A Graph Neural Network Based Approach for Predicting Wildfire Burned Areas

by

Ursula Das

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Applied Science
in
Electrical and Computer Engineering

Waterloo, Ontario, Canada, 2025

© Ursula Das 2025

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

Abstract

Wildfires annually cause substantial economic and environmental losses and has a detrimental impact on human lives and health due to the release of their harmful byproducts. Moreover, wildfire incidents have exhibited an alarming surge in frequency as well as severity in recent years due to increased urbanization near forested areas coupled with climate change, highlighting the need for advanced technologies to predict wildfire behavior in advance and mitigate its impact. In recent years, the enormous strides in machine learning research coupled with the increased availability of wildfire data through various sources such as remote sensing and the increased availability of computational resources have fueled the rise of data-driven approaches across all stages of wildfire management. Despite the growing adoption of machine learning-driven approaches in wildfire mitigation, the primary focus has been on analyzing historical patterns and identifying the causes leading to wildfire patterns rather than predicting wildfire behavior. The prediction of wildfire behavior over time, such as the burned area has been largely underexplored. This study aims to address this gap by advancing data-driven methods for predicting wildfire behavior during the active fire stage and aiding in resource allocation efforts.

This study adopts a Graph Neural Network based framework for predicting the burned area resulting from a wildfire ignition. While CNN-based architectures have been widely employed to model wildfire behavior, including spread prediction, as a semantic segmentation task, these architectures impose specific limitations on geospatial data due to their reliance on fixed-size inputs and local receptive fields. Graph Neural Network (GNNs), have shown success in capturing the long-range dependencies and irregular-sized inputs inherent in geospatial data, such as wildfires, making them a viable alternative to CNNs. To this end, a GNN-based approach is adopted to model wildfire burned area prediction. A framework is developed to represent spatial wildfire data and its influencing factors as homogeneous graphs followed by the development of three distinct GNN models based on different message-passing mechanisms to process the graph-structured data. The results obtained through various experiments illustrate the efficacy of Graph Neural Networks in modeling wildfire behavior. In terms of Precision, most GNN models outperform the segmentation models, with the highest achieving a score of 0.4536. For AUROC, all GNN models demonstrate superior performance, reaching a maximum of 0.9377. Based on AUPRC, the Graph Convolutional Network (GCN) model surpasses all others, including segmentation models, with a top score of 0.4787. These findings underscore the potential of Graph Neural Networks (GNNs) as a powerful tool for wildfire behavior modeling and supporting resource allocation initiatives.

Acknowledgements

I would like to express my sincere gratitude to my supervisor, Prof. Kshirasagar Naik, for his unwavering support throughout this entire journey. His promptness in addressing any questions, and his keen attention to detail has been instrumental in my professional growth. I would also like to express my gratitude to Dr. Marzia Zaman for her consistent encouragement and invaluable feedback throughout my thesis. I would also like to express my gratitude to Prof. Pin Han Ho for his invaluable time and insightful feedback on my thesis. Additionally, I would like to express my appreciation to Dr. Chung-Horng Lung at Carleton University, and Dr. Srinivas Sampalli at Dalhousie University for their resourceful suggestions. I also thank my colleagues in this project for their valuable insights. I would also like to express my gratitude to my friend, Dheeraj Khanna, for his invaluable assistance and feedback throughout the thesis. Last but not least, I would like to extend my heartfelt appreciation to my parents, sister, grandmother, and my entire family. Their unwavering support and belief in me have been instrumental in my journey to this esteemed institution.

Dedication

This thesis is dedicated to my parents, Urvashi and Anjan, my sister Sunniva, my grandmother Anu . I owe my entire existence and success to my beloved mother, Urvashi, whose countless sacrifices and unconditional love have shaped me into the person I am today. I am forever grateful to my father, Anjan, whose unwavering love, and belief in my potential have empowered me to pursue my dreams with confidence. To my sister, Sunniva, who has been my biggest cheerleader and whose boundless enthusiasm and support have been a constant source of strength. And to my grandmother, Anu, whose nurturing presence has been a cornerstone of my life. I also dedicate this to my late grandfathers Deuta Koka and Baba Koka, who may no longer be with us but continue to inspire me in many ways and who would have loved to see me reach this stage.

Lastly, I thank my aunts and uncles - Asha, Amlan, Nisha, Pinky, Siddhartha and, Upasana without whose support and motivation I wouldn't have reached this destination.

All of them have been my unwavering pillars of support throughout my life. Without their staunch belief in my abilities and their support and love, I would not have had the courage and resilience to relocate to a foreign country and start life anew or have the opportunity to pursue my education at one of the world's most prestigious institutions.

Table of Contents

Author's Declaration ii

Abstract iii

Acknowledgements iv

Dedication v

List of Figures ix

List of Tables xi

List of Symbols xii

1 Introduction 1

1.1 Motivation 1

1.2 Problem Formulation 3

1.3 Contributions 4

1.4 Outline of Thesis 5

2 Background & Related Work 8

2.1 Wildfire Datasets 8

2.2 Machine Learning in Wildfire Management 10

2.3 Traditional Models for Wildfire Spread & Burned Area Prediction 11
2.4 Semantic Segmentation in Wildfire Applications 12
2.5 Graph Neural Networks 13
2.5.1 Background 13
2.5.2 Categories of Applications for Graph Neural Networks 15
2.5.3 Graph Neural Networks in Wildfire Management 16

3 Exploratory Data Analysis **18**
3.1 Study Area 18
3.2 Dataset Description 19
3.3 Imbalance Analysis 22
3.4 Data Preprocessing 26

4 Methodology **29**
4.1 Graph Neural Network 29
4.1.1 Problem Formulation 30
4.1.2 Graph Dataset Creation 30
4.1.3 Model Architecture 33
4.2 Semantic Segmentation 37
4.2.1 Problem Formulation 37
4.2.2 Model Architectures 38

5 Experiments & Results **41**
5.1 Experimental Setup 41
5.2 Performance Metrics 43
5.3 Loss Functions for Imbalanced Class Distributions 46
5.3.1 Distribution-based Loss Functions 46
5.3.2 Region-based Losses 48

5.4 Comparative Analysis of Loss Functions for Handling Class Imbalance . . . 48
5.5 Model Configuration . . . 51
5.5.1 GCN . . . 53
5.5.2 GAT . . . 54
5.5.3 GraphSAGE . . . 55
5.6 Comparison of GNN Models . . . 58
5.7 Ablation Study: Feature Importance . . . 59
5.8 Comparison of Semantic Segmentation and Graph Neural Networks . . . 65

6 Conclusion & Future Work **69**
6.1 Conclusion . . . 69
6.2 Future Work . . . 70

References **72**

List of Figures

1.1 Visual Representation of Problem Formulation 5

3.1 Distribution of Wildfires by Year 19

3.2 A visual representation of the (a) slope and (b) the agricultural landcover fraction feature channels for a data sample. The colourbar indicates a continuous range of values 20

3.3 Visual representation of a sample (X, Y). Here, f and t represent the input feature variables and the target variable at the (i, j)-th location, respectively. 24

3.4 Visual representation of a single sample S illustrating all features listed in Table 3.1. 25

3.5 Histogram of Imbalance Ratios 26

3.6 Histogram of Burn Ratios 26

3.7 Boxplot of Burned Area Sizes 28

4.1 Visual Representation of GNN Problem Formulation 31

4.2 Graph Transformation Process for a Grid Subset. Red indicates burnt nodes (target class 1), while green represents unburnt nodes (target class 0) . . . 33

4.3 Overall Workflow and Model Architecture of GNN 36

4.4 Message Passing Mechanism for a specific GNN. α, d, and $d*$ denote the attention coefficient, normalized degree and generalized aggregation coefficient respectively. 37

4.5 Semantic Segmentation Workflow 38

4.6 Semantic Segmentation Pipeline 39

5.1 Predicted Probabilities for a Test Sample 51
5.2 Impact of Loss Functions on Model Performance 52
5.3 Performance Comparison of GCN Configurations 55
5.4 Performance Comparison of GAT Configurations 56
5.5 Performance Comparison of GraphSAGE Configurations 57
5.6 Target and Predicted Burned Areas for GNN models 61
5.7 Performance Comparison of GNN Models 62
5.8 Performance Comparison of Feature Ablations 64
5.9 Target and Predicted Burned Areas for Feature Ablations 65
5.10 Performance Comparison of GNN & Segmentation Models 68

List of Tables

1.1 List of Notations 7

3.1 List of all features in the Mesogeos Dataset [49] 23

5.1 Temporal Split of Dataset 42

5.2 Hyperparameters used in the training phase. 43

5.3 Notations for Performance Metrics 44

5.4 Configuration of GNN Models for Loss Comparison 49

5.5 Results for Loss Functions. (**Average*** indicates the average computed for columns marked with *) 51

5.6 List of Model Configurations 53

5.7 Test Results of GNN Model Configurations 60

5.8 Notations for Feature Category 63

5.9 Test Results for Feature Ablation 65

5.10 Results of Semantic Segmentation and GNN Comparison 67

List of Symbols

A Adjacency Matrix 30

C Number of 2D input features 3, 32, 36

D_{Tr} Training Dataset 3

E Set of all edges in a graph 30

$G = (A, X_g)$ Attributed Graph 32

$G = (V, E)$ Graph representation in terms of edges and nodes 30

L Total number of samples in the training dataset 3

N Spatial dimension of a sample 3, 4, 30

Q Segmentation mask 37

V Set of all nodes in a graph 30

(X, Y) A single sample 3, 30, 32

X_g Node Feature matrix of graph G 30, 32, 36

X Input corresponding to a sample 3, 4, 14, 20

Y_g Node Output Matrix of graph G 30, 32

$\hat{Y}$ Predicted output of a sample 4

Y Output corresponding to a sample 3

$e_{(v_{ij}, v_{i'j'})}$ Edge between nodes v_{ij} and $v_{i'j'}$ 30

f_S Function learned by segmentation model 37

f_C Function learned by classifier 30

f_G Function learned by GNN model 30

f Generalized prediction function learned by the Deep Learning Model 3, 4

h_{gv} Hidden embedding of node v belonging to graph G 30

(i, j) Coordinates of a cell in the spatial grid 4

$\mathcal{N}(v)$ Neighborhood of node v 30

$r_{(v_{ij}, v_{i'j'})}$ Edge weight between nodes v_{ij} and $v_{i'j'}$ 30

v_{ij} Node corresponding to coordinates (i, j) in the spatial grid 30, 32

x_{gv} Feature vector of node v belonging to graph G 30

x_{ij} Feature vector of the cell with coordinates (i, j) 4, 32

y_{gv} Target output class of node v belonging to graph G 30

$\hat{y}_{gv}$ Predicted output of node v belonging to graph G 30

$\hat{y_{ij}}$ Predicted output of the cell with coordinates (i, j) 4

y_{ij} Output of the cell with coordinates (i, j) 4, 32

Chapter 1

Introduction

1.1 Motivation

Wildfires, also known as forest fires or wildland fires, have seen a massive surge in frequency in recent years. Although wildfires are important for maintaining the natural order of the ecosystem, the increase in urbanization near Wildland Urban Interfaces (WUI) coupled with climate change has resulted in frequent and a higher number of extreme forest fires [79]. This has necessitated the improvement of existing methodologies used in wildfire management. The existing methods for managing wildfires can be classified into three major categories pre-fire, active-fire, and post-fire [41],[7]. The pre-fire phase encompasses the assessment of fire risk, analysis of fuel properties, and other activities aimed at preventing or reducing the risk of forest fires. On the other hand, the active fire phase includes activities that aim to reduce the impact of active forest fires, such as the detection and localization of active fires, the optimization of wildfire suppression methods, and the prediction of the behavior of active fires. Lastly, the post-fire phase involves analyzing the impact of forest fires on various aspects, such as the extent of damage in forest areas, post-fire regeneration of forests, impact on wildlife, economic losses, and health impacts of by-products produced during the burning of wildfires.

With increased access to computational power and the availability of large amounts of data, the use of Machine Learning techniques across all three stages of wildfire management has seen a steady increase in the past decade. Based on the objective to be achieved, the applications within each wildfire management stage can be further categorized as predictive, prescriptive, diagnostic, or descriptive. Descriptive tasks, such as identifying wildfire hotspots and susceptibility mapping based on historical wildfire data, are concerned with

the identification of patterns in past fire events whereas diagnostic tasks, such as understanding the relative importance of environmental features in wildfire occurrence, focus on identifying the underlying cause leading to specific patterns in wildfire occurrence. On the other hand, predictive applications, such as forecasting the burned area resulting from an ignition, are concerned with the prediction of future outcomes given historical data. Prescriptive applications go beyond prediction by recommending specific actions or decisions to optimize outcomes such as optimization of evacuation or firefighting routes. Although there has been a sharp increase in AI-related wildfire research, this increase has been far from uniform across the aforementioned areas. While machine learning has been extensively used in descriptive and diagnostic analytics, machine learning driven approaches have been largely under-explored in predictive tasks such as forecasting future wildfire events and the extent of the burned area resulting from a wildfire [41]. Hence, the proposed approach aims to advance data-driven methods for predicting wildfire behavior in the active fire stage, moving beyond descriptive and diagnostic analyses.

The active fire stage involves the precise identification of the ignition locations of the wildfires and the prediction of the active fire behavior. The objective of this stage is to enable effective crisis management by providing timely and actionable information to decision-makers. Although substantial literature exists in predicting the spread of wildfire in terms of the progression of the fire front [35], the spread rate [45], the height of smoke [87], the prediction of the extent or severity of the final burned area resulting from an ignition is limited. It is crucial to effectively predict the scale of the fire in the nascent stages of an ignition in order to effectively plan resource allocation and rescue efforts in at-risk areas. Although [85],[29], contribute to assessing the size and severity of the burned areas by classifying them into distinct classes and predicting the absolute burned area size, the spatial extent of fire spread is not taken into account. However, it is equally important to delineate at-risk regions and the spatial extent of wildfires to facilitate more effective resource allocation strategies. To the best of our knowledge, [49] and [15] are the only known existing methods that predict the final burned area pattern resulting from an initial ignition.

Semantic segmentation, a highly researched task in computer vision, involves classifying each pixel of an image and has been extensively applied in wildfire science. A majority of the existing literature focuses on the segmentation of active wildfires in RGB images acquired using UAVs or cameras, such as in [26], [27]. Since the advent of Convolutional Neural Networks, [52], it has been widely used in various tasks with image data sets such as image classification [50], instance segmentation [33] and, semantic segmentation [66], [10]. In wildfire-related applications, convolution-based architectures such as UNet [66] and DeepLabV3 [10] have been widely used for semantic segmentation in RGB images and

satellite imagery. The success of CNNs with image data can be attributed to their ability to effectively utilize the shift-invariance and local connectivity inherent in image data.

However, convolutional neural networks (CNN) have certain limitations when applied to geographical data due to their reliance on local receptive fields. While CNN performs well in capturing local spatial patterns in euclidean data, such as images, their capabilities in capturing long-range dependencies inherent in geographical data, such as in wildfires [94], is largely limited. Recently, Graph Neural Networks have proven to be successful in modeling complex relationships in geographical data such as weather [51] and El Niño forecasting [69]. By representing locations as nodes in a graph and interactions between them as edges, GNNs can efficiently model complex patterns of information flow, even between regions separated by large distances. Moreover, most machine learning algorithms require a fixed-size input, which, while suitable for image data, does not work well with geographical data due to irregular regional boundaries and varying formats in which such data is available[76], [94]. To this end, a Graph Neural Network based approach is adopted for predicting the final burned area resulting from a given ignition point.

1.2 Problem Formulation

The objective of the proposed approach is to estimate the extent of the final burned area resulting from an initial ignition point given the location of the ignition and other influencing features on the day of the ignition event. This objective is achieved by formulating the problem as a supervised machine learning task and developing a deep-learning model to accurately predict the final burned area.

In the context of supervised machine learning, prediction is defined as the process of learning a mapping from an input X to an output Y given a labeled training dataset D_{Tr} consisting of similar input-output pairs [60]. A single (X, Y) pair is referred to as a sample in the rest of the paper. Each sample corresponds to a specific geographical area centered around the ignition point of the fire event. For uniform spatial analysis across all samples with varying burned area patterns, we assume a uniform width and height of size N resulting in an area of $N \times N$ km^2 around the ignition point. Based on the aforementioned definition of prediction, in our proposed approach, each input $X \in R^{N \times N \times C}$ is a set of C two-dimensional grids, where each grid represents a feature and the corresponding output $Y \in R^{N \times N \times 1}$ is a single two-dimensional grid. A visual representation of a single sample (X, Y) is shown in Figure 3.3. The training dataset can then be represented by the set $D_{Tr} = \sum_{i=1}^{L} (X_i, Y_i)$ where L is the total number of samples in the training set and f is the function learned by the model during training. The function f learned from the trained

model is validated by predicting the output for samples not seen during training. The predicted output $\hat{Y}$ for any input X, either seen or unseen, can be computed as shown in Equation 1.1.

$$\hat{Y} = f(X) \tag{1.1}$$

A prediction task can be further narrowed down as a classification task if the output to be predicted is categorical in nature. The unit element of an $N \times N$ grid can be defined as a cell. The cell at the $(i, j)^{\text{th}}$ location of the grid can be represented by (x_{ij}, y_{ij}) where x_{ij} is the feature vector and y_{ij} is the output value. At the granular level, the proposed approach can be reformulated as a binary classification problem. The output $\hat{y_{ij}}$ predicted by the function f is the binary class $b \in \{0, 1\}$ that has the highest probability of being true for the cell (i, j). In other words, $\hat{y_{ij}}$ indicates whether the cell (i, j) is burnt(1) or unburnt(0), due to the given ignition point.

$$\hat{y}_{ij} = \arg \max_{b \in \{0,1\}} p(y_{ij} = b \mid x). \tag{1.2}$$

1.3 Contributions

The key contributions of this work are as follows:

- Creation of a Graph Neural Network based framework for the prediction of the final burned area resulting from a wildfire ignition. Develop a pipeline to transform a multidimensional wildfire dataset into a graph structured dataset. The framework includes the development of three distinct Graph Neural Network (GNN) architectures based on distinct message passing mechanisms. A comprehensive analysis is performed to compare the three architectures and assess their strengths and limitations in wildfire burned area prediction.
- Evaluate and address the data imbalance inherent in spatial wildfire datasets by utilising various loss functions tailored for data imbalance and assess the importance of choosing suitable performance metrics for imbalanced class distributions in wildfire datasets. This analysis offers valuable guidance for future research focused on handling skewed data in environmental modeling. Additionally, assess the importance of different feature categories in predicting the extent of wildfire burned area by conducting an ablation study.

- Conducting a comprehensive comparative analysis between the proposed GNN framework and well-established semantic segmentation architectures commonly used in image processing and computer vision. This comparison highlights the strengths and limitations of graph-based versus segmentation-based approaches for wildfire spread prediction, providing deeper insights into their respective applicability.

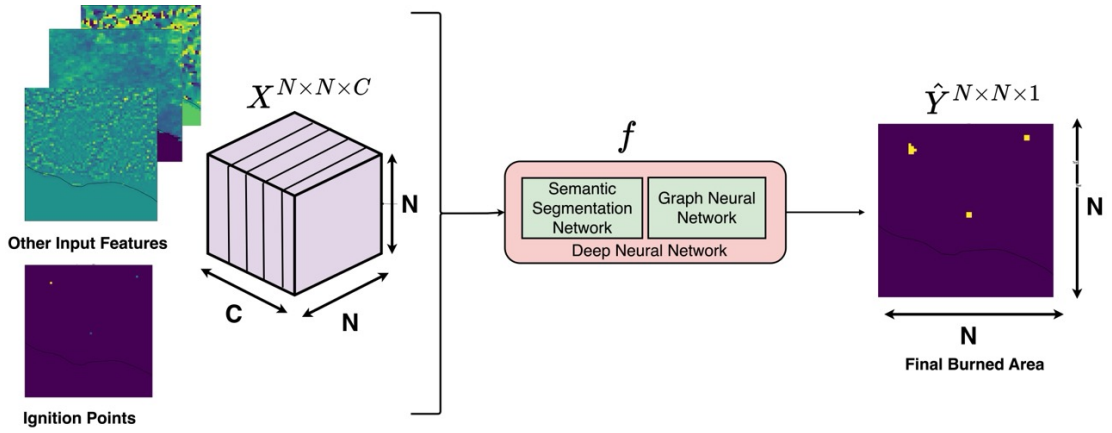


Figure 1.1: Visual Representation of Problem Formulation

1.4 Outline of Thesis

The remaining chapters of the thesis are structured as follows:

Chapter 2: This chapter provides a comprehensive review of the existing literature in traditional and data-driven wildfire behavior modeling, semantic segmentation, and graph neural networks. It highlights key developments and applications relevant to the proposed research.

Chapter 3: A detailed examination of the dataset is presented in this chapter, including its characteristics, statistical insights, and data preprocessing.

Chapter 4: This chapter outlines the proposed prediction methodologies, architectural designs, and the rationale behind the chosen approaches. It also includes details of the implemented models and their architectures.

Chapter 5: Here, the experimental results are presented, accompanied by a discussion of their implications. The performance of the proposed models is evaluated against standard approaches, with insights into their strengths and limitations.

Chapter 6: The final chapter summarizes the key findings, discusses the broader impact of the work, and proposes directions for future research.

Notation	Description
X	Input Matrix of sample
Y	Output Matrix of a sample
(X, Y)	Sample corresponding to (X, Y)
$N \times N$	Spatial dimension of a sample
L	Total number of samples in Training Dataset
$\hat{Y}$	Predicted Output Matrix
x_{ij}	Feature vector of the cell with location coordinates (i, j)
y_{ij}	Output value of the cell with coordinates (i, j)
$\hat{y_{ij}}$	Predicted output of the cell with coordinates (i, j)
f	Generalised prediction function learned by the Deep Learning model
f_g	Function learned by GNN Model
f_c	Function learned by Classifier
f_s	Function learned by Segmentation Model
G	Graph Representation of a sample
V	Set of all nodes in a graph
E	Set of all edges in a graph
v_{ij}	Node corresponding to coordinates (i, j) in the sample
$e_{(v_{ij}, v_{i'j'})}$	Edge between nodes v_{ij} and $v_{i'j'}$
$N(v)$	Neighbourhood of node v
V	Set of all nodes in a graph
A	Adjacency Matrix of Graph
$r_{(v_{ij}, v_{i'j'})}$	Edge weight between nodes v_{ij} and $v_{i'j'}$
X_g	Node Feature Matrix of Graph G
Y_g	Node Output Matrix of Graph G
h_u^k	Hidden embedding of node u at the k^{th} GNN layer
Q	Segmentation mask
$q_{i,j}^t$	Predicted probability for class t at coordinates (i, j)

Table 1.1: List of Notations

Chapter 2

Background & Related Work

This section presents a systematic overview of the literature related to the current landscape of machine learning-driven approaches in wildfire burned area prediction. Section 2.1 systematically outlines the most prominent datasets available for wildfire modeling. Section 2.2 provides an overview of the current landscape of machine learning based applications in wildfire management. Section 2.3 discusses the traditional approaches for wildfire spread and burned area prediction. Section 2.4 sheds light on the semantic segmentation-based approaches employed in wildfire management. Section 2.5 provides a background of different graph neural network models, and discusses the diverse application areas of graph neural networks and the current state of the literature regarding the utilization of GNN-based approaches in the wildfire domain.

2.1 Wildfire Datasets

Assessing and comparing the suitability of various predictive models to a particular task requires the availability of reliable benchmark datasets. However, the number of such datasets available for wildfire-related tasks is limited, particularly when it comes to comprehensive real-world datasets that consider the diverse factors governing the ignition and spread of wildfires. While a substantial number of image datasets exist for the detection and segmentation of wildfires [81],[74],[13], datasets that integrate historical wildfire data along with the environmental, meteorological, and anthropogenic factors governing fire dynamics are either scarce or not publicly accessible. Datasets that contribute to the modeling of wildfire dynamics either focus on the daily temporal progression of wildfires [36], [25] or focus on the cumulative burned area resulting from an ignition over the entire

duration or multiple days of wildfire [63], [49]. The following paragraph provides a description of the wildfire datasets that were found to be suitable for modeling wildfire spread and predicting the resulting burned area.

