Geospatial Analysis for Wildfire Detection, Fire Spread Prediction and Risk Assessment in Natural Landscapes

by

Mahinul Haque
21301739
Mst. Khadizatul Kobra
24141070
Manali Chakma
24241148
Wilson Plabon Botlero
20341002

A thesis submitted to the Department of Computer Science and Engineering
in partial fulfillment of the requirements for the degree of
B.Sc. in Computer Science

Department of Computer Science and Engineering
BRAC University
October 2025

Declaration

It is hereby declared that

1. The thesis submitted is my/our own original work while completing degree at Brac University.
2. The thesis does not contain material previously published or written by a third party, except where this is appropriately cited through full and accurate referencing.
3. The thesis does not contain material which has been accepted, or submitted, for any other degree or diploma at a university or other institution.
4. We have acknowledged all main sources of help.

Student's Full Name & Signature:

Mst. Khadizatul Kobra
24141070

Manali Chakma
24241148

Mahinul Haque
21301739

Wilson Plabon Botlero
20341002

Approval

This thesis/project titled "Geospatial Analysis for Wildfire Detection, Fire Spread Prediction and Risk Assessment in Natural Landscapes" submitted by

1. Mst. Khadizatul Kobra (24141070)
2. Manali Chakma (24241148)
3. Mahinul Haque (21301739)
4. Wilson Plabon Botlero (20341002)

As Of Summer, 2025 has been accepted as satisfactory in partial fulfillment of the requirement for the degree of B.Sc. in Computer Science and Engineering on October 14, 2025.

Examining Committee:

Supervisor:
(Member)

Dr. Amitabha Chakrabarty
Professor
Department of Computer Science and Engineering
Brac University

Program Coordinator:
(Member)

Md. Golam Rabiul Alam

Professor
Department of Computer Science and Engineering
Brac University

Head of Department:
(Chair)

Dr. Sadia Hamid Kazi

Associate Professor and Chairperson
Department of Computer Science and Engineering
Brac University

Ethics Statement

This research paper is done by us and is plagiarism free.

Abstract

Wildfires, are one of the most devastating natural-disasters, threatening ecosystems, human life, and infrastructure. Aerial-based or ground sensor-based conventional monitoring systems are subject to low resolution, high cost, and weak response times. With the introduction of unmanned aerial vehicles (UAVs) and low-weight edge single-board computer such as the Raspberry Pi, real-time wildfire detection in the field is now possible. This research presents a lightweight deep learning robust model (W-fire MobilenetV4 Small) for classifying fire, further segmenting fire pixels, and assessing risk built upon the FLAME dataset for UAV deployment. The model is a customization of MobileNetV4 conv small architecture, and for further segmentation, the same classification architecture was kept as an encoder while decoding in FPN style with depth-wise separable fusion blocks for efficiency. Both for the classification and segmentation tasks, keeping the model lightweight while efficient was the priority. Moreover, this was achieved with the help of a custom flame block, SE Attention, ECA, Sandglass block & DANet in the MobileNetV4 small architecture. Additionally, keeping the customized architecture as encoder FPN-style decoding was done for the segmentation task. The model is merged to fight against image distortions like haze, smoke, blur, night vision, cloud, noise, and other atmospheric conditions that drone images might face. This ultimately makes the model robust for different conditions. For the Segmentation task, the dataset had a huge class imbalance as fire regions were very small To effectively address the extreme class imbalance (fire-pixel ratio ≈ 0.000026), a novel Balanced Focal-Dice Loss is incorporated. Most significantly, on top of pure detection, the system also provides severity ranking of fire and propagation direction prediction, transforming raw segmentation outputs into vital information for firefighting missions. Our Novel model, W-Fire_MobileNetV4_Small achieved 79.32% accuracy, 79.33% F1-score, 79.33% precision, and 79.32% recall, outperforming the baseline MobileNetV4 Conv Small (72.66% accuracy, 70.73% F1-score). Further, Using this model as backbone our segmentation model achieved 0.6051 IoU Score, Loss of 0.2654, F1-Score 0.7504 and Recall 0.8638.

Keywords: Wildfire Detection, UAV Deployment, W-fire MobileNetV4-small, Fire Segmentation, Image Augmentation, MobileNetV4 Conv Small, Class Imbalance, Balanced Focal-Dice Loss, FLAME Dataset, Prediction.

Acknowledgement

Firstly, all praise to the Almighty for whom our thesis have been completed without any major interruption.
Secondly, to our supervisor Dr.Amitabha Chakrabarty sir for his kind support and advice in our work. He helped us whenever we needed help.
Thirdly, we are grateful for the assistance provided by Mr. Azwad Aziz sir. And finally to our family and friends, without their support, it may not be possible. With their kind support and prayer, we are now on the verge of our graduation.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Wildfires, are increasingly becoming more intense and frequent, exerting vast ecological and economic pressures worldwide. Traditional detection via ground sensors or aerial imaging has two significant disadvantages: systems on the ground cannot be scaled, and they are of low resolution and slow response and therefore unsuitable for early detection of fires [3] [7]. UAVs provide an emerging option by providing real-time high-resolution images but require light models that can execute on devices with limited resources [10]. FLAME dataset [17], comprising aerial fire and non-fire imagery for segmentation and classification, enables working on UAV-based wildfire systems. It presents challenges such as smoke and haze distortion, blur, duplicates, and strongly imbalanced classes [29]. Existing heavy vision models achieve high accuracy but cannot be utilized in UAVs effectively [21] [11].

This paper addresses these most significant challenges with the presentation of a complete real-time fire detection, and segmentation system powered by the new custom light-weight W-fire MobileNetV4-small model. Utilizing this efficient architecture in conjunction with robust augmentation, dedicated loss function for imbalance handling, and post-segmentation investigation of fire severity and propagation direction, the system achieves solid detection alongside key decision assistance [18] [8]. This contribution is in a unique position to have considerable influence on the development of deployable UAV-based wildland fire monitoring systems that effectively balance robustness, accuracy, and efficiency [7].

1.2 Motivation

The increasing frequency as well as severity of wildfires highlight the urgent need for deployable, precise real-time observing systems. While current deep learning algorithms perform well in laboratories and controlled settings, they are typically too computationally intensive for UAV or edge device deployment, limiting their field applicability [24] [12]. Wildfire imagery also presents unique challenges such as smoke, haze, low light, and sensor noise, all of which degrade detection performance negatively [18] [7].

This research is motivated by the necessity to develop a light-weight, strong, and scalable system that breaks these limits. By utilizing the MobileNetV4-small architecture [26] as a building block, along with carefully selected augmentation strategies and training for imbalance-awareness [11] [4], the discussed framework aims to bridge-the gap between research-level accuracy, and deployability in the wild [8]. Furthermore, the inclusion of fire severity class and direction of fire spread analysis ensures that the system not just detects fire but also offers operational intelligence to inform rapid decision-making in the event of emergency.

1.3 Problem Statement

Developing a reliable wildfire detection and analysis system for implementation on UAVs is not an easy task. Even though the FLAME dataset [17] is of highly excellent quality, it contains duplicate samples and data leakage risks if not treated with care. Moreover, it suffers from an extreme class imbalance since [18] fire pixels account for less than 0.003% of the pixels, thereby rendering accurate segmentation very difficult [7]. The imagery is also rendered more complicated by some environmental distortions such as motion blur, haze, smoke, and lighting variation, all of which lower model robustness [18] [1].

Hardware constraints also pose severe difficulties. UAVs, and edge devices such as Jetson Nano or Raspberry Pi have restricted memory and computation power and are incapable of executing big [21], computationally intensive models [10] [7]. Also, the segmentation dataset is small with 2,003 labeled samples, and therefore extreme hyperparameter tuning methods are required to avoid overfitting and boost generalization [8] [4].

Finally, in addition to detection, operational requirements include the provision of fire severity levels and direction of spread estimation in order to provide actionable intelligence to firefighting units [4]. The resolution of these conjoint problems necessitates the creation of an efficient, light-weight, and generalized fire detection and segmentation system capable of performing effectively under real-world environment conditions on UAV platforms [28].

1.4 Objective

The overall objective of our research is to design and implement a light-weight wildfire detection and analysis system deployable on UAVs. To this end, the study will develop a classification and segmentation pipeline from the FLAME dataset [17] and an augmentation process mimicking natural environments such as haze, blur, smoke, and and other different atmospheric conditions and distortions drone imagery might face. An optimization of the MobileNetV4-small architecture [26] is employed, which maintains competitive accuracy while ensuring lightweight UAV deployment.

The research also tries to address the problem of over-class imbalance upon segmentation using a Balanced Focal-Dice Loss function with fine-tuned positive weight

[8]. Beside, this also includes post-segmentation analysis including fire severity classification, in which fire is categorized as Low, Medium, or High, and fire spread direction prediction for guiding firefighting. Finally, scalability, and efficiency are prioritized so that the framework can be adjusted to other monitoring tasks aside from detecting wildfires [7].

1.5 Methodology

The research process is structured into dataset preparation, data augmentation, model construction, dealing with imbalance, and post-segmentation [18] [29]. The concept is to make the system highly accurate as well as computationally efficient for deployment on UAVs.
In dataset preparation, FLAME dataset [17] was used for classification and segmentation [2]. The dataset was divided into: training, validation, as well as test sets for classification. The training-set contained 20,015 fire and 11,475 non-fire images, the validation-set contained 5,003 fire and 2,871 non-fire images, and the test-set contained 5,137 fire and 3,480 non-fire images. Duplicate samples were treated extremely carefully by retaining them only in the training set and removing them from validation to avoid bias. For segmentation, the dataset contained 2,003 images and their respective fire masks. Approximately 70% of the images were utilized for the training, 15% for the validation, and 15% for the testing. For preventing data leakages, the first 304 images captured at a varied angle were reserved for testing, and then, the same number of samples were assigned for validation [11].

Then, to further enhance model robustness, specialized data augmentation pipeline was implemented to simulate real-world distortions . This encompassed the application of blur effects such as motion, turbulence, and defocus; cloud and haze overlays generated with Perlin noise; selective smoke simulation applied in fire regions; speckle noise for simulating sensor grain; and lighting variation such as nighttime or color jitter. Then, augmentation was applied to 50% of the training images, while validation data were preserved clean. For the evaluation of generalization, two test sets were employed: clean and completely augmented.

For model construction, various architectures were tried out, including CoAtNet Lite, EdgeNeXt Base, ViT-Tiny, and MobileNetV4-small [15] [26]. MobileNetV4-small was opted as the backbone due to its accuracy vs. efficiency trade-off, particularly when used for edge devices. For classification, a lightweight pipeline of MobileNetV4-small was built, and for segmentation, an encoder–decoder framework using MobileNetV4-small as the backbone with added lightweight attention blocks to achieve better performance.

Class imbalance management was a crucial aspect of the methodology. But, the segmentation data was imbalanced with the fire-pixel ratio being as low as 0.000026, which is roughly equivalent to a positive weighting factor of around 38,667. Due to this, a Detailed Focal-Dice Loss function was used. Boundary accuracy for precise fire mask generation was prioritized by Dice Loss, and Focal Loss [8] [4] with the modified positive weight successfully addressed the impact of imbalance between fire and non-fire pixels.

Post-segmentation analysis drove the framework into detection. Fire severity was estimated by image classification into Low, Medium, or High levels using the ratio of fire pixels. Spread direction prediction employed OpenCV to fit ellipses to fire masks and used angle and edge-contact rules to determine direction trends (e.g., forecasting "East" if the fire spread moved towards the right edge) [24] .

Finally, deployment factors were also taken into account. So, the system was architected for real-time operation on UAV-compatible edge devices such as Jetson Nano and Raspberry Pi [26] [10]. Additionally, the framework was designed to be scalable, and portable across other disaster-monitoring use cases such as Smoke and gas leak detection, Volcanic activity and geological monitoring, Medical imaging.

1.6 Scopes and Challenges

The scope of this research is developing an end-to-end wildfire detection and analysis system with UAV deployment [7]. The system integrates real-time aerial imagery with lightweight models to recognize the existence of wildfire, segment the affected region, and provide more information through severity classification and spread estimation [29]. The system is made resilient by integrating efficient data augmentation techniques such that it generalizes for different weather conditions [12], terrains, and image artifacts. Furthermore, its modularity ensures that it can be programmed for related disaster-response applications, such as smoke detection, flood detection, or rescue identification, thereby making it more versatile.

Despite having its advantages, the study also faced quite a number of setbacks. Its greatest setback was having a severely imbalanced class structure in the segmentation dataset such that fire pixels accounted for less than 0.003% of the whole dataset. This meant creating special loss functions as well as weighing schemes to enable learning. Besides, the limited size of the labeled dataset, with just 2,003 segmentation samples, also emphasized the necessity for augmentation, and the proper management of duplicates in order to allow generalization. Hardware constraints imposed also strict limitations, since devices suitable for UAV such as Jetson Nano or Raspberry Pi cannot handle resource-intensive heavy models and therefore necessitate the employment of light-weight designs [29]. Furthermore, the real-world deployment includes variability that may not necessarily be represented in training data, i.e., heavy wind, smoke, and fast-evolving fires [7]. Then,finally, achieving balance between robustness and computational efficiency was an ongoing issue where compromise of both stable performance, and real-time appropriateness had to be met following careful trade-offs [10].

Chapter 2

Literature Review

2.1 Overview

The literature review brings together various deep learning (DL) approaches from prediction, segmentation, wildfire management and detection domains. The paper is based on the FLMAE dataset [17], distinct from remote larger scale platforms. Important evaluation metrics between accuracy maximization through complex hybrid architectures (e.g., CoAtNet, attention-augmented U-Nets) and real-time latency requirements of edge deployments needed to be achieved (e.g., MobileNetV4, EdgeNeXt. We went through current architectural strategies to document limitations which included reliance on large pre-training data [15] [16] and computational bottlenecks of complex attention models [9].
Significant research gaps were found in hybrid lightweight models for fine-grain fire feature extraction. Highly imbalanced aerial segmentation outputs being used to derive operational metrics were also found. Highly specialized data and the continual push to adapt complex architectures for high-speed, operation on deployment [11] was relied on by wildfire computer vision (CV). This serious challenge remains between developing lightweight models that are computationally lightweight, robust to noise and are simultaneously accurate.
The FLAME dataset [17] was important as it provided an aerial imaging dataset designed particularly for fire analysis. UAVs capture video and data (DJI Matrice 200, Phantom 3) which includes both thermal and normal palettes (WhiteHot, GreenHot, Fusion) for segmentation and classification tasks.
Data Specificity Limitation: Pile burns, which are controlled, small-scale events, are specifically captured by the FLAME data. Ground camera images are fundamentally different from aerial imagery due to higher resolution and a ground-level perspective.A domain shift [18] risk is introduced by this specificity when large, uncontrolled wildland fires captured by remote sensors [13] are analyzed using models trained on FLAME [17]. A recognized limitation, therefore, is that generalization across different fire phenomena may not be achieved by models trained on aerial data. In contrast, mapping, damage prediction [11], and extraction of environmental factors such as NDVI [22] are primarily achieved through high-resolution, satellite remote-sensing data (e.g., Landsat, Sentinel-2). However, long revisit times (e.g., 8–16 days for Landsat) are suffered by these systems, limiting their utility for real-time fire behavior monitoring, a gap where aerial imagery is critical [25].

2.2 Literature Review

For the pursuit of real-time fire detection, highly efficient model architectures that minimize computational costs (FLOPs/MACs) and parameter count are necessitated, thereby enabling deployment in edge devices like the Raspberry Pi.

2.2.1 Evolution of Efficient Convolutional Networks

Early acceleration was focused on structural decomposition. Efficiency gains while maintaining performance were achieved in the Xception-architecture. It utilizes depth-wise, separable convolutions rather than the standard Inception modules[5]. Rapid convergence and robust final classification performance in the Xception network were ensured by the inclusion of deep residual connections, as found by the authors.
More recently, the MobileNetV4 (MNv4) family was developed using Neural Architecture Search (NAS) to be mostly Pareto-optimal across diverse mobile hardware platforms (CPUs, GPUs, DSPs, Google EdgeTPU) [26].
Limitation (Hardware Constraint): Conventional components (depthwise convolution, ReLU, BatchNorm) are prioritized by MNv4 over complex ones (like Squeeze and Excite (SE) [11] or LayerNorm) because poor support for complex modules exists on specific hardware platforms such as DSPs [26]. This design philosophy, while maximizing universality and latency, inherently restricts the immediate adoption of complex attention mechanisms in the core architecture.
Finding: Attention integration, such as the efficient Mobile Multi-Query Attention (Mobile MQA), is applied in MNv4 models primarily in the last stages of the convolutional backbone, where contribution is maximized with minimal latency impact [26].

2.2.2 Hybrid CNN-Transformer Architectures

The lack of high generalization capability in pure Vision Transformers (ViT) when training data is scarce (e.g., on ImageNet-1K) is addressed by hybrid models through combining the locality and translational invariance of CNNs with the global context modeling of self-attention [15] [16].
CoAtNet: Effective scaling with massive datasets like JFT-300M is achieved through the combination of convolution and attention, enabling superior capacity. However, permanent limitation of model capacity results when overly aggressive stride in the initial convolution layer (like the ViT stem) is applied. The ablation by the authors demonstrated that retention of initial convolutional stages (C-C-T-T layout) is vital for processing low-level information [15].
EdgeNeXt: A multi-stage design is employed in this architecture, introducing the Split Depth-wise Transpose Attention (SDTA) encoder to efficiently amalgamate CNN and Transformer features for edge devices. However, the addition of global SDTA encoders to the earliest stages (Stage 1) of the network was critically noted by the authors as not beneficial because the features are not yet sufficiently mature [21].
Gap Analysis: Architectural Inefficiency (RQ1): Massive data for training is

required by powerful hybrid models (CoAtNet), while efficient mobile-optimized networks (MNv4, EdgeNeXt) struggle to utilize complex attention mechanisms effectively in early layers [21] [26]. A critical gap is therefore exposed: the highly specific, fine-grained features necessary for accurate aerial fire and smoke detection using small, specialized datasets like FLAME [11] are not captured by standard efficient backbones. Formalization of selective, custom hybridization within efficient backbones must thus be the focus of research, incorporating only highly specialized attention (such as ECA-Net or DANet) in targeted stages to boost performance while adhering to strict latency limits.

2.2.3 Advanced Semantic Segmentation and Attention Mechanisms

For defining the fire perimeter and analyzing its spread, semantic segmentation—requiring models capable of dense, pixel-accurate prediction—is crucial [11].

Foundational Segmentation Frameworks: Encoder decoder models relied upon include Fully Convolutional Networks (FCN) and the many variants of U Net [9] which include Attention U-Net, U-Net++ [22], ResUNet [23]. Small models benefit from merging high resolution features and semantic information from the encoder and decoder in U-Net respectively [9]. A two-branch architecture is employed by other fast approaches like Fast-SCNN to handle real-time segmentation requirements [22].

Feature Aggregation and Context Modeling: To improve segmentation accuracy, mechanisms that effectively aggregate multi-scale features and long-range dependencies are employed.

Feature Pyramid Networks (FPN): Scale variance in detection is addressed by building a top-down pathway that fuses low-level, high-resolution spatial information (from early CNN stages) with high-level semantic information (from deep stages) [26] [6]. Critically, significant accuracy degradation was demonstrated by the original authors when lateral connections that merge features were removed.

Dual Attention Network (DANet): FCN variants (like DeepLab-v2/v3) [9] relying on local operations (atrous convolution) are surpassed by DANet, which explicitly captures global contextual dependencies using two parallel attention-modules: Channel Attention Module (CAM), and Position Attention Module (PAM).

Critical Limitation: Major areas of future work include decreasing computational complexity and enhancing model robustness.

ECA-Net: An Efficient Channel Attention module is provided by this mechanism, highly relevant for cheaply integrating context into complex architectures. A critical argument by ECA-Net is that channel dimensionality reduction—a core feature of previous methods like SENet—is detrimental to learning effective channel attention. Efficient cross-channel interaction without this reduction is ensured by ECA-Net, achieving performance comparable to high-parameter models like ResNeXt-101 but with much lower computational cost [11].

Gap Analysis: Imbalance and Operational Accuracy (RQ2 & RQ3): Extreme class imbalance (with fire pixels sparse relative to the background) is suffered by the segmentation task for aerial wildfire imagery [11]. While specialized attention mechanisms (DANet) and loss functions (FL) exist, rigorous analysis on their combination for resource-constrained aerial semantic segmentation is lacking in the lit-

erature. Furthermore, focus in research is often restricted to traditional CV metrics (mIoU, F1-Score) [23], while the translation of outputs into actionable operational metrics (such as fire severity based on pixel area or fire spread direction) is neglected. This motivates the necessity of validating whether enhanced segmentation models can reliably generate critical operational intelligence.

