orange line represents the validation loss, reflecting the model's performance on a separate set of data not seen during training. The goal is to minimize both losses; however, the spikes in validation loss, particularly around epoch 20, suggest some variability in the model's learning or potential overfitting to the training data. Overall, both losses decrease, which is a sign that the model is learning to generalize from the training data.

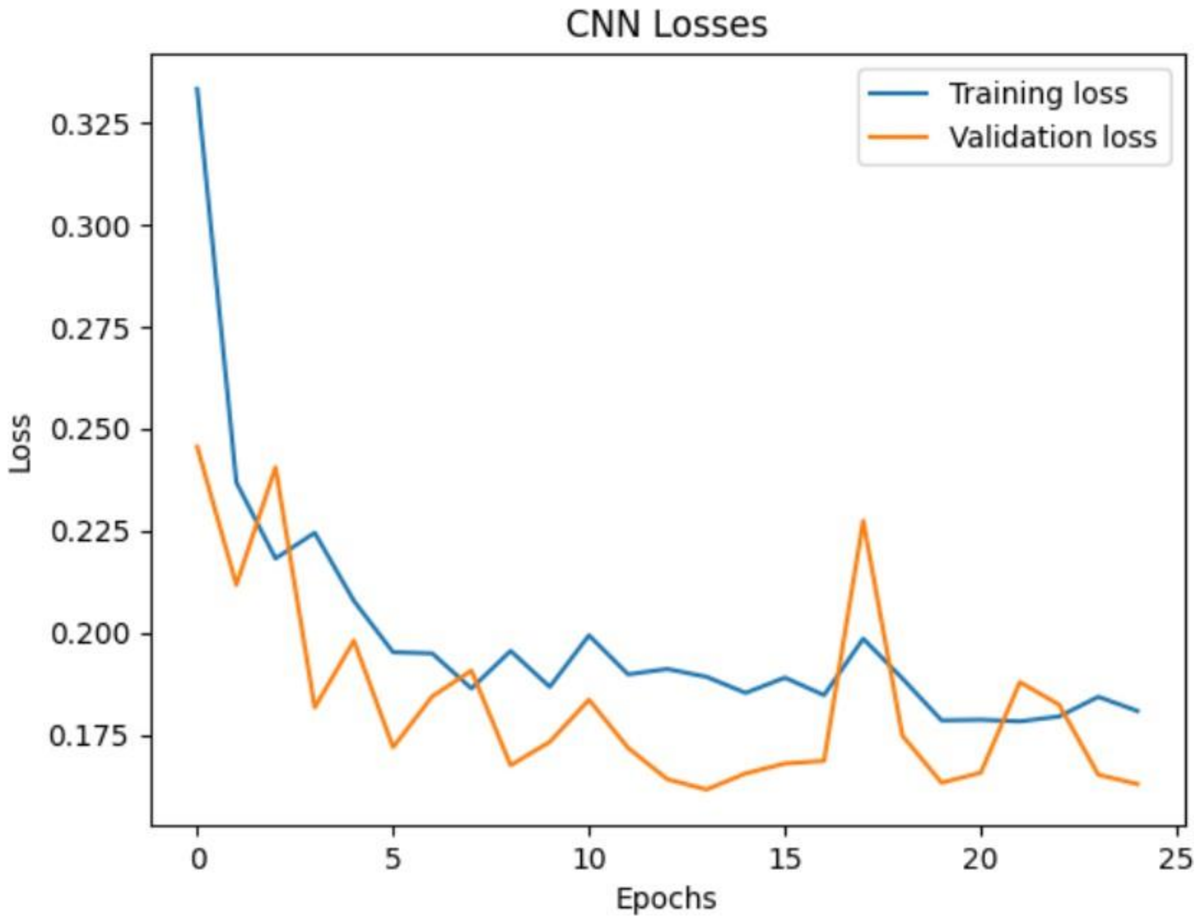

Figure 5.1. The graph illustrates the training and validation losses of a CNN over 25 epochs, showcasing the model's learning progress and potential overfitting concerns.

This graph illustrates the training and validation accuracies of a Convolutional Neural Network over 25 epochs. The training accuracy (blue line) and validation accuracy (orange line) are both above 90% and closely track each other, indicating good generalization. The graph shows an initial sharp increase in accuracy, followed by fluctuations that level out as the epochs progress, suggesting some variability but overall stable learning as demonstrated in the Figure 5.2. The close alignment of both lines throughout the training process indicates that the model is not overfitting to the training data.

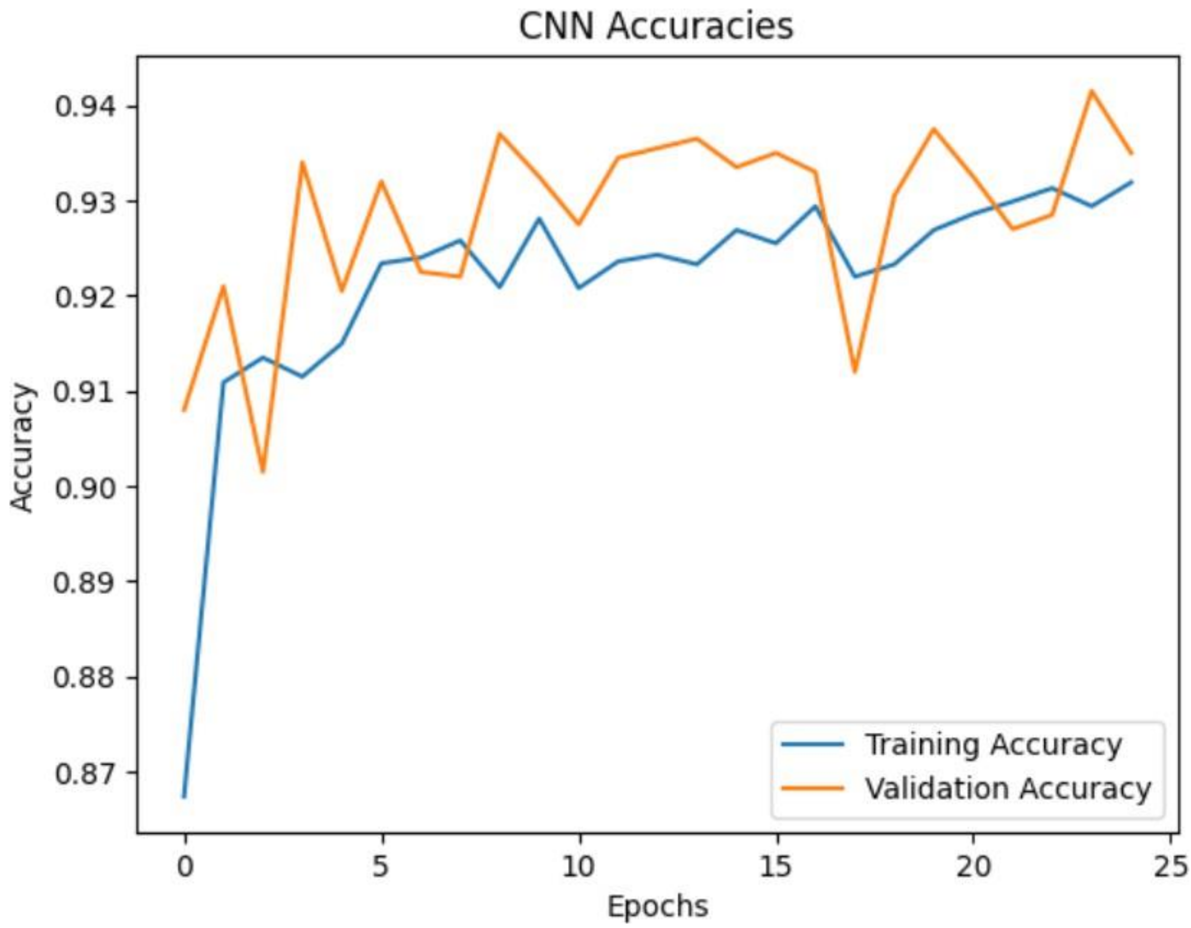

Figure 5.2. Stable training and validation accuracies above 90% suggest effective generalization in the Convolutional Neural Network.

Figure 5.3. Batch Training Image using CNN and Autoencoder.

Confusion Matrix of Proposed Model

This graph is a confusion matrix for a binary classification model. The matrix shows the number of correct and incorrect predictions. There are two classes, labelled as "0" and "1". For class "0", 911 instances were correctly predicted (true negatives), while 89 instances were incorrectly labelled as class "1" (false positives). For class "1", 959 instances were correctly predicted (true positives), and 41 instances were incorrectly labelled as class "0" (false negatives). The colour intensity represents the magnitude of the counts, with darker shades indicating higher numbers. This model demonstrates high accuracy with more correct predictions and a relatively low number of errors as depicted in the Figure 5.4.

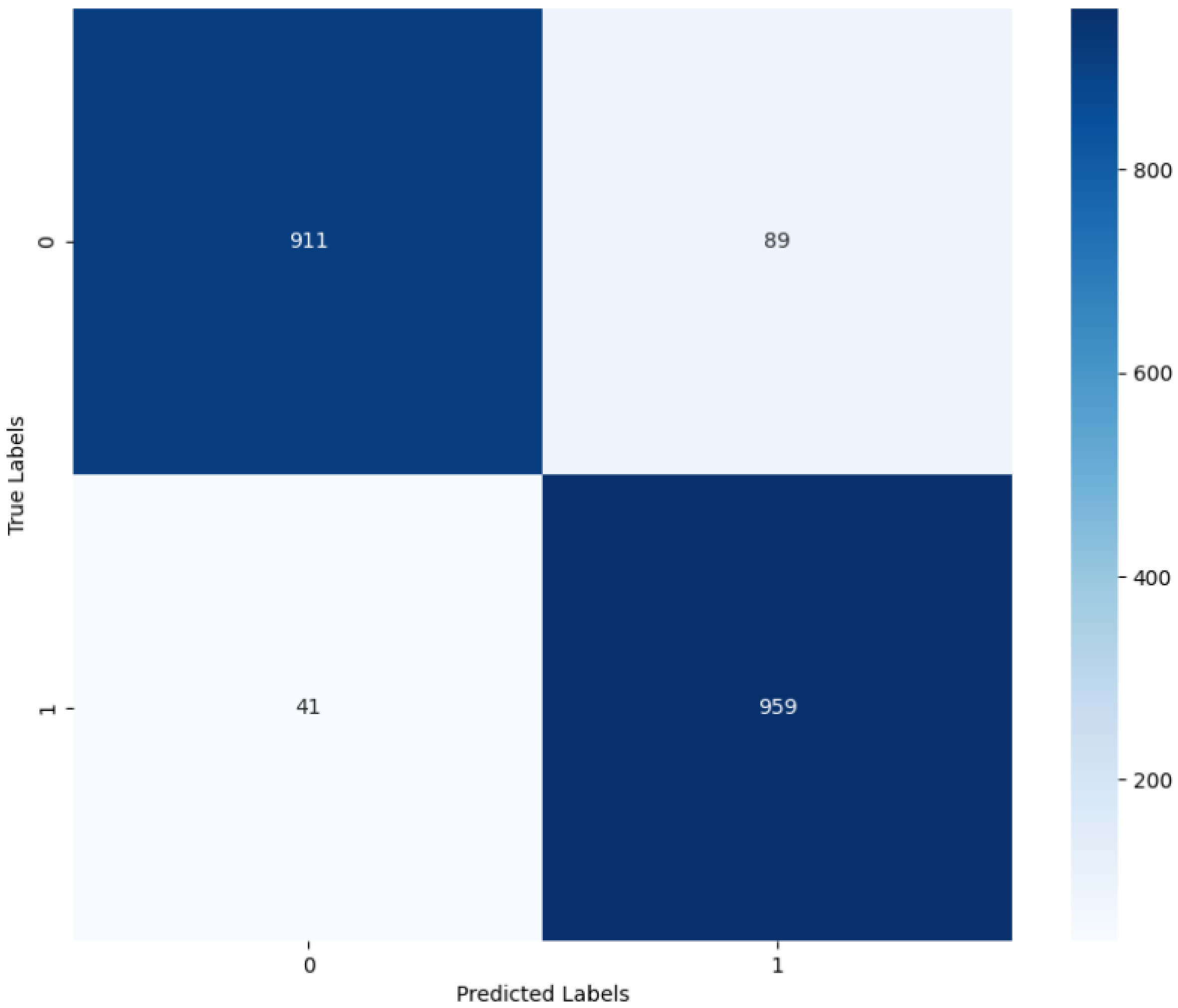

Figure 5.4. Binary classification confusion matrix illustrates high accuracy with minimal errors.