- **Wildfire DB:** WildfireDB is a spatio-temporal dataset, in tabular form, developed with the objective of identifying the relationship between wildfire spread and other influencing factors, including weather conditions, vegetation, and topographical data. The dataset is specifically focused on the continental United States region consisting of five years of data ranging from 2012-2017. The dataset provides information on the Fire Radiative Power (FRP) at a particular location along with the other fire drivers. Additionally, identical information is provided for a neighboring location corresponding to each cell, facilitating further research on finding the correlation between fire spread and other influencing factors. Although the authors validate the results using standard machine learning architectures, the performance is reported to be suboptimal due to the high sparsity present in the data and subsequent complexity in modeling the spread using simple machine learning architectures such as Logistic Regression, Random Forest and Artificial Neural Network [75].

- **Next Day Wildfire Spread:** Another dataset suitable for the wildfire forecasting task is the Next Day Wildfire Spread [36]. Unlike [75], which provides information of only the neighboring cells of a burned location, this dataset combines two-dimensional fire data with multiple fire drivers, such as topographical, vegetation, meteorological and, population data, aggregated over 2D regions. The dataset consists of 18545 fire events across the United States from 2012-2020 with a spatial resolution of 1 km. The paper demonstrates the usability of the dataset by utilizing it for the prediction of the fire progression on the next day given other influencing features and the current day's fire mask. The machine learning models include Convolutional Auto-Encoder (2-D data), Logistic Regression and, Random Forest(1-D data).

- **WildfireSpreadTS:** With a temporal extent from 2018 to 2021, the WildfireSpreadTS dataset [25] covers 607 wildfire events also spread across the contiguous United States with a spatial resolution of 385m. Drawing upon the work in [36], this spatiotemporal dataset is created with the objective of facilitating both mono-temporal forecasts and multi-temporal forecasts unlike [36] which provides historical fire data for only the previous day. The WildfireSpreadTS dataset consists of multi-day time series data corresponding to each fire event and predicts the spread of a wildfire at a temporal resolution of 24 hours based on a set of 22 influencing features of the active fire mask on the current day.

- **SeasFire Cube:** While most wildfire datasets focus on specific regions or countries, the Seasfire Cube dataset [63] is a highly comprehensive global spatiotemporal dataset consisting of 59 features, including oceanic indices and carbon emissions from fires, across a timeline of 21 years from 2001 to 2021. It has a spatial resolution and temporal resolution of 0.25 degrees and 8 days respectively. The coarser temporal resolution of 8 days offered by this dataset is specifically tailored for the prediction and analysis of wildfires on a seasonal or sub-seasonal basis. The authors further demonstrate the utility of the dataset by forecasting future burned area patterns using historical data from 8 days or multiples of 8 days prior to the target date.

2.2 Machine Learning in Wildfire Management

With the increased availability of wildfire data, whether it be from remote sensing data sources or multimodal public datasets, coupled with the exponential rise in the available computational resources, the utilisation of Machine Learning in wildfire management has also increased. Based on the stage of the wildfire, machine learning based applications can be broadly categorised as Pre-Fire, Active-Fire and Post-Fire applications. The pre-fire stage, which refers to the stage prior to a wildfire ignition, are primarily aimed at preventing wildfires in advance by identifying wildfire prone areas and analyzing the factors that contribute to wildfire ignitions. In [14], the authors develop a segmentation based framework to detect the number of dead trees in aerial images while [1] utilises both high resolution images coupled with biophysical and spectral data to develop surface fuel maps. In [39], the authors developed susceptibility maps for a specific Mediterranean region using various ML algorithms such as Random Forest(RF), and, Gradient Boosting Models such as XGBoost, AdaBoost and Gradient Boost. In the active fire stage, the primary aim is to facilitate effective crisis management by precise identification of active wildfire ignitions, prediction of the behaviour of active wildfires, and optimize fire suppression efforts. In [27] and [58], the authors adopt a segmentation-based approach using Vision Transformers and Graph Neural Networks respectively , to precisely detect and delineate the extent of a wildfire using an image dataset. In [98], the authors adopted a deep reinforcement learning based approach for path planning in wildfire emergencies. In [36] and [37], classical machine learning approaches such as Logistic Regression as well as Deep learning approaches are used to predict the daily spread of an active wildfire. Beyond the active fire stage, the applications are primarily aimed at evaluating the ecological, environmental and anthropogenic impacts post the wildfire. In [100], the authors adopted a machine learning approach to assess the contribution of wildfires to the $PM_{2.5}$ pollution

in a region and its effect on population health. While [19] uses a Gradient Boosting method to assess the recovery of forest covers after a wildfire.

2.3 Traditional Models for Wildfire Spread & Burned Area Prediction

Wildfire spread prediction has been researched extensively in the past and several models have been implemented over the last decades to study wildfire spread behaviour. Although, wildfire spread and burned area prediction are often used interchangeably in most literature, there exists a distinction between the two categories. The wildfire spread prediction task can assume either of two forms - it is either associated with the prediction of characteristics related to the spread behavior of the fire, such as the rate of spread or the intensity of the firefront, or it is related to the prediction of the spatial extent of the wildfire and subsequently the burned area resulting from it [62]. The burned area prediction can thus be categorized as a subfield of the wildfire spread prediction task.

The traditional models developed to predict the spread of wildfires, across all subfields, can be broadly categorised as Physical, Semi-Physical and Empirical Models. Physical models include those models which employ equations related to the physical and chemical process of combustion to predict wildfire spread. However, the strong reliance of these models on input parameters, such as flame gas viscosity, immeasurable outside the field renders them impractical in real-world scenarios. Semi-Physical models incorporate a combination of physical and empirical processes which makes them practically more feasible in comparison to physical models. The Rothermel model is a widely used semi-physical model used to predict the rate of spread in surface fires based on fuel, topography and, weather features. However, such models are incapable of modeling complex environments with mixed fuel and thus are not easily generalizable. On the other hand, Empirical models do not incorporate any physical processes in spread prediction and are solely based on statistical properties of experimental wildfires or attempts to predict the spatial extent of wildfires through computer simulations [62].

Simulation softwares are based on either Huygen's principle or the Cellular Automata model [78]. In Huygen-based models, each fire point is considered as an independent ignition point influenced by multiple external factors determining the spread boundary of the fire typically of an elliptical shape. Prometheus [82] is a deterministic wildfire growth simulation developed in Canada that uses Huygen principle to treat the fire perimeter as a collection of independent elliptical fires to model the behaviour of wildfire spread. Another

popular simulation model is FARSITE [23] which is a two-dimensional vector simulation that can simulate the spread of various types of fires such as surface fires, crown fires, and spotting fires, under different environmental conditions. In addition to Huygens' principle, another class of simulators is based on the Cellular Automata model [2]. This model is used to simulate spatial and temporal characteristics and assumes that fire propagates on a grid-based relationship.

2.4 Semantic Segmentation in Wildfire Applications

Semantic Segmentation is a widely researched computer vision task that refers to the classification of individual pixels in an image into distinct classes [56], [90]. Semantic segmentation is typically formulated as a supervised learning problem, where the target to be predicted is a mask containing pixel-level annotations indicating each pixel's class label. Unlike image classification, where images can be treated independently, semantic segmentation must account for the relationships between neighboring pixels often necessitating additional constraints [16].

With regard to predictive tasks in wildfire datasets, semantic segmentation is often used to segregate regions belonging to the fire class from the non-fire class as a binary mask. In [36], semantic segmentation is used to predict the progression of an active wildfire on the next day, given the input features and the fire mask from the previous day using a combination of classical Machine Learning architectures such as Logistic Regression and Random Forest, as well as a Convolutional Autoencoder neural network architecture. In [37], the wildfire segmentation task is approached in three ways - first by predicting the target day's fire mask using the two-dimensional input features from the previous day using a Convolutional Autoencoder network, in the second task, the target wildfire mask is aggregated across a week to capture the total burned area employing a UNet architecture, and lastly a spatiotemporal sequence of features from the seven preceding days is used for predicting the aggregated burned area employing a combination of the aforementioned models and a Convolutional LSTM architecture. In [63], [64], [65], the objective is to predict wildfire burned area at a larger temporal resolution to capture the weekly and sub-seasonal patterns. In [63], the authors introduce a global spatiotemporal wildfire dataset and validate its usability by predicting the future burned area in 8-day intervals using the UNet++ architecture. In [65], the wildfire segmentation task is also formulated as forecasting the burned area in eight-day intervals. However, it fuses both local and global features with data on oceanic indices to make predictions using a Vision Transformer-based architecture. [73] formulates spread forecasting as a semantic segmentation task and

evaluates the fire spread rate and area as a post-processing step along with assessing the predictive performance of the model in and around water bodies. It also uses historical data for dynamic features along with other static features as channels to forecast the burned area. [46] uses a novel modified UNet architecture, FU-NetcastV2, to forecast the perimeter of the burned area over a 24-h period using consecutive perimeters of wildfires extracted from the GeoMAC dataset, meteorological data, and satellite-derived vegetation data. In [49], the authors employ a UNet architecture to predict the final burned area resulting from an ignition point by formulating the problem as a semantic segmentation task. In [15], the authors adopt a decision tree based approach for predicting the final burned area size at the time of ignition. However, most semantic segmentation architectures ignore the long-range dependencies that are inherent in geographical data, such as wildfires, which can limit their ability to effectively capture spatial patterns and interactions critical for accurate predictions

2.5 Graph Neural Networks

The following subsections review the current state of literature on Graph Neural Networks (GNNs) and their applications in wildfire research. This section is organized as follows: Section 2.5.1 provides a background on the Graph Neural Network framework, their classifications, and underlying mechanisms, Section 2.5.2 explores the different categories of tasks in GNN research and their applications, while Section 2.5.3 focuses on the application of GNNs in wildfire management.

2.5.1 Background

Graph Neural Networks (GNNs) are a subcategory of Deep Neural Networks(DNN) which are designed to handle graph structured data. In GNNs, given a graph $G(V, E)$ where V and E represent the set of nodes and edges in the graph G respectively, the primary objective is to learn a representation of each node such that it incorporates both structural and feature information of the entire graph . A GNN framework is a combination of three primary modules namely, the propagation module, the sampling module, and the pooling module [97]. Among the three modules, the propagation module is the basic foundation of a GNN network, whereas the inclusion of the sampling and pooling modules depend on the task requirement and the data. The sampling module is used in conjunction with the propagation module to reduce the neighborhood size of nodes in extremely large graphs. It is utilized to mitigate issues related to the exponential rise of neighbors with the increasing

number of GNN layers. The Pooling module is used to reduce the dimensionality of large graphs and to attain a high level representation of the entire graph for downstream tasks such as Graph Classification. The propagation module is responsible for the transmission of information between nodes. The mechanism for the aggregation of information from nodes is determined by the propagation operator which can either be a convolutional operation or a recurrent operation. The convolution operation is the most widely used propagation mechanism in most GNN model and can be broadly classified as Spectral and Spatial convolutions.

- **Spectral Convolutions:** The spectral methods are based on graph signal processing and are designed to process the spectral representation of a graph. The graph is first transformed into the spectral domain by a graph fourier transform operation denoted by $\mathcal{F}$ and subsequent convolution operations are performed on this transformed representation. The result obtained from the convolved operations is retransformed into the original domain by means of an inverse graph fourier transform operation denoted by $\mathcal{F}^{-1}$. The mathematical formulation of the spectral convolution of a graph signal X is shown in Equation (2.1) with $U^T g$ as the convolutional filter in the spectral domain and U as the matrix of eigenvectors of the normalized graph Laplacian. If the convolutional filter is assumed to be a diagonal matrix g_w, which is learnable in nature, Equation (2.1) reduces to Equation (2.2) [97]. Depending on the nature of g_w, many spectral GNNs such as the Chebyshev Networks [17] and its variants such as GCN [48] have been developed. It should be noted that GCN can be represented as both a spectral and spatial convolution GNN. For simplicity, this study considers the GCN in its spatial domain [43].

$$\begin{aligned} g * x &= \mathcal{F}^{-1}(\mathcal{F}(g) \odot \mathcal{F}(x)) \\ &= U(U^T g \odot U^T x) \end{aligned} \tag{2.1}$$

$$g_w * x = (U g_w U^T x) \tag{2.2}$$

- **Spatial Convolutions:** The Spatial Convolution mechanism was developed with the aim of addressing the limitation posed by Spectral Convolutions. Spectral Convolutions require the recomputation of the graph Laplacian matrix if the graph topology changes, causing it to be computationally expensive in scenarios with dynamic graph topologies [43]. To overcome this challenge, Spatial convolution was proposed by GCN (Graph Convolutional Network) [48]. The equivalent equation for spatial

graph convolution is shown in Equation (2.3) where w refers to the learnable weight matrix. The degree matrix D is a diagonal matrix with each entry representing the sum of the weights of edges connected to a node, I_N represents an identity matrix, while the adjacency matrix A represents the connections between nodes in the graph.

$$g_w * x = w(I_N + D^{-1/2}AD^{-1/2})x \tag{2.3}$$

In addition to the GCN, other widely used spatial convolution based models include the Graph Attention Network (GAT) [83] and GraphSAGE [31] models. The GAT incorporates an attention mechanism in the propagation stage by specifying different weights to different nodes while aggregrating information from a node's neighbourhood. On the other hand, the GraphSAGE network introduces a neighborhood sampling mechanism to aggregate information from a fixed-size subset of neighbors rather than the entire graph, making it scalable to very large graphs. The propagation mechanisms of the GCN, GAT, and, GraphSAGE are discussed in further detail in Section 4.1.3.

2.5.2 Categories of Applications for Graph Neural Networks

A graph can be described as a mathematical structure that models the relationships between a set of entities or nodes. In recent years, research on applying machine learning to graph-structured data has seen a sharp rise due to the ubiquitous nature of graph data capable of modeling complex systems across diverse domains such as biological sciences, natural sciences, and computer sciences. Depending on the specific objective of prediction, the applications of machine learning on graph data can be broadly categorized into three categories: node-level, graph-level, and edge-level tasks.

- **Node-Level Tasks:** Node-level tasks focus on predicting attributes associated with the individual nodes in a graph. The task can assume the form of either node classification, regression, and clustering [96]. Node Classification involves predicting a discrete attribute associated with a node, whereas Node Regression is concerned with the prediction of continuous values. Clustering on the other hand is associated with recognizing groups of similar nodes in a graph. In natural sciences, forecasting of phenomena such as weather and carbon emissions using spatiotemporal graphs has been implemented as demonstrated in [51], [80]. In these approaches, geographic locations are represented as nodes within a graph, and neural networks are employed to predict

node-level statistics, capturing the temporal and spatial dependencies inherent in the data.

- **Edge-Level Tasks:** Edge-level tasks include edge classification and edge prediction. While edge classification is concerned with determining the type of edges existing between pairs of nodes, edge prediction aims to determine whether an edge exists between a given pair of nodes. In biochemistry, the prediction of the nature of interactions between different proteins by modeling it as an edge prediction and edge classification task has found great success such as in [86], [42] respectively.

- **Graph-Level Tasks:** Graph level tasks are concerned with predicting attributes associated with an entire graph, rather than focusing on individual nodes and edges. Based on the nature of the attribute to be predicted, graph level tasks can either be classification or regression similar to the node-level tasks. On the other hand, Graph matching involves establishing structural correspondences between graph-structured objects, either by finding optimal node-to-node mappings or by computing similarity metrics between graphs [54]. Graph-level predictions have been widely used in molecular property prediction such as protein-ligand binding affinity [21].

2.5.3 Graph Neural Networks in Wildfire Management

Although substantial literature focusing on the efficacy of Graph Representation Learning in natural sciences exists, research on the applicability of Graph Neural Networks (GNN) in wildfire-related tasks is largely limited. [58] uses the Graph Convolutional operator from the GraphSAGE model by [31] for wildfire segmentation in RGB images and videos captured by Unmanned Aerial Vehicle (UAV). The frames first undergo unsupervised segmentation using the SLIC algorithm which results in the creation of blocks of pixels or superpixels. The centroids of the resulting superpixels are treated as nodes and edges are established based on the Euclidean distance between nodes. While segmentation of wildfires in images is useful for the localization of active wildfires, it does not effectively capture the influence of other factors that affect short-term or long-term wildfire behavior. In [95], the authors employ a GNN architecture for temporal forecasting of wildfire danger in the Mediterranean region from a combination of three meteorological variables and three oceanic indices. To capture the intricate relationships between the various fire drivers, a causal graph is constructed between the features using the PCMCI algorithm. This causal graph is subsequently processed by a GNN to predict a binary output indicating whether the area will be burned or not. [68] extends beyond the temporal domain and proposes a spatiotemporal approach to predict the daily wildfire spread in the European region

based on historical data of daily burned areas and other wildfire drivers. The authors construct a dataset consisting of daily time series data curated from multiple historical data records and employ the Temporal Graph Convolutional Network (TGCN) architecture, which integrates multiple Graph Convolutional layers with a Gated Recurrent Unit (GRU), to predict the daily temporal progression of wildfire based on the fire drivers and burned area from the previous day. Similar to [95], the problem is formulated as a binary node classification task for a graph with nodes representing individual hexagonal grid cells and edges being defined between six neighboring grid cells. Similarly, [9] utilizes simulation data spanning 30 years from the JULES-INFERNO Dynamic Global Vegetation Models (DGVM) model to predict the monthly burned area using a combination of GCN and LSTM model architectures. The locations form the nodes of the input graph and the node features are derived from a combination of four meteorological features at the particular node on a specific day. The edges between nodes are defined by taking the correlation coefficient of the burned area across a sequence of time steps and dropping all edges with a correlation coefficient below the 10% quantile. In addition, a land mask is used to filter out pixels that do not correspond to the land surface. Although simulation data provides flexibility in modeling different scenarios and a complete representation of data across all regions unlike real-world data, it does not fully capture the intricate relationships between different influencing factors in wildfire behavior.

Chapter 3

Exploratory Data Analysis

This chapter provides an overview of the dataset, its analysis, and the preprocessing steps undertaken for modeling. The chapter is structured as follows: Section 3.1 discusses the geographical characteristics associated with the study area corresponding to the dataset. Section 3.2 provides a detailed description of the temporal and geographical extent, original data sources, resolution, and features of the dataset. Section 3.3 analyzes the imbalance in the class distribution inherent in the original dataset. Section 3.4 outlines the data transformation and preprocessing steps employed for data preparation.

3.1 Study Area

The vegetation in a specific region is primarily determined by the climatic factors that prevail in that area, including temperature and precipitation patterns. Based on the similarity in the predominant vegetation and climate, regions worldwide are classified into distinct categories known as Terrestrial Biomes [8]. The region of interest of the Mesogeos dataset, characterized by the Mediterranean biome, includes the wider Mediterranean region which is the geographical area encompassing the Mediterranean Sea and the lands surrounding it, spanning parts of Europe, Asia, and Africa. Mediterranean biomes are characterized by a strongly seasonal climate. The winters are characterized by cool temperatures and significant rainfall facilitating the growth of vegetation and allowing for the accumulation of substantial biomass. On the other hand, the summers that follow are warm and dry leading to the increased flammability of the vegetation and thus increasing the amount of potential wildfire fuel [4], [57]. Although the characteristic fire-prone climate makes wildfires a seasonal and, natural occurrence that is often vital for nutrient cycling and

the growth of numerous plant species [8], the surge in populations in the Mediterranean regions and the consequent increase in human-induced ignitions have led to an increased frequency of extreme fire events in these regions in recent years [57].

3.2 Dataset Description

Mesogeos [49], published in 2023, is a large-scale wildfire dataset developed with the aim of addressing the limited availability of open-source datasets suitable for wildfire modeling using deep learning methods. The complete dataset is available in the form of a spatiotemporal datacube consisting of the time, latitude and, longitude as the dimensions. With a resolution of 1 km $\times$ 1 km $\times$ 1 day, it covers a temporal period of 17 years from 2006 to 2022 and an area of 4714 km $\times$ 1753 km in the wider Mediterranean region. The distribution of samples across the years is shown in Figure 3.1, which reveals an upward trend in the number of wildfires, with peaks in 2017 followed by 2022, further corroborating the increase in wildfire activity in recent years. The dataset consists of 27 input features that are known to affect wildfire ignition and spread. The input features can be broadly categorized as belonging to meteorological, vegetation, topographical, and anthropogenic features. Additionally, two wildfire features, namely the ignition point indicating the location of the initial ignition corresponding to a fire event and the extent of the corresponding final burned area resulting from the ignition, are included in the dataset bringing the total number of features to 29. Table 3.1 provides a detailed list of all the features organized into the aforementioned categories.

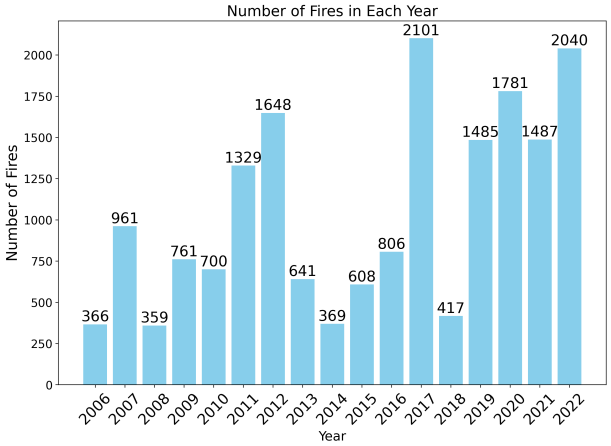


Figure 3.1: Distribution of Wildfires by Year

In addition to the spatiotemporal datacube, Mesogeos provides two additional subsidiary datasets extracted from it tailored for two specific machine learning tasks - wildfire danger forecasting and final burned area prediction. The dataset corresponding to the second task, i.e. final burned area prediction, is utilized for the proposed approach. Each sample in the Burned Area Dataset represents a historical fire event in the wider Mediterranean region. The samples are constructed from the original datacube by considering a $64\text{km} \times 64\text{km}$ area centered around the initial ignition point corresponding to the historical fire event and extracting all the features for the outlined geographical region. The samples are stored in an easy to access NetCDF format. In accordance to the description of the input, output and, problem formulation outlined in Section 1.2, the dimensions of the input X then becomes $64 \times 64 \times C$, where $C \in [1, 28]$ depending on the number of features used for training the model. The target variable Y to be predicted is of the dimension $64 \times 64 \times 1$. Each unit cell in the two-dimensional grid Y represents a 1 km $\times$ 1 km geographical region and assumes a binary value depending on whether or not it is burned due to the given ignition. Figure 3.4 gives a visual representation of a single sample in terms of its features. The "Cell ID" column in Table 3.1 represents the position of each feature in Figure 3.4. The dataset integrates data from multiple remote sensing sources - ERA5-Land Database for meteorological features, vegetation indices from MODIS, soil moisture from European Drought Observatory (EDO), Anthropogenic features from Worldpop, topographic data from the Copernicus DEM - Global Digital Elevation Model (COP-DEM). Land cover types are sourced from the Copernicus Climate Change Service, burned areas from EFFIS, and ignition cells and dates from the MODIS AF product.

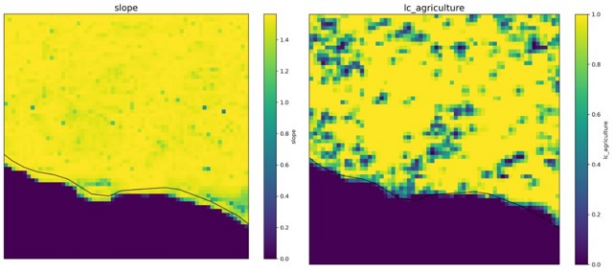


Figure 3.2: A visual representation of the (a) slope and (b) the agricultural landcover fraction feature channels for a data sample. The colourbar indicates a continuous range of values

Wildfire spread dynamics and the consequent burned area resulting from it are influenced by a complex interaction between different features. The influencing features can be broadly classified into four categories as described below.

1. **Anthropogenic Features**: With the increase in urbanization near forested regions in recent decades, the influence of human or anthropogenic factors on wildfire occurrences has led to a significant increase. Factors such as population density have a direct impact on wildfire occurrences as densely populated areas near forests are more susceptible to human-induced ignitions, which can subsequently lead to wildfires. Human-made structures also have a significant impact on the spread and occurrence of wildfires. On the one hand, the existence of human-made structures such as roads facilitates the slowing down of wildfire spread due to the non-combustible characteristics usually present in such structures [78]. On the other hand, the proximity to transportation networks or human-made structures can elevate the risk of human-induced ignitions due to easier access to forest areas for humans [28].