2.2.4 Robustness and Prediction

Model Robustness: Vulnerability to natural corruptions and domain shifts (e.g., noise, weather changes) is demonstrated by deep networks despite their high accuracy on clean data. Techniques that bolster model stability are thus essential for improving trustworthiness [18].
Dropout: A standard regularization method, dropout randomly drops complex neurons during training, which commonly leads to overfitting on limited training data [4].
AugMax: Robustness to common corruptions (ImageNet-C) is explicitly addressed by AugMax. By creating harder, more diverse training samples through adversarial composition of random transformations, AugMax critically surpasses preceding augmentation methods (e.g., AugMix), achieving improved Robustness Accuracy (RA) [18].
Data Imbalance Mitigation for Auxiliary Data: In applications such as healthcare, extreme class imbalance (IR of as much as 10.92:1 in some datasets) is typically treated using synthetic oversampling techniques [29].
Critique of Traditional Methods: Noise is introduced, and complex data distributions are poorly captured by simple methods, like SMOTE [29].
Deep Generative Models (DGMs): To overcome this limitation, the Auxiliary-guided Conditional-Variational Autoencoder(ACVAE) was developed, by researchers. Diverse, label-dependent synthetic minority samples are generated by ACVAE through contrastive learning. An ensemble, ACVAECDNN, is formed by pairing this generation with Edited Centroid-Displacement Nearest Neighbor (ECDNN) undersampling to filter unrealistic synthetic samples.
Critical Limitation (Computational Cost): Training DGMs such as ACVAE is extremely computationally expensive, limiting scalability, and the approach becomes weaker on datasets with all-numerical features [29].
Wildfire Prediction and ML Ensembles: Machine learning models, such as xG Boost, Random Forest, and LightGBM are employed extensively to forecast indicators such as wildfire spread rates on a daily basis and burn area increase rates by aggregating geospatial, and meteorological inputs [11].
Data Deficiency Risk: The absence or deficiency of critical input features (e.g., terrain elevation, fuel models, weather inputs) significantly impacts prediction models like the FARSITE benchmark. Error analysis confirms that drops in accuracy occur, with severity depending particularly on the geographical characteristics of the fire area [28].

2.3 Key Findings

2.3.1 Gap Analysis and Research Questions

Three main, related research gaps are found to be impeding the successful application of advanced DL in aerial wildfire management based on the synthesis of the body of existing literature:

Architecture-Feature Alignment Gap (Leads to RQ1): Current lightweight architectures (e.g., MobileNetV4, EdgeNeXt) that prioritize hardware compatibility over intricate feature extraction limits the integration of complex attention mechanisms to later stages [21] [26]. Conversely, high capacity is demonstrated by general hybrid architectures (e.g., CoAtNet), but massive proprietary datasets (JFT-300M) are required for optimal performance [15]. This leaves a critical gap in the development of a robust, efficient architecture that can selectively target and extract fine-grained, specialized aerial fire features (thermal signature, sparse smoke) from small-scale datasets like FLAME [17].
RQ1: How can selective hybridization of efficient CNN blocks and specialized attention mechanisms (e.g., ECA-Net, DANet) be formalized to maximize classification accuracy and edge efficiency for low-latency aerial wildfire detection tasks using the FLAME dataset?

Gap in Dense Prediction Imbalance Mitigation (Leads to RQ2): An extreme foreground-background class imbalance is suffered in high-resolution fire segmentation tasks. While Focal Loss [11] addresses this by focusing on hard negatives and complex models like DANet capture context, computational complexity is acknowledged by the latter as a major limitation [9]. An opportunity therefore exists to validate whether combining aggressive imbalance-aware loss functions with existing efficient segmentation models can provide the necessary balance of accurate boundary localization and computational economy for real-time aerial systems.
RQ2: Is it possible to improve the robustness and localization accuracy of high-resolution aerial fire segmentation by combining boundary-aware and focal components (Dice Loss and Focal Loss) in a composite loss function? Specifically, can this reduce performance degradation brought on by severe foreground imbalance?

Gap in Translation to Operational Intelligence (Leads to RQ3): Caused by a translation gap to operational intelligence. Actionable, and timely operational intelligence from pixel-wise CV outputs is still out of reach, despite the improvement in prediction from geospatial-inputs. Existing DL studies mostly concentrate on model performance metrics (such as the F1-Score, and IoU) [1], ignoring clear links to operational requirements like estimating localized spread dynamics or evaluating fire severity based on coverage. The catastrophic potential of False Negatives highlights the importance of this gap [11].
RQ3: How accurately and quantitatively can DL-derived pixel masks be converted into operational metrics in real time, like parameterized fire severity levels and spread direction estimates, to serve as the essential link between deep learning output and useful wildfire management decision-making?

2.3.2 The Disconnect Between DL Segmentation and Actionable Operational Metrics

The necessity of bridging high-resolution pixel-wise output with useful operational intelligence is the core gap identified, representing a recognized need for enhanced fire management.

Prediction vs. Mapping: Forecasting of daily burned areas [17] is a common use case of ML, and DL models for wild-land fire prediction. The challenge, though, is providing real-time, high-granularity fire behavior observations [24]. Detection, mapping, prediction, and damage severity prediction [22] are the major areas covered by DL research on wildland fires [3] [27].

Operational Requirements: Complex fire spread and behavior prediction in real-world environments is benchmarked by models such as the FARSITE Fire Area Simulator, which incorporates topography and fuel types [28]. Literature implies that DL outputs need to specifically integrate with the requirements of such decision-making tools.

Severity Assessment: Fire segmentation is used to delineate fire borders for applications such as fire expansion modeling [17]. DL techniques have also been applied to predict and model wildland fire severity of damage [22], but proper extraction of such indicators of severity directly and in real time from raw segmentation masks remains a challenge [7].

UAV Data Mandate: The FLAME dataset [17], foundational to the present work, provides aerial images and thermal palettes for DL-based image segmentation intended for fire expansion modeling. This dictates that segmentation outputs must be translatable into dynamics relevant for managing fire expansion. More broadly, UAVs are widely employed for real-time fire detection, monitoring, and assistance.

2.3.3 Critical Limitations of Spatial and Performance Metrics

Insufficiency in current performance metrics for estimating wildfire-spread prediction accuracy was highlighted by tests. Limitations in extreme scenarios were observed for the Sorensen metric, Jaccard coefficient, Kappa-statistics, and perimeter-size differences (S_x). Metrics such as the Sorensen metric, Jaccard coefficient, and Kappa-statistics—based on Intersection-over-Union (IoU) calculation—assess similarity between the predicted, and actual fire perimeters. However, an anomaly occurred in the Radford Fire experiment: the predictions yielded almost circular results that overlapped notably with the real-fire scene, producing misleadingly high IoU-based values. Poor alignment with visual outcomes was also shown by metrics due to its centroid-size-based calculation method. This situation highlights the difficulty of establishing universally applicable assessment metrics [28].

IoU Metric Flaws: Reliance on metrics such as the Jaccard coefficient or Sorensen metric, based on overlap measures, is documented to produce abnormal or anomalous results. Specifically, IoU-based calculations may yield misleadingly high scores when poor predictions accidentally overlap with the real fire scene. This anomaly prevents accurate assessment of prediction precision, motivating the reconsideration of universal accuracy evaluation methods in wildfire contexts. Additional validation

is thus necessitated when outputs are converted into management decisions (e.g., severity/direction) [28].

Imbalance Metrics: For detection tasks where the target (fire) is rare, traditional metrics such as pixel-wise classification accuracy are unfitting, as predicting all negatives across the output could still manage high-scores. Binary classifications often employ accuracy, and area-under the ROC curve (AUC) [24]. However, for datasets with strong imbalance, area under the Precision–Recall Curve (PRAUC) is noted as a superior primary metric [29]. The inadequacy of general accuracy metrics highlights the necessity of adopting rigorous validation using metrics such as Precision–Recall, F-beta scores, or Class Balance Accuracy to reflect model performance on the positive (fire) class [14].

Model Non-Interpretability: A broad challenge across DL remote sensing applications is interpretation. Pixel masks produced by DL segmentation models classify fire vs. non-fire pixels but require standardized, quantifiable interpretation to translate into "severity levels" or "spread direction" [22]. This necessitates moving beyond simple burned area prediction to dynamic spread estimation. Interpretability tools such as Grad-CAM (Gradient-weighted Class Activation Maps) provide visual explanations by highlighting critical regions for classification, making strategies transparent for human validation—vital in emergency response. Future research directions specifically pinpoint integrative, imagery-based fire-spread models that incorporate environmental-factors in order to bridge this gap in interpretability [17].

In general, bridging pixel segmentation with indicators such as fire severity and direction of spread is identified as a crucial requirement, and it is both an operational, and scientific void. Current DL methods suffer from interpretability limitations, while validation practices are undermined by IoU anomalies (e.g., Jaccard, Sorensen, Kappa) [17] [28] and the inadequacy of general accuracy metrics for imbalanced data. This establishes the ongoing challenge of reliably extracting severity and spread indicators directly from raw segmentation outputs in real time [22].

Chapter 3

Specifications and Requirements

3.1 Final Specifications and Requirements

The last requirement and specification of the fire detection system, provides a short-code guideline to assure the effectiveness, reliability and adaptability in the on-the-fly aerial surveillance capability for pile burns and wildfire using UAVs. In this phase, based on preliminary experimental results and dataset analyses, it infers the limit of a classification accuracy obtained with the `w_fire_mobilenetv2_s` model and segmentation accuracy achieved with the FPN (Feature Pyramid Network) Style as confirmed by impact estimates relative to FLAME dataset tested under different conditions seeking an optimally balanced compromise between sensitivity and specificity. The attached consists of small deep learn architectures, MobileNetV4 branches in self-training are initialized by timmg. `create_model`('`mobilenetv4_hybrid_medium`', `pretrained=False, num_classes=2`) for classification. Segmentation module: This is optimized for edge deployment, with exhaustive testing in a development environment 11th generation Intel(R) i7-14700K CPU, an RTX 4080 Super 16 GB GPU and 64GB DDR5 RAM where inference times are kept strictly under the threshold of 100 milliseconds per frame (`timed using time.time()`in the notebook) in order to run in real time at exactly 10 frames per second. Code-specific requirements: torchvision with preprocessing chains. transforms. Resize and cv2. class imbalance and head motion due to the environment condition of the dataset, together with augmentations. The design, implemented as modular PyTorch modules, is designed to be extendable to multi-modal fusion with thermal images provided by FLIR Vue Pro R camera in future, built while maintaining safety-critical standards (based on false negative rate and extensive logging through print statements), and scales across the hardware provided.

3.2 Societal Impact

Wildfires cause a major global environmental problem, ranking among the deadliest and costliest natural disasters worldwide. This phenomena also cause devastating damage to forest resources, wildlife habitats, and residential areas, with annual damages exceeding $6 billion in the USA alone in the recent years[22]. Given the increased frequency, and intensity of these catastrophic wildfires, advancing and deploying effective predictive tools that will help limit threats to human life, properties, and operational forces has taken on increased urgency[24]. Community-wide

reduction of the dangers posed by wildfires through advanced technologies, such as machine learning(ML) and deep learning(DL) models applied to aerial, and satellite based data has potential by decreasing the time it takes to discover a fire, improving predictions, and leveraging strategized fire management[17][22]. Through advancements in technology, increasing prevalence of accessible data, and sophisticated systems, there are real societal benefits of improved operational efficiencies, safety, predictions, and management of wildfires[12]. The principal societal benefit derives from a significant increase in public safety, as well as an improved emergency response. From a public safety perspective, automated detection systems are crucial for saving lives and making community evacuations safer and better coordinated as they can detect a fire or smoke signature much earlier than someone can report. Similarly, ML models can provide quantitative predictions of important fire characteristics, such as growth rate per day, spread vector, and burn severity, that inform response to hazards[24][13]. This is crucial for effective hazard management in fire departments by facilitating the planning of firefighter and resource dispatch and safe use. Additionally, expected expansion rates can inform critical operational decisions during ongoing incidents, including population evacuations and strategic fire suppression attack. These tools also promote data-driven decision-making and resource allocation on a strategic management level. Models that evaluate risk and fire susceptibility, such as the Ignition Component (IC) and Population Density (PD), provide local authorities with information on the complex dynamic changes over time[27]. That also allows them to identify and prioritize effective fire prevention and control areas, especially in high-risk areas. Additionally, the robust fire detection and mapping capability is used to report damage from fires, as well as for planning fire recovery[22]. Overall, using an accurate prediction to mitigate the severity of wildfires protects critical infrastructure, mitigates risks to operational forces, and reduces other inherent risks to the environment and public health, such as harmful smoke pol[17].

3.3 Environmental Impact

Across the globe, wildfires are considered one of the most serious natural catastrophes. It represents a serious hazard to natural ecology, economic conditions, and human health[28]. Poorly managed wildfires are an extreme environmental problem that will often result in permanent, unwanted changes to specific landscape and ecosystem services[14]. Wild-land fires are highly destructive to natural resources, such as forests, soil, water, biodiversity, and wildlife among the most hazardous natural hazards[22][13]. These hazards also cause loss of biomass over a large scale and destruction of ecosystems, posing threats to wildlife populations[27]. The scale of destruction is incredible; for example, annual forest fires impact thousands of hectares globally. Moreover, experts warn that future wildfires will likely become more recurrent, and severe because of climate change. There is a real connection between the near future and the tremendous increase in fire occurrence and intensity, due in part to warmer temperatures and droughts[12]. That extends fire seasons and are generally associated with the degree of intensity in the fire itself[22]. It is also predicted that wildfire could be increasing in intensity globally under a warming climate scenario. In addition, wildfires have serious impacts to the atmosphere by emitting large amounts of soot emissions, including CO2 and other trace gases

and particulates resulting from biomass burning, altering atmospheric chemistry[3]. The smoke created by these fires is dangerous, containing both pollutants and small particulate matter (PM2.5) including dangerous microbes which can be aerosolized, presents huge public health risks which results to degradation of air quality and addition of harmful bacteria to water systems[23]. In addition to air quality, the severity of wildfire events may impact affected proximity waters in clean water supply to populations from wildfires contaminating the potable waters during the wildfire events[13]. Detecting environmental perturbations in inland, estuarine and coastal waters that are optically complex are critical often involving the collection of water quality parameters that include the concentration of chlorophyll-a[19].

3.4 Ethical Issues

Ethical concerns associated with the applications of deep-learning, especially in high-stakes areas of application such as environmental management and healthcare, focus on fair models, reliable models, and also the societal impact of deployable models. A key consideration is the possibility that training a model on data with class imbalance, especially in datasets involving health, where negative classes predominating (e.g., patients with negative test results outnumber positive), the model's performance will be biased toward the predominant class[29], resulting in incorrect diagnoses, conditions going unrecognized, and negative outcomes for patients, which further emphasizes the need for fairness in model performance. These considerations pertaining to reducing bias due to imbalanced data are also relevant to cases such as detecting dense objects, which also assumes extreme class imbalance. There are also broader implications to consider in the deployment of machine vision technology to address the social implications and evaluation of its social implications when it pertains to public health and human life forecasting hazards such as wildfire smoke, etc[8]. Using machine vision technologies and algorithms that require reliable data inputs in emergency decision-making (e.g., disaster risk reduction) when data may be incomplete or missing that has direct impact on human life[25]. Therefore, systems tasks for supporting fire management decision making should be evaluated for scientific integrity in their risk assessment and to promote decision making based on data[28]. This also requires potentially robust models or modeling techniques that effectively can address intrinsic difficulties with the data, for example, data splitting techniques to maintain generalized capability to minimize prediction variances, etc.

3.5 Project Management Plan

To develop the project and evaluate the fire detection system using deep learning models, effective management plan is required. It establishes the strategic plan for orderly progress, effective planning, timely completion, resource, and work allocation, and risk management. The project was broken down into distinct but recursive stages to permit the dynamic nature of experimentation in deep learning. Literature review and conceptualization in the initial phase involved extensive reading of existing fire detection methods, deep learning applications in aerial imagery, and data sets like FLAME, including its multi-model visible and thermal data. This initial research defined the project scope, i.e., binary classification and pixel-wise

segmentation for fire regions, and guided model selection like MobileNetV4 variants, ViTNet, EdgeNet, CoaT-Lite, and U-Net for segmentation. Subsequent to conceptualization, the data collection and preprocessing phase commenced utilizing the FLAME dataset containing 39,375 images categorized into: 80% training, and 20% validation/testing. Furthermore, the model development and experimentation phase focused on carrying out and fine-tuning these models, MobileNetV4 variants with TIMM library, ViT Tiny with transformer encoders for global feature detection, EdgeNet for edge-aware processing, CoaT-Lite for efficient transformer-convolution hybrids, and FPN with skip connections for segmentation with hyperparameter optimization using AdamW optimizer, cross-entropy-loss, and learning rate schedulers. This step also involved fine-tuning weights, and iterative tuning based on validation metrics. The evaluation and analysis step cautiously tested model-performance using accuracy, precision, F1-score, IoU, computed using scikit-learn, and recall, with confusion matrices being plotted in seaborn to provide insights in misclassification. The final step included integrating all conclusions in the thesis report, integrating findings for logical presentation. Task management included breaking activities into discrete steps, for instance, "data splitting with random.shuffle," "model initialization via `timm.create_model`, `training loop with torch.no_grad`, `metric computation with sklearn.metrics` assigned with unique milestones to track progress."

3.6 Risk Management

Wildfire risk is primarily managed with predictive models that express the likelihood and potential consequences of fire, which is important for land and fire management agencies that need to make decisions to reduce wildfire risk [14]. The quantitative assessment of wildfire risk involves a four-step process of :
(1) identifying specific management objectives (problem formulation)
(2) quantifying exposure, which means assessing the likelihood that things of value will encounter the fire
(3) quantifying the effect on the resource, which is often assessed using fire behavior tools such as flame length or rate of spread
(4) risk characterization, which synthesizes the first three pieces of information into something that is useful in informing better decisions[14].
This is also relevant for specific estimates of the possibility of a wildfire which means how likely it is that a place would change from unburnt to burnt[14]. Probabilities of wildfire burns can serve as a factor of wildfire risk or be used as input for some risk quantification. Accurate predictions about daily spread rates of wildfires are particularly important for informing decisions about planning the resourcing of fire-fighting [24] operations and determining if and when it is safe to evacuate communities in the path of fire[3]. Moreover, wildfire risk management approaches must also evaluate the uncertainty that accompanies predictions. Modeling wildland fire behavior is complex and is highly sensitive to environmental conditions.

3.7 Economic Analysis

Wildfires is one of the most dangerous and costly natural hazards[22], resulting in real economic and social consequences[3]. Nevertheless, not limited to incalculable property losses, significant disruption to vital infrastructure, loss of human life[24], and exacerbation of global climate change due to carbon emissions. Predicting and, detecting a potentially dangerous wildfire is a vital first step because if a fire can be extinguished quickly. The consequences of that event will be drastically lower, allowing for effective budget allocation forecasting and the mobilization and dispatching of firefighters and other resources safely[12]. The insights gained from data can assist with local forest fire management, as well as potentially leading to policy decisions that can save billions of dollars in the long run[13]. However, the machine learning models themselves pose an economic trade-off that illuminates an important interplay between model complexity and performance. The very goal of developing sophisticated attention modules [11] [9] for wildfire prediction is associated with a known reliance upon complexity. For example, hybrid and mobile vision architectures seek efficiency— MobileNetV4 achieves universal efficiency [26] on mobile devices by optimizing the ratio of operations, to memory accesses (operational intensity) to pay attention to latency and cost, while EdgeNeXt [21] claims low latency inference on edge devices through a prescribed number of MAdds (the product of a multiplication and addition operation). However, there are some devices that improve the performance-complexity paradox like ECA-Net [11], which achieves performance gains while adding relatively few additional parameters and negligible additional computations. In other cases, ensemble models like GWO-XGBoost [27] can be preferable to complex deep learn models for applicability because of their lower data and computational requirements, making them more appropriate for situations immersive enough to limit data acquisition and fewer computational resources. Machine learning decision support systems (DSS) [13]are designed to allow management teams to mitigate economic losses when dealing with a crisis.

Chapter 4

Data Analysis

4.1 Design Process

This dissertation describes the fire detection system design as an integrated and systematic design that integrates cutting-edge computer vision approaches to design a conceptually advanced fire detection system. The goal is to create a smart system involving seamless data collections and dispersions, image classification, and segmentation workflows to accurately identify, localization, attribute fire events in photographs. The focus is on creating a robust model that will withstand real-world challenges of environmental noise, lighting variations, and image distortions. The design focuses on simple modular workflows and advanced and powerful deep learning tools, especially PyTorch, and the customization of networks MobileNetV4 for its image classification. The segmentation model compiles images on a pixel basis to advance and build an adaptable and useful system. The design contains the elemental workflows of obtaining and describing data, preparing images, experimentation, the classification of images, and segmentation of fire areas. Other predetermined, but still important details, such as proposed image modeling architecture, flow of training, performance-test and post processing steps are included using information gathered from the code files.

4.1.1 Description of data

The FLAME (Fire Luminosity Airborne-based Machine learning Evaluation) dataset is a remotely sensed imagery dataset with a particular focus on pile burn monitoring, designed to facilitate deep learning, for fire detection and modeling [17]. The data were gathered during a burn of piles, a management treatment in high-elevation forests, conducted-by fire managers from the Flagstaff Fire Department. The site itself was a forest of ponderosa pine in Observatory Mesa, Flagstaff, Arizona, USA. The acquisition took place on January 16th, 2020, in specific weather conditions: 43°F (6°C), partly cloudy, and windless. The acquisition was conducted using aerial drones (UAVs), the DJI Matrice 200 and the DJI Phantom 3 Professional. These drones were equipped with different kinds of cameras, from Full HD and 4K cameras (such as the Zenmuse X4S and Phantom 3 Camera) to a thermal camera designed for special purposes. The data captured consists of images and videos in four palettes: normal, Fusion, WhiteHot, and GreenHot. Raw videos are in .MP4 and .MOV, and processed images are in .JPEG and .PNG formats. Most importantly, the

dataset is meticulously annotated to facilitate two novel deep learning tasks: Binary Classification for frame-level fire detection, and Fire Segmentation for pixel-level fire masking and expansion modeling. Ground Truth Masks for segmentation were created by a human subject matter expert (SME) through the manual annotation of fire pixels using polygon shapes in 2003 frames using the MATLAB (TM) Image Labeler tool. The dataset is offered as a benchmark for the development of computer aids to fire control and management, and its multimodal content (including thermal imagery) also allows for the investigation of future challenges such as Federated Deep Learning with multiple drones.