Performance Measures of Proposed CNN Model

The performance of a proposed CNN model across different configurations based on the number of layers: 1 layer, 2 layers, 3 layers, and 4 layers is explained in the table. The evaluation metrics used to assess these models include precision, recall, F1 score, and accuracy, which are fundamental in measuring the effectiveness of classification models as shown in the Table 5.1.

Table 5.1. Effect of Layer Variation on Performance Metrics in Proposed CNN Models.

Proposed Model using Different Layers Effect				
Method and Layers	**Precision (%)**	**Recall (%)**	**F1 Score (%)**	**Accuracy (%)**
Proposed CNN Model **(1 Layer)**	75.40	65.37	75.35	69.42
Proposed CNN Model **(2 Layers)**	62.35	77.35	84.35	75.24
Proposed CNN Model **(3 Layers)**	85.35	84.34	59.35	85.54
Proposed CNN Model **(4 Layers)**	**91.51**	**95.90**	**93.65**	**93.50**

Precision reflects the accuracy of positive predictions made by the model. A higher precision percentage indicates that the model's positive predictions are more likely to be correct. For instance, the CNN model with 1 layer achieves a precision of 75.40%. This means that when this model predicts a positive instance, it is correct 75.40% of the time. However, the precision decreases as more layers are added to the model. The CNN model with 2 layers, for example, exhibits a precision of 62.35%, indicating a reduction in the accuracy of positive predictions compared to the 1-layer model. A higher recall value proposes that the model can effectively detention more of the true positives within the dataset. The CNN model with 4 layers achieves a notably high recall of 95.90%, indicating that it can successfully identify a large proportion of actual positive instances as demonstrated in the Figure 5.5. In distinction, the model with 1 layer has a recall of 65.37%, signifying a relatively lower ability to capture true positives associated to the deeper models.

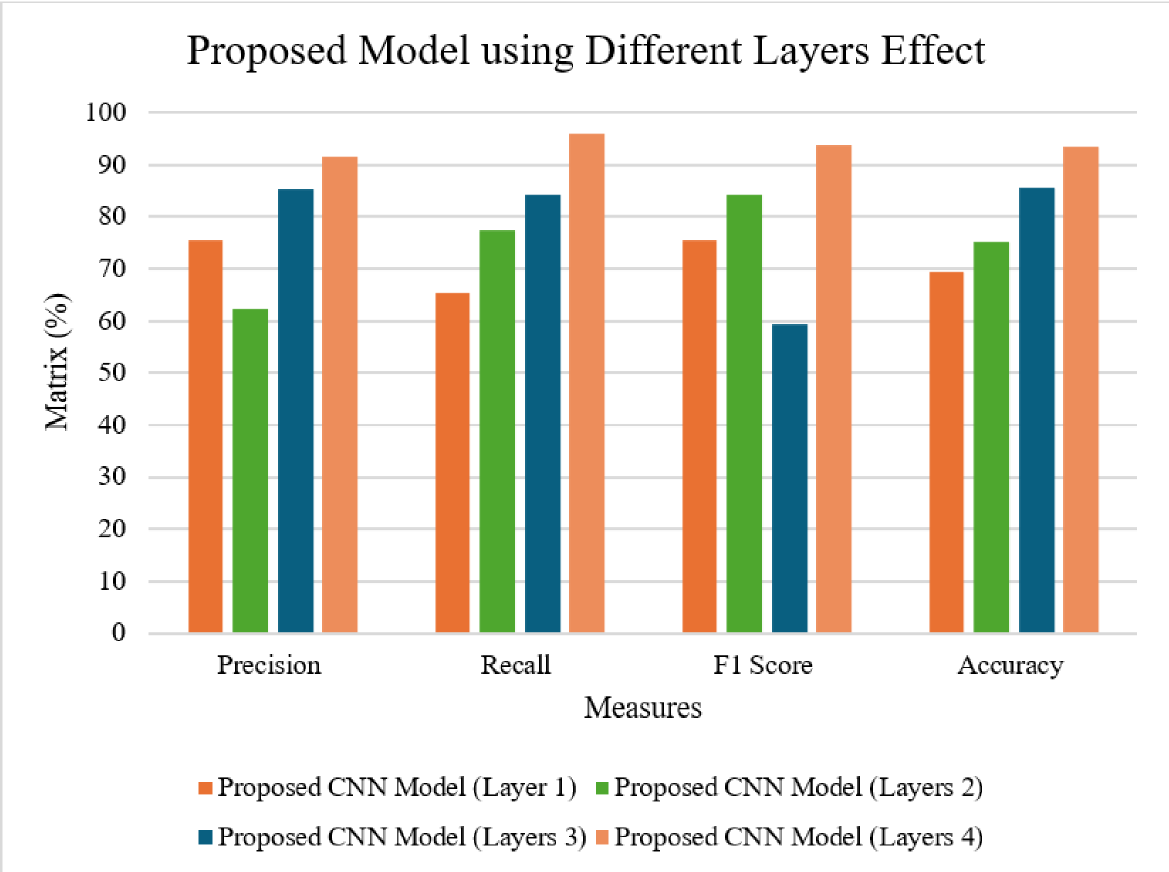

Figure 5.5. Effect of Layer Variation on Performance Metrics in Proposed CNN Models.

The F1 score, which is the harmonic mean of precision and recall, delivers a balanced valuation of a model's performance. Notably, the F1 score disagrees across different layer configurations. For occurrence, the CNN model with 2 layers attains a high F1 score of 84.35%, which reflects a good balance between precision and recall for that formation. Lastly, accuracy signifies the overall accuracy of the model's predictions crossways all classes. The CNN model with 4 layers achieves a high accuracy of 93.50%, representing strong overall performance. But it's essential to reflect on precision, recall, and F1 score alongside accuracy to improvement a comprehensive understanding of a model's behaviour, chiefly in scenarios where class differences exist, or exact performance goals are arranged.

ROC Curve and Learning Curve

The graph shows a Receiver Operating Characteristic (ROC) curve, which is a scheme demonstrating the diagnostic ability of a binary classifier as its judgement threshold is varied. The True Positive Rate (TPR) is on the y-axis, and the False Positive Rate (FPR) is on the x-axis. The orange line represents the ROC curve of the model, and the dashed blue line represents a random chance classifier. The area under the curve (AUC) is 0.99, indicating that the model has an excellent measure of separability with a high ability to distinguish between the two classes. The closer the ROC curve follows the left-hand border and then the top border of the ROC space, the more accurate the test. The ROC curve demonstrating excellent model discrimination with an AUC of 0.99 is given in the Figure 5.6.

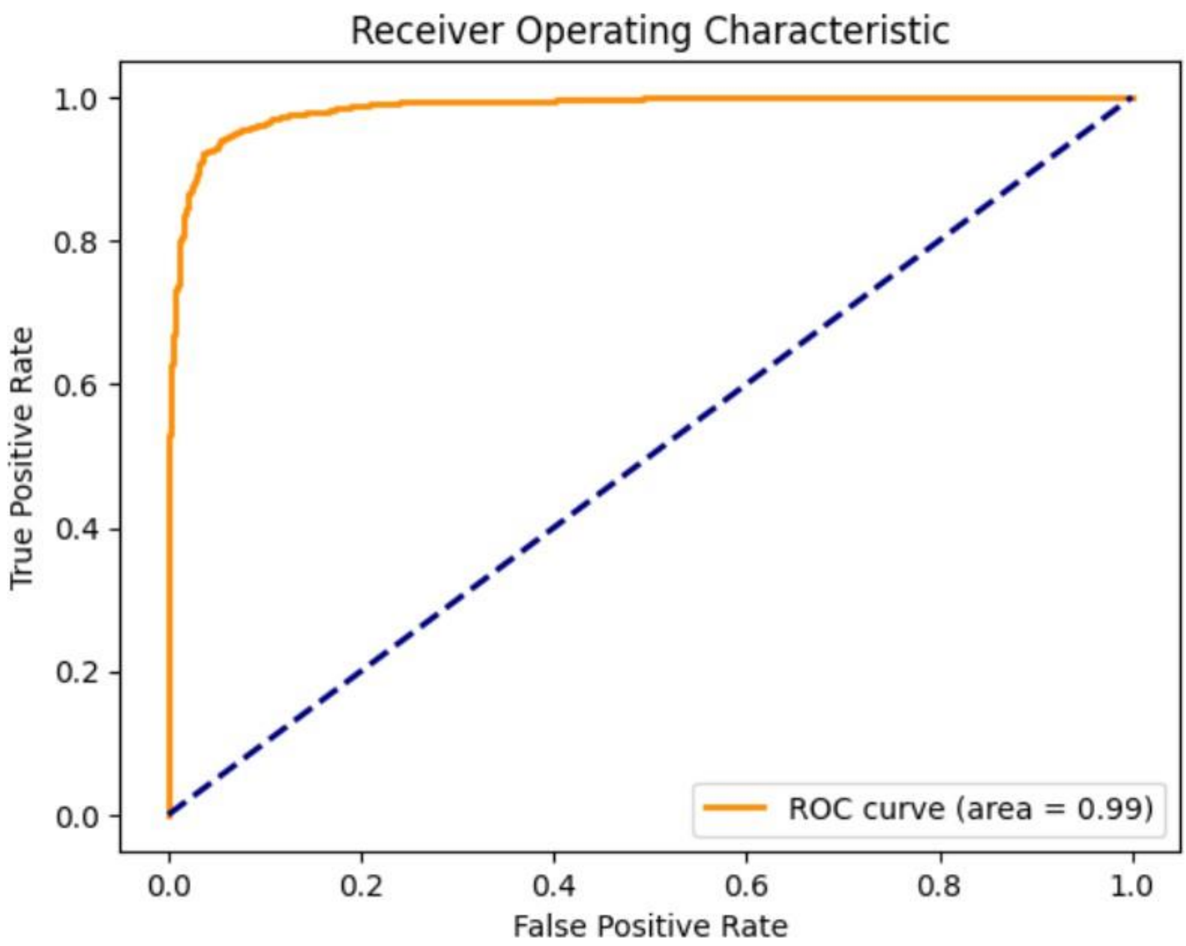

Figure 5.6. ROC curve demonstrating excellent model discrimination with an AUC of 0.99.