2. **Meteorological Features**: Meteorological features are a key driving factor of wildfire occurrence and spread. In a broader sense, they can be categorised into hydrometeorological, radiation, temperature and wind features. Hydrometeorological features such as humidity, soil moisture, and total precipitation are generally inversely correlated with wildfire conditions. As the magnitude of either of these increases, fuel moisture also increases and the subsequent risk of wildfire is lessened. Conversely, higher surface pressures facilitate stronger winds, which can further accelerate the spread of wildfire and aggravate the severity. The intensity of surface solar radiation (SSR) received directly influences the moisture content of vegetation. Consequently, high levels of SSR can lead to arid and fire-prone environments. Temperature is another primary meteorological feature that contributes to decreased humidity, thereby promoting dry conditions susceptible to wildfire occurrence. Wind features, such as wind speed and direction,influencethe real-time rate of spread of wildfires and their heat transfer. These factors can exacerbate the fire by causing the unburned fuel or vegetation to burn more rapidly and facilitate the transport of embers from the active fire to areas near the wildfire boundaries, resulting in new ignitions [78].

3. **Topographical Features**: Topographical factors represent the characteristics of a terrain and are static in nature. Elevation affects humidity levels, subsequently influencing the moisture content of vegetation. Typically, regions at high elevations exhibit low temperatures and humidity resulting in high moisture in the vegetation and reduced susceptibility to wildfires. The slope is another factor which influences wildfire occurrence, either through steep slopes resulting in accelerated drying after rainfall or by affecting the fire spread rate. Uphill and steep slopes accelerate spread rates as opposed to downhill and gentle slopes. Aspect determines the extent of sun exposure , resulting in sun-facing slopes, such as southern or western aspects in the

northern hemisphere, receiving substantial solar radiation and being more susceptible to fires compared to shaded slopes, such as north or eastern aspects in the southern hemisphere [78].

4. **Vegetation Features**: The type of vegetation cover, also known as Land cover, of a region has a significant impact on wildfire occurrence as it is inherently linked to the type of fuel available for wildfire ignitions. Based on the predominant vegetation type, land covers can be broadly classified as shown in Table 3.1. Shrublands and grasslands are generally more prone to wildfires owing to their intrinsic dryness and high flammability [61], [91]. Forests, while less fire-prone compared to shrublands and grasslands, are still more flammable than agricultural lands [61]. Naturally, wetlands and settlement areas are less susceptible to wildfires due to the absence of sufficient fuel. Furthermore, vegetation indices, such as NDVI and LAI, function as indicators of the type of vegetation present and the vegetation density, thereby determining the nature of potential fuel available in a region.

3.3 Imbalance Analysis

Imbalanced data can be described as any dataset in which there exists a significant disparity in the number of samples belonging to each class. In a binary classification task, one class, referred to as the majority class, significantly outnumbers the other class, known as the minority class [32]. In the context of semantic segmentation, imbalance generally assumes the form of a foreground-background imbalance problem where the foreground (region of interest) occupies a much smaller percentage of a two-dimensional sample as compared to the background [70]. Due to the skewed class distribution, deep learning algorithms prioritize loss minimization for the majority class resulting in a subpar performance on the minority class. Hence, to achieve optimal predictive performance on such imbalanced datasets, imbalance mitigation measures must be incorporated during training.

Imbalance Mitigation measures for Computer Vision, which often include two-dimensional data, can be broadly categorized into data-level, algorithmic-level, and hybrid approaches. Among data-level approaches, data augmentation, which involves oversampling the data through various image transformations such as cropping, rotating, and flipping, is widely used. Among algorithmic approaches, cost-sensitive learning finds extensive use, where a higher weight is assigned to minority classes in the loss function of the training process [70].

Category	Sub-category	Feature	Cell ID
Anthropogenic Features	-	Population	(1,1)
		Road Distance	(1,2)
Meteorological Features	Hydrometeorological Features	Relative Humidity (RH)	(1,3)
		Soil Moisture Index (SMI)	(1,4)
		Surface Pressure (SP)	(1,5)
		Total Precipitation (TP)	(1,6)
	Surface Solar Radiation	Surface Solar Radiation Downwards (SSRD)	(1,7)
	Temperature Features	Dew Point Temperature (D2M)	(2,1)
		Land Surface Temperature (LST Day)	(2,2)
		Land Surface Temperature (LST Night)	(2,3)
		Temperature at 2 m (T2M)	(2,4)
	Wind Features	Wind Direction	(2,5)
		Wind Speed	(2,6)
Topographical Features	-	Aspect	(2,7)
		Curvature	(3,1)
		Elevation (DEM)	(3,2)
		Slope	(3,3)
Vegetation Features	Land Cover Types	Agriculture	(3,4)
		Forest	(3,5)
		Grassland	(3,6)
		Settlement	(3,7)
		Shrubland	(4,1)
		Sparse Vegetation	(4,2)
		Waterbodies	(4,3)
		Wetland	(4,4)
	Vegetation Indices	Leaf Area Index (LAI)	(4,5)
		Normalized Difference Vegetation Index (NDVI)	(4,6
Wildfire Features	-	Ignition Points	(4,7)
		Burned Areas	Target

Table 3.1: List of all features in the Mesogeos Dataset [49]

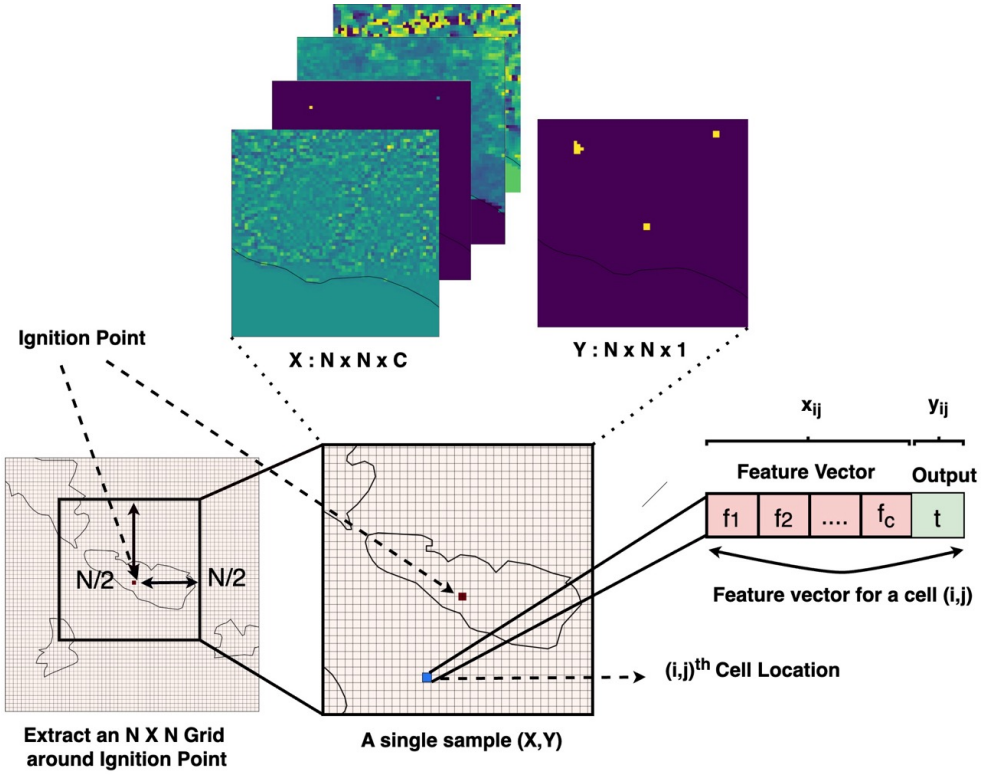


Figure 3.3: Visual representation of a sample (X, Y). Here, f and t represent the input feature variables and the target variable at the (i, j)-th location, respectively.

Additionally, the evaluation metrics used to assess model performance should take into account the imbalance in class labels by providing more importance to achieving accurate prediction results on the minority class[59], [32], [70].

The datasets used in wildfire related predictive tasks are highly imbalanced due to the rare occurrences of fire events or due to limitations in resolution imposed by data acquisition measures such as remote-sensing data sources. In the context of the problem, the imbalance assumes the form of a foreground-background imbalance originating from the coarse resolution of the source. The degree of imbalance is quantified by the Imbalance Ratio (IR), which represents the ratio of the majority class to the minority class [32]. In our dataset, the IR for a single sample S in the dataset is calculated as the ratio of the number of unburned cells to the number of burned cells in the output grid Y of each $N \times N$ sample as shown in Equation (3.1). The average imbalance ratio IR_{avg} across the entire dataset consisting of L samples is given by Equation (3.2). Additionally, the burn ratio

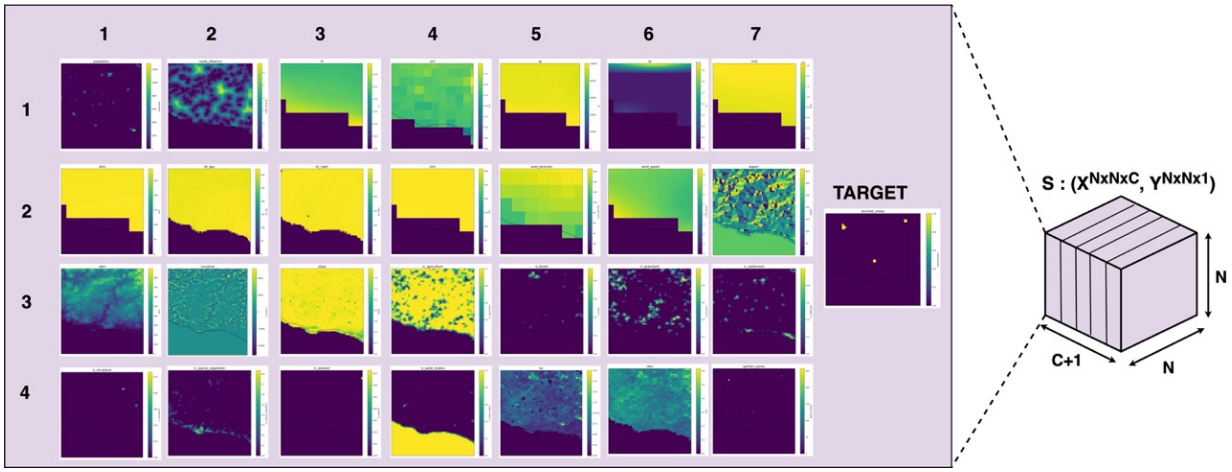


Figure 3.4: Visual representation of a single sample S illustrating all features listed in Table 3.1.

(BR) of a single sample is calculated as the ratio of burned cells to the total number of cells in a grid as shown in Equation (3.3) , and the corresponding average across the dataset is computed as shown in Equation 3.4

$$\mathrm{IR}^{(S)} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} \mathbb{1}(y_{ij} = 0)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \mathbb{1}(y_{ij} = 1)} \tag{3.1}$$

$$\mathrm{IR}_{(avg)} = \frac{1}{L}\sum_{S=1}^{L} \mathrm{IR}^{(S)} \tag{3.2}$$

$$\mathrm{BR}^{(S)} = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N} \mathbb{1}(y_{ij} = 1)}{\sum_{i=1}^{N}\sum_{j=1}^{N} \mathbb{1}(y_{ij} = 0) + \sum_{i=1}^{N}\sum_{j=1}^{N} \mathbb{1}(y_{ij} = 1)} \tag{3.3}$$

$$\mathrm{BR}_{(avg)} = \frac{1}{L}\sum_{S=1}^{L} \mathrm{BR}^{(S)} \tag{3.4}$$

The average imbalance ratio observed across the dataset is ≈ 571 reflecting the disparity between the majority and minority classes.The imbalance ratio varies widely, ranging from a minimum of ≈ 2.5 to a maximum of ≈ 4095, indicating the skewed distribution of the target variable. Notably, 88.6% of samples were observed to have an imbalance ratio

exceeding 100, while only 0.2% of the samples had an imbalance ratio ≤ 10. Additionally, the average burn percentage across the dataset is 0.496%, with a minimum and maximum burn percentages of 0.02% and 28.59% respectively, further underscoring the limited region of interest present in the majority of samples. The skewed distribution of imbalance ratios is visually represented in Figure 3.6 plotted on a logarithmic scale for visual clarity.

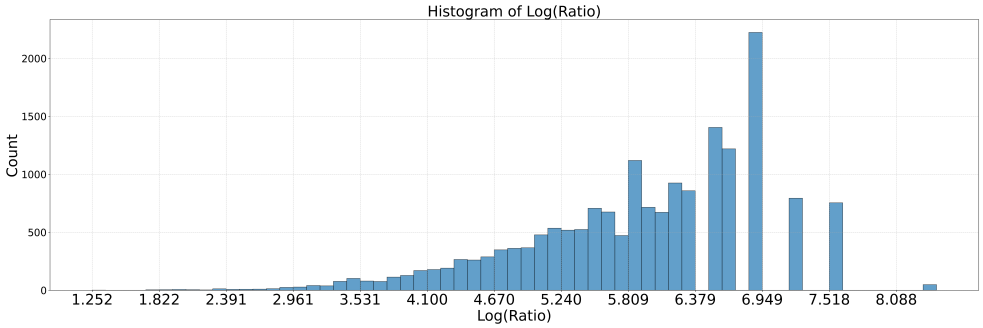


Figure 3.5: Histogram of Imbalance Ratios

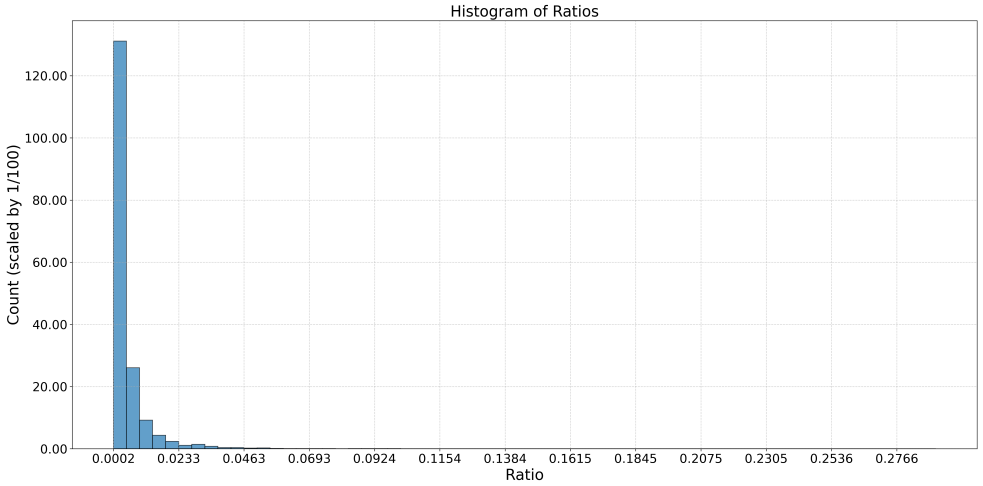


Figure 3.6: Histogram of Burn Ratios

3.4 Data Preprocessing

Several preprocessing steps are undertaken to prepare the data for subsequent modeling tasks. The dataset consists of multiple feature with varying ranges. In order to bring all

input features to a consistent scale, Min-Max Normalization is performed to scale all input features to a range of $[0, 1]$, with the exception of features that have fractional values, such as land cover data, and binary features, such as ignition points. The normalized output $x_{\hat{c}}$ of an input feature x_c for a data sample s, is computed using the normalization equation as shown in equation 3.5. Further, all null values are filled with 0 in all features to maintain the completeness and consistency of data.

$$x_{\hat{c}} = \frac{x_c - \min(x_c)}{\max(x_c) - \min(x_c)} \tag{3.5}$$

In the majority of samples, the ignition point and the resulting wildfire pattern is centered within the two-dimensional grid. To improve the generalizability of the model across non-centered burned area patterns, the samples are randomly cropped to 32×32 grids centered around the ignition point. The randomness in the cropping process refers to slight variations in the exact placement of the cropped region within the original grid while ensuring that the ignition point remains in the cropped region after cropping. Additionally, all ignition points and burned area values are binarized, with non-zero values set to 1 and zero values remaining as 0.

The relationship between wildfire ignition points and their resulting burned areas is crucial for understanding wildfire behavior. To maintain data consistency and focus on typical wildfire patterns, samples with burned area sizes falling outside a predefined range are excluded. The size of the burned area in hectares corresponding to each sample is calculated as the cumulative sum of the areas resulting from each ignition point in the sample. The distribution of burned area sizes in the dataset is highly skewed, with most samples concentrated within a relatively small range. The average burned area size is 796.84 ha , with values ranging from a minimum of 30 ha to a maximum of $107,602$ ha. Outliers are present above the calculated upper bound of $1,482$ ha, highlighting the presence of rare but extreme events. Most of the burned areas fall within the range of $[30, 648]$ ha, reflecting the concentration of smaller fires in the dataset, while the skewness is driven by a small number of large burned areas above the upperbound.

The boxplot in Figure 3.7 shows the distribution of burned area size in hectares. Given the skewed nature of the burned area sizes, Tukey's outlier removal method [34] is applied to identify extreme values of the size of the burned area and filter outliers. The Interquartile Range (IQR) is defined as the difference in the first and third quartiles of a dataset represented by $Q1$ and $Q3$ respectively and as shown in equation 3.6. $Q1$ of burned area size is the value (in hectares) below which 25% of the data points fall. Whereas, $Q3$ of burned area size is the value (in hectares) above which 25% of the data points fall. The

upper bound (up) and lower bound (low) for the burned area size are calculated using equations 3.7 and 3.8 respectively.

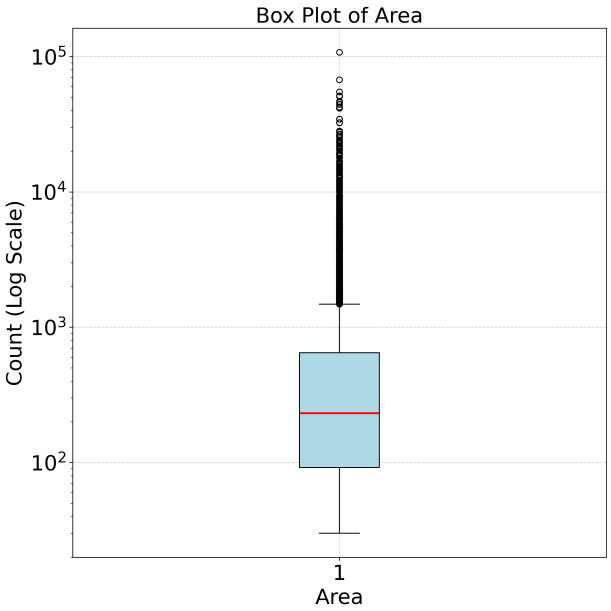


Figure 3.7: Boxplot of Burned Area Sizes

$$\text{IQR} = Q_3 - Q_1 \tag{3.6}$$

$$low = Q1 - 1.5 \times \text{IQR} \tag{3.7}$$

$$up = Q3 + 1.5 \times \text{IQR} \tag{3.8}$$

Data points that deviate from the range $[-742, 1482]$ are subsequently filtered and excluded from subsequent analysis.

Chapter 4

Methodology

This chapter outlines the methodology adopted in developing a predictive model for wildfire burned area prediction. The study utilized a Graph Neural Network based approach to accurately delineate the burned area extent resulting from an ignition point given the initial ignition point and the different categories of features outlined in Table 3.1. To evaluate its effectiveness, the proposed GNN approach is systematically compared against established Semantic Segmentation-based architectures. The chapter is structured as follows: Section 4.1 presents the problem formulation in the context of a graph neural network approach. It describes the methodology employed to transform the dataset into a graph-structured dataset. Additionally, the architecture of the GNN models is outlined. Section 4.2 begins with the formulation of the prediction task in terms of Semantic Segmentation. The subsequent section 4.2.2 describes the different categories of Semantic Segmentation architectures adopted in the proposed approach for comparison.

4.1 Graph Neural Network

Graph Neural Networks possess the ability to capture intricate spatial relationships between both short and long-range locations, which are inherent in wildfire data. This section discusses the graph-based framework for predicting the final burned area pattern resulting from an ignition point. The problem formulation based on Graph Neural Networks is specified in Section 4.1.1. In this approach, the dataset undergoes transformation into a graph-structured dataset, adhering to the methodology outlined in Section 4.1.2. Subsequently, three distinct architectures classified under Graph Neural Networks are employed, as outlined in Section 4.1.3.

4.1.1 Problem Formulation

A Graph Neural Network (GNN) architecture is a deep learning model specifically designed to perform predictions on graph-structured data. To meet this requirement of having graph structured samples, each $N \times N$ sample (X, Y) in the dataset is mapped to form a graph $G = (A, X_g)$, uniquely defined by the adjacency matrix $A \in R^{|V| \times |V|}$, the node feature matrix $X_g \in R^{|V| \times C}$ and, consisting the node output matrix $Y_g \in R^{|V| \times 1}$.

The proposed approach is formulated as a node classification task wherein the goal is to learn a hidden representation of the feature vector corresponding to each node in a given graph and subsequently perform classification on the learned hidden representation using a classifier. Mathematically, the aim is to learn a function f_G which maps the feature vector x_{gv}, corresponding to an individual node v, to a hidden representation h_{gv} as shown in Equation (4.1). Subsequently, the target output class y_{gv} of the node is predicted by performing binary classification on the learned representation using a classifier defined by the function f_C as shown in Equation (4.2) and the predicted output is denoted as $\hat{y}_{gv}$. Figure 4.1 provides a visual description of the problem formulation using GNN.

$$h_{gv} = f_G(G(A, X_g)) \tag{4.1}$$

$$\hat{y}_{gv} = f_C(h_{gv}) \tag{4.2}$$

4.1.2 Graph Dataset Creation

A graph is a structure consisting of a set of nodes, also known as vertices, that are interconnected by edges. The edges between a pair of nodes are established based on a specific criterion reflecting the relationship between nodes. Formally, a graph can be defined as $G = (V, E)$, where V corresponds to the set of nodes and E corresponds to the set of edges. An edge, denoted by $e_{(v_{ij}, v_{i'j'})}$, is said to be an edge of the graph that connects the nodes v_{ij} and $v_{i'j'}$ if $e_{(v_{ij}, v_{i'j'})} \in E$. The neighborhood of a node is the set of nodes that are directly connected to it. Formally, the neighborhood can be defined as $\mathcal{N}(v) = \{u \in V \mid (u, v) \in E\}$. The topology of a graph is mathematically represented as a square matrix known as an Adjacency Matrix. Formally, the adjacency matrix, denoted by A, is a $|V| \times |V|$ matrix such that $A_{(v_{ij}, v_{i'j'})} = r_{(v_{ij}, v_{i'j'})}$ if $e_{(v_{ij}, v_{i'j'})} \in E$ and 0 otherwise. Here, $r_{(v_{ij}, v_{i'j'})}$ is a scalar value corresponding to the weight of the edge.

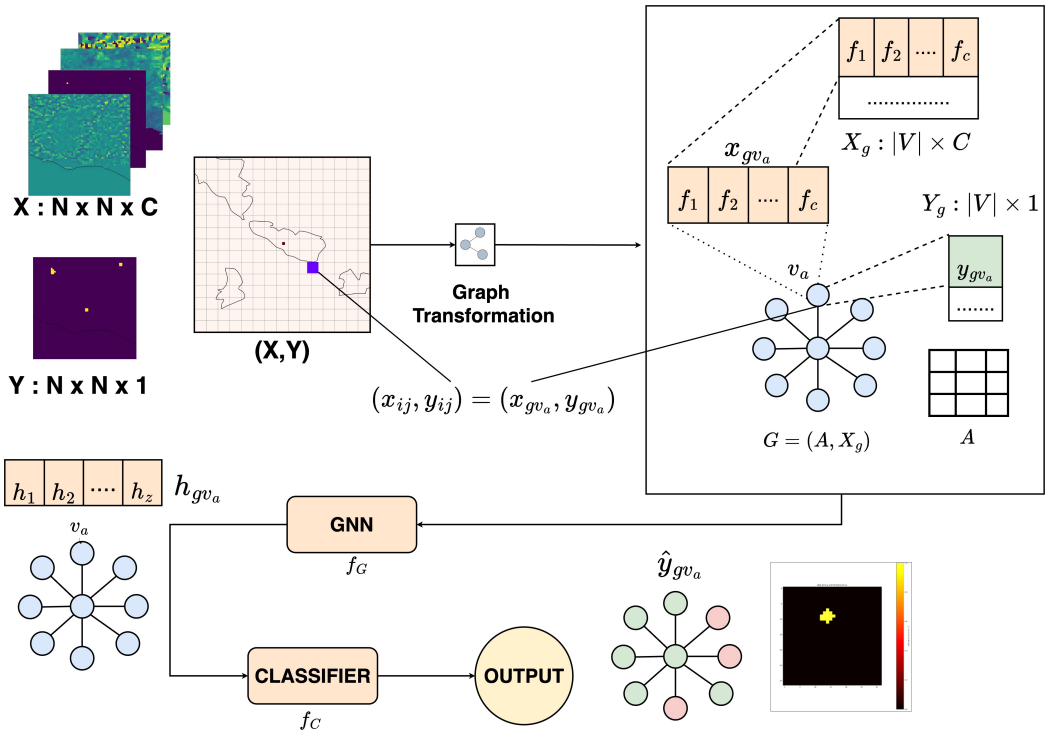


Figure 4.1: Visual Representation of GNN Problem Formulation

A graph can have features associated with each node and it is then known as an attributed graph. In all subsequent sections, the graph $G = (A, X_g)$ refers to an attributed graph. To incorporate node features into the graph representation, we extend the previous notation of a graph to $G = (A, X_g)$ where $X_g \in \mathbb{R}^{|V| \times C}$ is the corresponding node feature matrix and C represents the dimension of the feature vector corresponding to each node. For supervised node classification tasks, an additional matrix known as the node output matrix $Y_g \in \mathbb{R}^{|V| \times 1}$ is associated with each graph which holds the binary output class of each node. Based on this terminology, the proposed approach transforms each sample (X, Y) in the dataset as a graph $G = (A, X_g)$ and its corresponding output matrix Y_g and each cell (x_{ij},y_{ij}) in a sample as a node v_{ij}. The edges between nodes are formulated based on the pairwise connectivity of each node to its eight immediate neighbors. The node feature vector is a C-length vector that comprises the preprocessed features at a particular node location.