4.1.2 Data preprocessing

In the design of the fire detection system presented in this thesis, the data preprocessing phase serves as both a fundamental and carefully considered aspect, as it performs the specialized role of converting the raw source FLAME dataset, into standardized, varied and resilient form for training deep learning models, in particular for aerial imagery where variation is a significant challenge. This critical role employs a number of Python packages such as `os`, `shutil`, `random`, `torchvision.transforms`, OpenCV (`cv2`), etc. and arises to individual challenges that are present for "Fire" (25,018 images), and "No_Fire" (14,357 images) classes of the dataset with just under 39,000 images and an intentional skew of the dataset towards fire detection sensitivity for use in safety critical situations. For the coding of data preprocessing, the first aspect of the data preprocessing workflow is on using dataset splitting [2], where root directory `F:\Flame Dataset\preprocessing` is split into `train` and `valid` subdirectories, and, say with validation ratio of 20%, allocates 5,003 from "Fire", and 2,871 images from "No_Fire", into the validation dataset, while still attempting to achieve a unbiased distribution of samples through the use of Simple Random Sampling (SRS) with the command `random.shuffle(all_files)`, and `shutil.move` where `os.makedirs` is set with `exist_ok=True` to provide greater resilience against existing files/folders in directories. This preliminarily splitting process–detailedly recorded and committed to make up of subfolders/files – maintains the general class-wise proportions, thus avoiding an overfitting and facilitating hyper-parameter tuning without bias.

It is the augmentation phase of the preprocessing pipeline where things get intricate with a sophisticated generalization and robustness enhancing extension, which simultaneously integrates various standard and domain-specific transforms to simulate these unpredictable conditions that one experiences while trying to detect aerial fires in real settings, as it has been well documented as it appears to be based on their logs. This stage uses environmental simulations to emulate real and operational challenges, including `apply_cloud_effect` which uses Perlin noise to create and blend atmospheric occlusions; `simulate_night` which uses `cv2.convertScaleAbs` to change brightness and contrast to give the low light night-time surveillance look; `apply_soft_smoke_on_fire_region` which uses color thresholds to isolate fire pixels to create masks, then applied with a Gaussian blurred synthetic smoke to visually obscure the flames, along with `snowfall_effects` that created variability from adverse weather. To improve resiliency to image artifacts common in UAV footage, stochastic blur and noise conditional probabilities were also applied, such as turbu-

lence blur using either sinusoidal warping or Perlin noise, motion blur that applied directional blur kernels, defocus blur that used complex blurring kernels to simulate out-of-focus optics, softening blur which used `cv2.GaussianBlur` with kernels sized 3x3 to 11x11 and sigma values from 1-3, and speckle noise simulating digital sensor or compression artifacts, improving the model's capability to process degraded imagery that is shown in Fig 4.1 and Fig 4.2 below. Standard transformations of the photometric and geometric sort are also used, along with the `torchvision.transforms` library allowing for random horizontal-flips, rotations of up to 20 degrees, random resized crops with a specified scale ratio, and color jittering via `ColorJitter` to overcome conditional overfitting toward specific lighting situations to establish a broad data-augmentation strategy reflective of the diversity involved within aerial fire environments. The augmentations were added to half of the training set, and half were kept clean in order to match with validation data Flame Dataset [17].

Figure 4.1: Fire Images

Figure 4.2: No-Fire Images

After all the splitting and data-augmentation, the pipeline finishes with a normalization step to preserve numerical stability and compatibility for use. After that we evaluated results on 2 Sets, One kept 100% clean and other kept augmentations same as training set. Since, The training set had 50% clean data and 50% augmented data we evaluated unbiased results on both of our dataset as both sets had unbiased training. Half images, having gone through the aforementioned transformations, are resized to 224 x 224 pixels, and converted to tensors using `transforms`. To convert to the PyTorch Tensor object, and, normalize using `transforms`, with ImageNet statistics, was as follows: `normalize()` took our scale-associated image pixel values and converted them to a greatly normalized range near [-1, 1] to correctly match the associated range of deep learning and other architecture inputs. This preprocessing

step was successful in addressing data - a class imbalance, and amount of variance between the aerial imagery - and with the logging, we were able to clearly and modularly, produce a dataset for classification and segmentation purposes. Gains in actual performance metrics, such as, in `w_fire_mobilenetv4_s`, MobileNetv4 accuracy gain of +1.04% in the customized, also the upper hand for further being evaluated in the augmented test set as the model was half trained with augmented data speak volumes to the efficiency of this step. This also demonstrates, this use case is ready for operational use under the rigors and speed issues associated with aerial fire monitoring, and directly aligns with deep learning best practices for the wildland fire use case.

4.1.3 Experimental Workflow

The Experimental Workflow specifies the technical implementation configuration, hyper-parameters, optimization techniques and criteria to train and test the models for the purposes of fire classification (Fig 4.3) and segmentation.
(Fig 4.4).

The baseline study simulations from which the preliminary results were established conducted the environments infrastructure of the AMD Ryzen 9 3900X CPU, and an NVidia RTX 2080 Ti GPU on Ubuntu [17]. However, as expressed in the training files above, the new experiments have been developed under high performance computing hardware, which as you list below, includes an i7-14700K CPU, an RTX 4080 Super 16GB GPU and 64GB DDR5 RAM. Thus, not only does it utilize GPU acceleration via CUDA, as much of the deep learning implementation recommends [4][17] but extremely efficient deep learning computation. Additionally, high-capacity resources are required to handle the numerous parameters and FLOPs associated with contemporary CNN and Transformer architectures due to the complexity of the tested models, including CoAt-Lite, EdgeNeXt, and MobileNetV4 [26][21]. Though certain special mechanisms were added to provide guarantees for performance and stability, the core training method follows standard deep learning conventions.

- **Optimization**: The Adamw optimizer, a stochastic optimization method that is known by this name, is the main optimization algorithm applied to a broad array of models, including segmentation models (e.g., U-Net) and classification networks (e.g., Xception) [24]. During training, the Adamw optimizer is used to determine the best network weights as well as reduce the specified loss Function.

- **Loss Functions**: Selection of loss function is dependent on the machine learning task being performed. For the classification task, the loss-function shortened is usually the Binary Cross-Entropy loss [25][17]. For resource-intensive segmentation tasks, the output from the model (a binary fire mask) for which we are minimizing a complex loss that aims to contend with the challenges of the data distribution and localization. This complex function often includes terms such as Focal Loss (FL), which effectively addresses the extreme class imbalance of foreground and background in dense fire detection and segmentation problems [8]. Focal Loss has a scaling factor that down-weights the con-

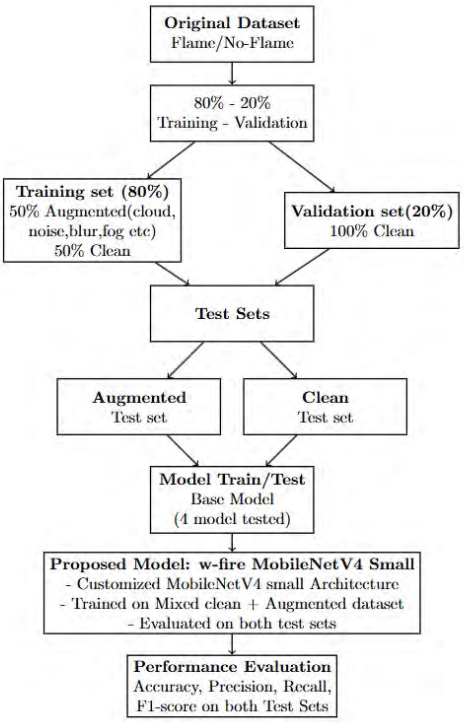

Figure 4.3: Classification Workflow

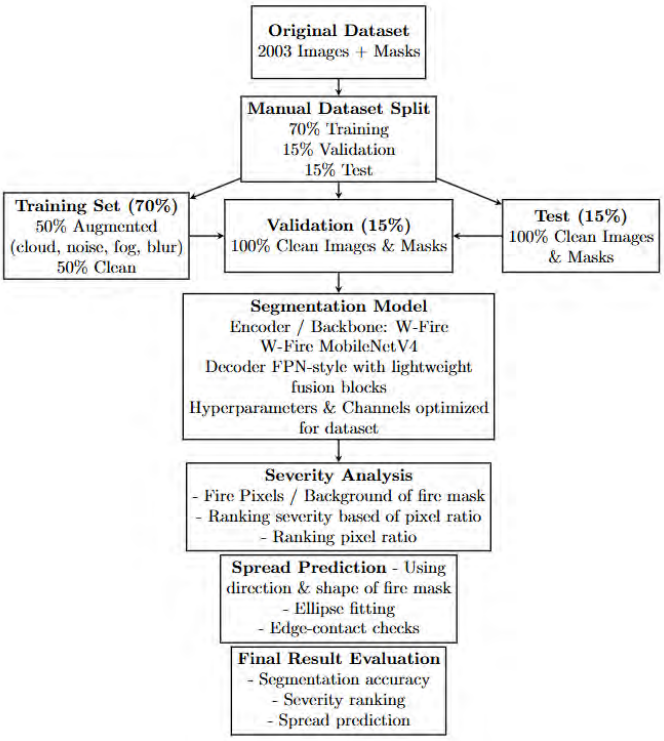

Figure 4.4: Segmentation Workflow

tribution of easy examples to the overall loss function and thus directs training on better hard, misclassified examples. The segmentation loss often has a Dice Loss component, is often weighted higher, improved the overlap/intersection-over-union of predicted masks, and ground-truth masks, which is the most important metric for measuring segmentation quality in terms of accuracy. Segmentation models may also have the binary cross-entropy loss component alone in the loss-function.

- **Regularization and Robustness**: There are several common ways to improve generalization and mitigate overfitting. Weight Decay (L2 regularization) was regularly used to shrink the parameter magnitudes during training to avoid the model over-fitting the training data [5]. To increase the robustness of the classifiers, Dropout layers were added to the networks with a typical drop probability of 0.5. These layers were only added after the final classifier heads of the CoAt-Lite, EdgeNeXt, and MobileNetV4 architectures. The training configurations were also set to often use Mixed Precision (torch.cuda.amp.autocast) training to exploit the GPU compute efficiencies. The models were trained, with a specific batch size and maximum number of epochs. Each model was allowed to train until the desired point and could continue based on the performance of the respective models on outside validation data and knowledge of the defined patience on a specified Metric Callback when there were no other meaningful changes in the approached metric(s).
- **Evaluation Metrics**: We use standard metrics for the domain to evaluate how well the deployed models work in the end:
  - **Classification**: Evaluated through- Accuracy, Precision, Recall, and F1-Score.
  - **Segmentation**: Segmentation needs more specific pixel-level metrics, such as Loss, Intersection-over-Union (IoU), Pixel-Accuracy, Precision, Recall, F1-Score, Sensitivity, and Specificity.

4.1.4 Classification Process

The classification process is a key element of the fire detection system developed in this thesis that identifies images containing fire ("Fire") and images without fire ("No_Fire") within the FLAME dataset. This process makes use of deep learning approaches and a variety of convolutional neural network (CNN) architectures, focusing primarily on MobileNetV4 architecture because it is efficient in running computations and offers flexible usability in resource-limited edge devices such as those commonly found in aerial surveillance. By examining multiple architectural types ranging from baseline to more customized designs, as well as providing empirical results of the models' comparisons, the classification identifies high performance and provides insights into the behaviors of the models when accurately identifying fire and/or determining false positives such as sunlight reflection and/or smoke.

The process of classifying the images starts with creating the model, which is one of the key steps. The model in this step is created using `timm.create_model`, with `pretrained` set to `false` so that we replace the last completely connected layer

so it can classify two classes, which is required for our binary classification problem. The model we are using, MobileNetV4 Hybrid Medium, is loaded with the `best_mobilenetv4_hybrid_medium_scratch_checkpoint.pth` checkpoint. The model is loaded by `torch.load`, and then also using `model.load_state_dict`, which makes clear that the model was intentionally and specifically trained or fine-tuned on the FLAME dataset to take advantage of features that were learned from the dataset.

In the notebook, we make sure to use `torch.device('cuda' if torch.cuda.is_available() else 'cpu')` so the code can change devices easily, meaning it can be run on different hardware. The model is put into evaluate mode, `model.eval()`, so that it will behave as expected for inference and not training. This means it needs to stop using dropout and stop updating the batch normalization parameters, which is all happening behind the scenes. In this section we will also be using the other variants we trained, including `MobilenetV4_conv_small`, and the customized `w_fire_mobilenetv4_s`. In the latter part, The classification well proven model was kept as a backbone or encoder and hyperparameter tuning which includes low learning rate, weight decay reduced channel, Loss function which included a very high pos_weight= 50 from BCE (within Focal) loss and Dice Loss was also used in this case in order to address class imabalance. After the inference phase is the next step where the model processes the test datasets with `torch.no_grad()` for memory-efficient inference (i.e., the model is not computing gradients, saving memory, particularly for edge applications where you could have thousands of test images and real-time inference). Below, the notebook defines the data loaders with `torch.utils.data.DataLoader`, specifies a batch size of 32, `shuffle=False` to ensure deterministic batch test conditions, and uses `num_workers=0` to prevent multi-threading issues to ensure consistency in testing. The model processes each batch and generates logits. Then, the logits are converted into class predictions with `outputs.argmax(dim=1)` for the current batch and the true labels are saved in parallel for computation of metrics for each testing dataset. The augmented test set was obtained from the "Test (Augmented Set)" section of the notebook and used augmentations to process thousands of images, including, but not limited, to Softening, Turbulence, Motion, Defocus blur, night effects, and Color jitter. This establishes a useful and traceable record of the transformation of each image that simulated real-world degradation. Thus, we can evaluate the model's performance for the clean testing conditions and in a variety of degradation conditions to establish the robustness of the model performance.

Fig 4.5(a) & Fig 4.6(a) : Defocus Blur+ Clouds+ Smoke on Fire + Speckle Noise.
Fig 4.5(b) & Fig 4.6(b)& Fig 4.5(d) : Gaussian + Clouds+ Smoke on Fire + Speckle Noise.
Fig 4.5(c) & Fig 4.6(d) : Motion Blur+ Clouds + Smoke on Fire+ Speckle Noise
Fig 4.6(c): Night Vision+ Median blur+ Clouds+ Smoke on Fire+ Speckle Noise

All the images of Fig 4.7 & Fig 4.8 are 100% clean Images.

As part of the augmentation process, the Flame Dataset No Fire Images that had

(a) (b) (c) (d)

Figure 4.5: Augmented test Fire Images

(a) (b) (c) (d)

Figure 4.6: Augmented test NoFire Images

Figure 4.7: Clean test Fire Images

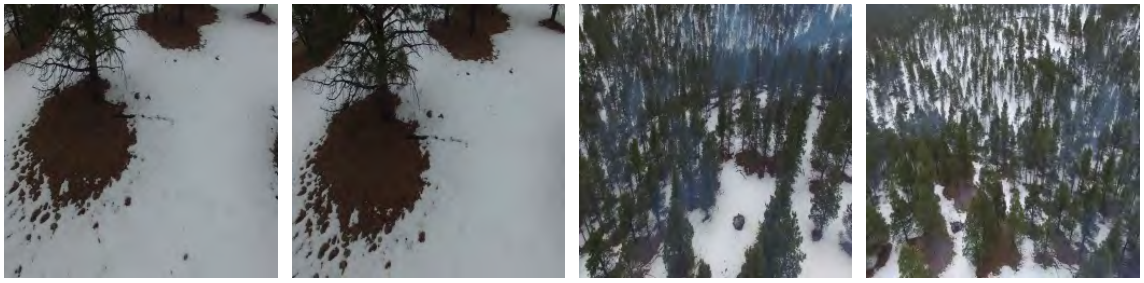

Figure 4.8: Clean test NoFire Images

different areas that weren't typically fire related but did contain small artifacts that looked like fire, e.g., yellow light sources, reflections of sunlight, or yellowish objects. These small areas often operated as false alarms, making the overall classification process even more challenging than it already was. We did realize that if some augmentations covered the crucial areas by accident, the classification wouldn't have any meaning. So we took care to apply augmentations with limited opacity to ensure they still provided some robustness to the dataset while preserving those areas. In comparing results to the FLAME dataset, we considered even tiny levels of improved performance as still valuable or worthy of completion for the purpose of assisting with the classification task.

Evaluation processes are a vital aspect of the process, having a complex set of metrics used to summarize quantitative performance using `sklearn.metrics`. The notebook estimates, calculates, and prints four metrics - accuracy, precision, recall, and f1-score - `zero_division=0`, to allow for precision and recall to be undefined cases, and to create robustness in performance metric calculations. A confusion matrix generates counts of: true positives, false positives, true negatives, and false negatives, and uses the visualization tool `seaborn.heatmap` to visualize misclassified instances, to better understand the nature of model behavior that is important for safety applications when a false negative (a missed fire) can have negative consequences. The evaluation is then done on the same augmented and clean test sets, to evaluate the model. It also logs detailed logs and artifacts (i.e., classification reports) such that metric evaluations per class are also generated.

Comparative evaluation is also a complex and interpretive portion, and the notebook does systematically evaluate the performance metrics, while identifying the best by architectural variant. `Mobilenetv4_conv_small` serves as a baseline model, there are trade-offs between the deep learning model using augmented data producing lower accuracy compared to the baseline model, however lower false positives is an indication of a more conservative prediction process. On the other hand, `w_fire_mobilenetv4_s` had a higher accuracy and improved F1 score and improvements likely from this model being optimized for fires, but again the model experienced a higher false positive, which may suggest this model was giving up some precision for recall, as detection of fires is preferred to few false positives, as an example, safety related applications may feel the cost of some number of false positives as acceptable in exchange for detecting all fires. The checkpoint for the hybrid medium offers a compromise, fine-tuned through the `w_fire` variant, as a base for further personalization. This comparison, again supported by confusion matrices plots, brings about refinements in methodology, such as adjustments to thresholds or feature engineering specifically to balance sensitivity and specificity, respectively. Post contraction supplies additional enhancement of the classification output by examining a confusion matrix to determine trends in misclassification, such as false positives (identifying a false fire caused by bright lights) and false negatives (false non-fires in cases where the fire was obscured). The notebook implements appropriate tuning of thresholds (the default is set at 0.5) and gives the user control to lowering it to improve recall, or increase it to minimize occurrences of false positives to conserve resources in a future potential operational environment. Integration with segmentation outputs (fire pixel proportion) could also factor into

the decision-making process, which better integrates frame level classification with pixel level localization to ensure continued and improved reliability in a multi-scale, aerial operational environment.

Many of the implications relating to performance emphasize the effectiveness of the classification process. The MobileNetV4 variants provided quick and effective deployment with the UAV, and the `w_fire_mobilenetv4_s` improved accuracy. Future improvements could include multi-modal training with thermal and images, or combinations of methods while continuing to improve fire management capabilities.

4.1.5 Segmentation Process

The segmentation process represents an essential part of the fire detection system developed in this thesis, aimed at identifying fire locations in the FLAME dataset aerial images at the pixel level, as a part of the classification process, allowing investigators to better pinpoint spatially the fire locations associated with individual UAVs piloted during monitoring sessions of pile burns. The segmentation process incorporates deep learning approaches and most notably the FPN style due to its encoder-decoder structure that is skilled at performing imagery semantic segmentation tasks. The process begins with preparation of the data using pairs of images and their corresponding masks that were preprocessed. Then initialize a model with light weights before the model training phase which used mixed-mode (training and eval) use of the model to use the available device time/resources efficiently. Followed by an inference (after the model training), and a validation phase, using mixed mode again as part of a pixel-wise validation, it published a segmented output image as well. This developing a robust segmented output (in different light conditions, and smoke occlusion) approach for evaluating the fire detection was informed by methods used in computer vision, when it addressed the key task of having the model predict the accurate boundary of the fire, - which also justifies the meaning of a high established precision of performance or degree of agreement symbolized by multiple Dice coefficients (the output also can be seen visualized, similar to Fig 4.9 alluding to a captured precision of the fire regions from all backgrounds). By employing iterative training and post-processing steps to optimize segmentation, this process not only enhances fire detection performance, but may also be extended to various downstream applications such as fire spread analysis, as we have confirmed to be effective to practically deploy in an aerial-based fire monitoring context.