This graph shown in the Figure 5.7 displays the learning rate of an algorithm over 25 epochs. The learning rate is constant as depicted by the flat blue line, indicating that a fixed learning rate strategy is being used during training. The value of the learning rate is set just above 0.00100, which determines the size of the steps the algorithm takes when updating the model weights. A constant learning rate can be beneficial for certain models and datasets, but in others, a dynamic learning rate that changes over time might be more effective. The consistency of the learning rate suggests that the trainer prioritized stability over dynamic adjustment for this training session.

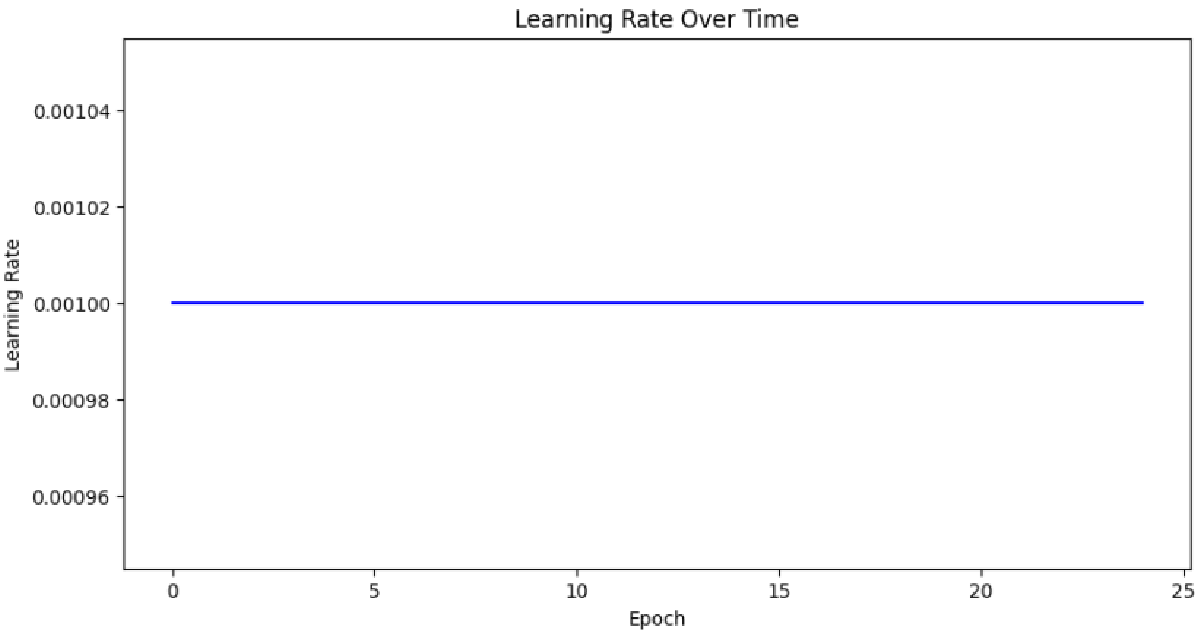

Figure 5.7. Stable learning rate strategy maintained throughout training, fostering consistency in model updates.

Training and Validation Loss of Autoencoder

This graph plots the training and validation losses of an Autoencoder model over 25 epochs. The losses sharply decrease at the beginning and then plateau, indicating rapid initial learning that stabilises as the model starts to converge. Both the training loss (blue line) and the validation loss (orange line) follow a similar trend, which suggests the model is generalising well. However, at epoch 25, there's a significant spike in both training and validation losses. This could be due to overfitting, an anomaly in the data, or a learning rate that's too high, causing the model's weights to overshoot optimal values suddenly as given in the Figure 5.8.

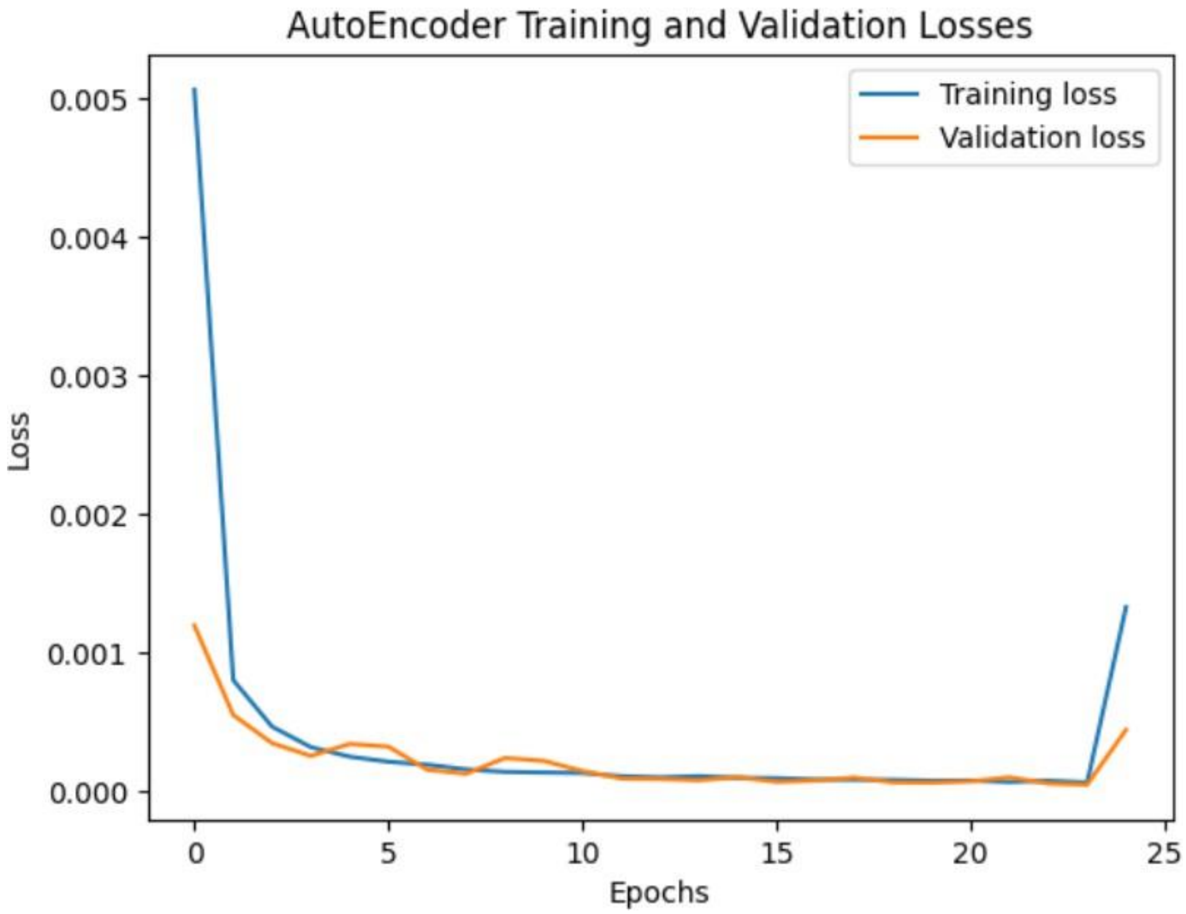

Figure 5.8. Autoencoder training and validation losses exhibit rapid initial decrease followed by stabilization.

Effect of Different Training Data on CNN Model

The performance metrics of a machine learning model trained on varying sizes of the dataset, specifically 60%, 80%, and 100% of the available training data is discussed in the Table 5.2.

Table 5.2. The performance metrics of a machine learning model trained on varying sizes of the dataset.

Training data size	Precision (%)	Recall (%)	F1 Score (%)	Accuracy (%)
60% training data	60.24	50.34	85.34	77.35
80% training data	75.35	85.35	65.35	88.24
100% training data	**91.51**	**95.90**	**93.65**	**93.50**

When trained with 60% of the data, the model achieves a precision of 60.24%. This indicates that about 60.24% of the instances predicted as positive by the model are indeed true positives. As the training data size increases to 80%, the precision improves significantly to 75.35%, suggesting that a larger training dataset contributes to more accurate positive predictions. Finally, when the model is trained on the entire dataset (100% of the data), the precision further increases to 91.51%, indicating a substantial enhancement in the model's ability to make accurate positive predictions as given in the Figure 5.9. With 60% of the training data, the recall is 50.34%, indicating that the model captures only about half of the actual positive instances. However, as the training data size increases to 80% and 100%, the recall improves significantly to 85.35% and 95.90%, respectively. This signifies that a larger training dataset allows the model to identify more of the actual positive instances, leading to improved recall rates.

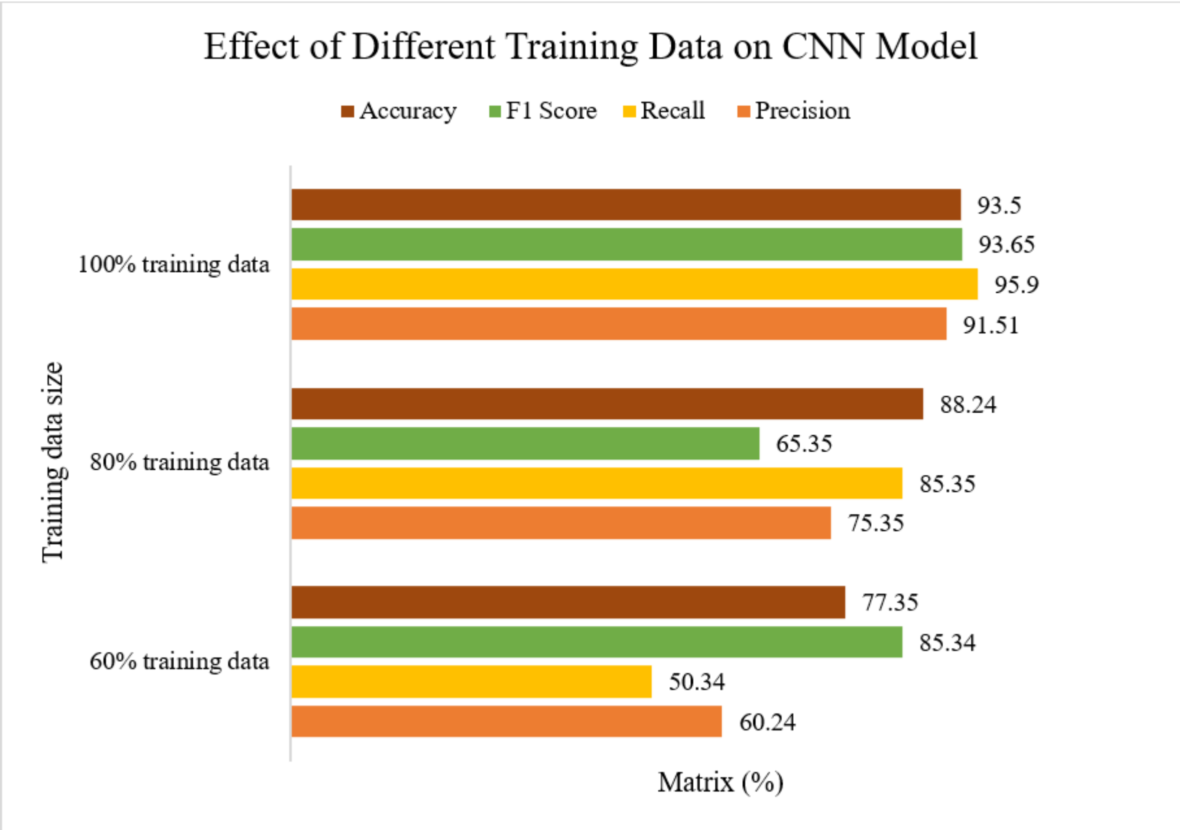

Figure 5.9. Training with larger datasets improves model precision and recall.