Graph structures for GNNs are formulated based on some heuristics such as the similarity in node attributes, inter-node connectivity, the functional dependency between nodes, or their spatial proximity [44]. In the case of geospatial data, spatial proximity and, inter-node connectivity measures have proven to be a suitable heuristic [24],[88]. In wildfire ignitions, the final burned area resulting from an ignition source is localized within a region surrounding it and hence, the neighboring locations which are in direct contact with the site of ignition are most likely to be consumed in the resulting wildfire. Hence, we adopt inter-node connectivity as the criterion for defining the edges between nodes. Each individual cell in a sample, representing a node, is connected to its 8-immediate neighboring nodes ($\mathcal{N}_8$) in the proposed approach. The mathematical equation governing the adjacency matrix value of a pair of nodes $(v_{ij}, v_{i'j'})$ in such a graph is defined in Equation (4.3). Given a pair of nodes with coordinates (i, j) and (i', j'), there exists an edge between the pair of nodes if their respective coordinates in the $N \times N$ sample satisfy the constraints defined in Equations (4.4), (4.5) and, (4.6). Figure 4.2 illustrates the transformation of a smaller subset (X', Y') of a sample (X, Y) to the graph $G'(A', X'_g)$. The algorithm for the graph transformation process is outlined in Algorithm 1.

$$A_{(v_{ij}, v_{i'j'})} = \begin{cases} 1, & \text{if } v_{i'j'} \in \mathcal{N}_8(v_{ij}), \\ 0, & \text{otherwise.} \end{cases} \tag{4.3}$$

$$i' \in \{i-1, i, i+1\} \tag{4.4}$$

$$j' \in \{j-1, j, j+1\} \tag{4.5}$$

$$(i', j') \neq (i, j) \tag{4.6}$$

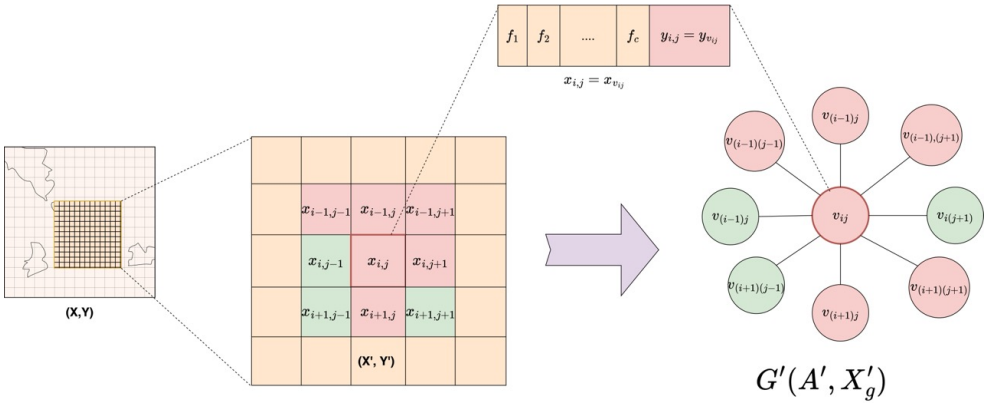


Figure 4.2: Graph Transformation Process for a Grid Subset. Red indicates burnt nodes (target class 1), while green represents unburnt nodes (target class 0)

4.1.3 Model Architecture

The primary objective of a Graph Neural Network (GNN) is to learn node embeddings , which are vector representations of nodes. These embeddings encode both the graph topology and the node's local features, capturing meaningful structural and relational information to enable downstream tasks such as node classification. The core of the GNN is the Message Passing (MP) function defined in equation (4.7). The goal of the message-passing function is to iteratively update node embeddings by aggregating and transforming information from a node's local neighborhood, enabling each node to incorporate relational information from its neighbourhood. The MP function is based on two operations known as Aggregate ($AGR(.)$) and Update ($UPD(.)$). In terms of the aforementioned operations, the equation of the MP function is defined in (4.7).

The Aggregate operation consolidates the information from the embeddings of a given node's neighbourhood while the Update operation computes a new hidden embedding for each node at the end of a message passing iteration . The equation corresponding to the Aggregate and Update operations are defined in Equations (4.7) and (4.8) respectively. After k iterations of message passing in a GNN, the hidden embedding $h_u^{(k)}$ of each node $u \in V$ is updated using the Update function to produce a new embedding $h_u^{(k+1)}$. The Update operation takes as input the hidden embedding of node u from the previous iteration and

Algorithm 1 Transformation to Graphs

Input: Dataset D' ▷ Preprocessed Dataset
Output: Graph Dataset D_g
Initialize:
$D_g \leftarrow \emptyset$ ▷ Graph Dataset
$\mathcal{N} \leftarrow \{(\Delta x, \Delta y) \mid \Delta x, \Delta y \in \{-1, 0, 1\}, (\Delta x, \Delta y) \neq (0, 0)\}$
▷ Neighbourhood Offsets
$A \leftarrow \emptyset$ ▷ Store Edge Connections
$R \leftarrow \{\}$ ▷ Store Edge Weights
$N \leftarrow \text{length}(X)$ ▷ Spatial grid dimension
for each $(X, Y) \in D'$ **do**
    $X_g, Y_g \leftarrow Reshape(X, (|V| \times C), Reshape(Y, (|V| \times 1)$
    for each i, j in range $0 < i \leq N$ and $0 < j \leq N$ **do**
        a $\leftarrow i \times N + j$
        for each $(\Delta x, \Delta y)$ in $\mathcal{N}$ **do**
            $(i', j') \leftarrow (i + \Delta x, j + \Delta y)$
            $b \leftarrow i' \times N + j'$
            $A \leftarrow A \cup (\text{a}, \text{b})$
            $R \leftarrow R \cup \{(a, b) \rightarrow 1\}$ ▷ Assign edge weight
        end for
    end for
    Construct Graph: $G(A, X_g), Y_g$
    $D_g \leftarrow D_g \cup \{G(A, X_g), Y_g\}$
end for

the aggregated message from its neighborhood $\mathcal{N}(u)$, as calculated using the $AGR(.)$ operation [30]. Each k^{th} message passing layer is parameterized by trainable weight matrices $W_{\text{self}}^{(k)}$ and $W_{\text{neigh}}^{(k)}$ which are applied to the node's own features and its neighbour's features, respectively. An element-wise nonlinearity σ, (such as tanh or ReLU), is applied to the final output from the message passing function to introduce non-linear transformations .

$$\begin{aligned} h_u^{(k+1)} &= \sigma\left(W_{\text{self}}^{(k+1)} h_u^{(k)} + W_{\text{neigh}}^{(k+1)} m_{\mathcal{N}(u)}^{(k)}\right) \\ &= \text{UPD}^{(k)}\left(h_u^{(k)}, m_{\mathcal{N}(u)}^{(k)}\right) \end{aligned} \tag{4.7}$$

$$m_{\mathcal{N}(u)}^{(k)} = \text{AGR}^{(k)}\left(\{h_v^{(k)} \mid \forall v \in \mathcal{N}(u)\}\right) \tag{4.8}$$

The architecture of a GNN largely depends on the design of the Aggregate and Update functions within the message passing framework defined in Subsection 4.1.3. Based on different mechanisms of message passing, various GNN architectures have been proposed to capture the complex relationships in graph structured data. In our proposed approach, we utilise the message passing mechanisms defined in the Graph Convolutional Network (GCN) [48], Graph Attention Network (GAT) [83], and, GraphSAGE network [31] and propose three different GNN node classification architectures. The overall workflow and the architecture of GNN is depicted in Figure 4.3. The message passing mechanism corresponding to a particular GNN variant among

GCN, GAT, and GraphSAGE is illustrated in Figure 4.4.

- **Graph Convolutional Network (GCN)** Graph Convolutional Networks (GCN) are based on the application of the principles of convolutional neural networks (CNN) to graph structured data. GCN introduces two additional features to the MP function in GNN - self loops and symmetric normalization as defined in equation (4.9). In a GCN, self loops are introduced in the graph so that the aggregregation is taken over the node itself along with its neighbourhood. The symmetric neighbourhood normalization in GCN is another feature which normalises the aggregation operation based on the degrees $|\mathcal{N}|$ of the nodes to prevent numerical instabilities in the training process.

$$h_u^{(k+1)} = \sigma\left(W^{(k+1)} \cdot \sum_{v \in \mathcal{N}(u) \cup \{u\}} h_v^{(k)} \cdot \frac{1}{\sqrt{|\mathcal{N}(u)| \cdot |\mathcal{N}(v)|}}\right) \tag{4.9}$$

- **Graph Attention Network (GAT)**

  The Graph Attention Network (GAT) is another variant of GNN in which attention mechanisms are incorporated into the aggregation process. An attention mechanism is a widely used strategy in deep learning to improve model predictions by focusing on relevant information. In the context of GNNs, attention mechanisms are used to weigh the importance of neighboring nodes in relation to a specific node during the aggregation step. The update operation, as described by equation 4.7, is modified to a weighted sum of the neighborhood node embeddings as shown in equation 4.10. The weight $\alpha_{u,v}$ corresponds to the attention weight of neighbor $v \in N(u)$ while performing aggregation at the node u and is defined in Equation (4.11)where a is a trainable attention vector, W is a trainable matrix, and $\|$ denotes the concatenation operation.

$$h_u^{(k+1)} = \sigma\left(\sum_{v \in \mathcal{N}(u) \cup \{u\}} \alpha_{u,v} W^{(k+1)} \cdot h_v^{(k)}\right) \tag{4.10}$$

$$\alpha_{u,v} = \frac{\exp\left(a^\top [Wh_u \| Wh_v]\right)}{\sum_{v' \in \mathcal{N}(u)} \exp\left(a^\top [Wh_u \| Wh_{v'}]\right)} \tag{4.11}$$

- **GraphSAGE**

  In [31], a framework was introduced to learn node embedding in dynamic graphs without retraining which is achieved through a process called inductive learning. Unlike traditional approaches such as GCN [48] which directly sum the features of neighboring nodes, GraphSAGE allows for generalized aggregation methods,denoted by AGR in Equation (4.12),

including mean, maximum and LSTM. It also employs concatenation for combining the aggregated neighborhood information with its own node embedding in the Update operation as shown in Equation (4.13). In the proposed approach, the maximum aggregation operation in [31] is used where the feature vectors of neighboring nodes are averaged to form a neighborhood representation. The propagation in the message passing function defined by the GraphSAGE aggregation operation is.

$$h_{\mathcal{N}(u)}^{(k+1)} = AGR^{(k)}\left(\left\{h_v^{(k)} \mid \forall v \in \mathcal{N}(u)\right\}\right) \tag{4.12}$$

$$h_u^{(k+1)} = \sigma\left(W^{(k+1)} \cdot \text{CONCAT}\left(h_u^{(k)}, h_{\mathcal{N}(u)}^{(k+1)}\right)\right) \tag{4.13}$$

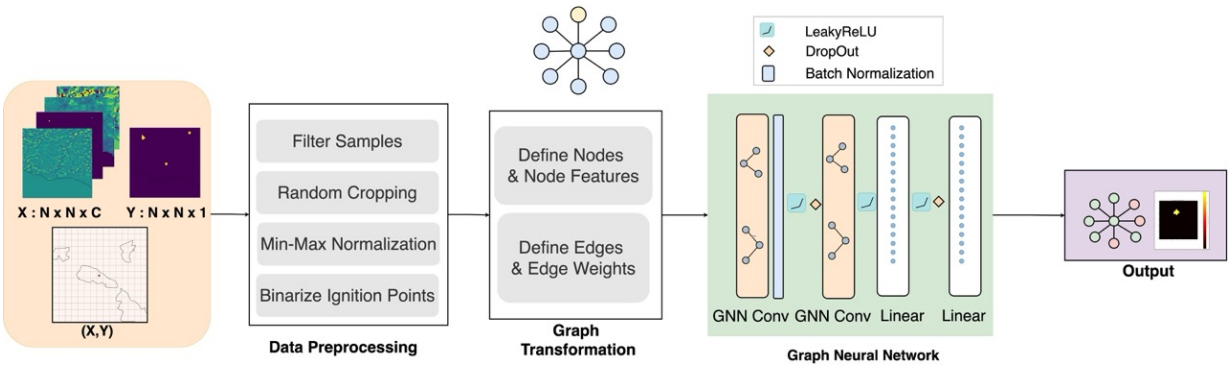


Figure 4.3: Overall Workflow and Model Architecture of GNN

In the proposed approach, three distinct architectures, based on the message passing mechanism of GCN, GAT, and GraphSAGE, are employed for binary node classification. The baseline architecture of each of the three distinct models comprises two GNN convolutional layers, each with a predefined embedding size, stacked together. The embedding size is a hyperparameter that was determined through careful evaluation across multiple experiments. Experiments are conducted with three sizes of hidden embeddings: $[32, 64, 128]$ and two depths $[2, 3]$, i.e., the number of GNN layers, for each of the three GNN models. The leaky ReLU activation function is subsequently applied after each GNN convolution layer to introduce non-linearity. The output from the final GNN convolutional layer and leaky ReLU activation function is subsequently passed through two linear layers, stacked together. Two dropout layers are incorporated to mitigate overfitting - one after the first GNN layer and the second after the final activation layer. A batch normalization layer is added after the first GNN convolution layer. The initial Graph Neural Network (GNN) layer receives the node feature matrix $X_g \in \mathbb{R}^{N^2 \times C}$, where $N^2 = 1024$ and C represents the number of input features. The adjacency matrix $A \in \mathbb{R}^{N^2 \times N^2}$ contains the pairwise

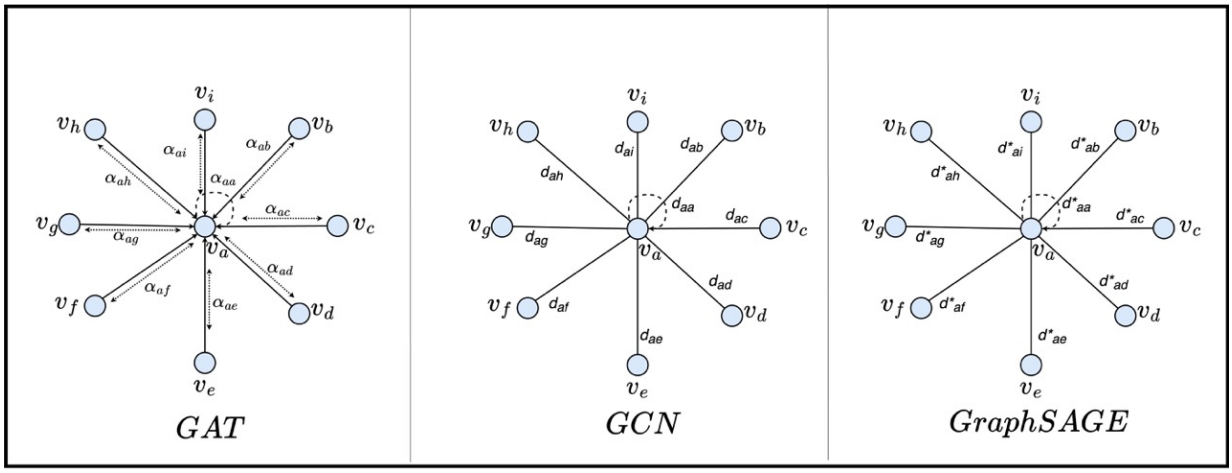


Figure 4.4: Message Passing Mechanism for a specific GNN. α, d, and $d*$ denote the attention coefficient, normalized degree and generalized aggregation coefficient respectively.

connections between nodes, and the weight for each element corresponds to the weight of each edge. These matrices are subsequently input to the GNN model. The output matrix obtained is a binary mask corresponding to the final burned area resulting from the provided outputs and is denoted by $\hat{Y}_g \in \mathbb{R}^{N\times N\times 1}$. The architecture of the baseline GNN model is depicted in Figure 4.3.

4.2 Semantic Segmentation

This section discusses the semantic segmentation based approach for predicting the final burned area pattern resulting from an ignition point. Section 4.2.1 presents the mathematical formulation of the problem defined in Section 1.2 in terms of semantic segmentation and 4.2.2 discusses the segmentation models used for comparison to the proposed GNN framework.

4.2.1 Problem Formulation

Semantic segmentation involves predicting the class for each cell in a given grid. Mathematically, the objective is to learn a function f_S that takes an input $X' \in \mathbb{R}^{M\times N\times C}$ and predicts a segmentation mask $Q \in \mathbb{R}^{M\times N\times T}$, where T is the number of distinct classes as shown in Equation (4.14). In the context of the proposed approach, the input is the grid $X \in \mathbb{R}^{N\times N\times C}$ defined in Section 1.2 and the goal is to predict the segmentation mask $Q \in \mathbb{R}^{N\times N\times T}$, where $T = 2$. The final predicted probability for a cell at the location (i, j) is the maximum of the probability

corresponding to each class $t \in [0, 1]$ as shown in Equation (4.15). The visual representation of the Semantic Segmentation workflow is shown in Figure 4.5.

$$Q = f_S(X) \tag{4.14}$$

$$\hat{y}_{i,j} = \arg \max_{t \in \{0,1\}} q_{i,j}^t \tag{4.15}$$

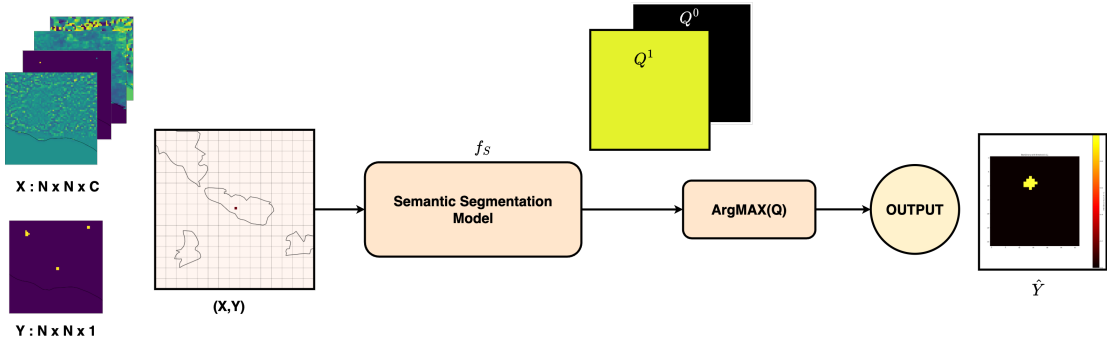


Figure 4.5: Semantic Segmentation Workflow

4.2.2 Model Architectures

Drawing upon popular neural network architectures such as CNNs, RNNs, Encoder-Decoders and, GANs and their architectural features such as skip connections,dilated convolutions and encoder-decoders, several architectures tailored for semantic segmentation have been developed. For the purpose of this study, we focus on 3 categories of semantic segmentation architectures, namely - Encoder-Decoder Architectures, Dilated Convolutional Models and Attention-based Models following the categorization in [56]

- **Encoder-Decoder Architectures**: As the name suggests, encoder-decoder architectures consist of two stages for effectively mapping samples from the input domain to the output domain. The encoder is used to capture the semantic information of the input by compressing it into a latent feature vector. The decoder, on the other hand, uses this latent representation of the input to predict the final output. One of the most popular semantic segmentation architectures originating in medical image segmentation is the U-Net family of architectures [66]. U-Net, initially introduced for medical image segmentation, employs an encoder-decoder structure where the encoder path extracts contextual features through

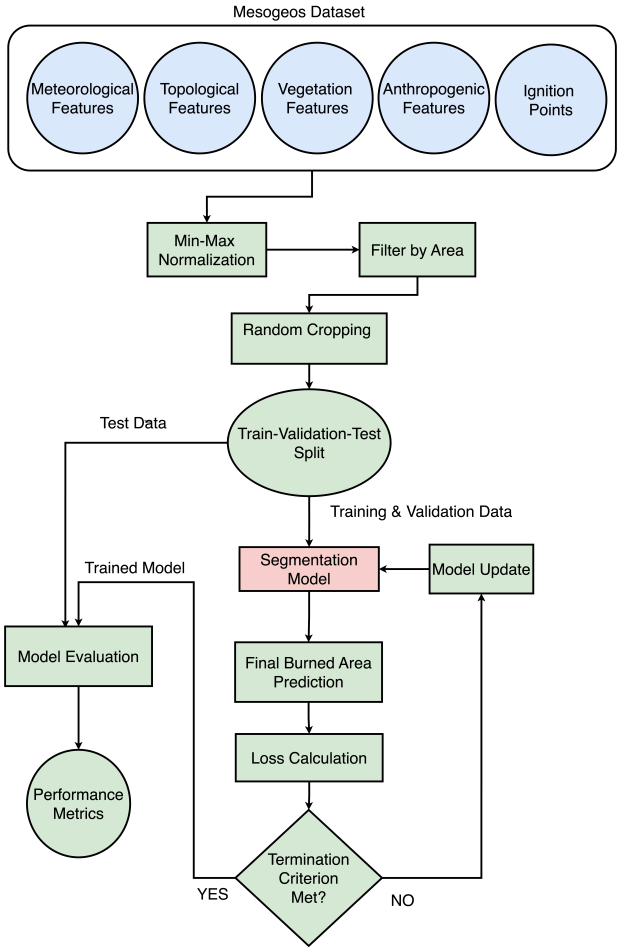


Figure 4.6: Semantic Segmentation Pipeline

successive convolutions and max-pooling layers, while the decoder path progressively upsamples these features to reconstruct the segmented image. Skip connections between the encoder and decoder preserve spatial information, improving localization. The architecture incorporates data augmentation techniques like elastic deformation to handle limited data and reduce overfitting. Building on U-Net, [99] introduced U-Net++, which modifies the skip connection mechanism for greater flexibility in feature aggregation. Instead of restricting skip connections to match the scale of features between the encoder and decoder, U-Net++ allows feature fusion across different scales. The model incorporates multiple U-Nets of varying depths, sharing a common encoder but with distinct decoders. This architecture enables the network to learn which decoder's feature maps are most relevant for segmentation, improving the model's ability to utilize multiscale features effectively.

- **Dilated Convolution Architectures**: Dilated convolutions are a subcategory of convolutional network modules developed to address the limitations of reduced feature resolution and loss of information in data that contains information at multiple spatial scales. In multiscale data, local and global information coexist, and dilated convolutions help capture both levels of information efficiently. Dilated Convolutions, also known as Atrous Convolutions, introduce another parameter called Dilation Rate which widen the receptive field of the network without increasing the number of parameters or losing resolution, allowing it to effectively process multiscale information [89]. The DeepLab architectures [10] are one of the most widely used semantic segmentation architectures which uses Dilated or Atrous convolutions as a fundamental architectural feature. DeepLabv3[11], in particular, utilizes Atrous Spatial Pyramid Pooling (ASPP) to aggregate multi-scale information by utilizing multiple distinct dilation rates. ASPP applies multiple parallel convolutions with varying dilation rates at different spatial resolutions, capturing features at multiple scales and improving segmentation accuracy, especially at object boundaries. DeepLabv3+[12] builds upon DeepLabv3 by incorporating an encoder-decoder architecture, where the DeepLabV3 encoder, enhanced by ASPP, is paired with a decoder module that refines coarse feature maps. This combination helps improve segmentation accuracy, particularly in tasks requiring precise boundary delineation. In the proposed approach, the DeepLabV3+ is used for the semantic segmentation of burned area patterns, to capture both local and global context effectively.