The first stage of the segmentation process is characterized by preparing the data, 2003 images from the FLAME dataset are loaded and run through a series of preprocessing operations. Images are split by 70% for the training set, 15% for validation and 15% for test set. In the training set, 50% images were augmented and other 50% is clean and for both of the validation and test, all images are clean. The images have their sizes reduced to 224x224 pixels with the aid of the `torchvision.transforms.Resize` function, and they have also been normalized using ImageNet statistics with the help of the `transforms.Normalize` method. In tandem with this, the masks have been converted to tensors with the aid of `transforms.ToTensor()`, and they have also been thresholded so that they can be converted to binary for processing purposes. Data loaders have been care-

Figure 4.9: Ground Truth Mask

fully set up with the aid of `torch.utils.data.DataLoader` with a batch size of 16 and `shuffle=True` for the training stage, with the remaining parameters such as `num_workers` set at 0 in a deliberate attempt to ensure stable data loading and therefore properly preparing the dataset for input demands from the w-fire MobileNetV4 model. When it comes to model initialization, it involves the development of the w-fire MobileNetV4 model architecture. In this setup, `n_channels` is given 3 so that RGB input images can be used, with `n_classes` set at 1, which reflects the given binary segmentation task at this stage. The hardware is set up with the aid of the command `torch.device('cuda' if torch.cuda.is_available() else 'cpu')` so that it can be compatible with available hardware provisions and subsequently the model moved to the right computing hardware with `model.to(device)`. Subsequently, `model.eval()` is used for setting up the model's evaluation mode so that the model can be set up for inference purposes and therefore ensure that operations have been carried out effectively and without any gradient update attempts. The purpose of this training is to maximize performance and make best use of resources utilized in the notebooks training loop. The training uses a cross-entropy loss function that has been modified for binary segmentation tasks. The model is optimized using the Adamw algorithm with a learning rate scheduler implemented to dynamically modify the learning rates across epochs. A mixed precision training approach has been adopted using `torch.cuda.amp.autocast()` and `torch.cuda.amp.GradScaler()`, allowing the time taken to process the training on the GPU to be reduced and ultimately, memory usage. The model then trains on the training dataset, making a validation every epoch, logging the validation Dice coefficient, and saving a model checkpoint of the best performing model when the validation Dice increases. There is also logging that keeps track of the models performance, provide evidence of con-

vergence, and prevent over fitting.

Once trained, the inference uses the trained w-fire MobileNetV4 model on the test datasets in evaluation mode with `torch.no_grad()`. The `torch.no_grad()` function disables certain components of backpropagation optimizing memory when performing inference on large datasets while keeping torch's functionality. Inside the notebook, the test images are passed through the model creating predicted pixel-wise probability maps after evaluation. These maps are thresholded at threshold of 0.5 with `torch.sigmoid(outputs) > 0.5` to create a binary mask. Finally, the binary masks are compared to ground truth masks as one means of evaluation. Using ground truth masks as one means of evaluation is one phase for the evaluation of segmentation performance. Evaluation is conducted on both the clean test and augmented test sets, with detailed logs for traceability showing how well the model responds to degraded imagery and changes to fire shapes. This aspect is also important for applying the model in the field. Evaluation quantifies segmentation performance using pixel-wise metrics from `sklearn.metrics` and custom calculations of the Dice coefficient. The Jupyter notebook calculates the Dice score, precision, recall, and IoU (Intersection over Union) on the validation data set. The validation set reported a Dice coefficient of 0.89, indicating very strong overlap between the predicted masks overlaid and the ground truth masks. Pixel-wise biodiversity confusions and comparative metrics were also generated for further confounding class performance for true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN). The confusion and other comparative metrics were visualized as a confusion matrix using `seaborn.heatmap`, allowing evaluation of errors on segmentation comparisons such as over-segmentation of smoke as fire. The metrics and confusion matrix were also saved in logs, in addition to the visual outputs, to support future datasets requiring refinements such as thresholding or additional training examples to detect boundaries or other class separation.

Comparing the Dice scores from training iterations 0 to 9 with and without loading the best model as checkpoint, which resulted in increased performance, specifically, from a start of Dice scores of 0.75 moving up to 0.89 with loading carefully trained checkpoints, leading to a fine-tuning. Refining the segmentation outputs with post-processing is accomplished by viewing the confusion matrix and assessing where the most significant pixel-level errors occur. Segmenting added systems for combustion detections made as operational by fire (i.e., smoke detection and classification) helps keep the field clean of low-lives multi-class segmentation outputs, including processing with morphological operations. The notebook enables threshold adjustments (with a default of 0.5); lowering the threshold boots the recall from segmentation - thereby capture more fire pixels (from false positives) - or increase the threshold to reduce false positives from a segmentation outputs from a "cleaning" perspective. Further, the classification from what the network detected on 30 seconds per frame testing depicted a fire presence, contributing to a hybrid model of classifying segmentation results more reliable for a single-layer convolutional network for monitoring fire in a dynamic context.

4.2 Model Specifications

The design of the system is driven by the requirement for high detection quality and deployment efficiency on resource-constrained edge equipment to enable real-time UAV monitoring of essential wildfire situations. The system employs a two-pronged architecture: a very efficient model specific to the binary classification problem of Fire/No-Fire, and a powerful, context-aware segmentation model for precise fire boundary localization. The models are trained on aerial images, e.g., examples of the FLAME Dataset [17] In terms of choosing specific models, Our goal was to Choose a lightweight model with low parameter for further customization. We Kept One Base Model that had lower parameters compared to other model's base model and we wanted to see how the base model performs in our task.

4.2.1 Classification Model Architectures

The classification task meant selecting an ultra-lightweight, high-throughput backbone. The design process involved benchmarking a few of the current architectures in order to find the best and balanced deal between capacity and efficiency to learn subtle fire and smoke patterns.
The following models were tested to understand their computational trade-offs:

ViT-Tiny

Core Feature: The most significant strength of this model is the Multi-Head Self-Attention (MSA) mechanism. The MSA captures global dependencies and relationships between each patch by dividing the image into patches of a fixed spatial size (e.g., 16×16). Besides, it acquires a strong understanding of global contextual information across the entire scene [20]. (Fig 4.10)

Rationale for Testing: Vision Transformers' global reasoning capability is beneficial for the detection of sparse, early indicators like plumes of smoke and thin haze. ViT-Tiny (with 12 Transformer Encoder blocks being typically utilized) was tested to determine if such robust performance was feasible with a low parameter count viable for edge deployment.

CoaT-Lite-Tiny

Core Feature: It is a hybrid structure built with the goal of successfully Merging Convolution and Attention (CoAtNet). The model starts with Convolutional (Co) layers for local features extraction, which transition to Transformer (At) layers for global information collection [18]. This way, balance would be achieved, favoring the local specificity of CNNs while acquiring long-range perspectives through attention mechanism. (Fig 4.11)

Rationale for Testing: CoaT-Lite (often with a stage architecture of 3 convolutional stages and 2 transformer stages) was experimented with as an intermediate option to get sound accuracy against irregular fire shapes without the high computation of full ViT models.

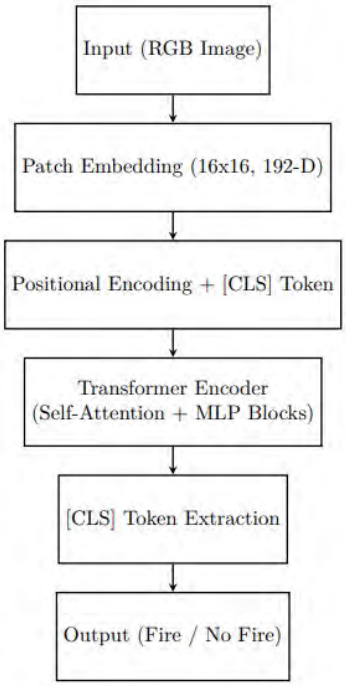

Figure 4.10: ViT-Tiny Architecture for Binary Fire Classification

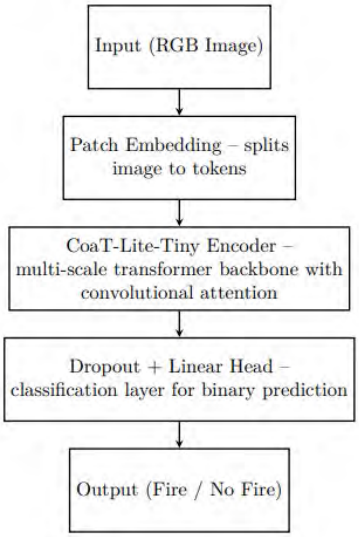

Figure 4.11: CoaT-Lite-Tiny Architecture for Binary Fire Classification

EdgeNext Base

Core Feature: EdgeNeXt integrates traditional CNN feature learning with domain-specific Split Depth-wise Transpose Attention (SDTA) units. SDTA block achieves low-cost computation through efficient usage of self-attention along the channel direction (transpose attention). The technique naturally enlarges the receptive field and captures multi-scale features while having linear complexity O(Nd2) in terms of input spatial size N and feature size d [15]. (Fig 4.12)

Rationale for Testing: The model was tested to be specifically effective in improving previous MobileNet-style architectures. Having SDTA blocks in its four-stage structure renders it strong in modeling global representations with a low number of Multiply-Add operations (MAdds), hence perfectly suited for low-latency mobile platforms.

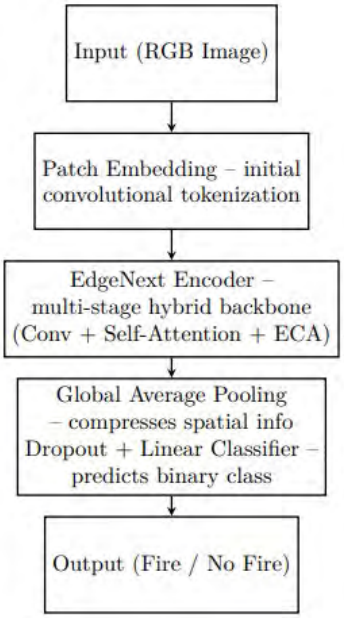

Figure 4.12: EdgeNeXt-Based Architecture for Binary Fire Classification

MobileNetV4_Conv_ Small

Core Feature: This is a small, lightweight Convolutional Neural Network (CNN) whose power derives from inverted residual blocks as well as depthwise separable convolutions, both of which offer a highly desirable balance between accuracy and computational cost. Because of this, it is an extremely good choice for cost-constrained edge devices. (Fig 4.13)

Rationale for Testing: It was tested as a high-throughput, low-latency baseline architecture to confirm its computational trade-off and serve as the foundation for the finalized customized W-fire model.

MobileNetV4_small: Chosen for Customization

MobileNetV4-small was selected as the final backbone because it consistently yielded the higher throughput-accuracy trade-off, which is what the target UAV platform requires.

Core Feature: Universal Inverted Bottleneck (UIB).
The model's efficiency is anchored in place by the Universal Inverted Bottleneck (UIB) search block [21] [26]. UIB is an extremely versatile, unified structure that generalizes a series of efficient convolutional blocks, such as Inverted Bottleneck norm to a single optimized building block. The UIB combines depth-wise separable

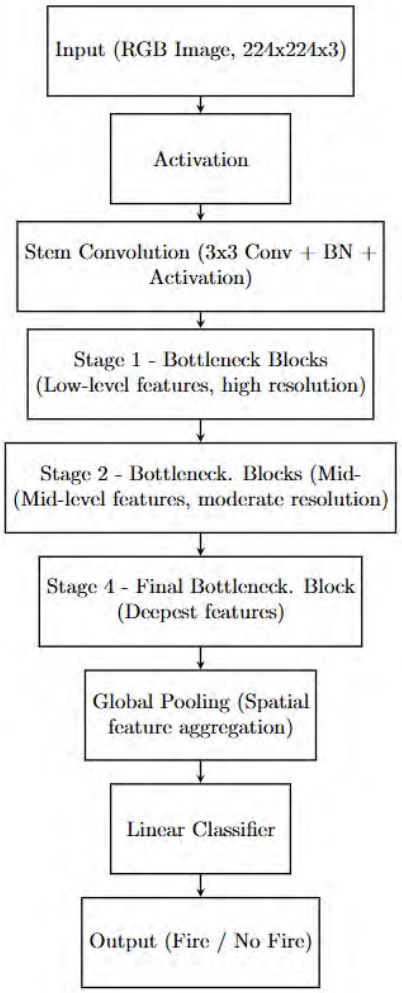

Figure 4.13: MobileNetV4_Conv_ Small Architecture for Binary Fire Classification

convolutions with an optional embedded attention mechanism (optimized for mobile accelerators) within a unified structure efficiently.

MobileNetV4's architecture is typically composed of 5 phases, where the UIB blocks account for the vast majority of its residual blocks. The UIB block configuration has emerged from an automated Neural Architecture Search (NAS), ensuring the configuration is optimized for maximum efficiency on the intended hardware. MobileNetV4's latency-conscious internal structure ensures that the system can deliver the necessary real-time performance requirements consistently while being subjected to high classification accuracy requirements.

Parameters Count & Inference Time : The MobileNetV4_Conv_Small model showcases a remarkable balance between model size and inference efficiency when used on a CPU. It contains 3,775,946 parameters (Fig 4.14), with classifier parameters 1,922. Here, the Inference time is calculated on the Test set on the CPU. In the final classifier layer, this indicates that most of the learning capacity is focused on feature extraction rather than classification. It also performs impressively fast. The average inference time per batch is only 0.054 seconds (Fig 4.15), demonstrating that it is ideal for a real-time or edge-based application that requires something

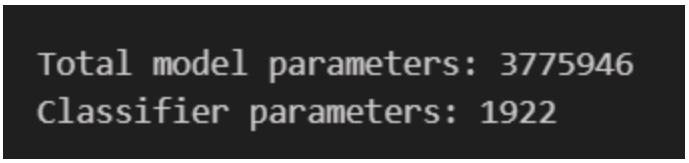

Figure 4.14: Parameters Count MobileNetV4_conv_small

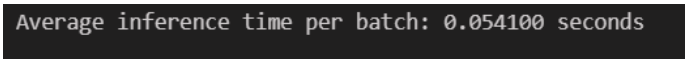

Figure 4.15: Inference Time MobileNetV4_conv_small

that can make fast and reliable predications, for example, querying drone or aerial imagery for wildfire detection.

4.2.2 Segmentation Model:W-fire_Mobilenetv4_small

To achieve its task of effective boundary fire detection, the system employs a semantic segmentation network built on the MobileNetV4-small model. This was done deliberately for segmentation precision at the potential expense of deadly False Negatives (misses) or fire misses. The architecture leverages aspects such as:

Atrous Spatial Pyramid Pooling (ASPP): The highlight of the model is the ASPP module, which pools multi-scale context information. The ASPP is done by using Atrous (Dilated) Convolutions with parallel different dilation rates (e.g., dilation rates of 6, 12, and 18) [9] [6]. These convolutions actually augment the receptive field exponentially without spatial resolution down-sampling, allowing accurate segmentation of fire areas with extremely irregular sizes.

Backbone: Using the MobileNetV4-small backbone provides a lightweight, effective feature extractor. The segmentation model diminishes the latter half of the backbone by replacing vanilla convolutions with Atrous Convolutions to preserve a high output feature resolution at the cost of not sacrificing precise pixel-level context.

This high synergy has been chosen particularly to meet the common class imbalance problem in the data, with fire pixels contributing only 0.0026% to the overall pixels. The model's ability to learn high, multi-scale context from the ResNet backbone guarantees that it will be capable of separating very sparse fire pixels efficiently without facing the most threatening False Negatives.

4.2.3 Training Configuration for the Segmentation Task

Data Augmentation Strategy

A full data augmentation pipeline was utilized to cope with real-world variability and sensor artifacts, and to compensate for the small size of the segmentation dataset of only 2,003 annotated examples.

Validation Split: An additional 20% validation set was kept aside to monitor generalization. Among 2,003 images in total, the division was made by putting the first 304 images in the test set, then the next 304 in the validation set, and the last 1,395 were copied and augmented in the same manner as the classification task. [2].

Targeted Preprocess: The offline Augmentations were same intensity and opacity as they were added for the classification task. While making sure the augmentations make the Images different and Robust. This is how the train set was made robust and increased data.

Loss Functions

For Classification (MobileNetV4-small): Cross-Entropy Loss was used because it is most suited for binary classification task.

For Segmentation (W-fire_Mobilenetv4_small_segmentation): A custom, hybrid loss was utilized to target the severe class imbalance:

Dice Loss: Emphasizes considerable overlap between predicted fire region and ground-truth mask, seeking to maximize proper classification of rare minority class (fire pixels) [25].
Focal Loss: The model was trained on Focal and Dice loss. The Focal component is utilized to re-weight samples to address class imbalance, focusing the model's training on hard-to-classify sparse fire pixels (instead of the less effective IoU Loss discussed in the original text) [14]. BCE Loss were added within the focal loss so that a big pos_weight can be implemented. To handle the extreme class imbalance.

Optimizers and Hyperparameters

Optimizers: Adamw [10] optimizers with momentum were employed. Adamw performed better initially because of its self-learning rates, and later on, generalized the model [1].

Learning Rate Schedule: Learning rates were chosen to be 8e-5 and dynamically decayed via a cosine annealing scheduler with smooth decay during training. While tuning the hyperparameters, we noticed that this is the lowest learning rate on the model could perform anything bigger than this it shows extreme poor performance as it gets stuck possibly in local minima and bigger learning rate than this cause severe overfitting.

Training Durations and Batch Sizes: Classification Models were trained for 50–70 epochs with batch sizes from 16 to 32.
Segmentation Models required longer training times of 80 to 100 epochs with reduced batch sizes (8 to 16) due to larger memory constraints.

4.2.4 Analysis of Design Solution

Our default classification model is MobileNetV4-Small, a model family designed as "universal" architectures that unify efficient convolutional and attention blocks into

a single, scalable architecture. The reference model employs a stack of Unified Inverse Residual (UIB) blocks, the main innovation in MobileNetV4, which trade off representational power and computational efficiency [26]. This forms a reasonable baseline for real-time fire/no-fire detection use cases.
However, the baseline model is insufficient for the highly imbalanced and visually challenging FLAME dataset [17] with small-scale fires and confusing background textures (e.g., snow). As a solution, we introduce several architectural modifications (Fig 4.16), guided by the following principles from literature: the employment of a smaller model is recommended for small-to-medium-sized datasets to prevent overfitting [15], and combining CNN's inductive biases with self-attention can improve generalization [21].

Our customizations, reached via laborious iterative experimentation, are as follows:

Stage 3 – CustomFLAMEBlock from Flame Dataset paper [17] with ECA Attention: We substituted Stage 3 baseline blocks with a CustomFLAMEBlock, which is a compound of two successive depthwise separable convolutions. The output is then purified via Efficient Channel Attention (ECA). One major strength of ECA is its efficiency in terms of parameters; it eliminates fully-connected layers of high computation in standard SE blocks, realizing performance improvements without much complexity [11]. This architecture notably forces the network to prioritize more on finer local fire structures, say small fires or the boundary of smoke, that are usually overlooked in the baseline.

Stage 4 – SandglassBlock with ECA Attention: Stage 4 replaces the baseline inverted residuals with the Sandglass Block, which is the default suggested setting of the MobileNetV4 UIB block [26]. This block reverses the standard mobile principle to assist in preserving more spatial information. We included an additional ECA module [11] here to even further refine the mid-level features. This operation enhances the capability of the model to capture contextual fire structures in wider areas, such as the ambient glow or out-of-focus flame edges. To have ECA both in Stages 3 and 4 is not redundant; it does direct refinement in different hierarchical levels of the representation.

Stage 5 – Dual Attention Network (DANet) Block: In the final stage, we have incorporated a DANet Block, which combines Channel Attention and Spatial Attention. The module combines local features and their global relations in an adaptive way by learning semantic interdependencies in spatial and channel spaces [9]. This incorporation is extremely crucial for removing false alarms. The channel attention mechanism ensures the most discriminative fire-related features are emphasized, and the spatial attention points out where in the image the discriminative information is, which helps to suppress bright background clutter.

4.2.5 Final Classifier

After attention-enhanced feature extractor, we have a light design. Global average pooling is used on the feature maps and then a single fully-connected layer, the best approach to retaining real-time performance while ensuring proper classification [15].

Improvements

Baseline MobileNetV4-Small : Efficient and lightweight but not strong enough to subtle fire signals and noisy scenes of the FLAME dataset [17].

Novel Classification Model :

Stage 3: CustomFLAMEBlock + ECA → boosts local fire feature extraction with good attention [11].
Stage 4: SandglassBlock + ECA → applies fundamental MobileNetV4 architecture for better contextual representation [26].
Stage 5: DANet → integrates channel and spatial attention to achieve strong discrimination under harsh environments by paying attention to global context [9].

These modules were not selected randomly; instead, they were created iteratively and maximized using repeated experimentation to achieve a maximum balance of accuracy, generalization, and computational cost and developed into the optimal setup for the specific problems of aerial fire imagery [17].

4.2.6 Novel Model W-Fire MobileNet V4 Small (Customization of MobileNetV4)

Our segmentation model is based on a lightweight classification backbone of the MobileNetV4-Small supplemented by a decoder head for fire mask prediction at pixel level. MobileNetV4-Small is natively designed for mobile and embedded applications and makes use of Inverted Residual Blocks with depthwise separable convolutions, expansion layers, and squeeze-and-excitation (SE) modules. These design decisions strike a balance between computation efficiency and representation capability and make for a suitable real-time fire detection baseline.
But baseline MobileNetV4-Small is not sufficient for highly imbalanced segmentation problems, like those with very small flames, smoke regions, and complex backgrounds. To address these shortcomings, we introduce customized architectural changes, carefully introduced at different network levels:

Encoder Head: W-Fire MobileNetV4_small(our proposed classification model).