At 60% training data, the F1 score is 85.34%, representative a balance between precision and recall for that dataset size. Though, the F1 score droplets to 65.35% with 80% training data and then intensifications to 93.65% when using 100% of the training data. This highpoint the influence of training data size on attaining a balanced presentation between precision and recall. The accuracy rises with larger training data sizes, attainment its highest value of 93.50% when the model is trained on the perfect dataset. This underscores the importance of enough and characteristic training data in improving model accuracy and overall presentation in organisation tasks.

Proposed Model Weights and Frequency

The diagram shown in the Figure 5.10 is a histogram that shows the delivery of weight values from the fully associated layer of a neural network model. The x-axis represents the weight values, and the y- axis represents the frequency of these weights within the model. The weight values range approximately from -0.2 to 0.3. The distribution is roughly symmetric around zero and suggests that the weights of the model are varied, with no extreme values, which is often a sign of a well-regularized model. This could imply that the network has learned a complex pattern without overemphasising any particular input feature.

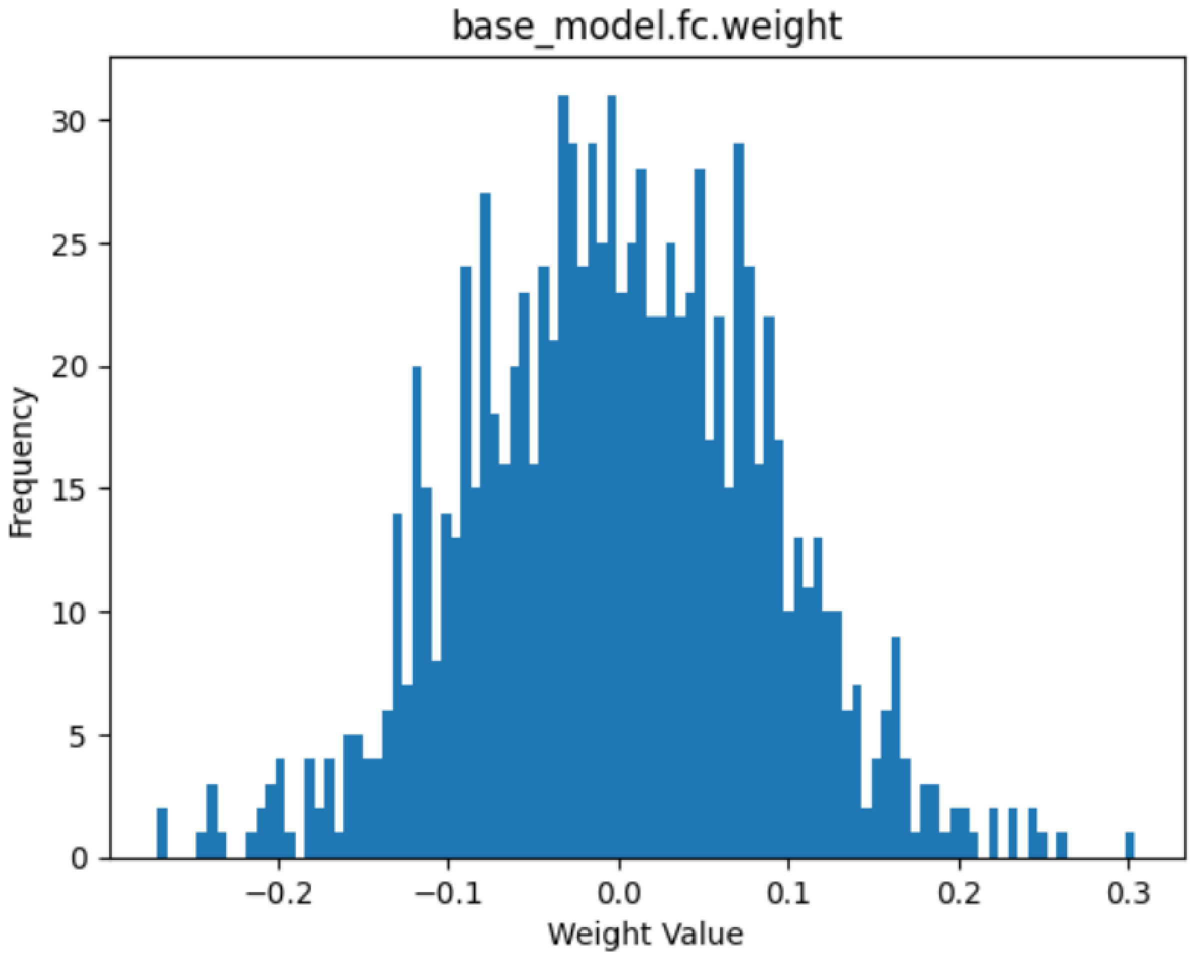

Figure 5.10. Symmetric weight distribution suggests a well-regularized neural network model.

This image given in the Figure 5.11 is a heatmap representation of the weights from a fully connected layer (often denoted as fc) in a neural network model, likely the final layer given the name base_model.fc.weight. The x-axis shows input features, and the y-axis represents the output features, which are typically the classes in classification tasks. In this case, it seems to represent weights connecting two features. The colours indicate the weight values, where purple signifies negative weights, and shades of red to beige indicate positive weights. The uniformity of the colour pattern suggests that all input features are equally contributing to both output classes, indicating either an uninitialised weight matrix or that the model has learned an unusual pattern where all features are equally weighted across the outputs.

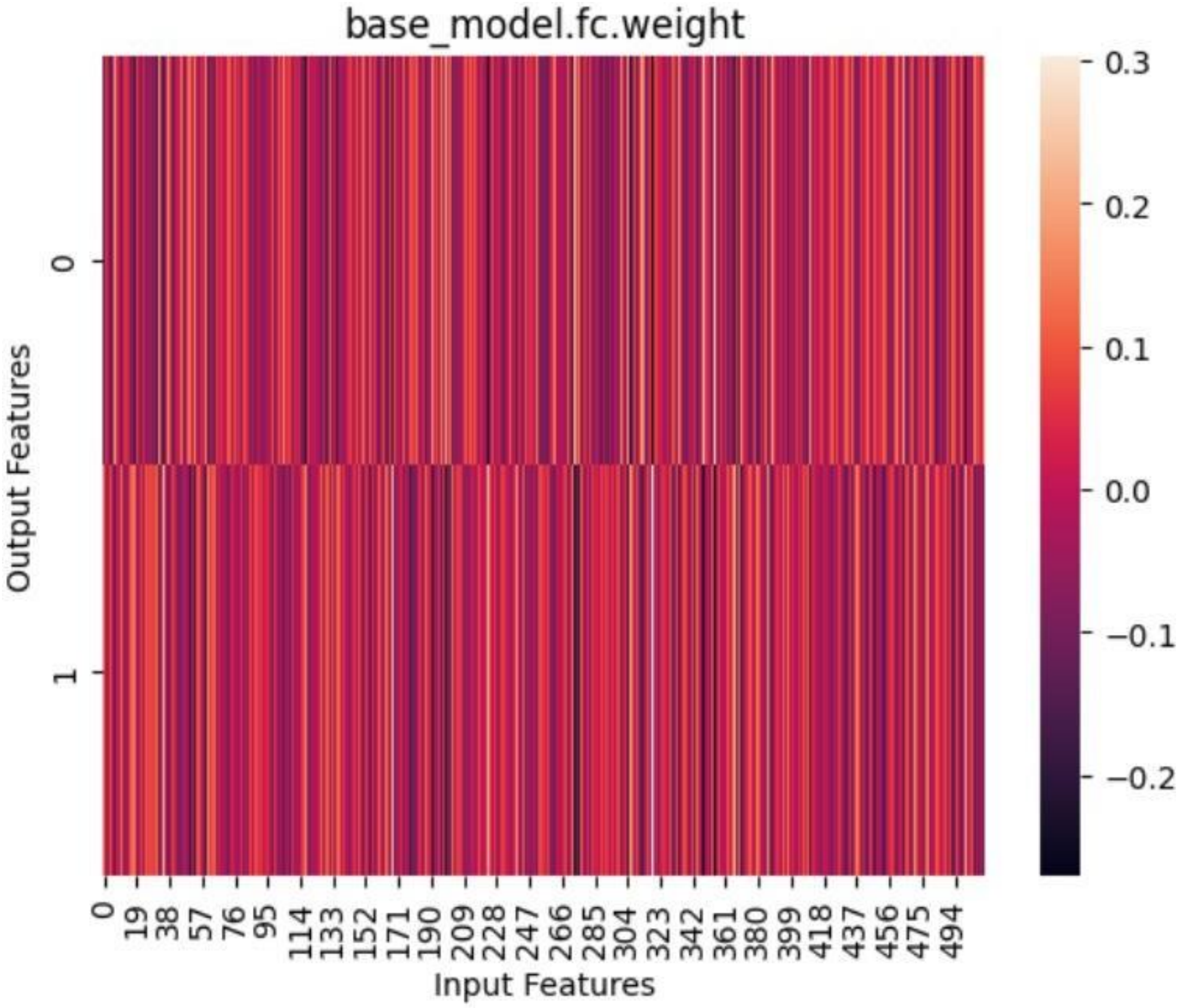

Figure 5.11. Uniform colour pattern in the heatmap suggests either uninitialized weights or equal contribution of input features to output classes in the fully connected layer of a neural network model.

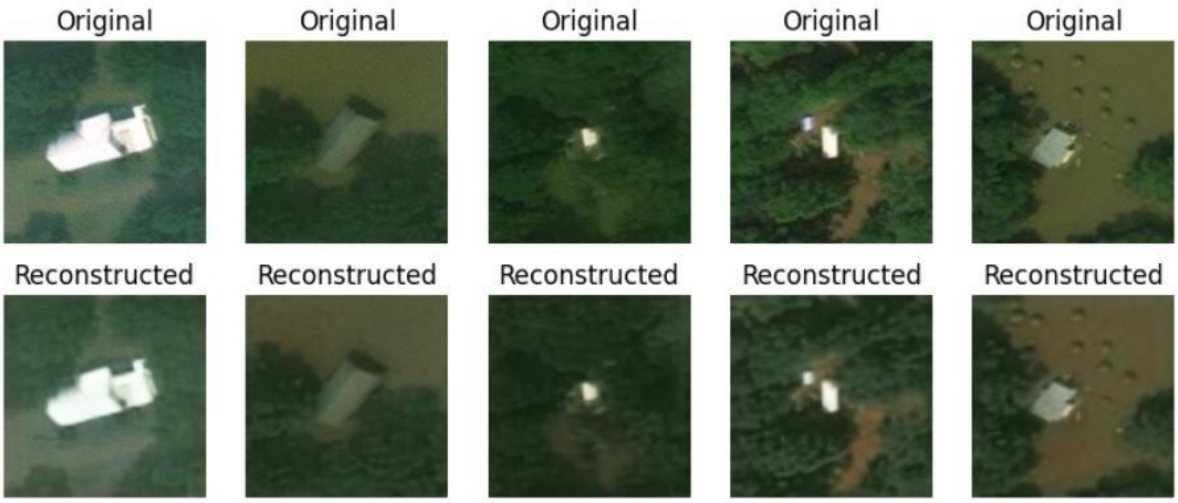

Figure 5.12. The originals and reconstructed images using CNN and Autoencoder

Feature Distribution and Histogram of CNN

The graph given in the Figure 5.13 shows a scatter plot representing the output of a dimensionality reduction or. Each point represents an encoded feature with two dimensions on the x and y axes. The colour scale indicates density or another variable, with yellow showing higher values. The distribution is somewhat linear, suggesting some correlation between the encoded dimensions.