- **Attention-based Architectures**: Attention mechanisms in semantic segmentation enhance the capture of long-range dependencies and focus on salient features. The Multi-scale Attention Network (MA-Net) is a notable architecture designed for liver and tumor segmentation [20] which also exhibit significant imbalance in class distributions. MA-Net introduces a multi-scale attention mechanism to effectively capture information at different scales. It consists of two key components: a multi-scale feature extraction module using dilated convolutions, and an attention fusion module. The feature extraction module captures features at various receptive fields, while the attention fusion module combines spatial and channel attention to selectively emphasize relevant features from different scales. This architecture, built upon a backbone network like ResNet, allows MA-Net to achieve state-of-the-art performance in medical image segmentation by effectively integrating local and global contextual information.

Chapter 5

Experiments & Results

This chapter presents the experiments conducted to evaluate the methodology described in Chapter 4 and provides a comprehensive analysis of the results obtained from the experiments. The chapter is organized as follows: Sections 5.1 to 5.3 serve as the foundational background for the experiments conducted, while Sections 5.4 to 5.8 detail specific experiments conducted to validate the methodology outlined in chapter 4. More specifically, Section 5.1 elucidates the details of the experimental setup, including details on the dataset split and hyperparameters used in subsequent experiments and Section 5.2 provides a detailed description of the performance metrics used, discussing their suitability to the imbalanced class distribution in the dataset. Section 5.4 evaluates the suitability of various loss functions to address the imbalance issue by assessing the performance of four loss functions on a uniform configuration of the Graph Attention Network (GAT), Graph Convolutional Network (GCN), and GraphSAGE based Graph Neural Network (GNN) models identified from the experiments. Section 5.5 examines the impact of the number of layers and the hidden embedding size across all GNN models and determines the optimal configuration for each architecture for further experiments. Section 5.6 performs a comparative analysis of the three GNN models while Section 5.7 investigates the importance of the various feature categories, defined in 3.1, through an ablation study on the best performing GNN models. The chapter concludes with Section 5.8 in which the performance of GNN models is compared to the standard segmentation architectures defined in Subsection 4.2.2 to assess their effectiveness in burned area prediction.

5.1 Experimental Setup

In order to split the dataset for training, validation and test, a temporal split was performed to mirror the year-wise split in [49]. A filtering of samples was performed as described in subsection

3.4 to exclude outliers i.e. extremely large or small burned area patterns. The number of samples and corresponding year(s) of interest in the training, validation, and, test datasets, prior to and post filtering, is outlined in Table 5.1.

Category	**Years**	**No. of Samples (Unfiltered)**	**No. of Samples (Filtered)**
Training	2006-2019	12551	11054
Validation	2020	1781	1610
Test	2021-2022	3527	3161
Total	2006-2022	17859	15825

Table 5.1: Temporal Split of Dataset

All models were trained on a single NVIDIA GeForce RTX 3060 Ti GPU. A uniform set of hyperparameters was used in the training phase of all the models. The Adam optimizer was employed with an initial learning rate of 0.001 and was complemented by a *ReduceLRonPlateau* learning rate scheduler with a factor of 0.5 to adjust the learning rate for faster convergence dynamically. Due to constraints in the available computational resources, a larger batch size was simulated using gradient accumulation with 8 accumulation steps and a baseline batch size of 64. A weight decay of 0.0005 was used to regularize the model training to mitigate overfitting and control gradient magnitudes. The loss function parameters were set as follows: for the weighted binary cross-entropy loss, a positive class weight of 17.85 was assigned; for the Tversky loss, the negative class was weighted at 0.3 and the positive class at 0.7; and for the focal loss, the weighting factor for the positive class was set to 0.75, while the focusing parameter was set to 2. The training of all models was performed with an epoch size of 200 and early stopping, with a patience of 10, was employed to prevent overfitting.

Hyperparameter	Value
Optimizer	Adam
Initial Learning Rate	0.001
Baseline Batch Size	64
Gradient Accumulation Steps	8
Weight Decay	0.0005
Epoch Size	200
Patience	10
Weighted Binary Cross Entropy Weight	Positive class: 17.85
Tversky Loss Weights	Negative class: 0.3, Positive class: 0.7
Focal Loss Weights	Positive class: 0.75, Focusing parameter: 2

Table 5.2: Hyperparameters used in the training phase.

The GNN model architectures were developed using the PyTorch Geometric Library [22]. The dataset was converted to a graph representation using the custom dataset class in PyTorch Geometric. The segmentation models used for comparison are sourced from the Segmentation Models PyTorch library [38].

5.2 Performance Metrics

In the proposed approach, the predictive performance of the trained models is evaluated using six metrics: Accuracy, Precision, Recall (Sensitivity), Dice Similarity Coefficient (DSC), Cohonen's Kappa (κ), Area Under the Precision-Recall Curve (AUPRC) and, Area Under the Receiver Operating Characteristic Curve (AUROC) . A classification threshold of 0.5 is applied to the sigmoid-generated confidence scores to produce binary predictions for computing the single threshold metrics i.e. Accuracy, Precision, Recall, DSC Coefficient (DSC), and, Cohonen's Kappa (κ). The prediction of the spatial extent of the burned area resulting from a wildfire mandates good performance in terms of both classification and localization accuracy. In large-sized grids, wildfires often occupy only a small portion of the grid in comparison to the unburnt area resulting in a considerable class imbalance. This leads to higher accuracy in predicting the negative or unburned cells in comparison to the burned cells. Hence, the usage of performance metrics that do not account for the imbalance overestimates the predictive performance of models trained on such datasets. To address this, more importance is given to the Precision, Recall, Dice Similarity Coefficient, Cohonen's Kappa, and AUPRC to assess model performance. Other metrics such as AUROC and Accuracy are also included to reflect the unsuitability of these metrics in imbalanced class distributions. The detailed descriptions of each of the aforementioned metrics

Notation	Description
TP	True Positives: Correctly predicted positive instances
FP	False Positives: Incorrectly predicted positive instances
TN	True Negatives: Correctly predicted negative instances
FN	False Negatives: Incorrectly predicted negative instances

Table 5.3: Notations for Performance Metrics

and their mathematical formulations are outlined below. The notations used in the mathematical formulations are outlined in 5.3.

- **Accuracy**: It is defined as the ratio of the number of correct predictions, including both the negative and positive classes, to the total number of predictions. Although it is relatively simple to compute and is widely used in classification tasks, accuracy is a highly unsuitable measure for imbalanced class distributions where due to the dominant negative class, the accuracy scores are highly overestimated even when the model performance is subpar.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.1}$$

- **Precision**: Precision is a measure of the ability of the model to reduce false positives. It is defined as the ratio of the number of true positives to the total number of positive predictions as defined in (5.2).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{5.2}$$

- **Recall**: Recall, also known as Sensitivity or True Positive Rate, is a measure of the capability of the model to accurately predict the true positive samples. In comparison to Precision, which indicates the ability of a model to reduce false positives, Recall is more suitable for binary classification with imbalanced classes, such as in wildfire segmentation, as it focuses more on achieving correct predictions on the true positive samples i.e. actual burned area cells. Recall is computed as the ratio of the true positives (correctly predicted positive instances) to the total number of actual positive instances in a sample as shown in (5.3).

$$\text{Recall} = \frac{TP}{TP + FN} \tag{5.3}$$

- **Dice Similarity Coefficient (DSC)**: Dice Similarity Coefficient is a widely used metric in segmentation that is used to measure the overlap between the predicted grid and the

ground truth. Mathematically, it is identical to the F1-Score for binary classification and hence can be represented as the harmonic mean of Precision and Recall as shown in 5.4.

$$\text{DSC} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2TP}{2TP + FP + FN} \tag{5.4}$$

- **Cohonens's Kappa (κ)**: Cohonen's Kappa (κ) is a statistical measure, originating in psychology, that is used for evaluating the degree of agreement between the annotated or ground truth classifications and the predicted classification of nominal or categorical variables. In addition to the observed probabilities, Cohonen's Kappa incorporates the random chance of agreement between the predicted and annotated classifications based on the distribution of classes in the dataset which makes it suitable for imbalanced data. The metric ranges from -1 to +1, with +1 indicating perfect overlap and 0 or lower representing poor overlap. It can be denoted by Equation 5.5 where p_e and p_o represent the expected and observed agreements respectively and are calculated as described in Equations 5.6 and 5.7

$$\kappa = \frac{p_o - p_e}{1 - p_e} \tag{5.5}$$

$$p_e = \left(\frac{TP + FP}{N} \times \frac{TP + FN}{N}\right) + \left(\frac{TN + FN}{N} \times \frac{TN + FP}{N}\right) \tag{5.6}$$

$$p_o = \frac{TP + TN}{TP + TN + FP + FN} \tag{5.7}$$

- **AUROC**: AUROC (Area Under the Receiver Operating Characteristic Curve) is a commonly used metric for classification tasks and is computed from the ROC plot. The ROC is a representation of the relationship between the False Positive Rate (FPR) and Recall or True Positive Rate (TPR) at different thresholds. The baseline of the AUROC is insensitive to class distributions and shows little to no change in value in balanced and imbalanced scenarios [71]. The value of AUROC is fixed at 0.5 for random classifiers and is 1.0 for a perfect classifier. Equation (5.8) shows the calculation of AUROC for n distinct thresholds.

$$\sum_{i=1}^{n-1} \frac{\text{TPR}_i + \text{TPR}_{i+1}}{2} \cdot (\text{FPR}_{i+1} - \text{FPR}_i) \tag{5.8}$$

- **AUPRC**: AUPRC (Area Under Precision-Recall Curve) is a metric used to evaluate model performance based on the relationship between precision and recall values of a model across a range of classification thresholds. While AUROC is a more commonly used metric in classification or segmentation tasks, it is not particularly suitable for imbalanced datasets

and often overestimates model performance in datasets consisting of highly imbalanced classes. Additionaly, the baseline of AUPRC, which represents the performance of a random classifier on the dataset, is not fixed unlike in AUROC. Instead, the baseline value changes with the distribution in classes and is given by the ratio of positive class labels to the total number of samples and can be computed as 0.5 for a balanced class [71]. Given n threshold values, the AUPRC can be calculated as a weighted sum of the Precision value Pr_i at the i^{th} threshold with the difference in recall at the current and previous threshold, $R_i - R_{i-1}$ acting as the weight as shown in (5.9)

$$\text{AUPRC} = \sum_{i=1}^{n} (R_i - R_{i-1}) \cdot Pr_i \tag{5.9}$$

5.3 Loss Functions for Imbalanced Class Distributions

Loss functions are a fundamental part of the model training process and play a key role in achieving optimal predictive performance in a deep-learning model. The loss function of a model quantifies the deviation between the actual and predicted values and the ultimate goal of the training process is to minimize this deviation. The selection of a suitable loss function is crucial for achieving accurate predictive performance and heavily depends on factors such as the nature of the data and the task to be performed. For regression tasks, loss functions such as Mean Squared Error (MSE) and Mean Absolute Error (MAE) are commonly used to measure the deviation between continuous values. On the other hand, for classification tasks functions such as Cross Entropy Loss are used to compute the deviation in actual and predicted discrete class labels. Classification tasks for spatial grids, such as in Semantic Segmentation, are analogous to pixel-level classifications,and loss functions specific to classification such as Cross Entropy are widely utilized. In classification-specific loss functions, the emphasis is often on minimizing the dissimilarity between data distributions [55], [40]. However, in the classification of spatial data, other aspects such as the intra-region and intra-boundary differences are also taken into account. Based on the specific aspect to be minimized, loss functions can be broadly categorized into four categories: distribution-based, region-based, boundary-based, and compound losses.

5.3.1 Distribution-based Loss Functions

Distribution-based loss functions are designed to minimize the dissimilarity between the actual and predicted data distributions. Cross Entropy loss is the most fundamental of these loss functions while others are simply its extension or modification.

- **Binary Cross Entropy**: Binary Cross Entropy (BCE) is a measure of the deviation between the predicted and ground truth distributions. However, binary cross-entropy is

unsuitable for imbalanced data where the majority class largely dominates the loss function resulting in a low loss despite poor predictive performance on the minority class. For a $N \times N$ grid, the Binary Cross Entropy loss can be given as defined in equation (5.10)

$$L_{\text{BCE}} = -\frac{1}{N^2}\sum_{i=1}^{N^2}\left(y_i \log(\hat{p}_i) + (1-y_i)\log(1-\hat{p}_i)\right) \tag{5.10}$$

where $y_i \in \{0, 1\}$ is the binary class label for grid location i, where 1 represents a burned location and 0 represents an unburnt location. The term $\hat{p}_i \in [0, 1]$ represents the predicted probability that a location i will be classified as burned, while $(1 - \hat{p}_i) \in [0, 1]$ represents the probability that a location i will be classified as unburnt.

- **Weighted Binary Cross Entropy**: To overcome the limitations of BCE loss in handling imbalanced class distributions, weighted binary cross entropy (WBCE) is used. In WBCE loss, the positive class is weighed by a coefficient β as shown in equation (5.11), a hyperparameter determined based on the imbalance ratio between the positive and negative class. By setting $\beta > 1$, the loss function prioritizes reducing false negatives [40], improving the model's performance on the minority class.

$$L_{\text{WBCE}} = -\frac{1}{N^2}\sum_{i=1}^{N^2}\left(\beta y_i \log(\hat{p}_i) + (1-y_i)\log(1-\hat{p}_i)\right) \tag{5.11}$$

- **Focal Loss:** Another variant of BCE loss, which is suitable for imbalanced class distributions, is the Focal Loss (FL). The rationale behind Focal Loss is to improve the predictive performance on samples that are difficult to classify i.e. produce high losses. To achieve this, a modulating factor $(1-p_t)^\gamma$ where p_t denotes the predicted probability of the positive class is used. It is used to downweigh samples that are easily classified by the model. The loss function has two tunable hyperparameters, γ and α. The hyperparameter $\gamma \geq 0$ determines the degree to which the contribution of easily classifiable samples will be reduced. On the other hand, α is used to balance the weight of the positive and negative classes and is especially useful for handling imbalanced class distributions [53].

$$L_{\text{FL}} = -\frac{1}{N^2}\sum_{i=1}^{N^2}\alpha_t(1-p_t)^\gamma \log(p_t) \tag{5.12}$$

5.3.2 Region-based Losses

Region-based loss functions are designed to maximize the region of overlap between the predicted and actual segmentation masks. By maximizing the overlap between the predicted and true masks, region-based loss functions encourage the model to focus on accurately capturing both the boundaries and extent of the regions of interest [55].

- **Tversky Loss**: The Tversky Loss is a variant of the Dice Loss [77] which takes into account class imbalance by assigning different weights to false positives and false negatives. The hyperparameter β is used to control the trade-off between the precision and recall in the predictions [72].

$$a = \sum_{i=1}^{N^2} \hat{p}_i y_i, \tag{5.13}$$

$$b = \beta \sum_{i=1}^{N^2} (1 - \hat{p}_i) y_i, \tag{5.14}$$

$$c = (1 - \beta) \sum_{i=1}^{N^2} (1 - y_i)(1 - \hat{p}_i) \tag{5.15}$$

$$L_{\text{Tversky}} = 1 - \frac{a}{a + b + c} \tag{5.16}$$

Overall, weighted loss functions that assign specific weights to the minority class in imbalanced distributions present a more promising approach for classifying burned areas. To evaluate this, we conduct experiments using all the loss functions discussed above.

5.4 Comparative Analysis of Loss Functions for Handling Class Imbalance

The selection of an appropriate loss function is paramount in the context of imbalanced datasets to prevent the model's performance from being skewed by the majority class. In the burned area prediction task, imbalance occurs as a foreground-background imbalance wherein burned cells occupy a small region of a larger unburned region as described in Section 3.3. The objective of this experiment is to investigate the effectiveness of different loss functions in addressing the challenge posed by an imbalanced dataset during training. To conduct this analysis, two categories of loss functions - distribution-based (Binary Cross Entropy (BCE), Weighted BCE, and Focal

loss) and region-based loss functions (Tversky loss), as described in Section 5.3, are considered. The BCE loss is used as a baseline, and the performance of the other loss functions is evaluated in comparison to it. All the experiments are conducted by considering a uniform architectural configuration in terms of the hidden size, the number of GNN layers, and the number of features for the GCN, GAT, and GraphSAGE models as defined in Table 5.4. The hyperparameters are consistent across all experiments and have been described in Section 5.1. The impact on the performance of the models for each loss function is analyzed and compared based on the performance metrics defined in Section 5.2. Additionally, the average of the Dice score, Kappa score, and, AUPRC is also included to assess the overall performance. The results shown in Table 5.5 demonstrate the importance of choosing a suitable loss function for datasets with an imbalance in the number of positive and negative class nodes.

No. of GNN Layers	**Hidden Embedding Size**	**No. of Heads (GAT)**	**Feature Category**
2	64	2	All

Table 5.4: Configuration of GNN Models for Loss Comparison

1. **Binary Cross Entropy (BCE) Loss:** The BCE loss achieves the highest precision score across all models but exhibited poor recall scores nearing zero for the GCN and GAT models and moderate performance on GraphSAGE. This trend reflects the bias towards reducing false positives but poor predictive capabilities on the actual positive class due to the imbalance in class distribution. The AUPRC scores, while the highest across all models, were predominantly impacted by precision. The models were able to correctly reduce false positive instances but failed to recognize many true positive instances. The Dice and Cohonen's Kappa scores for GCN are moderate, while those for GAT are close to zero. In contrast, GraphSAGE achieves considerably good scores. The high performance on the negative class inflates the AUROC and Accuracy scores reflecting the unsuitability of these metrics for imbalanced datasets. These results suggest that while BCE can achieve relatively high precision, it struggles with recall, making it less effective for imbalanced wildfire datasets where the identification of true positives is critical.

2. **Weighted Binary Cross Entropy (WBCE) Loss:** The weighted BCE loss exhibits the highest recall across all models, indicating improved prediction of true positive instances. This can be attributed to weights assigned to the positive class in computing the loss. However, the precision score drops significantly as compared to the BCE loss indicating an increase in false positives resulting from the weighting of positive class samples. The

AUPRC was high for the GCN and GraphSAGE models but was lower for GAT. Additionally, WBCE produced significantly higher Dice scores than BCE suggesting improved overall prediction.

3. **Tversky Loss:** The Tversky loss achieved moderately high scores for both precision and recall indicating a balanced overall performance on the negative and positive classes. The AUPRC scores were also high for GAT and GCN but exhibited a slight decrease for GraphSAGE. The model's performance was further reflected in the high Dice and Kappa scores, indicating a more effective prediction of both negative and positive classes as compared to BCE and WBCE. While the AUROC remained inflated, similar to all other losses, the overall performance of Tversky Loss was strong, particularly in terms of recall and model generalization.

4. **Focal Loss:** Focal Loss, designed to focus on hard-to-classify samples, performed poorly in terms of precision for the GCN model and moderately for the GAT and GraphSAGE models, though it still lagged behind Tversky Loss and BCE in terms of precision. One of the key issues with Focal Loss was its recall, which was close to zero for both GCN and GAT, indicating that these models struggled to predict positive instances. However, for GraphSAGE, recall was more comparable to that of Tversky Loss, showing that Focal Loss can improve performance when applied to certain models. AUPRC followed the same trend as precision, with poor results for GCN but moderate performance for GAT and GraphSAGE, reflecting the challenges imposed by Focal Loss in balancing precision and recall. The Dice and Kappa scores were low for GCN and GAT, but remarkably higher for GraphSAGE, indicating that Focal Loss's ability to handle hard examples improved performance on the positive class in the GraphSAGE model. The AUROC for GCN was poor, but for GAT and GraphSAGE, it was considerably higher, approaching 0.95, suggesting that Focal Loss can achieve high overall accuracy and model performance in certain cases.

Overall, across all GNN models, Tversky loss consistently outperformed other loss functions in terms of the Dice Score and Cohonen's Kappa, demonstrating its effectiveness in handling the imbalance in positive and negative class nodes. The Weighted BCE loss resulted in slightly lower metric scores but was close in performance to the Tversky loss. The BCE loss, which did not consider imbalance, exhibited a poor performance on the positive class. The Focal loss, was observed to depend heavily on the type of GNN model but performed poorly overall. Further, the predicted probabilities, before thresholding, were also seen to vary across the different models scores predicted as shown in Figure 5.1. The probabilities, which denote the confidence in the prediction for the positive class, were much higher for Tversky and Weighted BCE Loss but were only marginally higher than the threshold for the BCE and Focal losses, further indicating the strength of the aforementioned loss functions to the imbalanced node classification task. Based on the results obtained from this experiment, all further experiments have been conducted by using the Tversky loss function.