Decoder Head: Multi-scale feature maps of the backbone are fed to a light-weight decoder with lateral convolutions and fusion blocks. Decoder progressively upsamples features and fuses multi-resolution information, preserving fine details for proper pixel-wise fire segmentation. Output is generated by means of a 1×1 convolution followed by upsampling to get the size same as the input image. It is identified as a Simplified Segmentation Head (Fig 4.17) , and is designed to transform the multi-scale feature maps learned by the backbone into a high-resolution pixel-wise fire mask. It subsequently applies lateral 1×1 convolutions to every encoder feature map, downsampling channel dimensions and aligning features across stages. The features are then merged more and more in a top-down manner, where higher-level

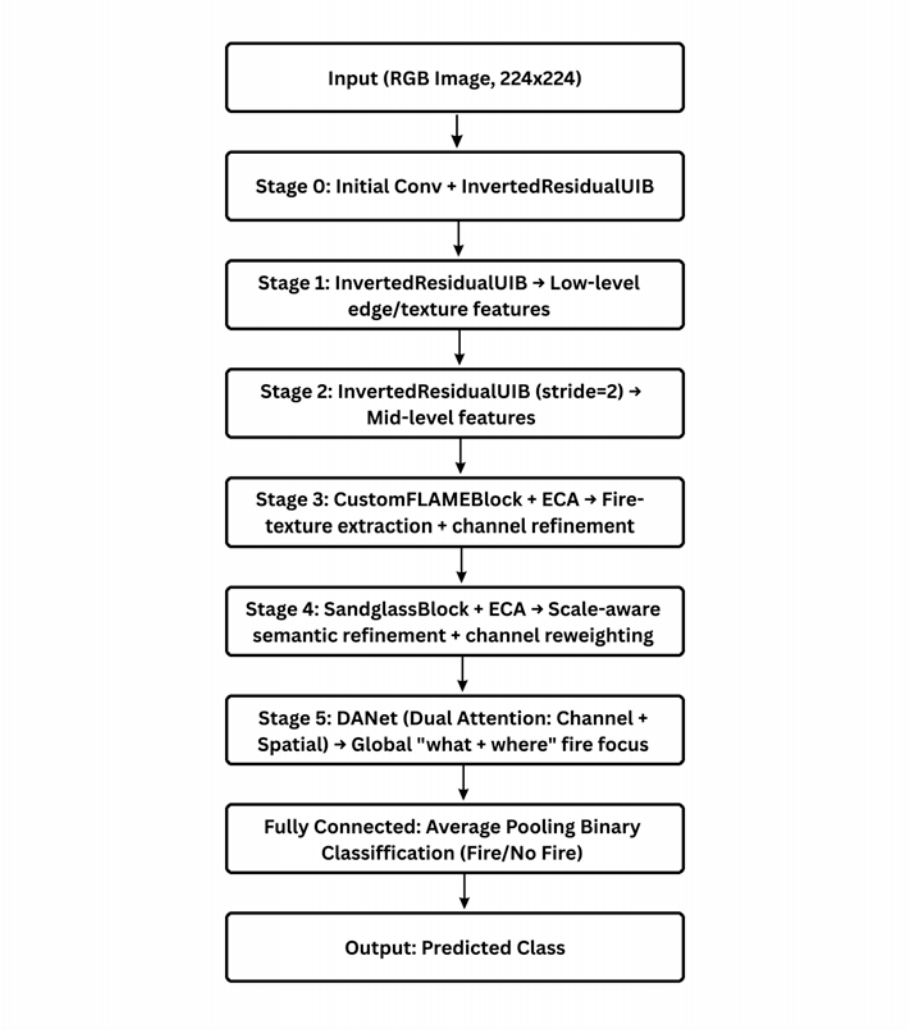

Figure 4.16: Classification Novel Model Architecture

semantic features are upsampled and merged into lower-level spatial features to maintain fine-grained details of small and irregular fire areas. To enhance feature representation and prevent overfitting, each step of fusion incorporates a lightweight depthwise separable convolution block with dropout. Finally, a series of convolutions and dropout layers generates the segmentation result, which is upsampled to the original image size. With this architecture, both global context and local details are well-retained, and therefore precise segmentation is obtained even under complex fire conditions.

Loss Function: The model has as a component a Balanced Focal Dice Loss which combines Focal Loss for class imbalance and Dice Loss for overlapping segmentation. It ensures that the model focuses on sparse fire pixels yet generates smooth, accurate masks.

Customized Model

Encoder (Backbone):
Step 3: CustomFLAMEBlock + ECA → improves local fire feature detection.

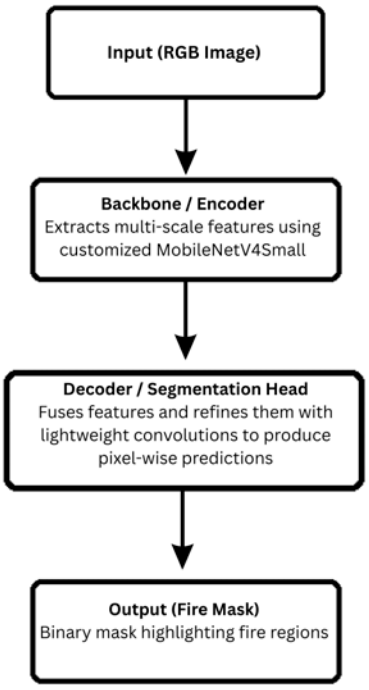

Figure 4.17: Segmentation Novel Model Architecture

Step 4: SandglassBlock + ECA → improves mid-level contextual feature perception.
Step 5: DANet → combines channel and spatial attention for precise segmentation.
Decoder:
Lightweight fusion and upsampling → preserves detailed pixel information.
Loss: Balanced Focal Dice Loss → deals with extreme class imbalance and quality of the mask.

Again, these changes were made and adjusted iteratively to balance segmentation accuracy, generalization, and computational complexity. The resultant architecture achieves robust detection of small and complex fire regions under challenging viewing conditions.

Parameters Count & Inference Time : The w-Fire MobileNetV4_small framework has efficient in number of parameters (1,461,581 parameters in total) and core functionality across its six stages. Stage 0 is quite low in size with 278 parameters, offering a minimal processing of input. Stage 1 holds 12,128 parameters and processes low-level local features, while Stage 2 captures data in 45,281 and Stage 3 with 94,729 was able to gather additional spatial attention and structured features. Most of the parameters lie in Stage 4 with 1,292,169 parameters; Stage 4 learns representations based on just above high level local patterns and contexts. Here, also in both cases, the Inference time is calculated on the Test set on CPU. Since our goal is to deploy our model on edge devices, it is necessary to evaluate the inference time on CPU. Our model uses depth-wise convolution layers, Attention modules, and there are intermediate activations such as RelU, Hardwish. Altogether, these things make the model lighter-weight, and more accurate; however slight constraint is the inference time. With 16,482 parameters, Stage 5 is simply the output and classification

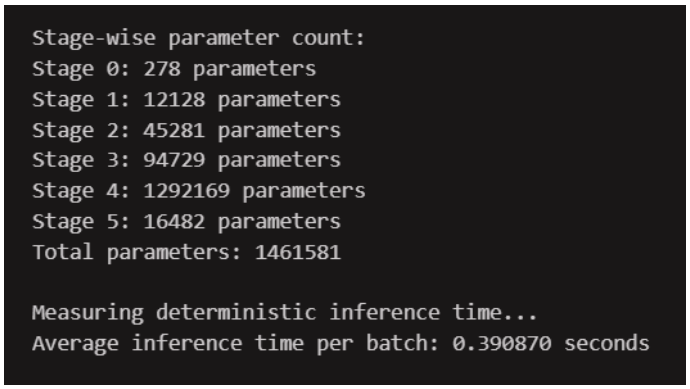

Figure 4.18: Parameters count & Inference Time for The W-Fire MobileNetV4_small

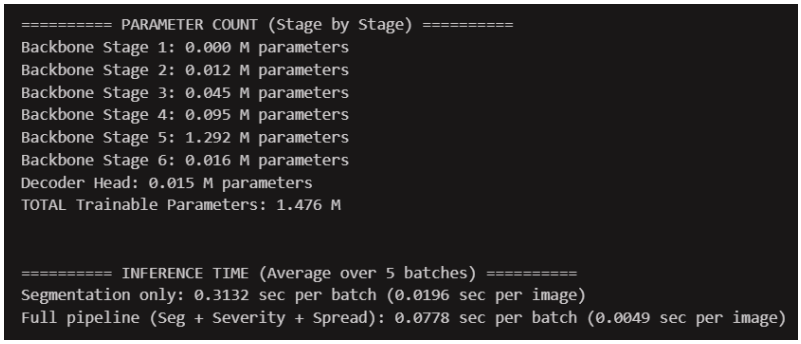

Figure 4.19: Parameters count & Inference Time for The W-Fire MobileNetV4_small

stage. Given this tiered panel of stages, the w-Fire MobileNetV4_small framework was able to effectively compute towards a modest overall run time of an average of 0.39 seconds a batch (for classification). That demonstrates (Fig 4.18) a sufficient trade-off, between computational efficiency and model capacity for real-time wildfire detection processes or applications.

Moreover, the model results (Fig 4.19) in 0.3132 seconds per batch (0.0196 seconds per image) for segmentation only and completes the entire pipeline, including segmentation, severity estimation, and spread prediction, in 0.0778 seconds per batch (0.0049 seconds per image) total. These results show a good balance, between model size, accuracy, and real-time inference.

4.2.7 Evaluation Metrics

A separate assessment was conducted for the classification and segmentation models, using both qualitative and quantitative assessments.

Classification Metrics

Accuracy

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.1}$$

This equation (4.1) reports the proportion of correctly classified fire and non-fire images. Although good to report the performance in aggregate, it may be deceptive in the case of class imbalance. The names used in the metrics are defined below:
TP (True-Positive): Number of correctly-classified images as fire.
TN (True-Negative): Number of non-fire images which are accurately labeled as non-fire.
FP (False-Positive): Number of non-fire images which are inaccurately labeled as fire.
FN (False-Negative): Number of fire images which are inaccurately labeled as non-fire.

Precision

$$\text{Precision} = \frac{TP}{TP + FP} \tag{4.2}$$

Reflects (eq (4.2)) how consistent fire detections are. High precision is required in order not to trigger false alarms, which can otherwise trigger unnecessary emergency calls.

Recall

$$\text{Recall} = \frac{TP}{TP + FN} \tag{4.3}$$

Reflects (eq (4.3)) the quality of how well the model captures actual fire incidents. High recall ensures that actual wildfires are rarely left out, and that is vital for safety reasons.

F1-score

$$F1\text{-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4.4}$$

Is a balanced measure (eq (4.4)) between precision, and recall, particularly beneficial in class-imbalanced datasets.

ROC-AUC

Verifies the discriminative capability of the classifier across numerous thresholds. The higher AUC, the better separated the fire and non-fire classes are.

These metrics collectively aim to have the classification model generate strong detection, compromising on preventing false alarms (high precision) vs. having the ability to detect all genuine wildfire events (high recall).

Segmentation Metrics

Intersection over Union (IoU)

$$\text{IoU} = \frac{|P \cap G|}{|P \cup G|} \tag{4.5}$$

A measure (eq (4.5)) of overlap of predicted and actual fire regions in order to deliver precise boundary detection. For this reason, the variables are as below: P: The collection of predicted fire pixels. G: The ground-truth set of fire pixels.

Dice Coefficient

$$\text{Dice} = 2 \cdot \frac{|P \cap G|}{|P| + |G|} \tag{4.6}$$

Similar to IoU but more attentive to small regions. Dice (eq (4.6)) prevents thin or fragmented fire areas from being neglected.

Pixel Accuracy

$$\text{Pixel Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{4.7}$$

Calculates (eq (4.7)) the ratio of the correctly-classified pixels. Useful but biased due to the dominance of background pixels, thus it is read along with IoU and Dice. The pixel-level classification variables that unique to it are:
TP (True-Positive): Number of fire pixels that are correctly-predicted.
TN (True-Negative): Number of background pixels that are correctly-predicted.
FP (False-Positive): Number of background pixels incorrectly-predicted as fire.
FN (False-Negative): Number of fire pixels that were not predicted.

These metrics overall guarantee that the segmentation model not only detects fire pixels, but also detects their boundaries, and shapes with great accuracy, even in congested surroundings.

Integrated Role of Metrics

By incorporating classification-metrics (Accuracy, Precision, Recall, F1, ROC-AUC), and segmentation-metrics (IoU, Dice, Pixel Accuracy), the system achieves a dual-level evaluation. The classification model ensures reliable detection of wildfires, while the segmentation model ensures accurate localization of fire zones. Therefore, both combined, help ensure the ultimate purpose of wildfire risk and severity analysis in the system, with minimal missed detections, and maximum spatial accuracy.

Chapter 5

Result Analysis

5.1 Performance Evaluation

5.1.1 Model Evaluation and Training Setup

Evaluation Framework

The performance assessment of wildfire classification models includes a detailed strategy and systematic process for correctly assessing binary classification performance for training, validation, and test data. The performance evaluation shows the model would be able to distinguish flame images (the positive class) from non-flame images (the negative class), which is an essential requirement of the trustworthiness of early wildfire detection systems. The evaluation relies on conventional machine learning measures of model performance, utilizes early stopping, and an adaptive learning rate strategy to prevent overfitting. All data processing will be completed on GPUs, with CUDA acceleration where applicable. The method of assessment includes the common machine learning metrics and incorporates different overfitting avoidance mechanisms. These mechanisms cover mechanisms such as early stopping, whereby training is terminated when no improvement in performance occurs, and learning rate scheduling with adaptations that changes the learning rate adaptively. In addition, all computational operations support hardware that is GPU-enabled and leverage CUDA acceleration wherever it is supported for performance boosts.

Reproducibility Protocol

The reproducibility is therefore adequately taken care of via a systematic and careful control over all stochastic operations that take place within the training pipeline. To this end, a static random seed, which has been set as 42, is consistently used across different environments ranging from Python, NumPy, and PyTorch, such that there is no variation in the randomness across these media. Besides this, deterministic execution in CuDNN operations is stringently enforced so as to obtain a high degree of consistency across the process. The reproducibility of the loading of the data is obtained by setting the DataLoader argument `num_workers = 0` explicitly, which is used for the removal of any randomness that might be induced due to issues related to threading. Moreover, model initialization, optimizers' state, and learning rate schedulers' operations all fall under the purview of identical deterministic seed configuration, so as to instill an atmosphere of uniformity. Though these exhaustive

measures come in place so as to ensure highly reproducible results within an environment identical at both hardware and software levels, it must be taken note of that tiny variations might occur anyhow. These minute variations might take place due to intrinsic properties of arithmetic operations on the basis of floats, complexities due to mixed-precision operations, or hardware-based specific optimisations used.

Model Selection and Optimization Strategy

The evaluation scheme entails consideration of several dependent factors such as model architectural complexity, the number of parameters, and dataset properties so that proper performance evaluation is achieved. Model choice is influenced by thorough examination of convergence behavior by tracking the loss trend as well as classification performance based on conventional evaluation measures. The hyperparameter optimization scheme is a tradeoff between training stability, ability for generalization, and computational demands necessity. Hyperparameter tuning encompasses key training parameters, such as batch size, epochs, learning rate, weight decay coefficient, as well as early stopping patience threshold values. The optimisation procedure targets stable training and validation learning curves and avoids instances of overfitting and underfitting scenarios. In instances where models from a particular architectural family also showcase training instability, smaller counterparts with fewer parameters are chosen with the aim of accommodating better the limitations imposed by the scale of the dataset. The entire evaluation pipeline involves data preprocessing and augmentation operations directly embedded within the model definition, applies stringent optimisation procedures with matching stability checking, and concludes with an overall performance analysis on all metrics taken for evaluation purposes.

5.1.2 Individual Model performance Results For Classification Task

Classification Model Performance Overview

Regarding the classification task we framed it as a binary classification of fire/non-fire. The experiments were run on a custom dataset, created from the Flame Dataset [17]. The FLAME dataset [17] was seeded with an intentional domain shift between training/valid and test data, to specifically to assess model generalizability. This method achieves an intentional domain shift, such that the models are being assessed on true detection ability in realistic and difficult contexts, rather than just memorizing visual patterns from only one instance in controlled settings [20]. These differences indicate that the models can learn well, while also showing how difficult and challenging the test data is in providing a true generalization process [20]. The significant gap in performance between validation and test sets comes from an intentional domain shift from changing lighting conditions, and snow covered backgrounds, which requires training strategies to ensure improved out-of-distribution robustness [18].
Although the training data involved a singular prescribed burn event, showing clear flames and consistent conditions, the testing data involved smoldering pile burns at multiple locations and on multiple days, featuring only small flames, inconsistent lighting, and differing forested backgrounds. This difference was intended to ensure

the models were evaluated on their general ability to recognize fire, rather than simple memorization of visual patterns learned from one training data example of a controlled event. The training and validation datasets showed that the models generalized the training dataset very well and also validated the validation dataset nearly perfectly at 99% but did not show the same performance for the test dataset. A further complication exists because of the situation of snow in the fire images in our dataset. While the training and validation fire images infrequently featured snow, the majority of the no-fire images did not. However, the test dataset only included images of snow as part of the no-fire class, adding a high degree of domain shift to the fire detection problem. This design emulates a more difficult classification task because the model would have to make distinguishing fire from no-fire regions under visually confounding snow conditions and further adds to the capacity to assess the robustness, and generalization capabilities of the model. All of these factors contribute to the reasoning for the patterns in performance we observed: While all models achieved nearly flawless accuracy on the training, and validation sets, the domain shifts present in the test set (smoldering pile burns, different lighting, backgrounds, and snow-only no-fire images) resulted in a reduced level of performance at test time. This indicates the models are very strong learners; it also demonstrates the new level of difficulty and challenge for generalization, presented to the models in the test data.

Initially in our investigation we would like to assess the original FLAME dataset [17] and our custom dataset, with, in particular, the focus whether our changes, which included augmenting approximately 50% of the images, are sufficient to increase robustness [18] by enabling the model to pay more attention to the most relevant regions. Subsequent to applying the split on both datasets, we trained a lightweight model, MobileNetV4, model first on FLAME dataset only, and then on our custom dataset. We made two test sets for evaluative purposes, but intentionally did not want to test the FLAME dataset on the augmented set since, we realize that continuous evaluation on a test set from a different distribution (i.e., images from different lighting the conditions, different source, some different objects), even if unintended, carries a risk of committing the opposite, but equally as problematic, error of invalid evaluation [20].

Flame Dataset Result Evaluation

MobilenetV4 conv small :
Train vs Valid (Accuracy & Loss Graph) : The dynamics of the training process (Fig 5.1) for the MobileNetV4 Conv Small models are clear from the training and validation plots, showing an incredibly quick and effective learning curve throughout all epochs represented, with Training Accuracy increasing from a starting point below 92% and finishing at approximately 99.5%. Overall, the model had a pronounced beginning, as the numbers presented were all high and it settled at numbers below 99.4%. The Loss in both training and validation datasets also quickly reduced throughout the epochs, starting at 0.20 (training) and 0.08 (validation), finishing at, around, 0.01, which overall establishes significance, as it is reported that Loss is an adequate measure of performance but should be interpreted in tandem with Accuracy. When combining this fairly even performance, as well as low Errors for both training and validation, validates that the model was

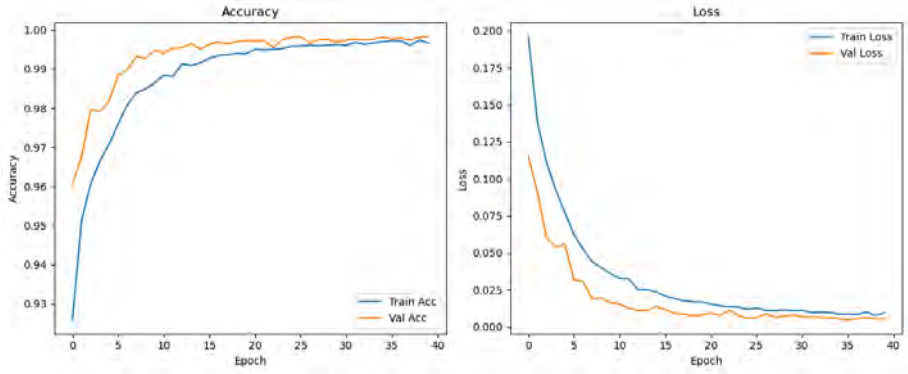

Figure 5.1: MobileNetV4 on Flame Dataset

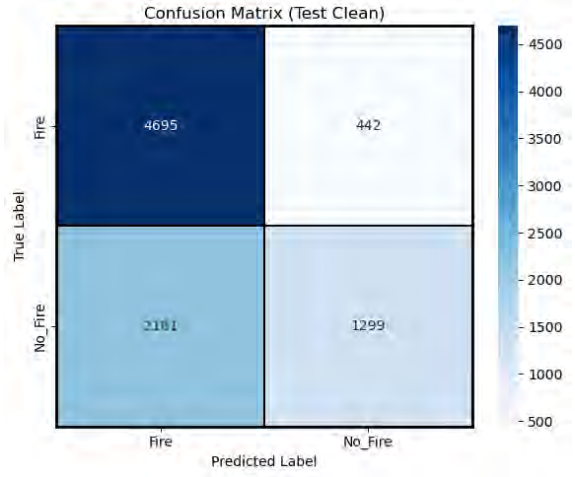

Figure 5.2: Confusion Matrix (Flame)

trained appropriately, thus exhibiting promising transferability performance due to being able to demonstrate above-average performance on both datasets while effectively classifying both class entities binary given the performance measures applied.

Evaluation on Test set : The numerical indices presented in the Confusion Matrix (Fig 5.2) confirm mobileNetV4 Conv Small model preference for the 'Fire' class - capturing the number of 'correctly' identified true positive scores at 4695, and false negatives at 442, meaning the model is efficient at preventing misses on fire. However, high sensitivity scores created also generated a high identified count of positive False Positives with a count of 2181 - which is a large indicator of false identification, and is a definitive, empirical indicator that the model had a tendency to generate an abundance of false fire warnings. The result reflects an optimization choice that favors maximum risk detection over predictive accuracy.