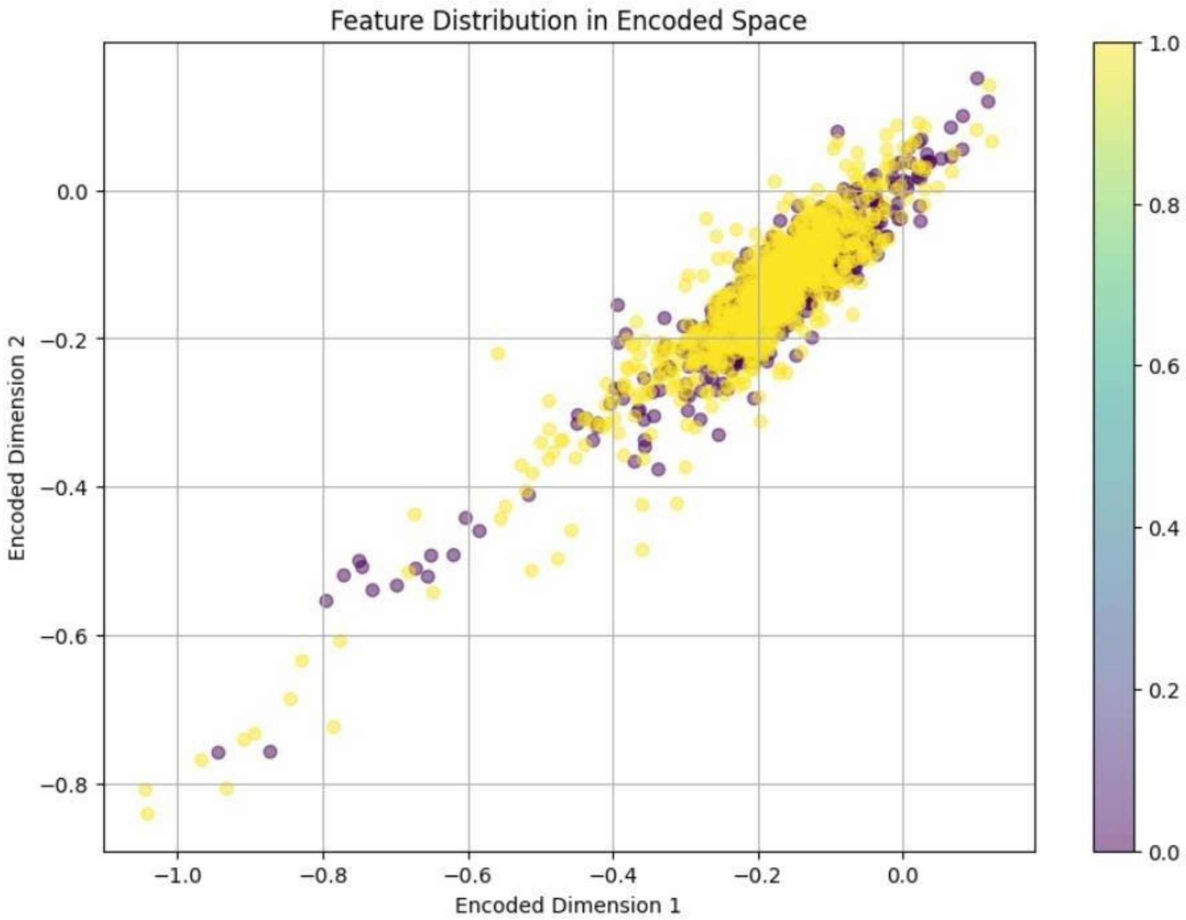

Figure 5.13. Scatter plot illustrates encoded features with two dimensions, showcasing a linear correlation between them, depicted through colour density.

The graph given in the Figure 5.14 is a histogram of model output scores, representing the confidence of a classification model's predictions. The x-axis displays softmax scores, which range from 0 to 1, and the y-axis shows the frequency of these scores. The bimodal distribution with peaks near 0 and 1 suggests the model is often very confident in its predictions, which is typical after applying a softmax function in a well-trained binary classifier.

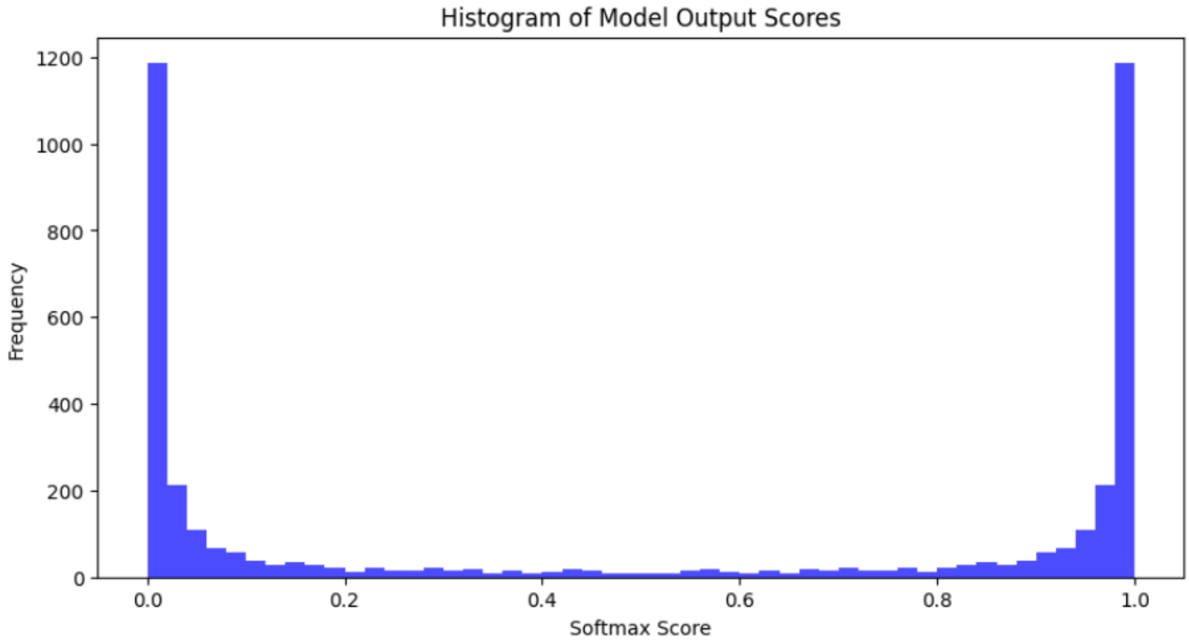

Figure 5.14. Bimodal histogram of model output scores indicates high confidence in predictions, typical for a well-trained binary classifier with softmax activation.

Comparison of Proposed Model with Latest Models

The performance metrics of various machine learning models applied in disaster management and related tasks is explained in the Table 5.3. Each row represents a different model or model architecture along with its respective evaluation scores.

Table 5.3. Performance metrics of various models for disaster management tasks.

Models	Precision (%)	Recall (%)	F1 Score (%)	Accuracy (%)
CNN architectures, including Inception v3 and DenseNet	63.78	51.37	59.37	78.14
CNN, Landslide Susceptibility Maps (LSMs), Decision Tree, Naïve Bayes	73.54	63.54	76.18	64.39
CNN+RNN for disaster management	52.34	70.35	56.74	53.67
Disaster prediction using Faster R-CNN model	87.39	78.35	67.91	60.37
Damage assessment using Hybrid CNN+ GRU	89.34	65.35	82.46	85.34
Natural language processing (NLP) model for disaster detection	90.36	85.39	80.25	90.35
Proposed CNN and Autoencoder Model for Disaster Management	**91.51**	**95.90**	**93.65**	**93.50**

Starting with the first entry, which involves CNN architectures such as Inception v3 and DenseNet, the model achieves a precision of 63.78%, a recall of 51.37%, an F1 score of 59.37%, and an accuracy of 78.14%. These metrics indicate moderate performance in terms of precision and recall, with a relatively higher accuracy score. Moving to the second row, which combines CNN, Landslide Susceptibility Maps (LSMs), Decision Tree, and Naïve Bayes, the model demonstrates improved precision (73.54%) and recall (63.54%), leading to a higher F1 score (76.18%), despite a slightly lower accuracy of 64.39%. This suggests effective performance in classifying landslide susceptibility using a combination of different techniques. The third entry involves a CNN+RNN (Recurrent Neural Network) hybrid model for disaster management, showing a precision of 52.34%, a recall of 70.35%, an F1 score of 56.74%, and an accuracy of 53.67% as shown in the Figure 5.15.

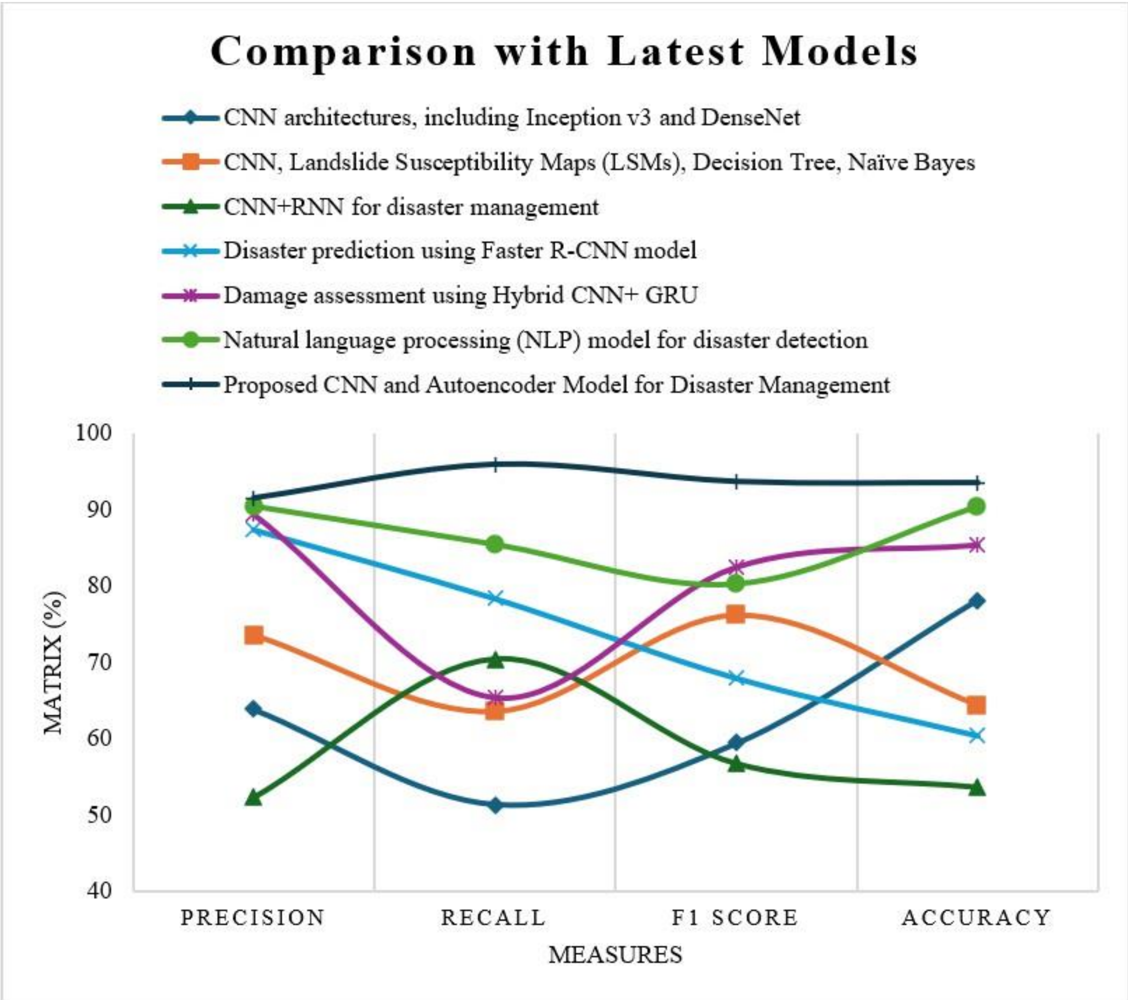

Figure 5.15. Performance metrics of diverse disaster management models.