Loss Function	Model	Precision	Recall	Dice*	Kappa*	AUPRC*	AUROC	Accuracy	Average*
BCE	GraphSAGE	0.5736	0.3461	0.4317	0.4276	0.4731	0.9421	**0.9915**	0.4441
BCE	GCN	**0.6984**	0.1657	0.2678	0.2651	0.4771	0.9404	0.9914	0.3367
BCE	GAT	0.6875	0.0040	0.0079	0.0078	0.4603	0.9447	0.9906	0.1587
Focal	GraphSAGE	0.4376	0.6344	0.5180	0.5125	0.4105	0.9377	0.9888	0.4803
Focal	GAT	0.4224	0.0207	0.0395	0.0386	0.4250	0.9396	0.9905	0.1677
Focal	GCN	**0.0001**	**0.0004**	**0.0002**	**0.0001**	**0.0150**	**0.6283**	0.9905	**0.0051**
Tversky	GCN	0.4328	0.6446	0.5179	0.5123	**0.4772**	0.9387	0.9886	**0.5024**
Tversky	GAT	0.4488	0.6354	0.5261	0.5208	0.4331	0.9377	0.9891	0.4933
Tversky	GraphSAGE	0.4567	0.6221	**0.5267**	**0.5215**	0.3679	0.9097	0.9894	0.4720
WBCE	GCN	0.3167	0.7910	0.4523	0.4448	0.4768	**0.9448**	0.9819	0.4580
WBCE	GraphSAGE	0.2736	0.8271	0.4112	0.4027	0.4497	0.9403	0.9776	0.4212
WBCE	GAT	0.2507	**0.8485**	0.3870	0.3780	0.3157	0.9431	**0.9746**	0.3602

Table 5.5: Results for Loss Functions. (**Average*** indicates the average computed for columns marked with *)

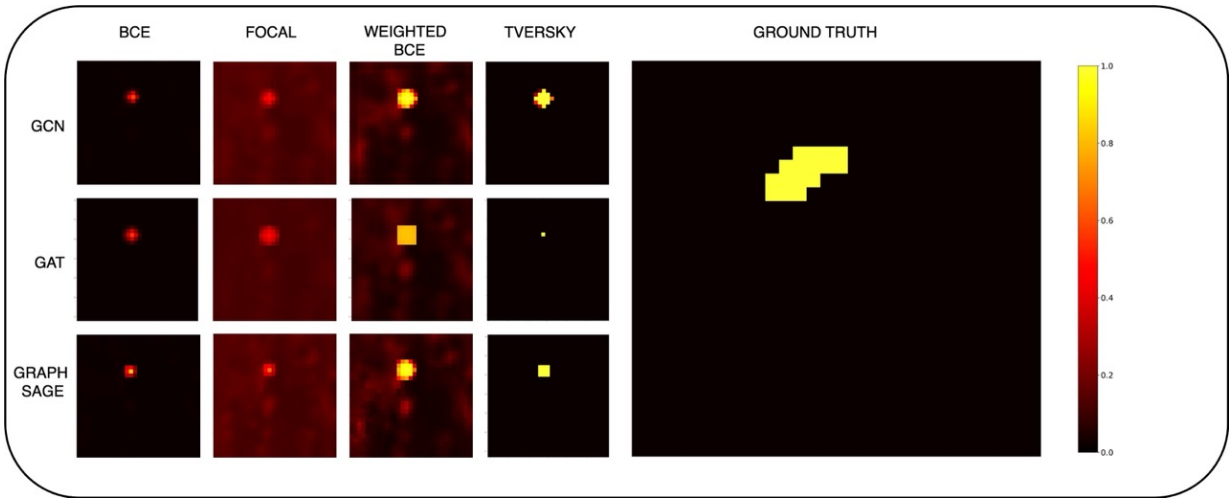
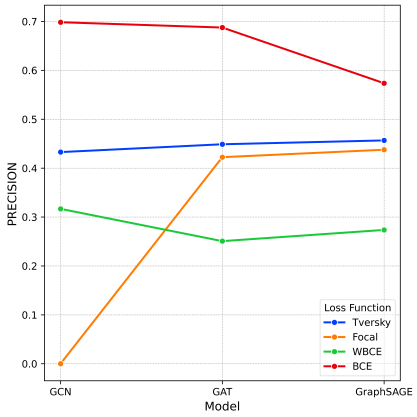


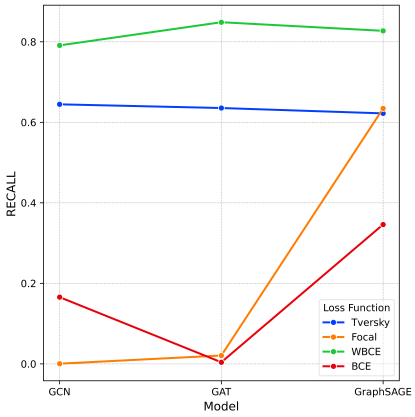
Figure 5.1: Predicted Probabilities for a Test Sample

5.5 Model Configuration

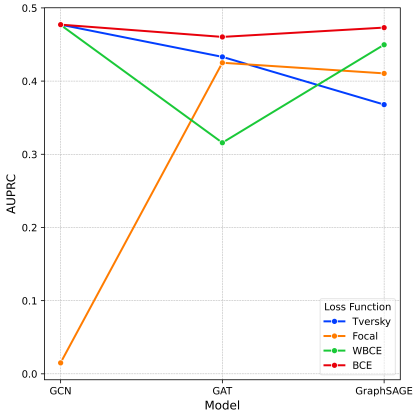
In order to achieve optimal model performance, architectural hyperparameters such as the number of layers and the size of embeddings in the hidden layers of GNN need to be selected with



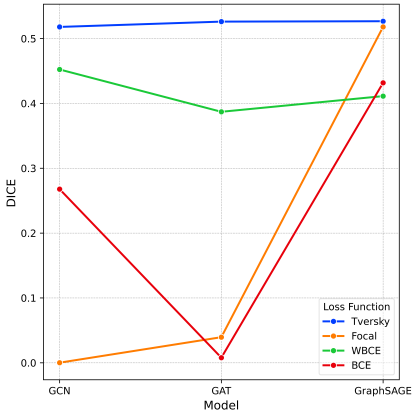
(a) Precision



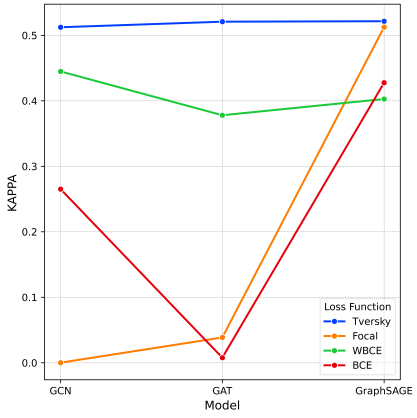
(b) Recall



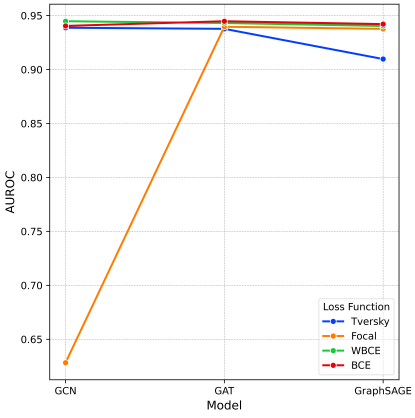
(c) AUPRC



(d) Dice Score



(e) Kappa Score



(f) AUROC Score

Figure 5.2: Impact of Loss Functions on Model Performance

Configuration ID	Embedding Size	No. of Layers
A	32	2
B	32	3
C	64	2
D	64	3
E	128	2
F	128	3

Table 5.6: List of Model Configurations

careful consideration. In this experiment, the objective is to analyse the impact of the increase or decrease in the number of GNN layers (depth) and the hidden embedding size to identify the best model configuration for each GNN model for further experiments. The practical implementations of Graph Neural Networks are hindered by certain phenomena, such as, oversmoothing and oversquashing. Oversmoothing is caused when the increase in the number of GNN layers leads to the convergence of node embeddings to the same constant value, causing the embeddings to be similar to each other and thus non-informative for downstream tasks such as node classification [67]. On the other hand, oversquashing arises when the increase in GNN layers leads to an exponential increase in the information flowing in from the nodes, and this information is compressed into a fixed-length vector, leading to the loss of information [3]. As a result, choosing the optimal hidden dimension may aid in curtailing the impact of oversquashing by retaining more information [6]. The motivation for this experiment is to choose the optimal number of layers and the optimal hidden embedding size without causing over-smoothing or oversquashing in the GNN models.

For the purpose of evaluation, experiments are conducted with three sizes of hidden embeddings: $[32, 64, 128]$ and two depths, i.e., the number of GNN layers, for each of the three GNN models (GCN, GAT, GraphSAGE). This resulted in a total of 18 trained models. All training and evaluations are conducted using all the input features (except burned areas) defined in Table Table 3.1. The loss function used for training all models is the Tversky loss. The results for all the models are provided in Table 5.7, where each of the 6 unique combinations of the hidden embedding size and the number of GNN Layers is denoted by a unique letter $[A - F]$ as shown in Table 5.6. The *Heads* column denotes the number of attention heads and is defined only for the GAT model as all other models do not utilize the attention mechanism.

5.5.1 GCN

The results of the various configurations of the GCN model are described below. The heatmaps depicted in Figure 5.3 offer a visual representation of the results, wherein performance progressively improves from the redder hues to the greener shades.

- **Effect of Layers:** It is observed that the precision improves with the addition of more layers across all hidden sizes indicating improved performance in decreasing the number

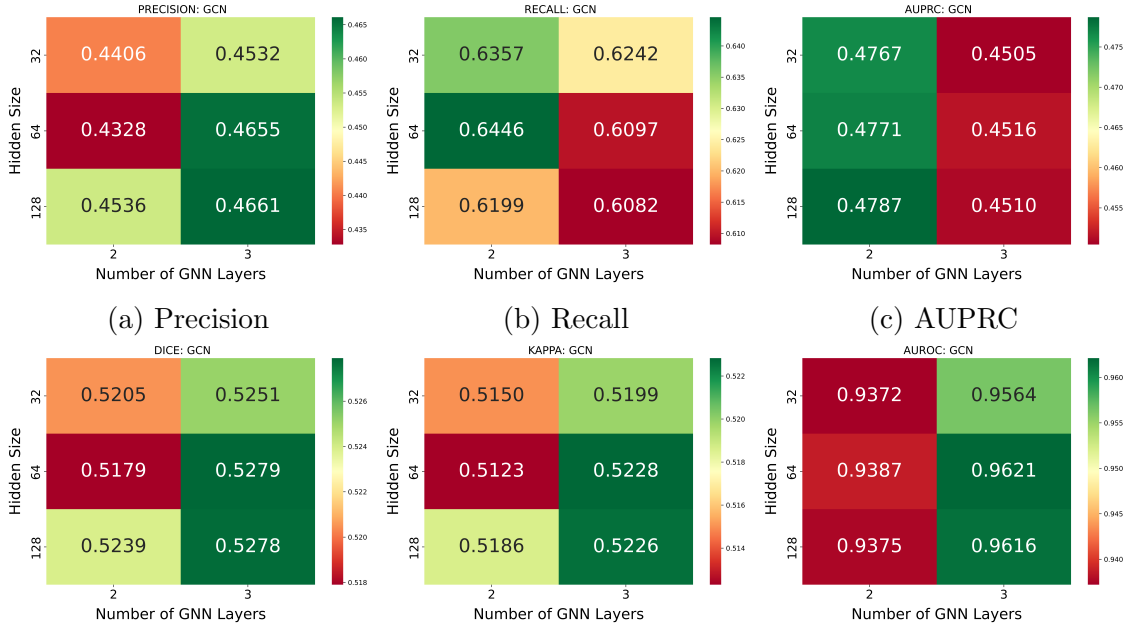
of false positives. However, the recall exhibits a contrasting trend and reduces with the increase in depth reflecting the degrade in the model's capabilities to correctly predict true positives which is crucial in predicting burned cells. The AUPRC follows a similar decreasing trend as recall and exhibits a drop in performance with depth. This suggests that the model is not effectively predicting the minority (positive) class at higher depths. On the other hand, Dice, Cohonen's Kappa and AUROC exhibit a consistent improvement with the addition of layers suggesting that the model's overall ability to distinguish between classes improves with depth, but at the cost of sensitivity to the minority class.

- **Effect of Hidden Size:** The Precision consistently improves with the increase in hidden sizes for both depths indicating a reduction in false positives. The Recall shows a varying trend for the two depths - at a depth of 2, it peaks at an intermediate embedding size of 64 but drops with further increase in size, while for 3 layers it decreases monotonically likely due to overfitting. The AUPRC, is dominated by precision at depth 2, and follows a similar trend by increasing with hidden size but shows a varying trend at a higher depth. The Dice and Kappa scores consistently improve with the increase in hidden size suggesting better overall predictive performance. AUROC shows a fluctuating trend but with minimal differences. Overall, larger hidden sizes consistently improve the precision, dice score, and kappa score, indicating better overall classification accuracy and reduction in false positives. However, the behavior of recall and AUPRC varies depending on the number of layers, highlighting that the optimal hidden size depends on the specific metric of interest and the architecture depth.

5.5.2 GAT

The results of the various configurations of the GAT model are described below. The heatmaps depicted in Figure 5.4 offer a visual representation of the results.

- **Effect of Layers:** The GAT architecture shows easily distinguishable trends across all metrics. The precision consistently improves with depth while the recall shows an opposite trend. The increase in precision across all hidden sizes suggests that GAT becomes more confident in reducing false positive predictions, however, the model misses more true positives, indicating a trade-off between precision and recall with the increasing number of layers. The AUPRC decreases monotonically across all hidden sizes indicating that the increase in precision is outweighed by the decrease in recall. The Dice score, Kappa score and, the AUROC increase with additional layers reflecting an overall improvement in predictive accuracy but degradation in predicting true positives.
- **Effect of Hidden Size:** The precision, recall and AUPRC show layer specific trends - for 2 layers the precision and AUPRC consistently improve while the recall consistently



(a) Precision (b) Recall (c) AUPRC

(d) Dice (e) Kappa (f) AUROC

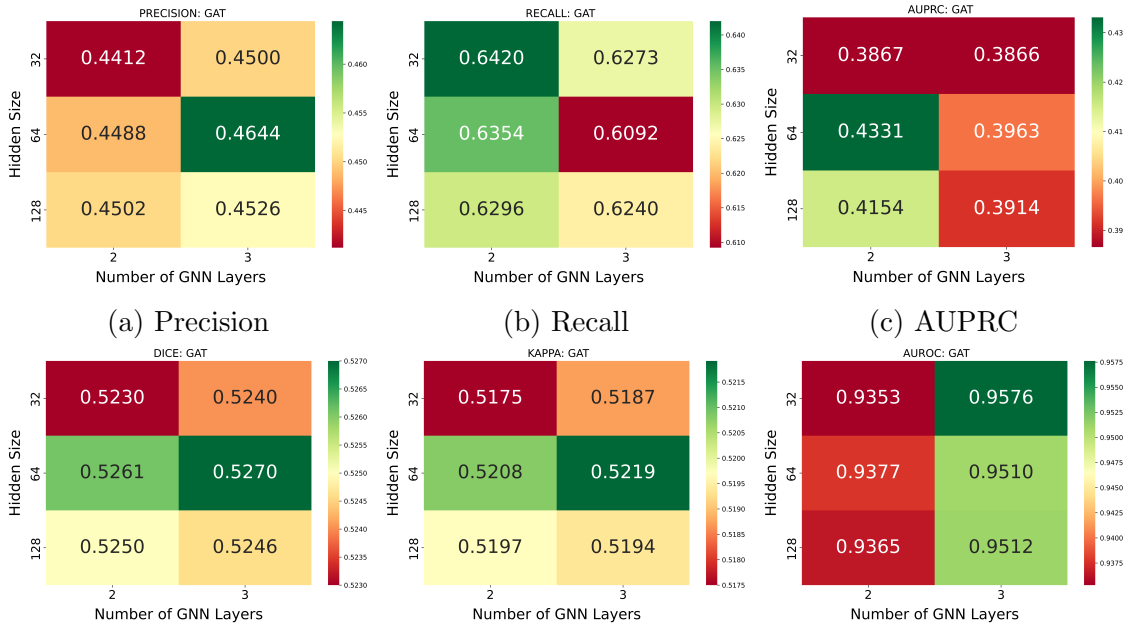
Figure 5.3: Performance Comparison of GCN Configurations

degrades with the increase in hidden size. For 3 layers, precision and AUPRC both improve initially and then drop, while recall decreases initially and then recovers, suggesting that the additional layer introduces complexity that requires careful tuning of the hidden size to balance these metrics. The Dice score, Kappa score, and the AUPRC follow non-monotonic trends. These metrics improve with hidden size up to a point and then decline for both 2 and 3 layers. This indicates that excessively large hidden sizes may lead to overfitting reducing the model's overall predictive performance.

5.5.3 GraphSAGE

The results of the various configurations of the GraphSAGE model are described below. The heatmaps depicted in Figure 5.5 offer a visual representation of the results.

- **Effect of Layers:** The precision worsens for all hidden sizes indicating higher number of false positives. On the other hand, the recall consistently improves meaning the model's ability to correctly predict true positives improves. The AUPRC exhibits a trend similar to



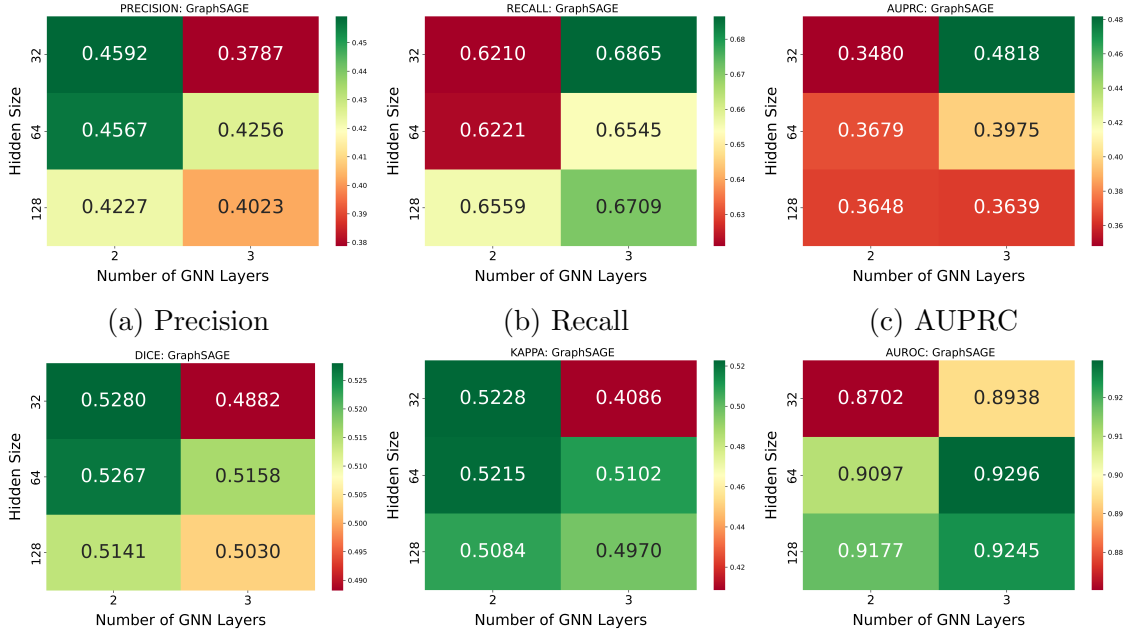
(a) Precision (b) Recall (c) AUPRC

(d) Dice (e) Kappa (f) AUROC

Figure 5.4: Performance Comparison of GAT Configurations

recall except at the embedding size of 128, where the gains due to the precision is outweighed by the decrease in precision. While the AUROC monotonically increases with depth, the Dice and Kappa score decreases with the increase in layers indicating that the worsening precision outweighs the gains in recall, leading to a decline in overall performance.

- **Effect of Hidden Size:** The effect on precision and recall vary with the layers but exhibit a contrasting trend. For two layers, the precision improves but the recall degrades while for three layers the precision initially improves and then dips while recall shows an opposite trend. The dice and kappa scores show non-monotonic trends- for 2 layers, both metrics decline consistently as the hidden size grows, reflecting reduced overall predictive quality. While for 3 layers, Dice and Kappa improve initially but then drop, suggesting that intermediate hidden sizes strike a better balance, while excessively large hidden sizes lead to overfitting . The AUROC and AUPRC exhibit layer specific performance, for 2 layers, the AUROC improves consistently with hidden size, indicating better separability of classes while AUPRC shows a fluctuating trend. For 3 layers, AUROC improves initially and then dips, while AUPRC consistently worsens, showing that deeper architectures may struggle with imbalanced data as hidden size increases.



(a) Precision (b) Recall (c) AUPRC

(d) Dice (e) Kappa (f) AUROC

Figure 5.5: Performance Comparison of GraphSAGE Configurations

5.6 Comparison of GNN Models

To evaluate the impact of different message-passing mechanisms in a Graph Neural Network, three distinct GNN architectures were trained and evaluated for their performance namely Graph Convolutional Network (GCN), Graph Attention Network (GAT), and GraphSAGE. In this section, a comparison of the different models are presented. Figure 5.7 showcases the performance comparison of the three models in terms of the performance metrics discussed in Section 5.2 and Figure 5.6 gives a visual representation of the results obtained using the best configuration of each model. The best configuration is selected on the basis of the average of the Dice, Kappa and, AUPRC scores due to their better suitability to the imbalanced class distribution as compared to AUROC and Accuracy. The Precision and Recall are not utilised in computing the average as the AUPRC already accounts for both Precision and Recall. The results of the comparative analysis are presented in Table 5.7. The best configurations for each model are highlighted in bold, while the highest and lowest values for each metric are marked with boldface and bold underlined text, respectively.

1. **Precision:** In terms of precision, GCN(F) exhibits the highest score of 0.4661 while GraphSAGE(B) performs the worst with a score of 0.3787. Overall, the GCN models demonstrate relatively consistent performance with most models achieving scores > 0.45 except configurations A and C, outperforming the majority of GAT and GraphSAGE models. On the other hand, GraphSAGE models exhibit greater variability in performance. While configurations A and C achieve considerably high values > 0.456, the performance drops for other configurations with the score plummeting to < 0.38 for configuration B. The GAT models show the most consistent results, although their precision values are moderate, typically falling within a narrow range of $(0.44, 0.47)$ that does not exhibit extreme highs or lows. Overall, GCN models provide the most favorable balance of consistent and high precision performance compared to GAT and GraphSAGE models.

2. **Recall:** The Precision-Recall tradeoff is clearly demonstrated by the contrasting trends observed in recall in comparison to precision. The majority of GraphSAGE models outperform the other models achieving recall scores > 0.65, with the exception of configurations A and C. In contrast, GCN models show variability in performance, with recall values ranging between 0.644 and 0.60. As compared to GCN, the GAT models exhibit more consistency across different configurations although their performance remains moderate and they do not achieve top-tier performance.Overall, GraphSAGE demonstrates good recall performance, with most models outperforming GCN and GAT.

3. **AUPRC:** In terms of AUPRC, GraphSAGE(B) achieves both the best and worst performance with a score of 0.4818 and 0.3480 respectively demonstrating its sensitivity to model configuration. On the other hand, GCN consistently outperforms all other models. GAT exhibits slightly inferior performance compared to GCN, yet maintains its consistent

performance across various configurations. In terms of AUPRC, most GraphSAGE models exhibit suboptimal performance, with the exception of B, which achieves the highest score, and D, which has a moderate value of 0.515. Overall, the precision appears to have a stronger influence on AUPRC, as the trends observed in AUPRC closely mirror those seen in Precision.

4. **Dice:** Similar to AUPRC, GraphSAGE shows extreme performances and achieves both the best and worst performance with a score of 0.528 and 0.4882 respectively. However, the majority of the GraphSAGE models underperform as compared to GAT and GCN. The GCN models also exhibit variability in performance, with a high of 0.5279 and a low of 0.5179, but the fluctuations are mild. GAT shows a similar trend to GCN, with scores consistently falling between 0.523 and 0.527, reflecting stable but moderate performance.

5. **Cohonen's Kappa:** The Cohonen's Kappa score shows a trend similar to Dice with extreme values for GraphSAGE, moderate fluctuations in GCN and, mild fluctuations in GAT. The top score however is achieved by GCN with 0.5228 while GraphSAGE is at a close second with 0.52283. The Kappa scores follow a similar trend to Dice Score, with GraphSAGE displaying significant fluctuations between configurations. GCN shows moderate variation in Kappa scores, while GAT exhibits minimal fluctuations. The highest Kappa score is achieved by GCN (0.5228), closely followed by GraphSAGE (0.52283), demonstrating the competitive performance between these models.

6. **AUROC:** In terms of AUROC, all the configurations of GraphSAGE underperform in comparison to other models. Both GCN and GAT show slight variations in performance, with GCN achieving better overall results. GCN has an AUROC score range from a high of ≈ 0.96 to a low of ≈ 0.94, while GAT scores range from ≈ 0.96 to ≈ 0.93, indicating moderate performance across its configurations.

7. **Accuracy:** All models display similar accuracy values, consistently close to 0.989. This high accuracy, however, highlights the inflated performance often associated with accuracy as a metric in imbalanced datasets, making it unsuitable for assessing model performance in such contexts.