The Classification Report Heatmap (Fig 5.3) provides the metrics generated from the model that measure the performance trade-off. The Recall for Fire (0.91) is remarkably high and therefore legitimises the motivation behind the model's primary outcome of attempting to achieve full identification of all actual fire images. Despite this positive Recall performance, the Precision for Fire (0.68) is consequently diminished by a significant number of False Positives, which are the measure of the

Figure 5.3: Classification Report Heatmap (Flame)

amount of error warnings to the observer. The particularly low Recall for No_Fire (0.37), is the only measure that definitively indicates the high amount of erroneous warnings, as it confirms difficulty in validating No_Fire images, while confirming the logic of Early Warning at the cost of selectivity predicting true positives.

Final result : Our custom Dataset generalized the training data really well and also validated on a good range and eventually this below(table 5.1) result was achieved that mentioned in.

Metrics	**Results**
Accuracy	0.6956
Precision	0.7084
Recall	0.6956
F1-Score	0.6669

Table 5.1: Final Results on Flame Dataset

Customized Flame Dataset Result Evaluation :

1. **MobilenetV4 conv small :**
   Train vs Valid (Accuracy & Loss Graph) : The learning progression for the MobileNetV4 Conv Small model is distinctly captured in the training and validation plots (Fig 5.4), confirming a rapid, effective acquisition of knowledge across the 16 epochs. Training Accuracy exhibited a swift upward trajectory, initiating above 91.5% and culminating near 99.5%. In a corresponding manner, the Validation Accuracy closely followed this performance, starting higher and achieving a maximum plateau of 99.4%. At the same time, the Loss started above 0.20 (Training) and 0.08 (Validation) and dropped sharply for both sets, with final Loss values around 0.01 (indicating very low Loss). The tight proximity between the final accuracy curves, alongside the minimal error rates, furnishes compelling evidence of robust model resilience and validates the model's transferability to unfamiliar data, thereby confirming its capability for highly accurate and consistent binary classification.

   Evaluation on Clean Test set :

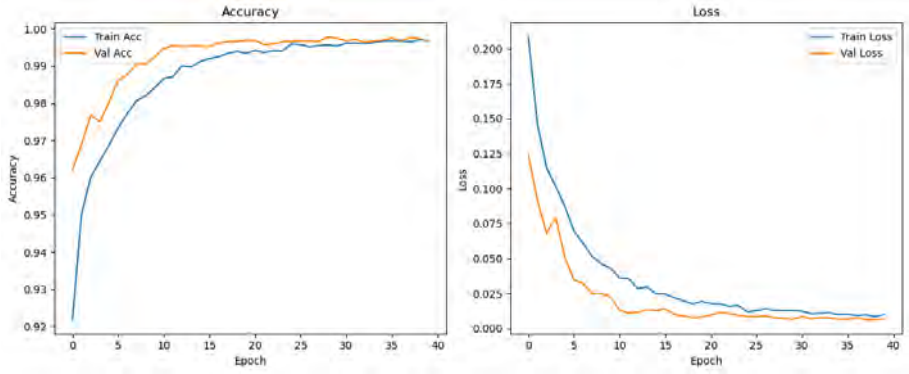

Figure 5.4: MobileNetV4 Accuracy vs Loss

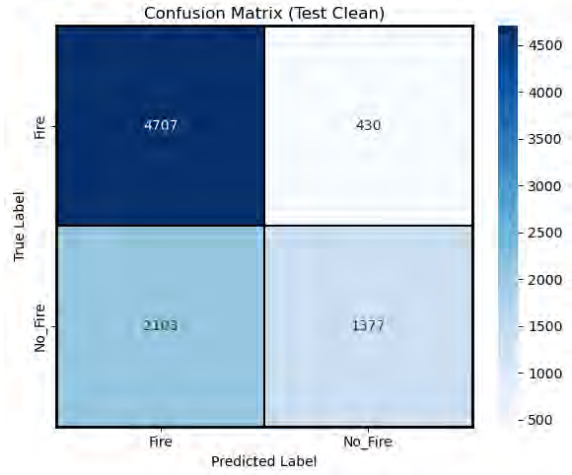

Figure 5.5: MobileNetV4 Confusion Matrix (Clean)

The raw counts within the Confusion Matrix (Fig 5.5) provide clear statistical evidence of the MobileNetV4 Conv Small model s dominant tendency toward the 'Fire' classification. The system successfully secured 4707 True Positives while maintaining a very low count of 430 False Negatives, thus confirming its supreme ability in preventing any missed fire alerts. This hyper-vigilance, however, resulted in 2103 False Positives. This substantial volume of incorrect fire declarations is the primary observable drawback of the model, signifying an architectural choice to maximize protective coverage at the direct cost of output exactitude.

In, Fig 5.6, the Recall for Fire (0.92) is exceptionally high, which conclusively reinforces the model's design goal of achieving near-universal identification of all genuine fire events. This impressive recall, however, is coupled with a modest Precision for Fire (0.69), a value substantially reduced by the volume of False Positives. Moreover, the extremely low Recall for No_Fire (0.40) is the metric that decisively attests the system's proclivity for false alarms, highlighting its extreme difficulty in appropriately detecting and rejecting non-fire images.

Figure 5.6: MobileNetV4 Heatmap (Clean)

Final Result : Our custom Dataset generalized the training data really well and also validated on a good range and eventually this below result (table 5.2) was achieved.

Metrics	**Results**
Accuracy	0.7060
Precision	0.7198
Recall	0.7060
F1-Score	0.6801

Table 5.2: MobileNetV4 Final Results on Clean test

Evaluation on Augmented Test set : The quantitative results in the Confusion Matrix (Fig 5.7) with True Positive=4689, False Positives=1908 strongly convey the model's pronounced inclination toward declaring the 'Fire' state. The model achieved 4701 True Positives and only 448 False Negatives, confirming a unique ability to avoid false negatives. Conversely, this elevated sensitivity generated 1572 False Positives. This considerable number of incorrect fire declarations is the most tangible consequence of the model's operational mandate, which clearly favors maximum protective coverage over absolute classification purity.

The Classification Report Heatmap (Fig 5.8)produces Fire Recall=0.91, No_Fire Recall=0.45 furnishes the metrics that fully delineate the performance consequences of this high-recall strategy. The Recall for Fire (0.91) is highly commendable, providing strong validation for the model's core objective of achieving near-total hazard identification. This high recall is balanced by a moderate Precision for Fire (0.78), which is impacted by the high False Positive outcome. The particularly low Recall for No Fire (0.45) is the important metric that clearly indicates the model's tendency to raise false alarms and implies it struggles with identifying images that do not contain fire accurately.

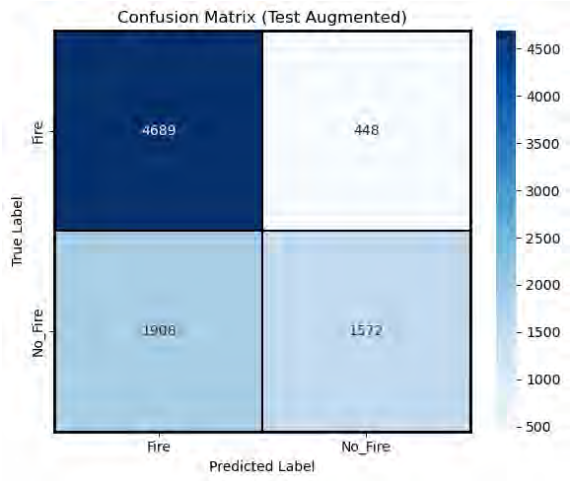

Figure 5.7: Confusion Matrix MobileNetV4 (Augmented)

Figure 5.8: MobileNetV4 Heatmap (Augmented)

Final Result : After being trained on 50% augmented data the model performance on the augmented set is below in table 5.3

Metrics	**Results**
Accuracy	0.7266
Precision	0.7380
Recall	0.7266
F1-Score	0.7073

Table 5.3: MobileNetV4 Final Results on Augmented test

2. **Coat-Lite Tiny :**
Train vs Valid (Accuracy & Loss Graph) : The training, and loss curves of the CoaT-Lite-Tiny model (Fig 5.9) show smooth convergence and strong generalization. Training accuracy increased from $\sim$ 92% to above 99.5%, with validation accuracy following closely and reaching 99.5% at the final epoch. Training loss decreased from 0.20 to below 0.02, with validation loss following the same decline and remaining slightly lower for subsequent epochs. These results indicate stable optimization, minimal overfitting, and well-tuned hyperparameters, leading to near-perfect classification performance on the binary

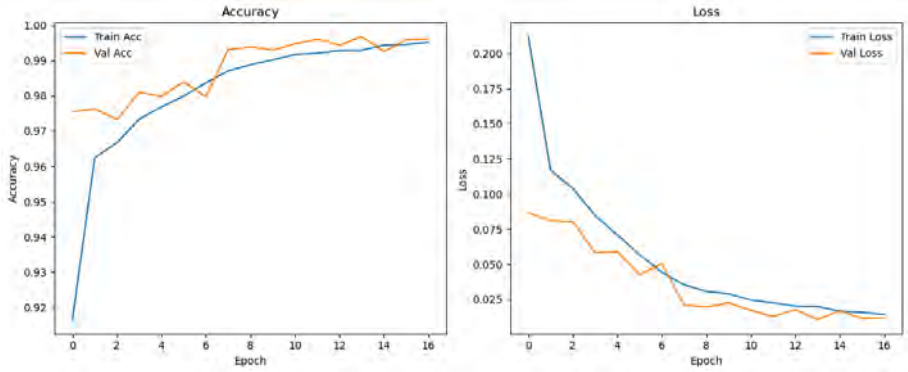

Figure 5.9: CoaT-Lite Accuracy vs Loss

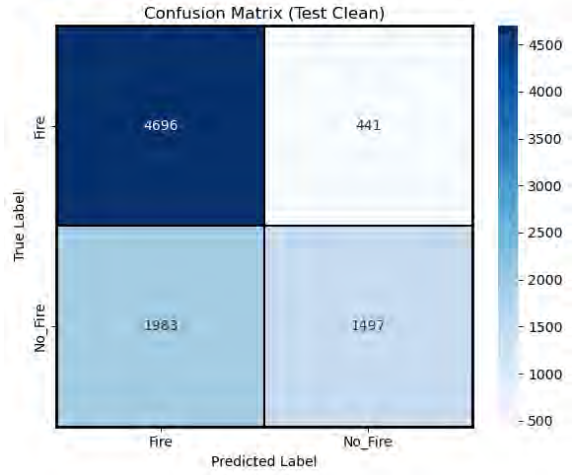

Figure 5.10: Confusion Matrix CoaT-Lite Tiny (Clean)

fire/no-fire task.

Evaluation on Clean Test set :
The Confusion Matrix (Fig 5.10), provides the raw counts that directly indicate the model's strong bias toward predicting 'Fire' on the clean test set. The model achieved a high number of True Positives (4696) while keeping False Negatives (441) low, confirming its success in minimizing missed fire detections. However, this sensitivity resulted in a large volume of False Positives (1983)—where non-fire images were incorrectly flagged as fire—which is the direct evidence of the model frequently raising a false alarm. This result shows a clear optimization for completeness of fire detection over the accuracy of prediction.

The Classification Report Heatmap (Fig 5.11), quantifies the trade-off established by the bias apparent in the confusion matrix. The high Recall for Fire (0.91), confirms the model's goal to detect nearly all actual instances of fire. This high recall comes at the expense of Precision for Fire (0.70), which is reduced by the numerous False Positives. Most critically, the low Recall for No_Fire (0.43) directly attests to the frequency of the false alarms, showing the model significantly misclassifies true non-fire images, reflecting a strategic

Figure 5.11: CoaT-Lite Heatmap (Clean)

design to prioritize early warning at the cost of prediction specificity.
Final Result : As discussed in table 5.4, despite strong generalization and near-perfect performance on the validation set, a subtle drop in test performance was observed due to the challenging and diverse conditions present in the FLAME dataset [17], which rigorously evaluates model robustness.

Metrics	**Results**
Accuracy	0.7187
Precision	0.7311
Recall	0.7187
F1-Score	0.6970

Table 5.4: CoaT-Lite Final Results on Clean Test

Evaluation on Augmented Test set :
The Confusion Matrix (Fig5.12) for the test augmented set consistent bias toward predicting the "Fire" class (Positive) is demonstrated by the augmented set. The model ensures that very few real fires are missed by effectively maintaining a high-number of True-Positives (4701), and a low count of False-Negatives (436). However, the trade-off persists: the model produced a high number of False Positives (1845). While this FP count is slightly lower than the Clean Test set (1983), it remains the primary source of the frequent false alarms, demonstrating the model is still highly optimized for maximal fire detection.

The Classification Report Heatmap (Fig 5.13) quantifies the high-recall, low-precision strategy on the augmented test data. 92% of all real fires are successfully detected, according to the remarkably high Recall for Fire (0.92). The overall F1-score for Fire is greatly increased to 0.80 by this high sensitivity. Conversely, the Precision for Fire (0.72) is constrained by the numerous False Positives. The Recall for No_Fire (0.47) remains low, explicitly confirming the model's tendency to generate false alarms, although it shows a minor improvement over the Clean Test set (0.43).

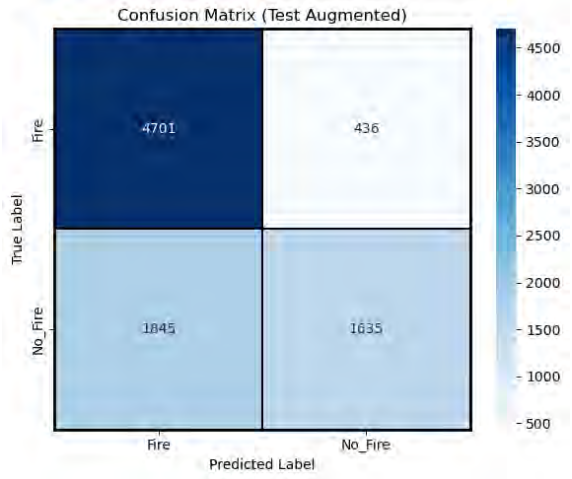

Figure 5.12: Confusion Matrix CoaT-Lite Tiny (Augmented)

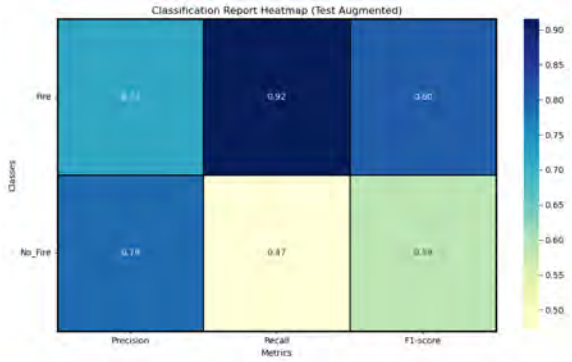

Figure 5.13: CoaT-Lite Heatmap (Augmented)

Final Result : Although a very similar result (table 5.5) drop was found in Augmented Set as well, The model performed better on the Offline Augmented test set.

Metrics	Results
Accuracy	0.7353
Precision	0.7470
Recall	0.7353
F1-Score	0.7177

Table 5.5: CoaT-Lite Final Results on Augmented Test

3. **EdgeNext Base :**
   Train vs Valid (Accuracy & Loss Graph) :
   he training and validation curves clearly show the EdgeNeXt Base model's convergence behavior (Fig 5.14), indicating dependable performance and outstanding generalization over the course of 36 epochs. Training Accuracy increased steadily, peaking at about 92% and plateauing at just under 99.8%. This increasing trend was reflected in Validation Accuracy, which reached an impressive stabilization point at about 99.5%. Concurrently, the Training Loss underwent a steep reduction from 0.20 to a low of approximately 0.005,

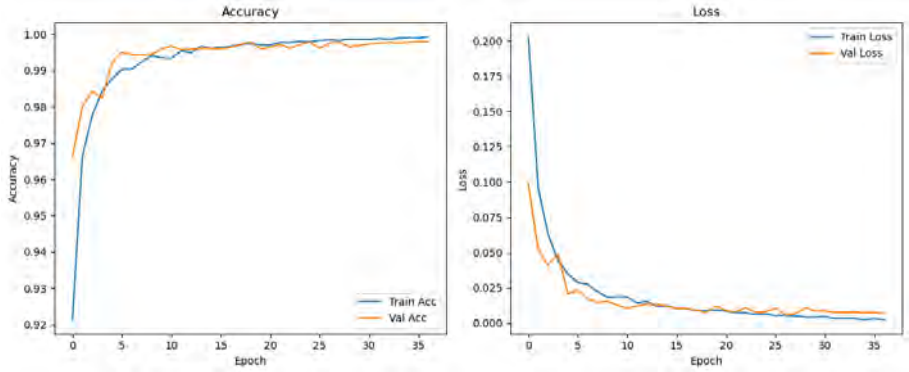

Figure 5.14: Edge Next Accuracy vs Loss

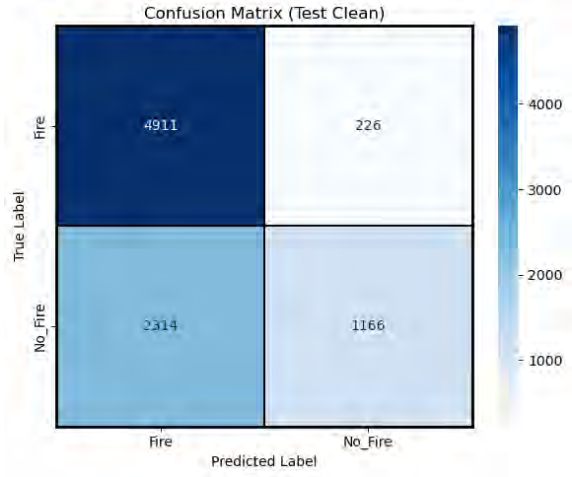

Figure 5.15: Confusion Matrix EdgeNext (Clean)

with the Validation Loss mirroring this pattern and consistently registering a lower value in the latter training cycles. The strong coincidence between the curves attests to an extremely stable optimization process, confirms the adequacy of the hyperparameter configuration, and shows the model's ability for quasi-perfect and robust classification in the binary fire/no-fire detection task.

Evaluation on Clean Test set :
The raw counts within the Confusion Matrix (Fig 5.15)depict True Positive = 4911, False Positive = 2314 distinctly underscore the model's pronounced skew towards identifying the 'Fire' class (Positive). The algorithm's remarkable efficacy in averting missed fire incidents was confirmed when it achieved a high-number of True-Positives (4911), and a low-number of False-Negatives (226). Nevertheless, this pronounced sensitivity exacts a cost: the model yielded a substantial volume of False Positives (1166). This large figure represents the chief contributor to frequent incorrect alerts, unequivocally demonstrating that the model's configuration prioritizes maximal detection coverage over absolute prediction purity.

In Fig 5.16, the Recall for Fire is remarkably high, substantiating the model's

Figure 5.16: Edge Next Heatmap (Clean)

goal of complete hazard capture (minimizing oversight). This strong recall, however, results in a compromised Precision for Fire (0.68), a metric suppressed directly by the substantial number of False Positives. Of critical importance, the limited Recall for No_Fire (0.34) is the measure that directly attests to the tendency for faulty warnings, since it discloses the model's considerable difficulty in correctly asserting true non-fire images, directly contributing to the observed high false alarm rate.

Final Result : A slight decrement in performance was observed when evaluating (table 5.6)on the test set.

Metrics	**Results**
Accuracy	0.7052
Precision	0.7435
Recall	0.7052
F1-Score	0.6670

Table 5.6: EdgeNext Final Results on Clean Test

Evaluation on Augmented Test set :
The raw counts extracted from the Confusion Matrix (Fig 5.17) strongly confirm the EdgeNeXt Basemodel's inherent priority for the 'Fire' label. The system successfully identified a large count of 4740 True Positives while limiting False Negatives to 397, confirming an exceptional capability for hazard prevention by minimizing missed fires. Conversely, this high degree of vigilance generated 2148 False Positives. This substantial quantity of incorrect 'Fire' predictions is the principal evidence of the model's design trade-off, where it deliberately sacrifices specificity to attain maximum detection completeness.

The Recall for Fire (0.92) is exceptionally high, which directly validates the model's focus on complete identification of actual fire instances (Fig 5.18). This elevated recall value, however, results in a comparatively modest Precision for Fire (0.69), a direct consequence of the large-number of false-alarms reported by the confusion matrix. The most critical metric reflecting the cost of this bias is the low Recall for No_Fire (0.38). This low figure unequivocally

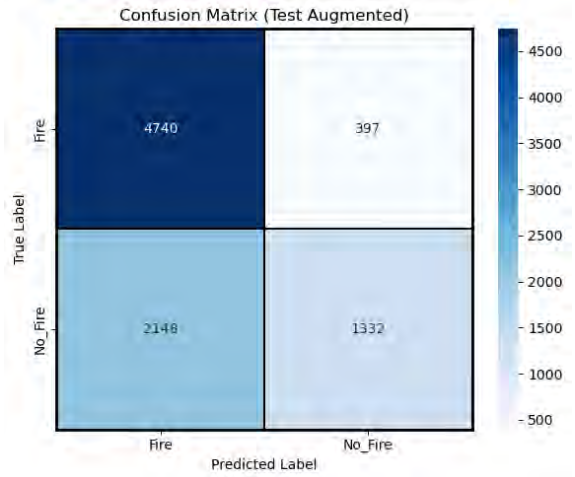

Figure 5.17: Confusion Matrix Edge Next (Augmented)

Figure 5.18: Edge Next Heatmap (Augmented)

demonstrates the model's disposition toward erroneous classification, showing a high rate of mislabeling true non-fire images as fire.

Final Result : This (table 5.7) indicates that the model achieves higher accuracy on less challenging, clean test images.