In the fourth row, disaster prediction is tackled using a Faster R-CNN (Region-based Convolutional Neural Network) model, achieving a high precision of 87.39% and a decent recall of 78.35%. However, the F1 score (67.91%) and accuracy (60.37%) are comparatively lower, suggesting a potential trade-off between precision and recall. Next, damage valuation is achieved using a Hybrid CNN+GRU (Gated Recurrent Unit) model, representative strong precision (89.34%) but lower recall (65.35%), resultant in a balanced F1 score of 82.46% and a high accuracy of 85.34%. The sixth entry involves a Natural Language Processing (NLP) model for disaster detection, attaining imposing precision (90.36%), recall (85.39%), F1 score (80.25%), and accuracy (90.35%). This model leverages text-based data efficiently to detect and classify disaster-related satisfied with high accuracy and recall. Lastly, a planned CNN and Autoencoder Model for Disaster Management exhibits remaining performance crossways all metrics, with precision (91.51%), recall (95.90%), F1 score (93.65%), and accuracy (93.50%) dazzling its vigorous capability in disaster management tasks, likely connecting image-based data and feature learning using autoencoders.

Chapter 6 – Conclusion and Future Work

Conclusion

Achievements

This dissertation concludes by highlighting the significant difficulties encountered in disaster recovery, particularly with regard to flood occurrences, where damage assessment and supply allocation are intricate and time sensitive. The inadequacy of manual inspection procedures that are outdated in terms of speed and efficacy highlights the need for powerful artificial intelligence techniques to transform disaster management protocols. This study successfully enhanced a deep learning-based system that combines CNN and custom autoencoder models for disaster management by utilising the FloodNet dataset.

The study's findings support the effectiveness of the designed model, which shows excellent presentation metrics in terms of counting precision (91.51%), recall (95.90%), F1-score (93.65%), and accuracy (93.50%) in the assessment of flood damage. The model enhances the rapid assessment of flood-related damage and contributes to better situational awareness and proactive disaster management plans by utilising FloodNet photos and cutting-edge AI techniques.

This research highlights the potential of AI-driven solutions to change disaster rejoinder skills by using extraordinary state-of-the-art models. This will enable more effective resource distribution and mitigation measures in areas that are prone to flooding. In the end, attractive community resilience against natural catastrophes like floods is achieved with the advanced use of deep learning and anomaly detection, which represents a substantial advancement in harnessing knowledge for real- time flood monitoring and response efforts. This work highlights the critical role that skill plays in building adaptable and responsive disaster response frameworks, and it paves the way for future developments in AI-driven disaster management systems.

Future Work

Future research using cutting-edge AI methods for disaster rejoinder offers stimulating predictions for additional invention and change. Expanding upon the groundwork conventional by this study, other inquiries may discover multiple pathways to improve the efficiency and appropriateness of artificial intelligence-powered systems in flood-related disaster extenuation.

First, a junction towards model optimization and improvement may result in momentous developments in real-time applicability and performance measures. Enhancing the CNN and autoencoder models would increase their strength and generalizability by regulating them to accommodate a greater variety of flood situations and conservational situations. Exploring more complex deep learning architectures, like graph neural networks or commitment mechanisms, may help to improve anomaly detection skills and harvest more complex valuations of flood-related damage. Including multi-modal data foundations, such as sensor data or public broadcasting feeds, in adding to satellite and drone photos, could improve the input to AI models and deliver more thorough situational consciousness during crisis incidences.

Second, in directive for AI-driven disaster retrieval systems to be extensively used, it is authoritative that deployment and scalability issues be addressed. Succeeding research events may essence on augmenting computational ability and resource management to enable immediate processing and judgement in ever-changing disaster circumstances. The implementation of AI models in resource- constrained contexts might be made conceivable by drifting cloud-based solutions or edge computing systems, which would authenticate convenience and responsiveness during crucial response stages. The human-in-the-loop frameworks that take benefit of emergency answer and help establishments' proficiency could improve AI-driven systems' usability and reaction, inspiring collective decision- making and adaptive rejoinder methods.

Chapter 7 – Student Reflections

Starting this project was a noteworthy revolving point in my academic career by bestowing me with tasks never seen before in footings of scope and complexity. The project looked devastating at first, but it eventually turned into a life-changing opportunity for skill expansion and personal growth, teaching me inestimable lessons that will confidently inspiration my future academic and professional endeavours. Throughout the endeavour, I travelled across unexplored terrain and overpowered every challenge with resolute resolve and plasticity. Time restrictions instigated some minor nonconformities from my original plan, but I still advanced project management systematically, following my timeline accurately and seeking unvarying advice from my supervisor, whose knowledge and assistance were invaluable in fine-tuning the particulars of the research process.

Absolutely one of the solidest things I had to deal with was really putting the project into repetition. Protracted trial processing durations and hardware reserve constrictions, including limited online GPU access, posed significant obstacles. But I overcame these challenges with perseverance and the help of my aristocracies and manager, and in the end, I ended the project on time and within the quantified scope. Magnificent on this exploit, I allow the plethora of relational and technical talents I have selected up along the technique. The project gave me a rich and immersive learning experience that improved everything from my time management skills to my problem-solving abilities, giving me invaluable tools for navigating my future academic and specialised endeavours.

References

Agrawal, T. and Meleet, M. (2021) Classification of natural disaster using satellite & drone images with CNN using transfer learning, in: *2021 International Conference on Innovative Computing, Intelligent Communication and Smart Electrical Systems (ICSES)*. pp. 1–5.

Bashir, M. H., Ahmad, M., Rizvi, D. R. and El-Latif, A. A. A. (2024) Efficient CNN-based disaster events classification using UAV-aided images for emergency response application, *Neural Computing and Applications*, pp. 1–14.

Blomeier, E., Schmidt, S. and Resch, B. (2024) Drowning in the Information Flood: Machine-Learning- Based Relevance Classification of Flood-Related Tweets for Disaster Management, *Information*, 15 (3),
pp. 149.

Damaševičius, R., Bacanin, N. and Misra, S. (2023) From sensors to safety: Internet of Emergency Services (IoES) for emergency response and disaster management, *Journal of Sensor and Actuator Networks*, 12 (3), pp. 41.

Daniels, J., Bailey, C. P. and Liang, L. (2022) Filling Cloud Gaps in Satellite AOD Retrievals Using an LSTM CNN-Autoencoder Model, in: *IGARSS 2022-2022 IEEE International Geoscience and Remote Sensing Symposium*. pp. 2758–2761.

Deborah, N., Rajiv, A., Vinora, A., Sivakarthi, G. and Soundarya, M. (2024) Machine Learning Algorithms for Natural Disasters, in: *Internet of Things and AI for Natural Disaster Management and Prediction*. IGI Global, pp. 188–212.

ES, P. K., Thatha, V. N., Mamidisetti, G., Mantena, S. V, Chintamaneni, P. and Vatambeti, R. (2023) Hybrid deep learning model with enhanced sunflower optimization for flood and earthquake detection, *Heliyon*, 9 (10).

Feng, J. and Sun, Y. (2024) Multiscale network based on feature fusion for fire disaster detection in complex scenes, *Expert Systems with Applications*, 240, pp. 122494.

GHOSH, B., Garg, S. and Motagh, M. (2022) Automatic flood detection from Sentinel-1 data using deep learning architectures, *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 3, pp. 201–208.

Hakim, W. L., Rezaie, F., Nur, A. S., Panahi, M., Khosravi, K., Lee, C. W. and Lee, S. (2022) Convolutional neural network (CNN) with metaheuristic optimization algorithms for landslide susceptibility mapping in Icheon, South Korea, *Journal of Environmental Management*, 305, pp. 114367.

Hernández, D., Cecilia, J. M., Cano, J. C. and Calafate, C. T. (2022) Flood detection using real-time image segmentation from unmanned aerial vehicles on edge-computing platform, *Remote Sensing*, 14 (1),
pp. 223.

Hussain, A., Latif, G., Alghazo, J. and Kim, E. (2024) Flood detection using deep learning methods from visual images, *AIP Conference Proceedings*, 3034 (1).

Islam, M. A., Rashid, S. I., Hossain, N. U. I., Fleming, R. and Sokolov, A. (2023) An integrated convolutional neural network and sorting algorithm for image classification for efficient flood disaster management, *Decision Analytics Journal*, 7, pp. 100225.

Jackson, J., Yussif, S. B., Patamia, R. A., Sarpong, K. and Qin, Z. (2023) Flood or non-flooded: a comparative study of state-of-the-art models for flood image classification using the FloodNet dataset with uncertainty offset analysis, *Water*, 15 (5), pp. 875.

Jamali, A., Roy, S. K., Beni, L. H., Pradhan, B., Li, J. and Ghamisi, P. (2024) Residual wave vision U-Net for flood mapping using dual polarization Sentinel-1 SAR imagery, *International Journal of Applied Earth Observation and Geoinformation*, 127, pp. 103662.

Jiang, Z., Wang, M. and Liu, K. (2023) Comparisons of convolutional neural network and other machine learning methods in landslide susceptibility assessment: a case study in Pingwu, *Remote Sensing*, 15 (3), pp. 798.

Khan, S. D. and Basalamah, S. (2023a) Multi-Scale and Context-Aware Framework for Flood Segmentation in Post-Disaster High Resolution Aerial Images, *Remote Sensing*, 15 (8), pp. 2208.

Khan, S. D. and Basalamah, S. (2023b) Multi-Scale and Context-Aware Framework for Flood Segmentation in Post-Disaster High Resolution Aerial Images, *Remote Sensing*, 15 (8), pp. 2208.

Kumar, T. V, Yeruva, A. R., Kumar, S., Gangodkar, D., Rao, A. L. N. and Chaturvedi, P. (2022) A New Vehicle Tracking System with R-CNN and Random Forest Classifier for Disaster Management Platform to Improve Performance, in: *2022 2nd International Conference on Technological Advancements in Computational Sciences (ICTACS)*. pp. 797–804.

Kumbam, P. R. and Vejre, K. M. (2024) FloodLense: A Framework for ChatGPT-based Real-time Flood Detection, *ArXiv Preprint ArXiv:2401.15501*.