5.7 Ablation Study: Feature Importance

Wildfire ignitions and the burned area resulting from an ignition are governed by a complex interplay of several factors which can be broadly categorised into five categories as specified in Table 3.1. To evaluate the significance of a particular category of features in predicting the final burned area, a systematic ablation study is conducted by removing a set of features and

Model	Embedding Size	No. of Layers	Configuration ID	Heads	Precision	Recall	Dice*	Kappa*	AUPRC*	AUROC	Accuracy	Average*
GAT	**64**	**2**	**C**	2	0.4488	0.6354	0.5261	0.5208	0.4331	0.9377	0.9891	0.4933
GAT	32	2	A	2	0.4412	0.6420	0.5230	0.5175	0.3867	0.9353	0.9889	0.4757
GAT	128	2	E	2	0.4502	0.6296	0.5250	0.5197	0.4154	0.9365	0.9892	0.4867
GAT	32	3	B	2	0.4500	0.6273	0.5240	0.5187	0.3866	0.9576	0.9892	0.4765
GAT	64	3	D	2	**0.4644**	0.6092	0.5270	0.5219	0.3963	0.9510	0.9897	0.4818
GAT	128	3	F	2	0.4526	0.6240	0.5246	0.5194	0.3914	0.9512	0.9893	0.4784
GCN	128	3	F	-	0.4661	**0.6082**	0.5278	0.5226	0.4510	0.9616	0.9897	0.5005
GCN	**128**	**2**	**E**	-	0.4536	0.6199	0.5239	0.5186	0.4787	0.9375	0.9893	**0.5071**
GCN	64	3	D	-	0.4655	0.6097	0.5279	0.5228	0.4516	0.9621	0.9897	0.5008
GCN	32	2	A	-	0.4406	0.6357	0.5205	0.5150	0.4767	0.9372	0.9889	0.5041
GCN	64	2	C	-	0.4328	0.6446	0.5179	0.5123	0.4771	0.9387	0.9886	0.5024
GCN	32	3	B	-	0.4532	0.6242	0.5251	0.5199	0.4504	0.9564	0.9893	0.4985
GraphSAGE	64	2	C	-	0.4567	0.6221	0.5267	0.5215	0.3679	0.9097	0.9894	0.4720
GraphSAGE	32	3	B	-	**0.3787**	**0.6865**	**0.4882**	**0.4086**	**0.4818**	0.8938	0.9863	0.4595
GraphSAGE	128	2	E	-	0.4227	0.6559	0.5141	0.5084	0.3648	0.9177	0.9883	0.4624
GraphSAGE	32	2	A	-	0.4592	0.6210	**0.5280**	**0.5228**	**0.3480**	0.8702	0.9895	0.4663
GraphSAGE	**64**	**3**	**D**	-	0.4256	0.6546	0.5158	0.5102	0.3975	0.9296	0.9884	0.4745
GraphSAGE	128	3	F	-	0.4023	0.6709	0.5030	0.4970	0.3639	0.9245	0.9875	**0.4547**

Table 5.7: Test Results of GNN Model Configurations

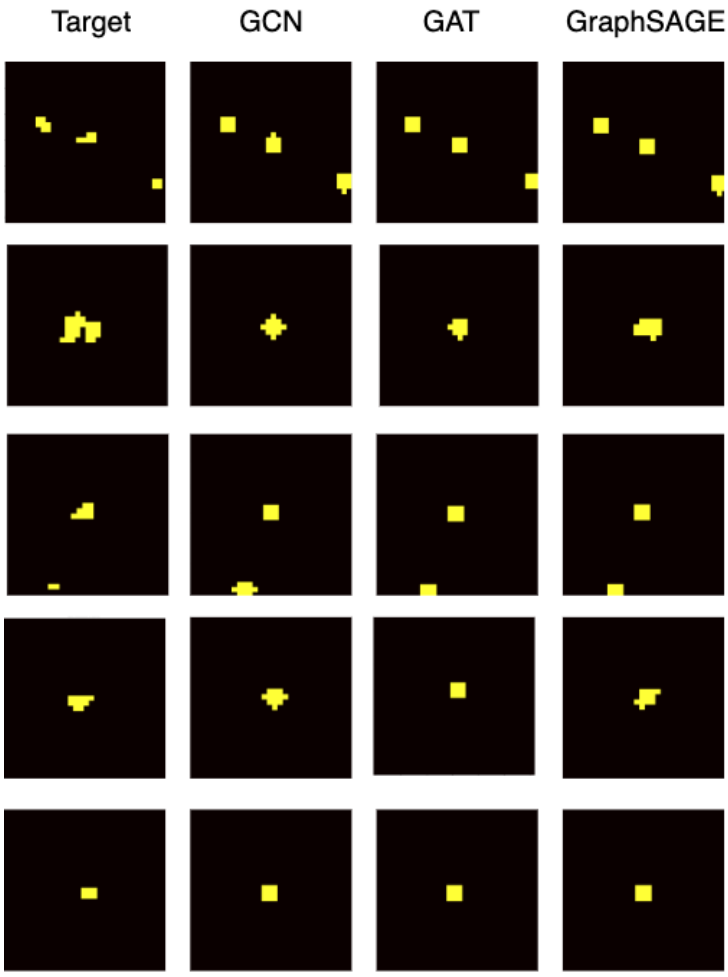
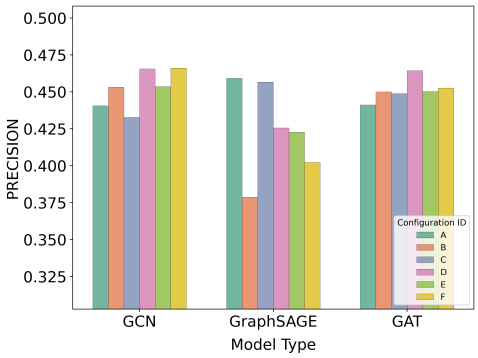


Figure 5.6: Target and Predicted Burned Areas for GNN models

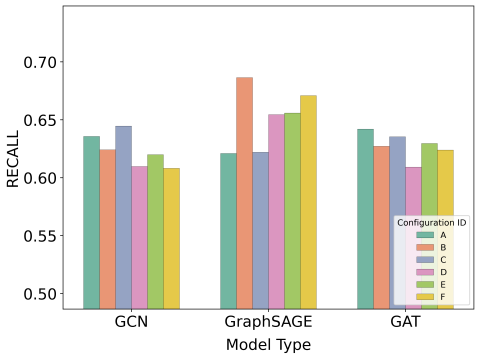
subsequently re-evaluating the model's performance. The ablation is conducted by choosing two of the best performing models from Table 5.7 based on two criteria - the highest Dice score and the highest average of the Dice, Kappa and AUPRC score.

The results of the ablation study are outlined in Table 5.9. A visual comparison of the results are shown as a set of barplots in Figure 5.8. The x-axis represents the feature category in Table 3.1 that was removed for retraining the models and the y-axis represents the score corresponding to each metric. The notations for the feature categories are defined in Table 5.8. Additionally, a visual comparison of the ablation is shown in Figure 5.9.

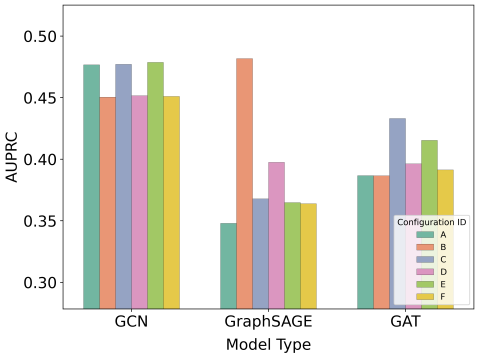
1. **Meteorological Features:** The exclusion of meteorological features results in a slight



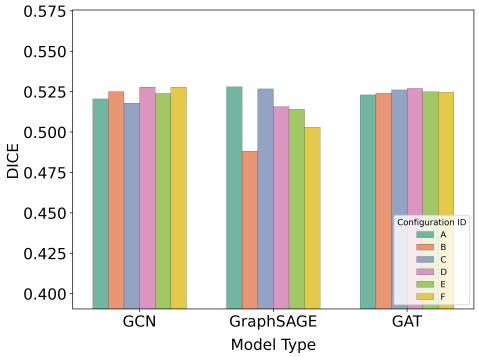
(a) Precision



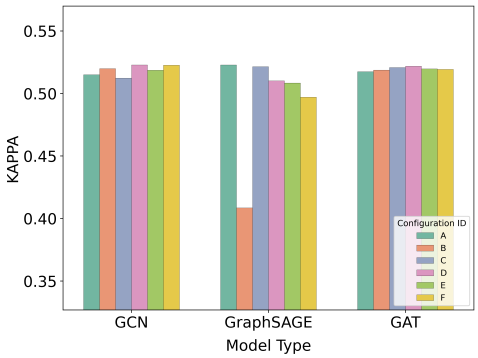
(b) Recall



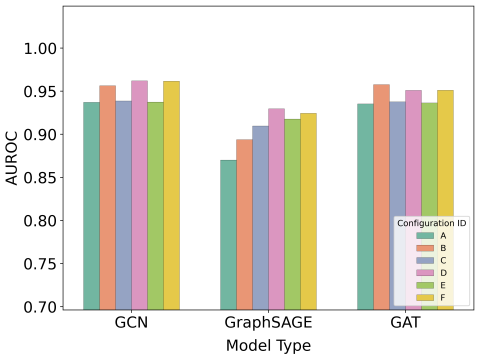
(c) AUPRC



(d) Dice Score



(e) Cohonen's Kappa



(f) AUROC

Figure 5.7: Performance Comparison of GNN Models

Feature Category	Notation
MET	Meteorological
ANT	Anthropogenic
TOP	Topographical
VEG	Vegetation
IGN	Ignition Points

Table 5.8: Notations for Feature Category

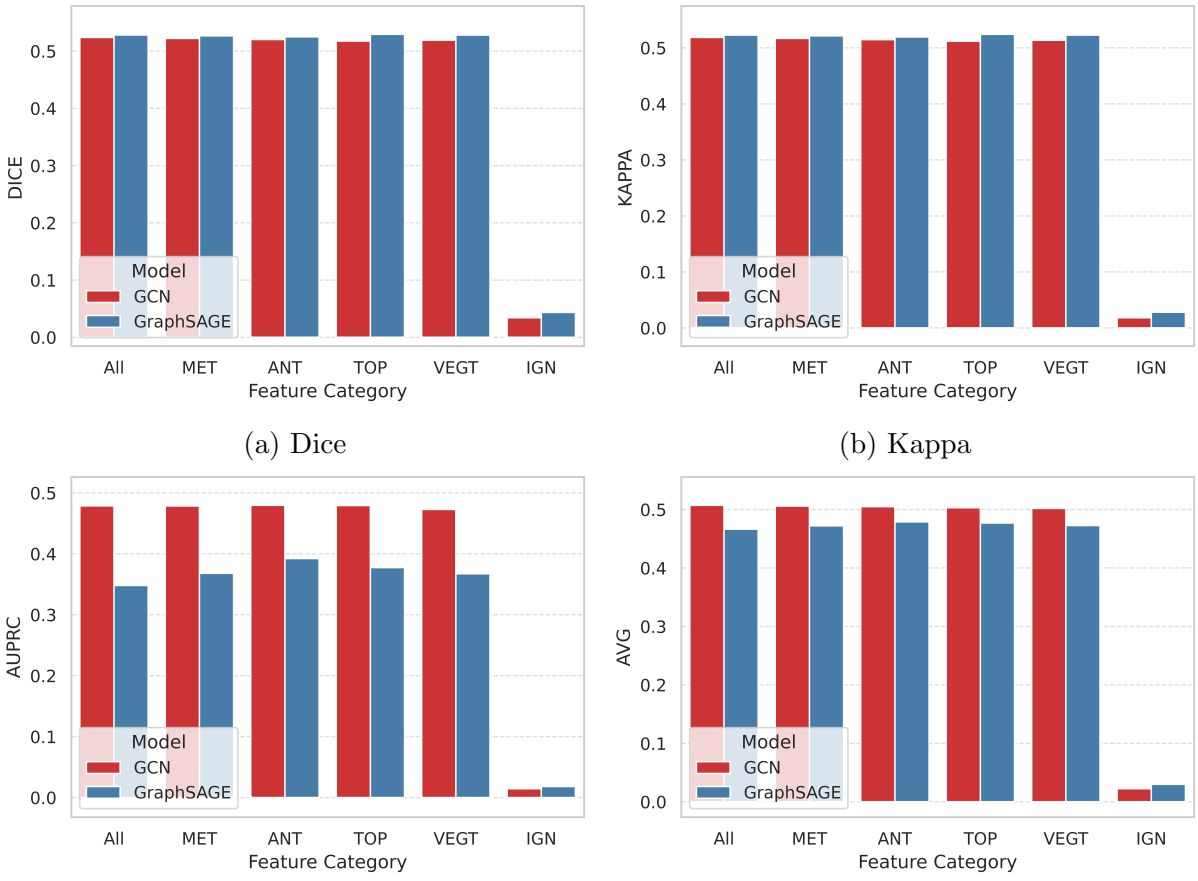
decrease in the AUPRC score for the GCN model. This indicates that GCN relies more heavily on the meteorological features for its predictions On the contrary, for the GraphSAGE model the AUPRC score increases by 0.02 units. Both the models exhibit a drop in the dice and kappa scores due to the removal of the meteorological features which indicates that meteorological features are critical to the accurate delineation of the burned areas in both models. On an average, the performance of GCN degrades slightly while GraphSAGE shows improved overall performance with the removal of meteorological features.

2. **Vegetation Features:** The removal of vegetation features affects the models in a manner similar to the meteorological features. The AUPRC and the average score both show a decline for the GCN model but a rise for the GraphSAGE model. The Dice and Kappa scores degrade slightly for both the models indicating that vegetation features have an impact on the overall delineation of burned areas.

3. **Anthropogenic Features:** The exclusion of anthropogenic features lead to a higher AUPRC in both the models but a lower Dice and Kappa score for both. The drop in AUPRC indicates that these features might contribute to noise in predicting the positive class but are important in the spatial delineation as reflected in the improved Dice and Kappa scores. On an average, the GCN model exhibits a decreased performance while GraphSAGE performs better as indicated by the increase in the average of the three metrics.

4. **Topographical Features:** The removal of topographical features results in a better AUPRC score for both the models indicating possible noise in predicting the positive classes. The Dice and Kappa scores degrade for the GCN model but exhibit an opposing trend for the GraphSAGE model in which both the scores increase with the removal of these features. On an average, the GCN still exhibits a degraded performance and GraphSAGE exhibits an improved performance.

5. **Ignition Feature:** The ignition point data is observed to be the most critical feature for all models and across all the three metrics. The metric scores plummet with the removal of the ignition point locations reaching near zero values. This indicates that the model struggles



(a) Dice

(b) Kappa

(c) AUPRC

(d) Average Score

Figure 5.8: Performance Comparison of Feature Ablations

to predict the final burned area without the foundational information of the location of the initial ignition.

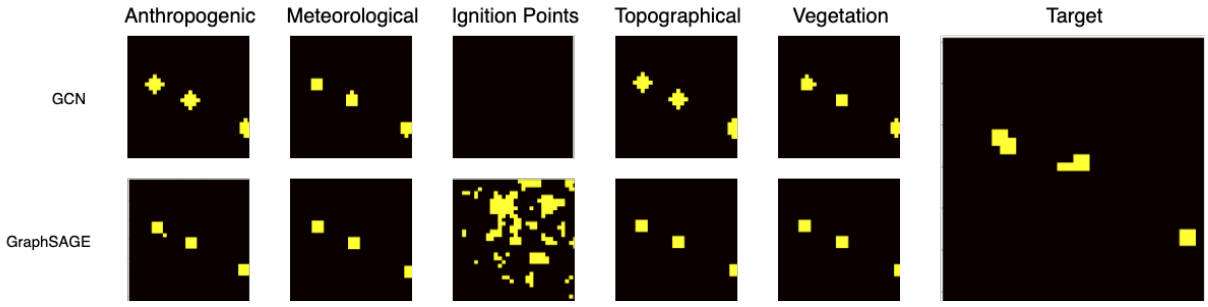


Figure 5.9: Target and Predicted Burned Areas for Feature Ablations

Feature Category	Model	Precision	Recall	Dice	Kappa	AUPRC	AUROC	Accuracy	Average
Anthropogenic	GCN	0.4385	0.6389	0.5201	0.5146	0.4795	0.9407	0.9888	0.5047
Anthropogenic	GraphSAGE	0.4480	0.6334	0.5248	0.5194	0.3923	0.9165	0.9891	0.4788
All	**GCN**	**0.4536**	**0.6199**	**0.5239**	**0.5186**	**0.4787**	**0.9375**	**0.9893**	**0.5071**
All	**GraphSAGE**	**0.4592**	**0.6210**	**0.5280**	**0.5228**	**0.3480**	**0.8702**	**0.9895**	**0.4663**
Ignition Points	GraphSAGE	0.0259	0.1353	0.0435	0.0280	0.0181	0.6511	0.9437	0.0299
Ignition Points	GCN	0.0199	0.1227	0.0342	0.0182	0.0146	0.6073	0.9345	0.0223
Meteorological	GCN	0.4435	0.6348	0.5222	0.5168	0.4785	0.9390	0.9890	0.5058
Meteorological	GraphSAGE	0.4567	0.6218	0.5266	0.5214	0.3680	0.9100	0.9894	0.4720
Topographical	GCN	0.4304	0.6489	0.5175	0.5120	0.4792	0.9428	0.9886	0.5029
Topographical	GraphSAGE	0.4652	0.6137	0.5292	0.5241	0.3774	0.9126	0.9897	0.4769
Vegetation	GCN	0.4401	0.6322	0.5189	0.5135	0.4729	0.9318	0.9889	0.5018
Vegetation	GraphSAGE	0.4662	0.6083	0.5278	0.5227	0.3672	0.8947	0.9897	0.4726

Table 5.9: Test Results for Feature Ablation

5.8 Comparison of Semantic Segmentation and Graph Neural Networks

Semantic Segmentation has been widely used in wildfire tasks to predict the wildfire progression and burned area delineation. In this section, the performance of standard Semantic Segmentation

architectures are compared to the best configuration of each GNN model type. The best GNN model configuration is selected based on the average of the AUPRC, Dice and Cohonen's Kappa score. For comparison, four semantic segmentation architectures are selected - UNet, UNet++, DeepLabV3+ and, MANet with the EfficientNet-b1 as the encoder for all the four models. All the models are trained using the Tversky Loss and hyperparameters as defined in Section 5.1. The results of the comparison are presented in Table 5.10 and a visual comparison is shown through a series of barplots in Figure 5.10

1. **Precision:** In terms of the Precision score, the majority of GNN models, excluding GraphSAGE, demonstrate superior performance compared to the segmentation models. The highest Precision score achieved is 0.4536, which is achieved by GCN. GAT closely follows with a score of 0.4488, while GraphSAGE exhibits a lower score of approximately 0.4256. Notably, the segmentation models exhibit comparable performance to GCN and GAT. MANet achieves a score of 0.4473, followed by UNet and UNet++ with scores of 0.4392 and 0.4388, respectively. DeepLabV3+ demonstrates the lowest performance with a score of 0.0332.

2. **Recall:** The Recall shows a clear trend with all the segmentation models outperforming the GNN models. The highest scores are achieved by the UNet variants, with UNet achieving 0.6698 and UNet++ scoring 0.6684, followed by DeepLabV3+ and MANet with scores of 0.6601 and 0.6599, respectively. Although the Recall is lower for all GNN models, the difference is marginal, with GraphSAGE achieving 0.6545, GAT achieving 0.6354,and GCN achieving 0.6199.

3. **Dice Score:** The segmentation architectures, except DeepLabV3+, are observed to outperform the GNN models in terms of the Dice score. The MANet model achieves the highest Dice score of 0.5332 and is closely followed by the UNet and UNet ++ models with scores of 0.5303 and 0.5301, respectively. Notably, the DeepLabV3+ exhibits the worst performance with a Dice score of 0.4429. Although the majority of segmentation models perform better than the GNN models, the disparity in the scores is marginal, with the GAT model achieving a score of 0.5261, followed by GCN with 0.5239, and GraphSAGE achieving 0.5158.

4. **Kappa Score:** The Kappa score mirrors the trends observed in the Dice Score, with MANet outperforming all other models with a score of 0.5279 and is followed by the UNet and UNet++ models with scores of 0.5248 and 0.5247, respectively. The GNN models slightly underperform but are only marginally worse, with GAT achieving 0.5208, followed by GCN with a score of 0.5186 and GraphSAGE with 0.5102. Similar to its performance in terms of the Dice score, DeepLabV3+ performs the worst with a score of 0.4358.

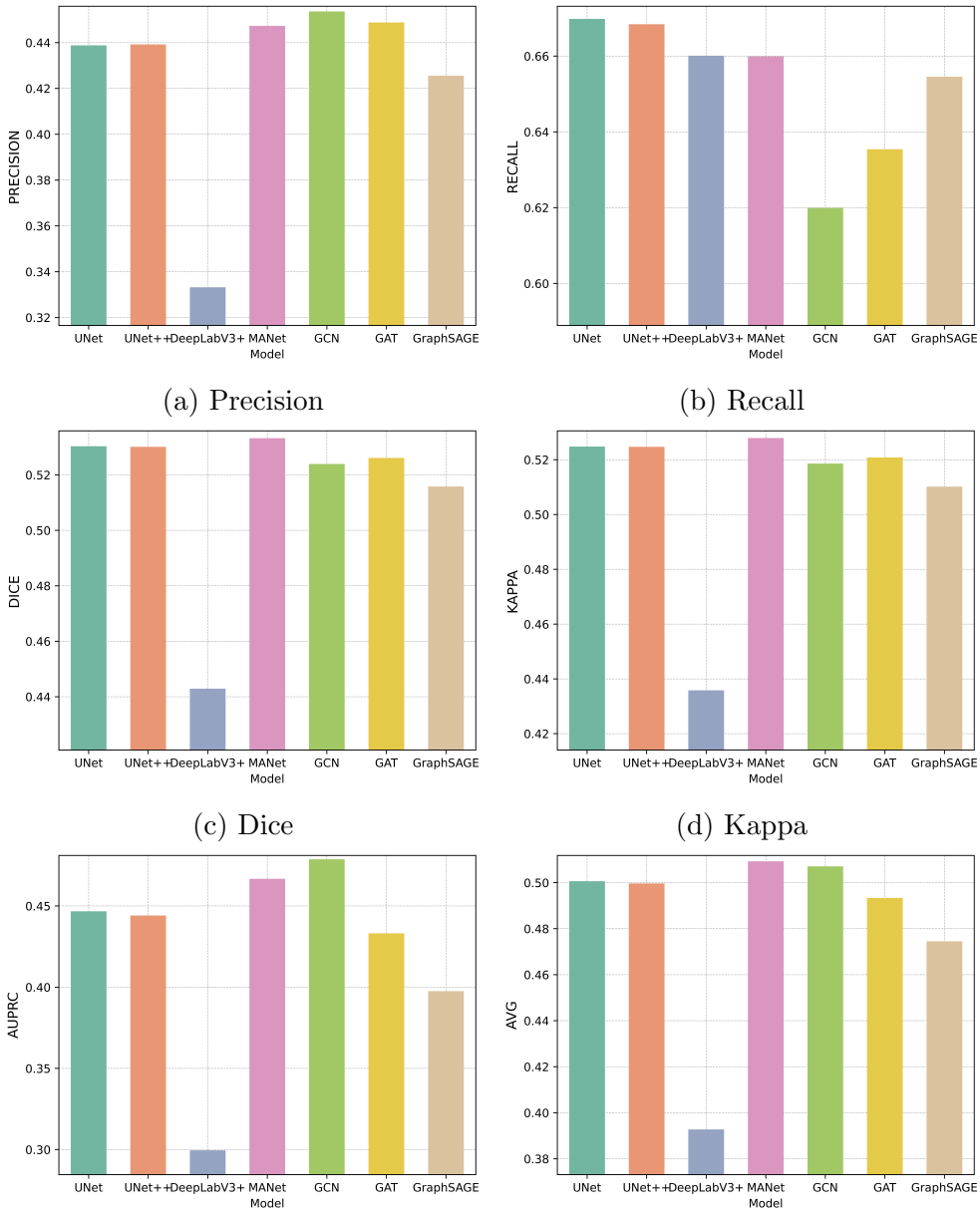
5. **AUPRC:** The AUPRC score shows a variable trend with the GCN model outperforming all models with 0.4787, including the segmentation models, but GAT and GraphSAGE still fall

behind a majority of the segmentation architectures. The MANet model performs the best among the segmentation architectures with a score of 0.4667, while UNet and UNet++ exhibit similar performance with scores of 0.4467 and 0.4441, respectively. While GAT achieves comparable performance to both UNet variants with a score of 0.4331, GraphSAGE underperforms considerably with a score of 0.3975. DeepLabV3+ continues to show poor performance with a low score of 0.2995. All the GNN models, except GraphSAGE, further outperform the UNet model proposed in [49] in terms of the AUPRC score however it should be noted that the training conditions and size of the dataset vary.

6. **Average:** On average, the segmentation models outperform the GNN models, with the exception of GCN, which exhibits a marginal advantage. The MANet model achieves the highest score of 0.5093, followed closely by GCN with 0.5071. UNet obtains a score of 0.5006, while UNet++ and GAT achieve similar scores of 0.499633 and 0.4933, respectively. GraphSAGE performs the least well with a score of 0.4745. The DeepLabV3+ model exhibits the most significant performance degradation with a score of 0.3927.

Model	**Precision**	**Recall**	**Dice**	**Kappa**	**AUPRC**	**AUROC**	**Accuracy**	**Average**
DeepLabV3+	0.3332	0.6601	0.4429	0.4358	0.2995	0.9208	0.9843	0.3927
GAT	0.4488	0.6354	0.5261	0.5208	0.4331	0.9377	0.9891	0.4933
GCN	0.4536	0.6199	0.5239	0.5186	0.4787	0.9375	0.9893	0.5071
GraphSAGE	0.4256	0.6546	0.5158	0.5102	0.3975	0.9296	0.9884	0.4745
MANet	0.4473	0.6599	0.5332	0.5279	0.4667	0.8869	0.9891	0.5093
UNet	0.4388	0.6698	0.5303	0.5248	0.4467	0.8830	0.9888	0.5006
UNet++	0.4392	0.6684	0.5301	0.5247	0.4441	0.8522	0.9888	0.4996
UNet [49]	-	-	-	-	0.418	-	-	-

Table 5.10: Results of Semantic Segmentation and GNN Comparison



(a) Precision

(b) Recall

(c) Dice

(d) Kappa

(e) AUPRC

(f) Average Score

Figure 5.10: Performance Comparison of GNN & Segmentation Models

Chapter 6

Conclusion & Future Work

This chapter presents a concise overview of the proposed methodology for predicting the final burned area resulting from wildfire ignitions using Graph Neural Networks. It details the results obtained from this approach and provides an in-depth discussion of the key findings and their implications. Additionally, the challenges encountered during the research are analyzed, along with potential directions for future work.