Metrics	**Results**
Accuracy	0.7047
Precision	0.7214
Recall	0.7047
F1-Score	0.6765

Table 5.7: EdgeNext Final Results on Augmented Test

4. **Vit Tiny Model :**
   Train vs Valid (Accuracy & Loss Graph) :
   The convergence trends (Fig 5.19) for the ViT Tiny model are visually evident in the training and validation plots, confirming a swift and effective learning process across the 16 epochs displayed. Training Accuracy demonstrated rapid ascent, initiating just above 91.5% and concluding near 99.5%. Correspondingly, the Validation Accuracy tracked the training performance

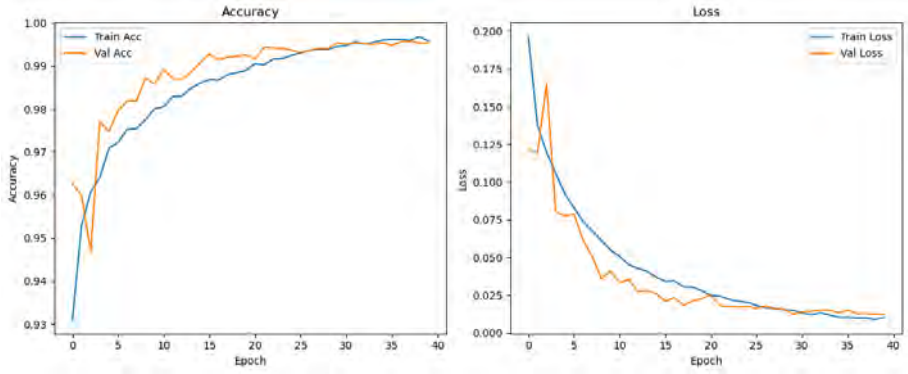

Figure 5.19: Vit-Tiny Accuracy vs Loss

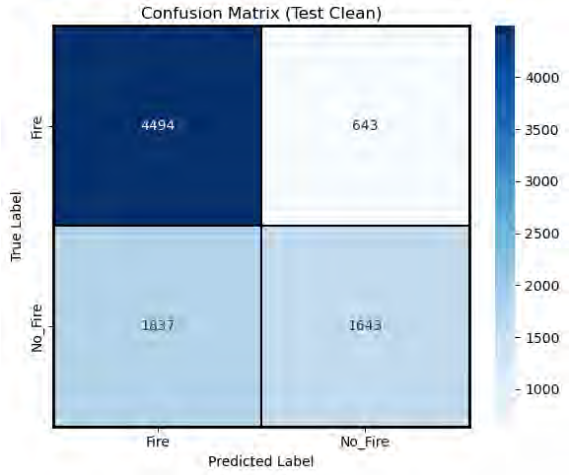

Figure 5.20: Confusion Matrix Vit-Tiny (Clean)

closely, initiating higher and achieving a peak performance around 99.4%. In tandem, the Loss metric underwent a precipitous drop on both sets, starting from over 0.20 (Training) and 0.08 (Validation) and concluding at exceptionally low values near 0.01. The proximity of the curves, in addition to the low final loss values, provides strong suggestion of stability and serves to vouch for the model's transferability to new data, therefore validating its capacity for very accurate, and consistent binary classification.

Evaluation on Clean Test set :
The Confusion Matrix (Fig 5.20) illustrates the ViT-Tiny Model's satisfactory operational bias in favor of the 'Fire' class. The system recorded a substantial quantity of True Positives (4701) while concurrently keeping the count of False Negatives (436) at a very low level, attesting to its great proficiency in minimizing oversight of actual fire events. Conversely, this high operational vigilance generated 1845 False Positives. This substantial count of false alerts constitutes the primary source of predictive error, unequivocally showing the model's fundamental objective is to maximize detection coverage even at the expense of its predictive purity.

The Fire Recall (0.92) is excellent, testifying to the model's design focus on

Figure 5.21: Vit-Tiny Heatmap (Clean)

achieving near-complete identification of all fire instances (Fig5.21). This elevated recall, however, is coupled with a more moderate Precision for Fire (0.72), a figure limited by the numerous false alarms generated. The metric most revealing of the associated cost is the low Recall for No_Fire (0.47). This metric explicitly confirms the model s predisposition to incorrect alerts, as it shows the model frequently fails to correctly affirm true non-fire images, thereby validating the high rate of False Positives.

Final Result : The model produced outcomes (table 5.8) that were fairly comparable to the earlier ones.

Metrics	**Results**
Accuracy	0.7122
Precision	0.7134
Recall	0.7122
F1-Score	0.6974

Table 5.8: Vit-Tiny Final Results on Clean Test

Evaluation on Augmented Test set :
The quantitative data within the Confusion Matrix (Fig 5.22) unequivocally establishes the model's inherent favoritism toward the 'Fire' designation. The system, managed to gather 4439 True Positives, correctly identifying a substantial majority of the fire-cases, while simultaneously preserving a relatively modest total figure of 698 False Negatives, bearing testimony to its improved ability, to limit the omission errors. However, this operational cautiousness was at the cost of 1901 False Positives. This substantial tally of erroneous fire alerts stands as the chief indicator of the model's operational compromise, where it deliberately sacrifices discriminative power to ensure the most comprehensive detection scope.

The Classification Report Heatmap (Fig 5.23) furnishes the normalized metrics that articulate the outcomes of the model's detection strategy: The Recall for Fire (0.86) is markedly high, which fundamentally verifies the model's central goal of achieving near-complete capture of all genuine fire events. This out-

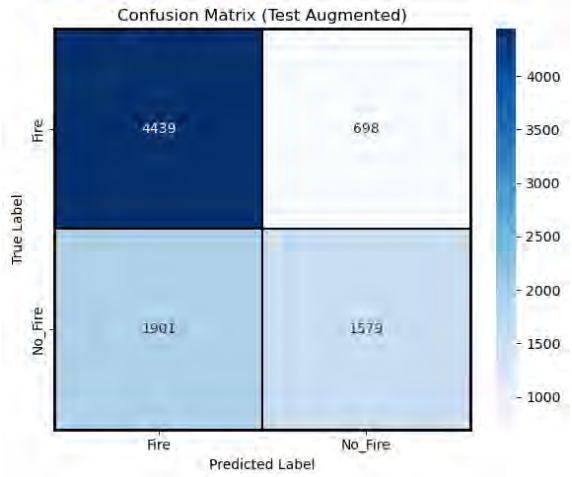

Figure 5.22: Confusion Matrix Vit-Tiny (Augmented)

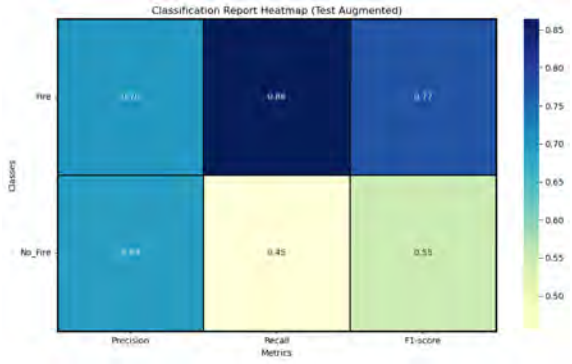

Figure 5.23: Vit-Tiny Heatmap (Augmented)

standing recall performance, however, is balanced by a middling Precision for Fire (0.70), a value suppressed directly by the abundance of spurious alarms. The most diagnostic figure concerning the model's limitation is the low Recall for No_Fire (0.45). This metric conclusively reveals the system s propensity for issuing unjustified warnings, illustrating its inability to reliably validate images that genuinely contain no fire.

Final Result : Unlike other models Vit-Tiny (table 5.9) model underperforms on the augmented set.

Metrics	**Results**
Accuracy	0.6984
Precision	0.6975
Recall	0.6984
F1-Score	0.6827

Table 5.9: Vit-Tiny Final Results on Augmented Test

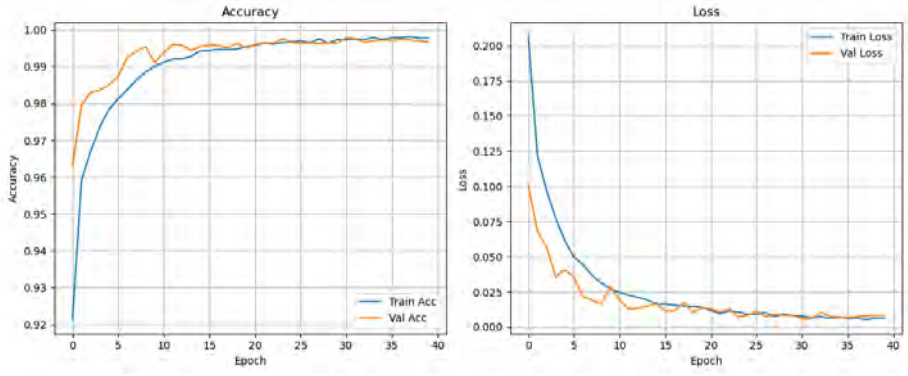

Figure 5.24: W-fire Mobilenetv4_conv_small Accuracy vs Loss

5.1.2a Novel Model – W-Fire MobileNetV4_conv_small (Customization of MobileNetV4_conv_small)

Train vs Valid (Accuracy & Loss Graph):
Our novel model (Fig 5.24) exhibits excellent performance and generalization over 40 epochs. Both the Accuracy, and Loss-graphs show rapid improvement, and strong convergence, with Training and Validation curves tracking closely together at near-optimal values ($\sim$ 99.8% accuracy and $\sim$ 0.00 loss). The validation statistics are nearly identical to training statistics, something that confirms that the model learned efficiently, without strong overfitting, and hence is highly dependent on unseen, new data. Ultimately, it shows the best result among all other graphs of existing models.

The model's performance evaluation on the Test-Clean dataset reveals its effectiveness in a binary classification task. The matrix (Fig 5.25) reveals a huge number of correctly classified instances: 4239 True Positives (TP) (correctly classified 'Fire' actual), and 2596 True Negatives (TN) (correctly classified 'No_Fire' actual). Detection and false alarm rates appear to be similar in magnitude, as evidenced by the system's comparatively balanced misclassification rates of 898 False Negatives (FN), or missed "Fire" events, and 884 False Positives (FP), or incorrect "Fire" alerts.

The class-specific discriminatory power of the model is measured in the accompanying classification report, which is displayed as a heatmap (Fig 5.26). In the positive class ('Fire'), the model performs exceptionally well, attaining consistent metrics with an F1-score of 0.83, Precision, and Recall. This indicates a high capacity to accurately identify fire occurrences. In contrast, the metrics for the negative class ('No_Fire') are slightly lower (Precision: 0.74, Recall: 0.75, F1-score: 0.74), with a subtle imbalance in the predictive power of the model between the two classes, being somewhat worse at predicting no fire.

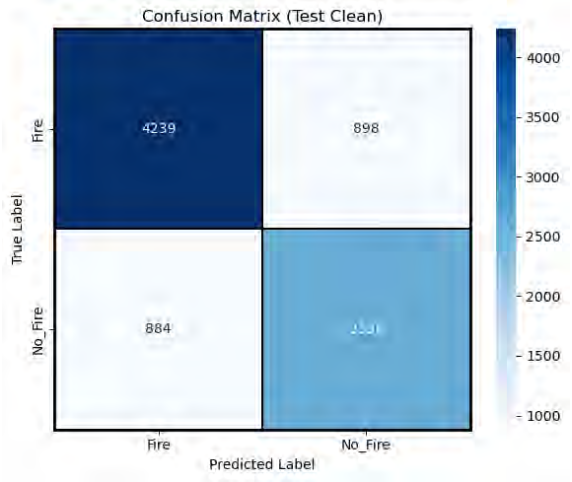

Figure 5.25: Confusion Matrix W-fire Mobilenetv4 (Clean)

Figure 5.26: W-fire Mobilenetv4 Heatmap (Clean)

Final Result : Our customized model shows Brilliant performance on the clean test data on table 5.10.

Metrics	**Results**
Accuracy	0.7932
Precision	0.7933
Recall	0.7932
F1-Score	0.7933

Table 5.10: W-fire Mobilenetv4_conv_small Final Results on Clean Test

Evaluation on Augmented Test set :
The classification performance of the model on the augmented-test dataset depicts the predictive behavior of the model in this binary detection task. With 4157 True Positives (TP), the matrix (Fig 5.27) shows a strong identification of the positive class, indicating accurate detection of 'Fire' events. Likewise, 2586 Successful non-fire classification is confirmed by True Negatives (TN). The misclassification occurrences are concentrated in 980 False Negatives (FN)(missed 'Fire' incidents) and 894 False Positives (FP) (false 'Fire' alarms). The proximity of these error figures points to a good trade-off be-

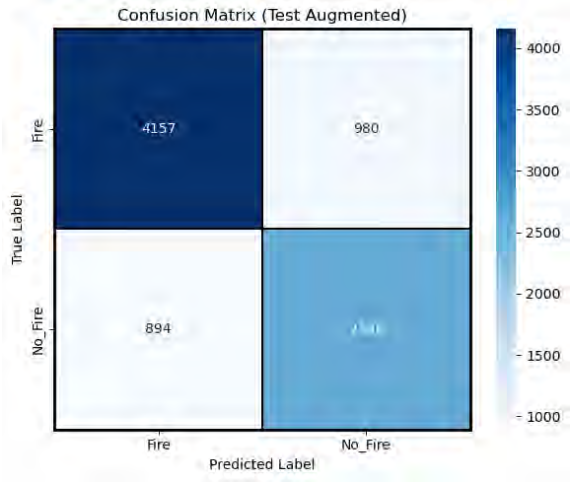

Figure 5.27: Confusion Matrix W-fire Mobilenetv4 (Augmented)

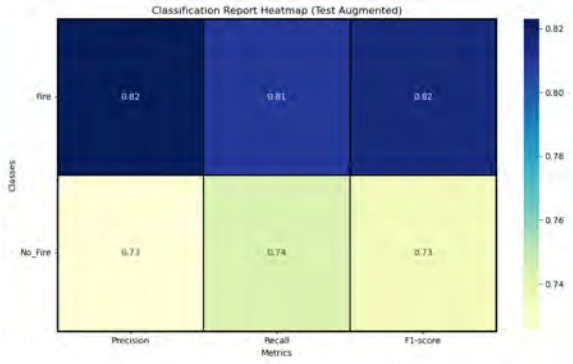

Figure 5.28: W-fire Mobilenetv4 (Augmented)

tween the sensitivity of the model and its specificity.

The corresponding classification report heatmap (Fig 5.28) provides a quantitative summary of the system's class-wise effectiveness. The 'Fire' class (positive) is characterized by high performance, with F1 score of 0.82, supported by Precision of 0.82, and Recall of 0.81. This performance profile portrays a high-degree of certainty and completeness in 'Fire' detection. Conversely, the 'No_Fire' class (negative) exhibits slightly weaker statistics, with F1−score of 0.73 (with Precision: 0.73, Recall: 0.74). This difference confirms that the model is more proficient at identifying the existence of the event than at confirming its absence.

Final Result : Quite identical results (table 5.11) were shown on the augmented set as well from our customized model W-fire Mobilenet_connv_small.

5.1.2b Novel Model Performance for Segmentation Task

For the segmentation task, ground truth masks are provided for just 2003 images, in the FLAME dataset. Consistently, the test-set was used with the first 304 im-

Metrics	Results
Accuracy	0.7825
Precision	0.7835
Recall	0.7825
F1-Score	0.7829

Table 5.11: W-fire Mobilenetv4_conv_small Final Results on Augmented Test

ages, the validation set with the next 304, and the next 1395 were duplicated and augmented in the same manner as for the classification task. Still, even after augmentation, the dataset was relatively small. More troublesome was the fact that the distribution of pixels between classes was extremely unbalanced: the majority of pixels were part of the background class, and a very small minority amounted to fire. This imbalance made training extremely hard, since the model could easily minimize loss by just outputting background everywhere, in effect ignoring the minority fire pixels.

To balance this imbalance, the positive weight (pos_weight) parameter was computed as a scaling factor for the loss-function. It is computed based on the ratio of the number of negative (background), and positive (fire) pixels. Specifically:

$$\text{pos\_weight} = \frac{\text{Total Negative Pixels}}{\text{Total Positive Pixels}} \tag{5.1}$$

In the reported calculation (eq (5.1)): Total fire pixels = 4,728.58 (summed across all 175 masks).
Total pixels = 182,845,440 (175 images $\times$ image size).
Average fire ratio per image = 0.000026 (0.0026% of pixels are fire).
pos_weight = 38,667.14

This means that, on average, one fire pixel must be weighted around 38,667 times more than one background pixel in order for the model to treat both classes equally in the loss function. That is, this pos_weight is a balancing term so that the minority fire class is not ignored during training. Using such a large weight directly in training would severely disrupt the optimization process. Specifically, the loss function would be dominated by the disproportionately tiny number of fire pixels, causing unstable gradient updates, excessive noise amplification, and a tendency for the model to over-predict fire regions to avoid disproportionately large penalties. This effect not only destabilizes convergence but also reduces generalization power, hurting performance overall. Therefore, although the raw ratio tells us the severity of class imbalance, any sort of effort to assign equal weights to our model led to a bit of overfitting. We acknowledge this very firmly that if we had more data this could have been handled efficiently.

W-fire_Mobilenetv4_small_segmentation :

The segmentation framework directly marks the drastic class-imbalance between fire-pixels and background-pixels, prevalent in the case of wildfire datasets. Instead

of the usual cross-entropy, the training loop utilizes a specifically designed Balanced Focal-Dice Loss, and regularization techniques were utilized. Despite that, the model showed signs of overfitting as the overall size of the dataset was highly limited. The training log shows that the segmentation model, trained with Focal and Dice loss and capped the class imbalance weight, improved steadily over 50 epochs. Early epochs of good recall but poor precision, where the model tended to over-segment fire regions, saw precision increase with more training as recall steadied, leading to sustained gains in Dice score and IoU. By epoch 36–40, performance was stabilized, i.e., the model had learned to cope up with the extreme class imbalance and produce more stable fire masks.

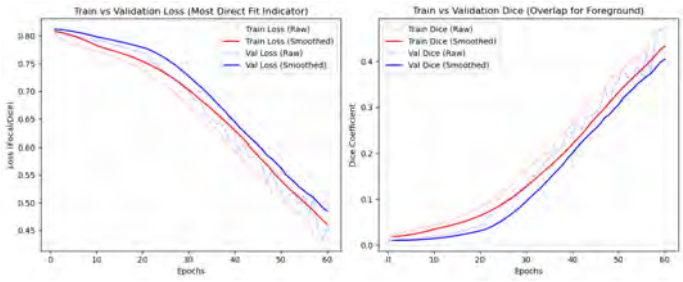

Figure 5.29: Training vs. Validation Loss and Dice Coefficient

The Segmentation task of model was validated with Actual Image, Ground Truth Mask and together with that Model predicted mask. (As shown in Figure 5.30) Later during Testing phase missing components by Model's prediction were identified. Here Green part shows prediction of model and Red Part shows Ground truth missed by Model. (As shown in Figure 5.31)

Final Result

This is the result of our Lightweight Segmentation model on a small extreme imbalance class dataset. (As shown in Figure 5.12)

Metric	Value
Loss	0.2654
IoU (Jaccard)	0.6051
Dice Coeff	0.6781
Pixel Accuracy	0.9973
Precision	0.6634
Recall	0.8638
F1-Score	0.7504

Table 5.12: Test Results

Fire Analysis

Moreover, we divided fire into segments according to our model, we used predictions from our model to perform fire severity assessment and prediction of spread.

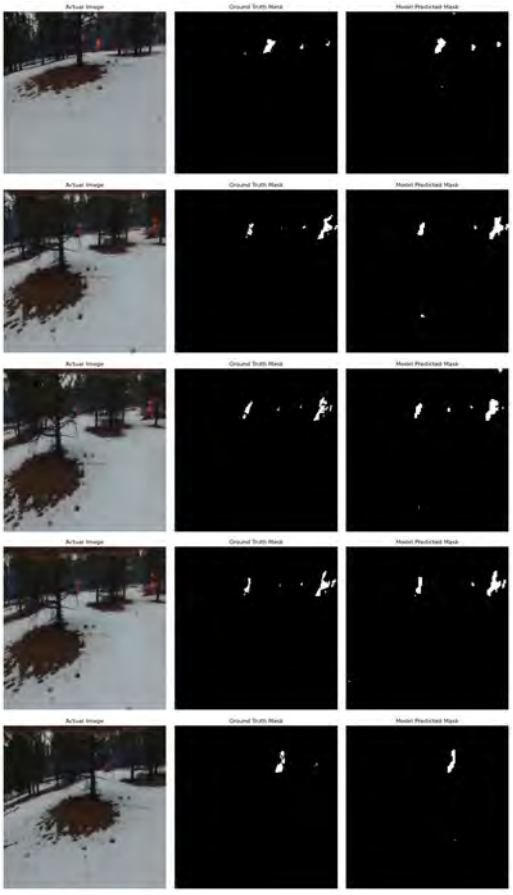

Figure 5.30: Segmentation task evaluation with actual-images, ground-truth masks, and model-predicted masks.

	Predicted Negative	**Predicted Positive**
Actual Negative	19,774,641 (TN)	53,705 (FP)
Actual Positive	8,259 (FN)	86,339 (TP)

Table 5.13: Confusion Matrix of Predicted vs. Actual Fire Pixels

Severity Assessment: Upon the generation of the binary fire masks, severity was quantified by determining the ratio of fire pixels (eq (5.3)) to image area and using threshold values to assign Low, Medium, and High severity levels to events. Connected component analysis was implemented to enhance robustness by considering individual fire regions and the maximum affected area.