Linardos, V., Drakaki, M., Tzionas, P. and Karnavas, Y. L. (2022) Machine learning in disaster management: recent developments in methods and applications, *Machine Learning and Knowledge Extraction*, 4 (2).

Li, Z., Sermet, M. Y. and Demir, I. (2024) EarthObsNet: A Comprehensive Benchmark Dataset for Data- Driven Earth Observation Image Synthesis.

Michail, E., Giannakeris, P., Koulalis, I., Vrochidis, S. and Kompatsiaris, I. (2024) Innovative Visual Analysis Solutions to Support Disaster Management, in: *Responding to Extreme Weather Events*. pp. 199–223.

Munawar, H. S., Ullah, F., Qayyum, S., Khan, S. I. and Mojtahedi, M. (2021) UAVs in disaster management: Application of integrated aerial imagery and convolutional neural network for flood detection, *Sustainability*, 13 (14), pp. 7547.

Pi, Y., Nath, N. D. and Behzadan, A. H. (2021) Detection and semantic segmentation of disaster damage in UAV footage, *Journal of Computing in Civil Engineering*, 35 (2), pp. 4020063.

Puthran, R. (2024) Spatio-Temporal Analysis of Hybrid CNN-GRU Model for Prediction of Earthquake for Disaster Management, *International Journal of Intelligent Systems and Applications in Engineering*, 12 (3s), pp. 270–281.

Qin, H., Wang, J., Mao, X., Zhao, Z. A., Gao, X. and Lu, W. (2024) An Improved Faster R-CNN Method for Landslide Detection in Remote Sensing Images, *Journal of Geovisualization and Spatial Analysis*, 8 (1), pp. 2.

Rahnemoonfar, M., Chowdhury, T. and Murphy, R. (2023) RescueNet: a high resolution UAV semantic segmentation dataset for natural disaster damage assessment, *Scientific Data*, 10 (1), pp. 913.

Rahnemoonfar, M., Chowdhury, T., Sarkar, A., Varshney, D., Yari, M. and Murphy, R. R. (2021) Floodnet: A high resolution aerial imagery dataset for post flood scene understanding, *IEEE Access*, 9, pp. 89644– 89654.

Raj, T. A. B., Pushpalatha, C. and Ahilan, A. (2023) An optimized profound memory-affiliated de-noising of aerial images through deep neural network for disaster management, *Signal, Image and Video Processing*, 17 (8), pp. 3983–3991.

Ramasubramanian, P. (2021) Disaster management using deep learning on social media, *International Journal of Applied Science and Engineering*, 18 (2), pp. 1–8.

Reis, H. C. and Turk, V. (2023) Detection of forest fire using deep convolutional neural networks with transfer learning approach, *Applied Soft Computing*, 143, pp. 110362.

Reis, H. C., Turk, V., Karacur, S. and Kurt, A. M. (2024) Integration of a CNN-based model and ensemble learning for detecting post-earthquake road cracks with deep features, *Structures*, 62, pp. 106179.

Ro, S. H., Li, Y. and Gong, J. (2024) A Machine learning approach for Post-Disaster data curation,
Advanced Engineering Informatics, 60, pp. 102427.

Rumapea, H., Zarlis, M., Efendy, S. and Sihombing, P. (2024) Improving Convective Cloud Classification with Deep Learning: The CC-Unet Model, *International Journal on Advanced Science, Engineering & Information Technology*, 14 (1).

Sener, A., Doğan, G. and Ergen, B. (2024) A novel convolutional neural network model with hybrid attentional atrous convolution module for detecting the areas affected by the flood, *Earth Science Informatics*, 17 (1), pp. 193–209.

Seydi, S. T., Saeidi, V., Kalantar, B., Ueda, N., van Genderen, J. L., Maskouni, F. H. and Aria, F. A. (2022) Fusion of the multisource datasets for flood extent mapping based on ensemble convolutional neural network (CNN) model, *Journal of Sensors*, pp. 1–20.

Shakeel, M., Itoyama, K., Nishida, K. and Nakadai, K. (2021) Detecting earthquakes: a novel deep learning-based approach for effective disaster response, *Applied Intelligence*, 51 (11), pp. 8305–8315.

Shakibaei, H., Farhadi-Ramin, M. R., Alipour-Vaezi, M., Aghsami, A. and Rabbani, M. (2023) Designing a post-disaster humanitarian supply chain using machine learning and multi-criteria decision-making techniques, *Kybernetes*.

Subhashini, R., Thomas, J. J., Sivasangari, A., Mohana, P., Vigneshwari, S. and Asha, P. (2022) Artificial intelligence–based intelligent geospatial analysis in disaster management, in: *Advances of Artificial Intelligence in a Green Energy Environment*. Academic Press, pp. 203–221.

Tan, W., Qin, N., Zhang, Y., McGrath, H., Fortin, M. and Li, J. (2024) A rapid high-resolution multi-sensory urban flood mapping framework via DEM upscaling, *Remote Sensing of Environment*, 301, pp. 113956.

Varsha, V. R., Naganandini, S. and Hariharan, C. (2024) Utilizing AI and Machine Learning for Natural Disaster Management: Predicting Natural Disasters With AI and Machine Learning, in: *Internet of Things and AI for Natural Disaster Management and Prediction*. IGI Global, pp. 279–304.

Yu, X., Wu, K., Yang, Y. and Liu, Q. (2024) WaRENet: A Novel Urban Waterlogging Risk Evaluation Network, *ACM Transactions on Multimedia Computing, Communications and Applications*.