6.1 Conclusion

In this research, a graph-based data-driven approach was explored to accurately predict the final burned area resulting from an ignition. Graph structures provide the flexibility to define the relationships between different spatial locations while considering both long-term and short-term dependencies. A graph transformation framework was developed to effectively restructure spatial wildfire data into graphs. To find the optimal architecture for an imbalanced dataset, several graph neural network (GNN) architectures were designed and trained. A prominent challenge in wildfire datasets is the imbalance in class distributions of the burned and unburned classes. To address this issue, several loss functions were explored. It was found that weighted loss functions considerably improved the performance of the models, both in terms of achieving training convergence and predictive capabilities. Overall, Tversky loss proved to be the most stable and effective loss function with weighted BCE following closely. The intermodel comparison between the GNN architectures further helped in identifying the most suitable architecture. In general, GCN proved to be the most stable and consistent while GraphSAGE showcased variable performance but achieved high performances with careful tuning. However, the attention mechanism in GAT did not prove to be as useful for the dataset, and its performance remained moderately effective overall. In terms of overall performance, GNNs were comparable to the performance of

the segmentation models. GNNs exhibited strong potential for modeling wildfires with most GNN models surpassing segmentation models in terms of AUPRC, AUROC and Precision. For other metrics, although segmentation models showed better performance, the differences were minimal. Considering the reduced training convergence time and number of parameters in GNNs compared to CNN-based segmentation architectures, GNNs exhibited strong potential for modeling wildfires.

6.2 Future Work

This research demonstrated the efficacy of representing spatial wildfire data as graph structures and exhibited that Graph Neural Networks can effectively predict the burned area resulting from an ignition. However, several potential avenues exist to improve upon the proposed approach which has been discussed in this section.

1. **Inclusion of Temporal Information:** The dataset considered for this thesis included information regarding the location of the initial ignition while excluding any information regarding the intermediate progression of the wildfire between the day of wildfire ignition and the end of the wildfire, resulting in the final burned area. Inclusion of intermediate fire progression data can potentially improve the prediction accuracy by allowing the models to learn the complex patterns better. Additionally, including temporal information prior to the day of ignition can be potentially useful in identifying conditions that lead to the ignition and the subsequent burned area resulting from it. This potential avenue leads to another potential improvement where sequence models can be employed in conjunction with graph neural networks to capture both the temporal and spatial information and forecast burned area several days in advance.

2. **Domain-informed Graph Structures:** In the proposed research, the graphs are modelled based on the proximity of nodes , wherein edges are established between neighbouring nodes. While effective for capturing burned area patterns due to its strong dependence on physical connections, it has room for improvement by including additional information such as edge features based on the underlying factors that contribute to short-term wildfire spread, such as wind speed, slopes etc. Thus, defining the graph structure based on domain-informed heuristics has tremendous potential for improving the predictive performance of Graph Neural Networks for wildfire burned area prediction.

3. **Learnable Graph Structures:** In most existing graph-based approaches in wildfire-related tasks, the intermediate graph dataset resulting from the original dataset is designed manually based on some deterministic heuristic mandating human intervention which may not generalize well to diverse geographical regions. In order to capture more complex

patterns inherent in the data, especially with regard to different geographical locations, a potential avenue is to incorporate learning-based approaches for transforming the dataset to a graph-structured dataset, eliminating the intermediate step for designing the graph topology and allowing a deep learning model to design the optimal graph structure for accurate predictions.

References

[1] Mohamad Alipour, Inga Puma, Joshua Picotte, Kasra Shamsaei, Eric Rowell, Adam Watts, Branko Kosovic, Hamed Ebrahimian, and Ertugrul Taciroglu. A multimodal data fusion and deep learning framework for large-scale wildfire surface fuel mapping. *Fire*, 6:36, 01 2023.

[2] Rodolfo Maduro Almeida and Elbert EN Macau. Stochastic cellular automata model for wildland fire spread dynamics. In *Journal of Physics: Conference Series*, volume 285, page 012038. IOP Publishing, 2011.

[3] Uri Alon and Eran Yahav. On the bottleneck of graph neural networks and its practical implications. In *International Conference on Learning Representations*, 2021.

[4] Enric Batllori, Marc-André Parisien, Meg A Krawchuk, and Max A Moritz. Climate change-induced shifts in fire for m editerranean ecosystems. *Global Ecology and Biogeography*, 22(10):1118–1129, 2013.

[5] Lilian Berton, Pedro Oliveira, and Didier Vega Oliveros. Graph-based semi-supervised classification for similar wildfire dynamics. In *Proceedings of the 38th ACM/SIGAPP Symposium on Applied Computing*, SAC '23, page 499–506, New York, NY, USA, 2023. Association for Computing Machinery.

[6] Mitchell Black, Zhengchao Wan, Amir Nayyeri, and Yusu Wang. Understanding over-squashing in gnns through the lens of effective resistance. In *International Conference on Machine Learning*, pages 2528–2547. PMLR, 2023.

[7] Karol Bot and José G Borges. A systematic review of applications of machine learning techniques for wildfire management decision support. *Inventions*, 7(1):15, 2022.

[8] Michael L. Cain, William D. Bowman, and Sally D. Hacker. Terrestrial biomes. Nature Education Knowledge. Accessed on January 06, 2025.

[9] Dayou Chen, Sibo Cheng, Jinwei Hu, Matthew Kasoar, and R. Arcucci. Explainable global wildfire prediction models using graph neural networks, 02 2024.

[10] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PP, 06 2016.

[11] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *ArXiv*, abs/1706.05587, 2017.

[12] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *European Conference on Computer Vision*, 2018.

[13] Xiwen Chen, Bryce Hopkins, Hao Wang, Leo O'Neill, Fatemeh Afghah, Abolfazl Razi, Peter Fulé, Janice Coen, Eric Rowell, and Adam Watts. Wildland fire detection and monitoring using a drone-collected rgb/ir image dataset. *IEEE Access*, 10:121301–121317, 2022.

[14] Chia-Yen Chiang, Chloe Barnes, Plamen Angelov, and Richard Jiang. Deep learning-based automated forest health diagnosis from aerial images. *IEEE Access*, 8:144064–144076, 2020.

[15] Shane Coffield, Casey Graff, Yang Chen, Padhraic Smyth, Efi Foufoula-Georgiou, and James Randerson. Machine learning to predict final fire size at the time of ignition. *International Journal of Wildland Fire*, 28, 09 2019.

[16] Gabriela Csurka, Riccardo Volpi, Boris Chidlovskii, et al. Semantic image segmentation: Two decades of research. *Foundations and Trends® in Computer Graphics and Vision*, 14(1-2):1–162, 2022.

[17] Michaël Defferrard, Xavier Bresson, and Pierre Vandergheynst. Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29, 2016.

[18] Elizar Elizar, Mohd Asyraf Zulkifley, Rusdha Muharar, Mohd Hairi Mohd Zaman, and Seri Mustaza. A review on multiscale-deep-learning applications. *Sensors*, 22:7384, 09 2022.

[19] Margaret Epstein, Carl Seielstad, and Chris Moran. Impact and recovery of forest cover following wildfire in the northern rocky mountains of the united states. *Fire Ecology*, 20, 06 2024.

[20] Tongle Fan, Guanglei Wang, Yan Li, and Hongrui Wang. Ma-net: A multi-scale attention network for liver and tumor segmentation. *IEEE Access*, 8:179656–179665, 2020.

[21] Evan N. Feinberg, Debnil Sur, Zhenqin Wu, Brooke E. Husic, Huanghao Mai, Yang Li, Saisai Sun, Jianyi Yang, Bharath Ramsundar, and Vijay S. Pande. Potentialnet for molecular property prediction. *ACS Central Science*, 4:1520 – 1530, 2018.

[22] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[23] Mark A Finney. *FARSITE, Fire Area Simulator–model development and evaluation*. Number 4. US Department of Agriculture, Forest Service, Rocky Mountain Research Station, 1998.

[24] Xu Geng, Yaguang Li, Leye Wang, Lingyu Zhang, Qiang Yang, Jieping Ye, and Yan Liu. Spatiotemporal multi-graph convolution network for ride-hailing demand forecasting. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 3656–3663, 2019.

[25] Sebastian Gerard, Yu Zhao, and Josephine Sullivan. Wildfirespreadts: a dataset of multimodal time series for wildfire spread prediction. In *Proceedings of the 37th International Conference on Neural Information Processing Systems*, NIPS '23, Red Hook, NY, USA, 2024. Curran Associates Inc.

[26] Rafik Ghali, Moulay Akhloufi, Marwa JMAL, Wided Mseddi, and Rabah Attia. remote sensing wildfire segmentation using deep vision transformers. *Remote Sensing*, 13:3527, 09 2021.

[27] Rafik Ghali, Moulay Akhloufi, and Wided Mseddi. Deep learning and transformer approaches for uav-based wildfire detection and segmentation. *Sensors*, 22, 03 2022.

[28] Katarzyna Grala, Robert K Grala, Anwar Hussain, William H Cooke III, and J Morgan Varner III. Impact of human factors on wildfire occurrence in mississippi, united states. *Forest policy and economics*, 81:38–47, 2017.

[29] Dale Hamilton, Enoch Levandovsky, and Nicholas Hamilton. Mapping burn extent of large wildland fires from satellite imagery using machine learning trained from localized hyperspatial imagery. *Remote Sensing*, 12:4097, 12 2020.

[30] William Hamilton. Graph representation learning. *Synthesis Lectures on Artificial Intelligence and Machine Learning*, 14:1–159, 09 2020.

[31] William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 1025–1035, Red Hook, NY, USA, 2017. Curran Associates Inc.

[32] Haibo He and Edwardo A. Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[33] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[34] David Hoaglin. John w. tukey and data analysis. *Statistical Science*, 18, 08 2003.

[35] Jonathan Hodges and Brian Lattimer. Wildland fire spread modeling using convolutional neural networks. *Fire Technology*, 55, 03 2019.

[36] Fantine Huot, R. Lily Hu, Nita Goyal, Tharun Sankar, Matthias Ihme, and Yi-Fan Chen. Next day wildfire spread: A machine learning dataset to predict wildfire spreading from remote-sensing data. *IEEE Transactions on Geoscience and Remote Sensing*, 60:1–13, 2022.

[37] Fantine Huot, R Lily Hu, Matthias Ihme, Qing Wang, John Burge, Tianjian Lu, Jason Hickey, Yi-Fan Chen, and John Anderson. Deep learning models for predicting wildfires from historical remote-sensing data. *arXiv preprint arXiv:2010.07445*, 2020.

[38] Pavel Iakubovskii. Segmentation models pytorch. https://github.com/qubvel/segmentation_models.pytorch, 2019.

[39] Muzaffer Iban and Aliihsan Sekertekin. Machine learning based wildfire susceptibility mapping using remotely sensed fire data and gis: A case study of adana and mersin provinces, turkey. *Ecological Informatics*, 69, 07 2022.

[40] Shruti Jadon. A survey of loss functions for semantic segmentation. In *2020 IEEE conference on computational intelligence in bioinformatics and computational biology (CIBCB)*, pages 1–7. IEEE, 2020.

[41] Piyush Jain, Sean Coogan, Sriram Subramanian, Mark Crowley, S.W. Taylor, and Mike Flannigan. A review of machine learning applications in wildfire science and management. *Environmental Reviews*, 07 2020.

[42] Kanchan Jha, Sriparna Saha, and Hiteshi Singh. Prediction of protein–protein interaction using graph neural networks. *Scientific Reports*, 12(1):8360, 2022.

[43] Guangyin Jin, Yuxuan Liang, Yuchen Fang, Zezhi Shao, Jincai Huang, Junbo Zhang, and Yu Zheng. Spatio-temporal graph neural networks for predictive learning in urban computing: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 2023.

[44] Ming Jin, Huan Yee Koh, Qingsong Wen, Daniele Zambon, Cesare Alippi, Geoffrey I Webb, Irwin King, and Shirui Pan. A survey on graph neural networks for time series: Forecasting, classification, imputation, and anomaly detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2024.

[45] Sadegh Khanmohammadi, Associate Prof. Mehrdad Arashpour, Emad Golafshani, Miguel Cruz, Abbas Rajabifard, and Yu Bai. Prediction of wildfire rate of spread in grasslands using machine learning methods. *Environmental Modelling Software*, 156:105507, 08 2022.

[46] Fadoua Khennou and Moulay A Akhloufi. Improving wildland fire spread prediction using deep u-nets. *Science of Remote Sensing*, 8:100101, 2023.

[47] Jinah Kim, Taekyung Kim, Joon-Gyu Ryu, and Jaeil Kim. Spatiotemporal graph neural network for multivariate multi-step ahead time-series forecasting of sea temperature. *Engineering Applications of Artificial Intelligence*, 126:106854, 2023.

[48] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations (ICLR)*, 2017.

[49] Spyridon Kondylatos, Ioannis Prapas, Gustau Camps-Valls, and Ioannis Papoutsis. Mesogeos: A multi-purpose dataset for data-driven wildfire modeling in the mediterranean. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 50661–50676. Curran Associates, Inc., 2023.

[50] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25, 2012.

[51] Remi Lam, Alvaro Sanchez-Gonzalez, Matthew Willson, Peter Wirnsberger, Meire Fortunato, Alexander Pritzel, Suman Ravuri, Timo Ewalds, Ferran Alet, Zach Eaton-Rosen, Weihua Hu, Alexander Merose, Stephan Hoyer, George Holland, Jacklynn Stott, Oriol Vinyals, Shakir Mohamed, and Peter Battaglia. Graphcast: Learning skillful medium-range global weather forecasting, 12 2022.

[52] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[53] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327, 2020.

[54] Xiang Ling, Lingfei Wu, Chunming Wu, and Shouling Ji. Graph neural networks: Graph matching. In Lingfei Wu, Peng Cui, Jian Pei, and Liang Zhao, editors, *Graph Neural Networks: Foundations, Frontiers, and Applications*, pages 277–295. Springer Singapore, Singapore, 2022.

[55] Jun Ma, Jianan Chen, Matthew Ng, Rui Huang, Yu Li, Chen Li, Xiaoping Yang, and Anne L Martel. Loss odyssey in medical image segmentation. *Medical Image Analysis*, 71:102035, 2021.

[56] Shervin Minaee, Yuri Boykov, Fatih Porikli, Antonio Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(7):3523–3542, 2022.

[57] Francisco Moreira, Davide Ascoli, Hugh Safford, Mark A Adams, José M Moreno, José MC Pereira, Filipe X Catry, Juan Armesto, William Bond, Mauro E González, et al. Wildfire management in mediterranean-type regions: paradigm change needed. *Environmental Research Letters*, 15(1):011001, 2020.

[58] Yunjie Mu, Liyuan Ou, Wenjing Chen, Tao Liu, and Demin Gao. Superpixel-based graph convolutional network for uav forest fire image segmentation. *Drones*, 8(4):142, 2024.

[59] Dominik Müller, Iñaki Soto-Rey, and Frank Kramer. Towards a guideline for evaluation metrics in medical image segmentation. *BMC Research Notes*, 15(1):210, 2022.

[60] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[61] Sandra Oliveira, Francisco Moreira, Roberto Boca, J. San-Miguel-Ayanz, and José Pereira. Assessment of fire selectivity in relation to land cover and topography. a comparison between southern european countries. *International Journal of Wildland Fire*, 23:620–630, 01 2013.

[62] GLW Perry. Current approaches to modelling the spread of wildland fire: a review. *Progress in Physical Geography*, 22(2):222–245, 1998.

[63] Ioannis Prapas, Akanksha Ahuja, Spyros Kondylatos, Ilektra Karasante, Lazaro Alonso, Eleanna Panagiotou, Charalampos Davalas, Dimitrios Michail, Nuno Carvalhais, and Ioannis Papoutsis. Deep learning for global wildfire forecasting. In *NeurIPS 2022 Workshop on Tackling Climate Change with Machine Learning*, 2022.

[64] Ioannis Prapas, Akanksha Ahuja, Spyros Kondylatos, Ilektra Karasante, Eleanna Panagiotou, Lazaro Alonso, Charalampos Davalas, Dimitrios Michail, Nuno Carvalhais, and Ioannis Papoutsis. Deep learning for global wildfire forecasting. *arXiv preprint arXiv:2211.00534*, 2022.

[65] Ioannis Prapas, Nikolaos-Ioannis Bountos, Spyros Kondylatos, Dimitrios Michail, Gustau Camps-Valls, and Ioannis Papoutsis. Televit: Teleconnection-driven transformers improve subseasonal to seasonal wildfire forecasting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3754–3759, 2023.

[66] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[67] T Konstantin Rusch, Michael M Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks. *arXiv preprint arXiv:2303.10993*, 2023.

[68] Moritz Rösch, Michael Nolde, Tobias Ullmann, and Torsten Riedlinger. Data-driven wildfire spread modeling of european wildfires using a spatiotemporal graph neural network. *Fire*, 7:207, 06 2024.

[69] Salva Rühling Cachay, Emma Erickson, Arthur F C Bucker, Ernest J Pokropek, Willa Potosnak, Salomey Osei, and Björn Lütjens. Graph neural networks for improved el niño forecasting. In *NeurIPS 2020 Workshop on Tackling Climate Change with Machine Learning*, 2020.

[70] Manisha Saini and Seba Susan. Tackling class imbalance in computer vision: a contemporary review. *Artificial Intelligence Review*, 56:1–57, 07 2023.

[71] Takaya Saito and Marc Rehmsmeier. The precision-recall plot is more informative than the roc plot when evaluating binary classifiers on imbalanced datasets. *PLoS ONE*, 10, 2015.

[72] Seyed Sadegh Mohseni Salehi, Deniz Erdogmus, and Ali Gholipour. Tversky loss function for image segmentation using 3d fully convolutional deep networks. In *International workshop on machine learning in medical imaging*, pages 379–387. Springer, 2017.

[73] Dmitry Shadrin, Svetlana Illarionova, Fedor Gubanov, Ksenia Evteeva, Maksim Mironenko, Ivan Levchunets, Roman Belousov, and Evgeny Burnaev. Wildfire spreading prediction using multimodal data and deep neural network approach. *Scientific Reports*, 14, 01 2024.

[74] Alireza Shamsoshoara, Fatemeh Afghah, Abolfazl Razi, Liming Zheng, Peter Z Fulé, and Erik Blasch. Aerial imagery pile burn detection using deep learning: The flame dataset. *Computer Networks*, 193:108001, 2021.

[75] Samriddhi Singla, Ayan Mukhopadhyay, Michael Wilbur, Tina Diao, Vinayak Gajjewar, Ahmed Eldawy, Mykel Kochenderfer, Ross Shachter, and Abhishek Dubey. Wildfiredb: An open-source dataset connecting wildfire occurrence with relevant determinants. In *NeurIPS Thirty-fifth Annual Conference on Neural Information Processing Systems*, 2021.

[76] Adam J Stewart, Caleb Robinson, Isaac A Corley, Anthony Ortiz, Juan M Lavista Ferres, and Arindam Banerjee. Torchgeo: deep learning with geospatial data. In *Proceedings of the 30th international conference on advances in geographic information systems*, pages 1–12, 2022.

[77] Carole H Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M Jorge Cardoso. Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: Third International Workshop, DLMIA 2017, and 7th International Workshop, ML-CDS 2017, Held in Conjunction with MICCAI 2017, Québec City, QC, Canada, September 14, Proceedings 3*, pages 240–248. Springer, 2017.

[78] Jingjing Sun, Wenwen Qi, Yuandong Huang, Chong Xu, and Wentao Yang. Facing the wildfire spread risk challenge: Where are we now and where are we going? *Fire*, 6:228, 06 2023.

[79] Fantina Tedim, Gavriil Xanthopoulos, and Vittorio Leone. Forest fires in europe: Facts and challenges. In *Wildfire hazards, risks and disasters*, pages 77–99. Elsevier, 2015.

[80] Mengfan Teng, Siwei Li, Jia Xing, Chunying Fan, Jie Yang, Shuo Wang, Ge Song, Yu Ding, Jiaxin Dong, and Shansi Wang. 72-hour real-time forecasting of ambient pm2.5 by hybrid graph deep neural network with aggregated neighborhood spatiotemporal information. *Environment International*, 176:107971, 05 2023.

[81] Tom Toulouse, Lucile Rossi, Antoine Campana, Turgay Celik, and Moulay A Akhloufi. Computer vision for wildfire research: An evolving image dataset for processing and analysis. *Fire Safety Journal*, 92:188–194, 2017.

[82] Cordy Tymstra, RW Bryce, BM Wotton, SW Taylor, OB Armitage, et al. Development and structure of prometheus: the canadian wildland fire growth simulation model. *Natural Resources Canada, Canadian Forest Service, Northern Forestry Centre, Information Report NOR-X-417.(Edmonton, AB)*, 2010.

[83] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. *International Conference on Learning Representations*, 2018.

[84] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and Philip S. Yu. A comprehensive survey on graph neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 32(1):4–24, 2021.

[85] Ying Xie and Minggang Peng. Forest fire forecasting using ensemble learning approaches. *Neural Computing and Applications*, 31(9):4541–4550, 2019.

[86] Fang Yang, Kunjie Fan, Dandan Song, and Huakang Lin. Graph-based prediction of protein-protein interactions with attributed signed graph embedding. *BMC bioinformatics*, 21:1–16, 2020.

[87] Jiayun Angela Yao, Sean Raffuse, Michael Brauer, Grant Williamson, David Bowman, Fay Johnston, and Sarah Henderson. Predicting the minimum height of forest fire smoke within the atmosphere using machine learning and data from the calipso satellite. *Remote Sensing of Environment*, 206:98–106, 03 2018.

[88] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: a deep learning framework for traffic forecasting. In *Proceedings of the 27th International Joint Conference on Artificial Intelligence*, IJCAI'18, page 3634–3640. AAAI Press, 2018.

[89] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. 05 2016.

[90] Xiaohui Yuan, Jianfang Shi, and Lichuan Gu. A review of deep learning methods for semantic segmentation of remote sensing imagery. *Expert Systems with Applications*, 169:114417, 2021.

[91] Jun Zhai, Zhuo Ning, Ram Dahal, and Shaoyang Yang. Wildfire susceptibility of land use and topographic features in the western united states: Implications for the landscape management. *Forests*, 14:807, 04 2023.

[92] Xiaojian Zhang, Xilei Zhao, Yiming Xu, Daniel Nilsson, and Ruggiero Lovreglio. Situational-aware multi-graph convolutional recurrent network (sa-mgcrn) for travel demand forecasting during wildfires. *Transportation Research Part A: Policy and Practice*, 190:104242, 2024.

[93] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239, 2016.

[94] Shan Zhao, Zhaiyu Chen, Zhitong Xiong, Yilei Shi, Sudipan Saha, and Xiao Xiang Zhu. Beyond grid data: Exploring graph neural networks for earth observation. *IEEE Geoscience and Remote Sensing Magazine*, pages 2–35, 2024.

[95] Shan Zhao, Ioannis Prapas, Ilektra Karasante, Zhitong Xiong, Ioannis Papoutsis, Gustau Camps-Valls, and Xiao Xiang Zhu. Causal graph neural networks for wildfire danger prediction. *arXiv preprint arXiv:2403.08414*, 2024.

[96] Jie Zhou, Ganqu Cui, Shengding Hu, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. Graph neural networks: A review of methods and applications. *AI open*, 1:57–81, 2020.

[97] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. Graph neural networks: A review of methods and applications. *ArXiv*, abs/1812.08434, 2018.

[98] Wen Zhou, Chen Zhang, and Siyuan Chen. Dual deep q-learning network guiding a multiagent path planning approach for virtual fire emergency scenarios. *Applied Intelligence*, 53, 06 2023.

[99] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. Unet++: A nested u-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018,*

Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4, pages 3–11. Springer, 2018.

[100] Yufei Zou, Susan O'Neill, Narasimhan Larkin, Ernesto Alvarado, Robert Solomon, Clifford Mass, Yang Liu, Mehmet Odman, and Huizhong Shen. Machine learning-based integration of high-resolution wildfire smoke simulations and observations for regional health impact assessment. *International Journal of Environmental Research and Public Health*, 16:2137, 06 2019.