Appendix 1

```
import os
import torch
import torch.nn as nn
from torch.utils.data import DataLoader, Dataset
from torchvision import transforms, models
from PIL import Image
from torchvision.datasets import ImageFolder
import tqdm

# Define the directory paths
dataset_root = '/kaggle/input/satellite-images-of-hurricane-damage'
train_dir = os.path.join(dataset_root, 'train_another')
val_dir = os.path.join(dataset_root, 'validation_another')
test_dir = os.path.join(dataset_root, 'test')

# Define your transformations
train_transforms = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.RandomHorizontalFlip(),  # Data augmentation
    transforms.RandomRotation(10),
    transforms.ToTensor(),
])

val_test_transforms = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.ToTensor(),
])

# Load the datasets
train_dataset = ImageFolder(root=train_dir, transform=train_transforms)
val_dataset = ImageFolder(root=val_dir, transform=val_test_transforms)
test_dataset = ImageFolder(root=test_dir, transform=val_test_transforms)

# Create data loaders
train_loader = DataLoader(train_dataset, batch_size=32, shuffle=True)
val_loader = DataLoader(val_dataset, batch_size=32, shuffle=False)
test_loader = DataLoader(test_dataset, batch_size=32, shuffle=False)
class DamageCNN(nn.Module):
    def __init__(self):
        super(DamageCNN, self).__init__()
        # Using a pretrained model as a feature extractor
        base_model = models.resnet18(pretrained=True)
        # Freeze the layers
        for param in base_model.parameters():
            param.requires_grad = False
        # Replace the classifier layer
        num_features = base_model.fc.in_features
        base_model.fc = nn.Linear(num_features, 2)  # 2 classes: damaged
and not damaged

        self.base_model = base_model

    def forward(self, x):
        return self.base_model(x)
```

```
model = DamageCNN()
class DisasterAutoencoder(nn.Module):
    def __init__(self):
        super(DisasterAutoencoder, self).__init__()
        self.encoder = nn.Sequential(
            nn.Conv2d(3, 16, 3, stride=2, padding=1),
            nn.ReLU(),
            nn.Conv2d(16, 32, 3, stride=2, padding=1),
            nn.ReLU(),
            nn.Conv2d(32, 64, 7)
        )
        self.decoder = nn.Sequential(
            nn.ConvTranspose2d(64, 32, 7),
            nn.ReLU(),
            nn.ConvTranspose2d(32, 16, 3, stride=2, padding=1,
output_padding=1),
            nn.ReLU(),
            nn.ConvTranspose2d(16, 3, 3, stride=2, padding=1,
output_padding=1),
            nn.Sigmoid()  # Ensuring the output is between 0 and 1
        )

    def forward(self, x):
        encoded = self.encoder(x)
        decoded = self.decoder(encoded)
        return decoded

autoencoder = DisasterAutoencoder()
# Check if we have a GPU available and if so, use it
device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")
print(f"Using {device} device for training.")
# Define the loss function and optimizer
model = DamageCNN().to(device)
criterion = nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=0.001)

# Function for training and validation phases
from tqdm import tqdm
# Assuming you've already defined your model, loss criterion, and
optimizer

def train_model(model, train_loader, val_loader, criterion, optimizer,
num_epochs=150):
    model.to(device)
    cnn_metrics = {
        'train': {'losses': [], 'accuracies': []},
        'val': {'losses': [], 'accuracies': []}
    }

    for epoch in range(num_epochs):
        # Training phase
        model.train()
        total_train_loss = 0
```

```
        correct_train_preds = 0
        total_train_samples = 0

        for inputs, labels in tqdm(train_loader, desc="Training",
leave=False):
            inputs, labels = inputs.to(device), labels.to(device)
            optimizer.zero_grad()
            outputs = model(inputs)
            loss = criterion(outputs, labels)
            loss.backward()
            optimizer.step()

            _, preds = torch.max(outputs, 1)
            total_train_loss += loss.item() * inputs.size(0)
            correct_train_preds += torch.sum(preds == labels.data)
            total_train_samples += inputs.size(0)

        train_loss = total_train_loss / total_train_samples
        train_accuracy = correct_train_preds.double() /
total_train_samples
        cnn_metrics['train']['losses'].append(train_loss)
        cnn_metrics['train']['accuracies'].append(train_accuracy.item())

        # Validation phase
        model.eval()
        total_val_loss = 0
        correct_val_preds = 0
        total_val_samples = 0

        with torch.no_grad():
            for inputs, labels in tqdm(val_loader, desc="Validation",
leave=False):
                inputs, labels = inputs.to(device), labels.to(device)
                outputs = model(inputs)
                loss = criterion(outputs, labels)

                _, preds = torch.max(outputs, 1)
                total_val_loss += loss.item() * inputs.size(0)
                correct_val_preds += torch.sum(preds == labels.data)
                total_val_samples += inputs.size(0)

        val_loss = total_val_loss / total_val_samples
        val_accuracy = correct_val_preds.double() / total_val_samples
        cnn_metrics['val']['losses'].append(val_loss)
        cnn_metrics['val']['accuracies'].append(val_accuracy.item())

        print(f'Epoch [{epoch + 1}/{num_epochs}]: Train Loss:
{train_loss:.4f}, Train Accuracy: {train_accuracy:.4f}, Val Loss:
{val_loss:.4f}, Val Accuracy: {val_accuracy:.4f}')

    return model, cnn_metrics


dataloaders = {'train': train_loader, 'val': val_loader}
```

```
# Train the model
model, cnn_metrics = train_model(model, train_loader, val_loader,
criterion, optimizer, num_epochs=150)
import matplotlib.pyplot as plt
import torchvision

def show_images_batch(loader):
    # Get a single batch of images
    images, labels = next(iter(loader))

    # Make a grid from batch
    img_grid = torchvision.utils.make_grid(images, nrow=8)  # Adjust nrow
to fit the batch size

    # Convert tensor to numpy for displaying
    img_grid = img_grid.numpy().transpose((1, 2, 0))

    # Normalizing the pixels to [0, 1] just for visualization
    mean = [0.485, 0.456, 0.406]  # Assuming ImageNet mean
    std = [0.229, 0.224, 0.225]   # Assuming ImageNet std
    img_grid = img_grid * std + mean  # Denormalize
    img_grid = np.clip(img_grid, 0, 1)  # Clip to make sure it's valid
image data

    # Display images
    plt.figure(figsize=(12, 12))
    plt.imshow(img_grid)
    plt.axis('off')
    plt.title('Batch of Training Images')
    plt.show()

# Example usage with your DataLoader
show_images_batch(train_loader)
import matplotlib.pyplot as plt

# Plot training and validation losses
plt.plot(cnn_metrics['train']['losses'], label='Training loss')
plt.plot(cnn_metrics['val']['losses'], label='Validation loss')
plt.title('CNN Losses')
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()

# Plot training and validation accuracies
plt.plot(cnn_metrics['train']['accuracies'], label='Training Accuracy')
plt.plot(cnn_metrics['val']['accuracies'], label='Validation Accuracy')
plt.title('CNN Accuracies')
plt.xlabel('Epochs')
plt.ylabel('Accuracy')
plt.legend()
plt.show()
from sklearn.metrics import confusion_matrix
```

```
import seaborn as sns
import matplotlib.pyplot as plt

# Assuming model is your trained classification model
# Assuming val_loader is your DataLoader for the validation set
# Ensure model and data are moved to the correct device
model.to(device)
model.eval()

all_preds = []
all_true = []
with torch.no_grad():
    for images, labels in val_loader:
        images, labels = images.to(device), labels.to(device)
        outputs = model(images)
        _, preds = torch.max(outputs, 1)
        all_preds.extend(preds.view(-1).cpu().numpy())
        all_true.extend(labels.view(-1).cpu().numpy())

# Compute the confusion matrix
cm = confusion_matrix(all_true, all_preds)

# Plot the confusion matrix
plt.figure(figsize=(10, 8))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues')
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.show()
from sklearn.metrics import f1_score, precision_score, recall_score,
roc_curve, auc

# Assuming model is your trained classification model
# Assuming val_loader is your DataLoader for the validation set
model.eval()

all_preds = []
all_probas = []
all_true = []
with torch.no_grad():
    for images, labels in val_loader:
        images, labels = images.to(device), labels.to(device)
        outputs = model(images)
        _, preds = torch.max(outputs, 1)
        probas = outputs.softmax(dim=1)[:, 1]  # Probability for the
positive class
        all_preds.extend(preds.cpu().numpy())
        all_probas.extend(probas.cpu().numpy())
        all_true.extend(labels.cpu().numpy())

# Calculate precision, recall, and F1 score
precision = precision_score(all_true, all_preds)
recall = recall_score(all_true, all_preds)
f1 = f1_score(all_true, all_preds)
```

```
print(f'Precision: {precision:.4f}')
print(f'Recall: {recall:.4f}')
print(f'F1 Score: {f1:.4f}')

# Compute ROC curve and AUC
fpr, tpr, thresholds = roc_curve(all_true, all_probas)
roc_auc = auc(fpr, tpr)

# Plot ROC curve
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label=f'ROC curve (area =
{roc_auc:.2f})')
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic')
plt.legend(loc='lower right')
plt.show()
def plot_learning_rate(optimizer, num_epochs):
    lrs = [param_group['lr'] for param_group in optimizer.param_groups
for _ in range(num_epochs)]
    plt.figure(figsize=(10, 5))
    plt.plot(lrs, color='blue')
    plt.title("Learning Rate Over Time")
    plt.xlabel("Epoch")
    plt.ylabel("Learning Rate")
#     plt.grid(True)
    plt.show()

plot_learning_rate(optimizer, 25)
import torch
import torch.nn as nn
from torch.utils.data import DataLoader
from torchvision import datasets, transforms, models
import os

# Check for a GPU
device = torch.device("cuda:0" if torch.cuda.is_available() else "cpu")

# Define the transformations
transform = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.ToTensor(),
])

# Create DataLoaders
def create_dataloader(root_dir, subfolder):
    full_path = os.path.join(root_dir, subfolder)
    dataset = datasets.ImageFolder(root=full_path, transform=transform)
    is_train = 'train' in subfolder  # Assuming that 'train' in the name
denotes a training set
    return DataLoader(dataset, batch_size=32, shuffle=is_train)

# Assuming dataset_root is your path to the dataset
```

```
dataset_root = '/kaggle/input/satellite-images-of-hurricane-damage'
train_loader = create_dataloader(dataset_root, 'train_another')
val_loader = create_dataloader(dataset_root, 'validation_another')

# Instantiate the model
autoencoder = DisasterAutoencoder().to(device)

# Define loss function and optimizer
criterion = nn.MSELoss()
optimizer = torch.optim.Adam(autoencoder.parameters(), lr=0.001)

# Training function
# Training function with progress bars
def train_autoencoder(autoencoder, train_loader, val_loader, criterion,
optimizer, num_epochs=25):
    metrics = {'train_losses': [], 'val_losses': []}

    for epoch in range(num_epochs):
        print(f'Epoch {epoch+1}/{num_epochs}')

        # Training Phase
        autoencoder.train()
        train_loss = 0.0
        # Wrap the train_loader with tqdm for a progress bar
        for inputs, _ in tqdm(train_loader, desc="Training",
leave=False):
            inputs = inputs.to(device)
            optimizer.zero_grad()
            outputs = autoencoder(inputs)
            loss = criterion(outputs, inputs)
            loss.backward()
            optimizer.step()
            train_loss += loss.item() * inputs.size(0)
        train_loss /= len(train_loader.dataset)
        metrics['train_losses'].append(train_loss)

        # Validation Phase
        autoencoder.eval()
        val_loss = 0.0
        # Wrap the val_loader with tqdm for a progress bar
        for inputs, _ in tqdm(val_loader, desc="Validation",
leave=False):
            inputs = inputs.to(device)
            outputs = autoencoder(inputs)
            loss = criterion(outputs, inputs)
            val_loss += loss.item() * inputs.size(0)
        val_loss /= len(val_loader.dataset)
        metrics['val_losses'].append(val_loss)

        print(f'Train Loss: {train_loss:.4f}, Val Loss: {val_loss:.4f}')

    return autoencoder, metrics

# Run the training process
```

```
autoencoder, autoencoder_metrics = train_autoencoder(
    autoencoder, train_loader, val_loader, criterion, optimizer,
num_epochs=25
)

# Plot training and validation losses
plt.plot(autoencoder_metrics['train_losses'], label='Training loss')
plt.plot(autoencoder_metrics['val_losses'], label='Validation loss')
plt.title("AutoEncoder Training and Validation Losses")
plt.xlabel('Epochs')
plt.ylabel('Loss')
plt.legend()
plt.show()


for name, param in model.named_parameters():
    if 'weight' in name and param.requires_grad:
        plt.figure()
        plt.title(name)
        plt.hist(param.cpu().detach().numpy().flatten(), bins=100)
        plt.xlabel('Weight Value')
        plt.ylabel('Frequency')
        plt.show()
import seaborn as sns

for name, param in model.named_parameters():
    if 'weight' in name and param.requires_grad and len(param.size()) >
1:
        plt.figure()
        sns.heatmap(param.cpu().detach().numpy().squeeze(), annot=False)
        plt.title(name)
        plt.xlabel('Input Features')  # Adjust this label to match the
context
        plt.ylabel('Output Features')  # Adjust this label to match the
context
        plt.show()
def visualize_reconstructions(model, loader, device, n_images=5):
    model.eval()
    images, _ = next(iter(loader))  # Get a batch of images
    images = images.to(device)
    with torch.no_grad():
        reconstructions = model(images)

    images = images.cpu().numpy()
    reconstructions = reconstructions.cpu().numpy()

    plt.figure(figsize=(10, 4))
    for i in range(n_images):
        # Display original images
        ax = plt.subplot(2, n_images, i + 1)
        plt.imshow(images[i].transpose(1, 2, 0))
        plt.title("Original")
        plt.axis('off')
```

```
        # Display reconstructed images
        ax = plt.subplot(2, n_images, i + 1 + n_images)
        plt.imshow(reconstructions[i].transpose(1, 2, 0))
        plt.title("Reconstructed")
        plt.axis('off')
    plt.show()

visualize_reconstructions(autoencoder, val_loader, device)
import numpy as np
def plot_encoded_space(model, loader, device):
    model.eval()
    encoded_samples = []
    labels = []
    for images, label in loader:
        images = images.to(device)
        with torch.no_grad():
            encoded_imgs = model.encoder(images)
            encoded_imgs = encoded_imgs.view(images.size(0), -1)
            encoded_samples.extend(encoded_imgs.cpu().numpy())
            labels.extend(label.numpy())

    encoded_samples = np.array(encoded_samples)
    labels = np.array(labels)

    plt.figure(figsize=(10, 7))
    scatter = plt.scatter(encoded_samples[:, 0], encoded_samples[:, 1],
c=labels, alpha=0.5, cmap='viridis')
    plt.colorbar(scatter)
    plt.xlabel('Encoded Dimension 1')
    plt.ylabel('Encoded Dimension 2')
    plt.title('Feature Distribution in Encoded Space')
    plt.grid(True)
    plt.show()

# Uncomment to use it if your encoded space is 2D and small enough to
visualize
plot_encoded_space(autoencoder, val_loader, device)
def plot_output_histogram(model, dataloader):
    model.eval()
    softmax = nn.Softmax(dim=1)
    all_outputs = []

    with torch.no_grad():
        for images, _ in dataloader:
            images = images.to(device)
            outputs = softmax(model(images))
            all_outputs.extend(outputs.cpu().numpy())

    all_outputs = np.array(all_outputs)
    plt.figure(figsize=(10, 5))
    plt.hist(all_outputs.flatten(), bins=50, color='blue', alpha=0.7)
    plt.title('Histogram of Model Output Scores')
    plt.xlabel('Softmax Score')
    plt.ylabel('Frequency')
```

```
    plt.show()

# Example usage:
plot_output_histogram(model, val_loader)
```

Appendix 1: MSc Dissertation Ethics Form

Division of
Computer Science & Informatics
School of Engineering

CHECKLIST FOR ETHICS REVIEWS

Project Supervisor:	Francis Babayemi
Project Title:	Damage Assessment (Disaster Management)
Student Name:	.Asaduzzaman
Student Number:	4142378

In the planning and design of your project ethical issues must always be considered. If your project does not involve testing and/or evaluating software with end users, or any other contact with people, then the general code of ethics for computing will apply.

If your project does involve contact with people then the issues outlined in this form and expressed in detail in the ACM code of ethics must be considered.

For all projects you must complete this form, discuss it with your supervisor and have it signed off. You may also be required to produce more details.

When the form is complete and signed off your supervisor will pass it onto the departmental ethics committee. You do not have ethical approval to proceed until you have received a reply from the committee via the supervisor.
If your project plan changes then ethical approval may need to be reviewed.

Tick this box to confirm that you have read the ACM Code of Ethics and Professional Conduct. https://ethics.acm.org/	✓
Tick this box if your project does not involve any contact with people. (If so **DO NOT** complete the rest of the form.Q1-Q14)	✓

Student Name:	Signature	Date
Asaduzzaman	Asad	19/03/2024
Supervisor Name:	**Signature**	**Date**
Francis Babayemi	FBabayemi	20/03/24
Ethics Approval:	**Signature**	**Date**