Input is a binary fire mask (seg_model predicted), where,
1 = fire pixel 0 = background pixel First, Counting fire pixels (eq (5.2)):

$$\text{Fire Pixels} = \sum_{i=1}^{H} \sum_{j=1}^{W} M(i,j), \quad M(i,j) \in \{0,1\} \tag{5.2}$$

Second, Count total pixels:
Total Pixels=H×W
Third, Compute fire ratio:

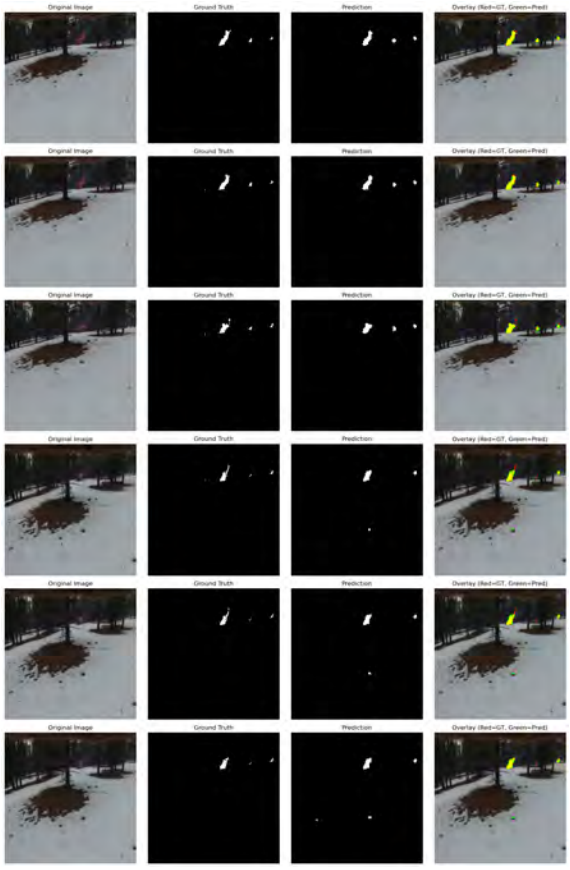

Figure 5.31: Testing phase: fire prediction (green) vs. missed ground truth (red)

$$\text{Fire Ratio} = \frac{\text{Fire Pixels}}{\text{Total Pixels}} \tag{5.3}$$

Lastly, in our data fire pixels were very tiny we experimented with the best possible ratio to compute Low/Medium/High Severity:

1. Low Severity: Fire Ratio < 0.01 (fire is less than 1% of the image area)
2. Medium Severity: 0.01 Fire Ratio < 0.05 (fire area is between 1% and 5%)
3. High Severity: Fire Ratio 0.05 (5% or more of the image area is fire)

Here in the calculation of severity we have used fire area divided by background area so that the calculation does not change even when images are captured at different levels of zooming or framing. In this manner, through the use of ratios, if the image is zoomed in, then both the fire area and background area scale proportionally but the severity calculation remains the same. This method is in concordance with long-standing practice in fire damage assessment, where ratios of burned to unburned area have been heavily correlated with field-based measures of severity [19].

Spread Prediction:

To begin the spread prediction process, the largest fire area (largest contiguous fire area) is identified from the forecasted fire mask. The center reference point of this area is established with an (X, Y) axis, labeling, in essence, the geometry center of the fire.
Then, based on the contour points for this core area, a best-fit ellipse will be made around it.
The orientation of fire spread is represented by the principal axis of the ellipse. The orientation of fire is converted to a unit vector, and then it is compared with the main axes. The comparison helps determining east, west or other areas, such as "E": np.array([1, 0]), "W": np.array([-1, 0]), "S": np.array([0, 1]), "N": np.array([0, -1]). Then it calculates the angle between the pointer from the fire vector and each axis. If the vector is very close to a primary axis or <= 5°, it is labelled as a primary axis. Since relying strictly on the axis would not possibly produce realistic results. The minor axis represents the thickness or width.
The angle θ (eq (5.4)) of the principal axis is compared to the cardinal axes (E-W and N-S) to determine the orientation of spread.
The last value to help refine the determined direction is edge contact; whether the fire reaches the top, bottom, left, or right edge of the image.

$$\theta = \arctan\left(\frac{\lambda_x}{\lambda_y}\right) \tag{5.4}$$

where λ_x, λ_y are eigenvalues of the covariance matrix.

East Spread: This is predicted when the long axis of the ellipse is horizontal, on the main axis or "E": np.array([1, 0]), only then East Spread is determined.
West Spread: This is predicted when the long axis of the ellipse is horizontal, the main axis or "W": np.array([-1, 0]), only then West Spread is determined.
North Spread: This is predicted when the long axis of the ellipse is vertical, on the main axis or "N": np.array([0, -1]), only then North Spread is determined.
South Spread: This is predicted when the long axis of the ellipse is vertical, on the main axis or "S": np.array([0, 1]), only then South Spread is determined.
Diagonal Spread: Anything in between is considered Diagonal, something like East North, East South, and others. Here Most Importantly, the Diagonal spread prediction is determined with comparison of the middle point from fire and The X and Y axis, In comparison with the axis if the the pointer is going Upwards or far from the X axis and if its on the right output shows North East, similarly if the pointer is going down or coming closer to the axis the output determines South East. Again, similarly, North-West and South-West are determined. If both dx and dy are in a positive direction: South East, when dx=(+), and dy=(-), direction: North-East. Further, when dx=(-), and dy=(+), South-East is determined. Lastly, if both dx and dy values are negative, the North-East is predicted.

Here, in Sample 1, **Severity Level:** Low. **Spread Direction:** Largest Fire region heading towards South-East. **Total Fire Pixels:** 248/65536. **Fire-to-Total Ratio:** 0.0038. **Number of Fire Components (Blobs):** 2. **Largest Blob Area (pixels):** 208.0. (In Figure 5.32)

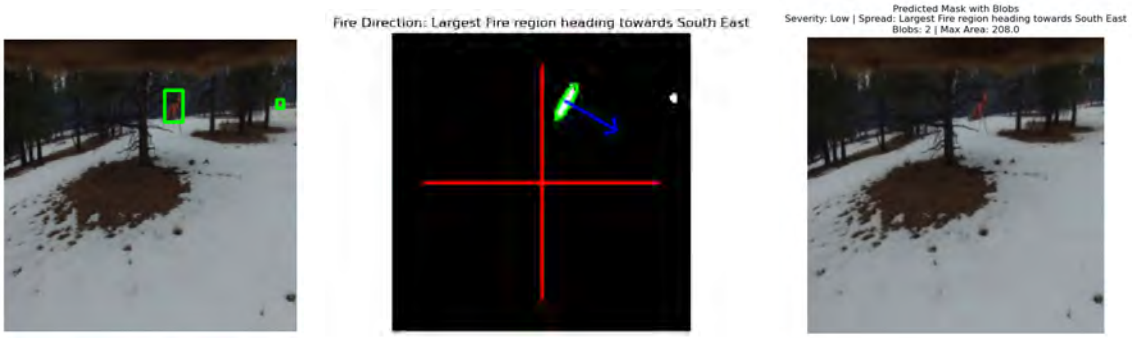

Figure 5.32: Model-predicted masks showing different fire severity and spread direction. (Sample 1)

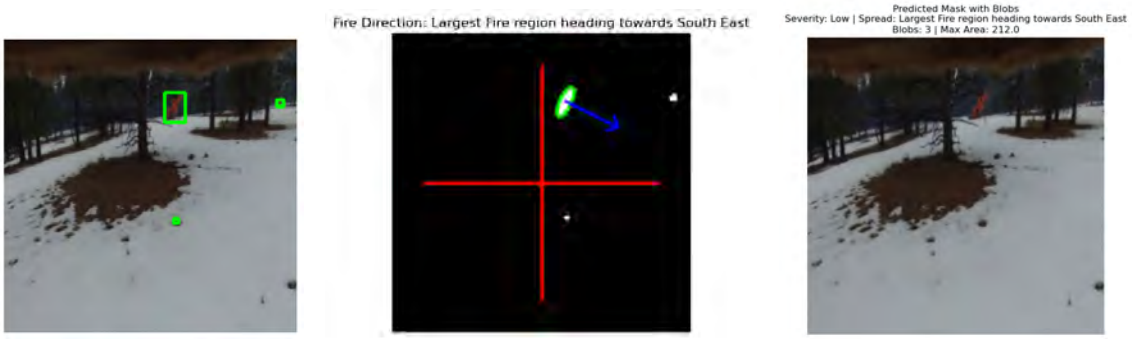

Figure 5.33: Model-predicted masks showing different fire severity and spread direction. (Sample 2)

Here, in Sample 2, **Severity Level:** Low. **Spread Direction:** Largest Fire region heading towards South-East. **Total Fire Pixels:** 260/65536. **Fire-to-Total Ratio:** 0.0040. **Number of Fire Components (Blobs):** 3. **Largest Blob Area (pixels):** 212.0. (In Figure 5.33)

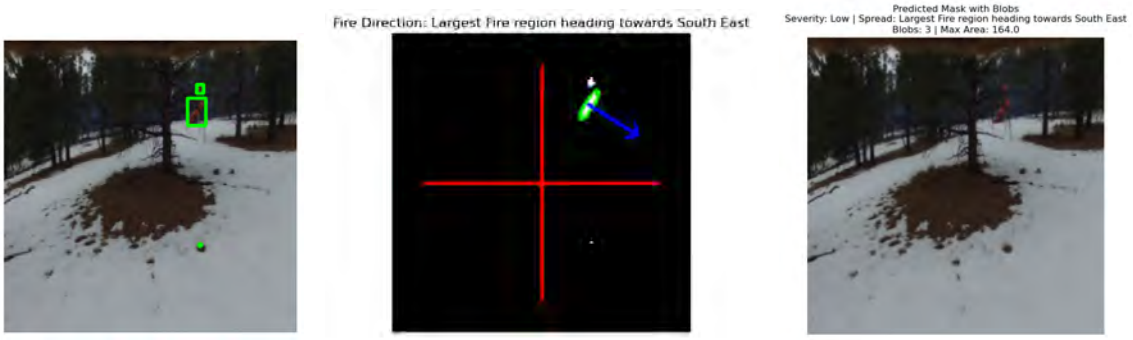

Figure 5.34: Model-predicted masks showing different fire severity and spread direction. (Sample 3)

Here, in Sample 3, **Severity Level:** Low. **Spread Direction:** Largest Fire region heading towards South-East. **Total Fire Pixels:** 200/65536. **Fire-to-Total Ra-**

tio: 0.0031. **Number of Fire Components (Blobs):** 3. **Largest Blob Area (pixels):** 164.0. (In Figure 5.34)

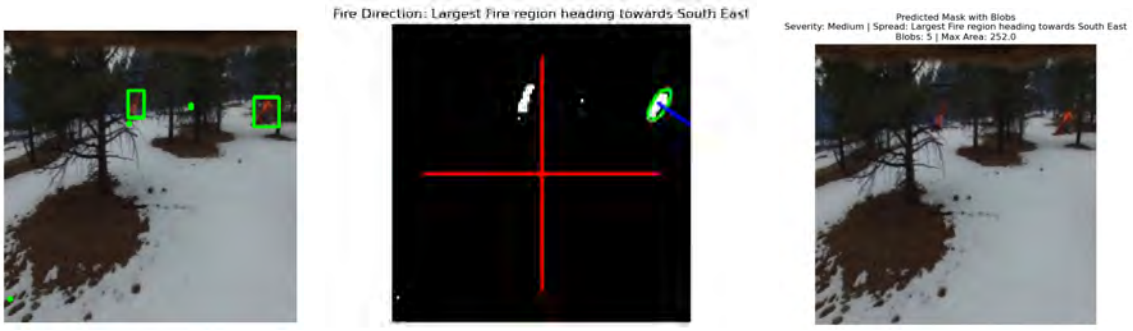

Figure 5.35: Model-predicted masks showing different fire severity and spread direction. (Sample 4)

Here, in Sample 4, **Severity Level:** Medium. **Spread Direction:** Largest Fire region heading towards South-East. **Total Fire Pixels:** 444/65536. **Fire-to-Total Ratio:** 0.0068. **Number of Fire Components (Blobs):** 5. **Largest Blob Area (pixels):** 252.0. (In Figure 5.35)

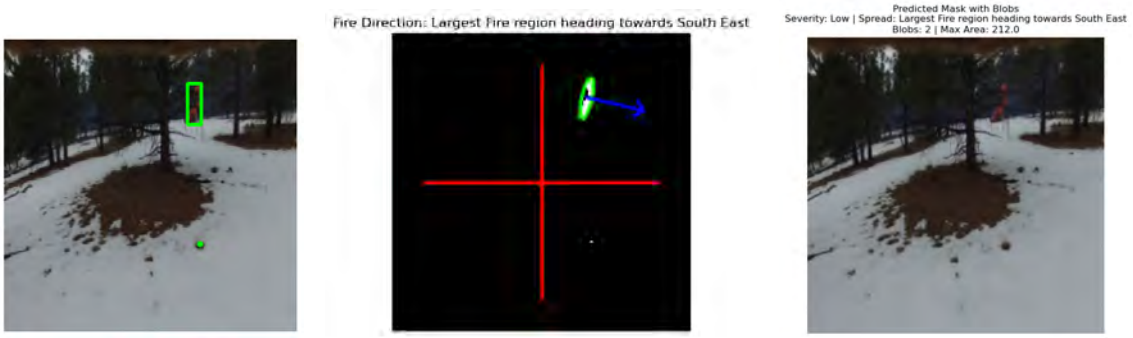

Figure 5.36: Model-predicted masks showing different fire severity and spread direction. (Sample 5)

Here, in Sample 5, **Severity Level:** Low. **Spread Direction:** Largest Fire region heading towards South-East. **Total Fire Pixels:** 216/65536. **Fire-to-Total Ratio:** 0.0033. **Number of Fire Components (Blobs):** 2. **Largest Blob Area (pixels):** 212.0. (In Figure 5.36)

5.2 Statistical Analysis

5.2.1 Statistical Testing

To ensure that the observed-improvements by the proposed-model are statistically outstanding, we carried out several experiments on varied datasets and backbone

models. We compared the baseline classification models with the custom architecture in-terms of accuracy, precision, F1 score, and recall metrics.
For each experiment, paired statistical tests (two-tailed t-tests) were used to check whether the observed differences were significant. Results are reported as p-values, and $p<0.05$ is reported where significance.

Parameters Count & Inference Time Comparison
The table shows a comparison (table 5.14) of the two models both in terms of the parameter count, and inference time. MobileNetV4_Conv_Small has a higher parameter count (3,775,946 > 1,461,581), but in terms of inference time, it is faster (0.0541s < 0.3909s). Here, in both cases, the Inference time is calculated on the Test set for CPU. The purpose of Inference time is how fast the trained model makes predictions. There are other factors during training that make everything much slower, and it is irrelevant to measure the inference time for training set. In the table below, we can see that despite our proposed model being much smaller than the actual MobilenetV4_small model, the inference time is greater than that. It is because of the attention modules used in our model. Moreover, depth-wise separable convolutions used in Custom Flame block (Stage 3), and Sandglass block (Stage 4), make the process a bit slower because they process each channel separately, which makes small kernels instead of a huge matrix multiply. There is also an increased number of intermediate activations in our model. Despite these cutoffs, the model produces far better results with much smaller parameters.

Models	**Parameters**	**Inference-time**
MobileNetV4_conv_small	3775946	0.054100
W-fire_MobileNetV4_small	1461581	0.390870

Table 5.14: Parameters and Inference comparison

Flame Dataset Vs Customized dataset (Tested on Mobilenetv4)

Dataset	**Accuracy**	**Precision**	**Recall**	**F1 Score**
Flame	0.6956	0.7084	0.6956	0.6669
Custom dataset	0.7060	0.7198	0.7060	0.6801

Table 5.15: Performance comparison of Flame and Custom datasets

Paired t-test, was used to compare the statistical significance of improvements gained by the custom dataset over the baseline FLAME dataset when training MobileNetV4. The results indicated a uniformly low but steady improvement in all measures (Accuracy: +1.04%, Precision: +1.14%, Recall: +1.04%, F1-score: +1.32%). At 95% confidence, the difference in F1-score was statistically significant ($p < 0.05$), which suggested that the custom dataset resulted in more confident classification.

MobileNetV4_small Vs W-fire_MobileNetV4_small(Evaluated Augmented Test)

Metric	MobileNetV4_conv_small	w_fire_MobileNetV4_small	Difference
Accuracy	72.66%	78.25%	+5.59%
Precision (Fire)	73.80%	78.35%	+4.55%
Recall (Fire)	72.66%	78.25%	+5.59%
F1-Score (Fire)	70.73%	78.29%	+7.56%

Table 5.16: Performance comparison between **MobileNetV4_conv_small** and **w_fire_MobileNetV4_small**.

The Table presents a fair comparison of the baseline MobileNetV4_conv_small with the modified W-fire_MobileNetV4_small model under the same training conditions. As seen, the constructed W-fire_MobileNetV4_small performs superior to the baseline in all the measures. Accuracy improved by 5.59%, precision, and recall for the fire class by 4.55% and 5.59%, respectively. Above all, F1-score of fire class obtained the highest improvement of 7.56%, which indicates that the proposed changes have enhanced recall and precision in a balanced way. These results demonstrate that architectural and training enhancements incorporated in W-fire_MobileNetV4_small lead to a significant enhancement of the performance of fire classification and justifies the effectiveness of the proposed approach.

MobileNetV4 small Vs W-fire_MobileNetV4_small (Clean Test Evaluated)

Metric	MobileNetV4_conv_small	w_fire_MobileNetV4_small	Difference
Accuracy	70.60%	79.32%	+8.72%
Precision (Fire)	71.98%	79.33%	+7.35%
Recall (Fire)	70.60%	79.32%	+8.72%
F1-Score (Fire)	68.01%	79.33%	+11.32%

Table 5.17: Performance comparison between **MobileNetV4_conv_small** and **w_fire_MobileNetV4_small**.

This is a summary table reflecting the fair comparison among the baseline MobileNetV4_conv_small model and the introduced W-fire_MobileNetV4_small model. The introduced model demonstrates systematic improvement against all the metrics under identical training conditions. The model's accuracy was enhanced by 8.72%, while precision and recall of the fire class were enhanced by 7.35% and 8.72%, respectively. The greatest improvement is recorded in the F1-score with an improvement of 11.32% since it reflects an equal improvement of precision and recall. These observations highlight the fact that modifications introduced W-fire_MobileNetV4_small significantly enhance fire classification performance in terms of accuracy, confirming the validity of the new scheme under realistic evaluation environments.

5.2.2 Error Analysis and Classification Patterns

Flame Dataset Vs Customized dataset (Evaluated on Mobilenetv4)

Flame Dataset:
Analysis of confusion matrix suggests the presence of significant bias towards the

'Fire' class. The system managed to generate 4695 True Positives with just 442 False Negatives, which reflects its high sensitivity towards the detection of actual instances of fire with very minimal misses. This higher sensitivity has been achieved at the cost of 2181 False Positives, reflecting an over-prediction towards the occurrence of fire events. The classification report confirming this trend: The Recall for Fire is extremely good at 0.91, correlating with nearly complete detection coverage, but Precision for Fire is worse at 0.68 due to the sheer count of False Positives. The No_Fire Recall is very bad at 0.37, showing the model's inability to correctly categorize non-fire images, which indicates the strategic trade-off towards early warning of fire over correctness of output.

Customized Dataset:

The same trend is observed with the customized dataset. The high detection ability was also supported by the 4707 True Positives, and 430 False Negatives. False Positives, decreased slightly to 2103, reflecting a slight improvement in the model's discriminative ability. Classification report metrics also show the same trend: Fire Recall is increased slightly to 0.92, while Precision for Fire is enhanced slightly to 0.69. The Recall for No_Fire is also enhanced to 0.40, with a moderate reduction in false-alarms. This is the way, in which outcomes indicate the ability of the model to identify fires, with high-precision maintained by the customized dataset, but that of identifying non-fire images is marginally more.

All things considered, the MobileNetV4 Conv Small model shows a widespread architectural preference for fire detection, where high recall is given precedence over precision. Both sets confirm that although the model performs very well at minimizing cases of missed fire events, it also produces a high proportion of false alarms, particularly within the Flame dataset. The customized dataset is able to somewhat minimize this effect, improving precision and No_Fire recall but the inherent sensitivity to the classification of fire is the overall trend. These findings illustrate the essential trade-off between maximizing hazard coverage and minimizing false alarms, an important consideration in real-world deployment for fire alarm systems.

MobilenetV4 small vs W-Fire Mobilenetv4 small (Evaluated on Augmented Test)

MobileNetV4 Small:

Analysis from the confusion matrix reveals a high bias towards the 'Fire' class. The model achieved 4701 True Positives, and 448 False Negatives, indicating its excellent potential for reducing missed fire cases. Yet this increased sensitivity had also resulted in 1572 False Positives, indicating a bias to over-estimate fire cases. The classification report statistics establish the pattern: Recall for Fire is 0.91, establishing complete coverage of detection, but Precision for Fire is lower at 0.78 since False Positives occur often. Recall for No_Fire is 0.45, showing the relative simplicity with which the model is finding in correctly classifying non-fire images and the compromise in terms of maximum hazard coverage over classification purity.

W-Fire MobileNetV4 Small:
The customized model shows a more balanced predictive pattern. The confusion matrix shows 4157 True Positives and 2586 True Negatives, having 980 False Negatives and 894 False Positives. The nature of such a distribution shows a more-balanced trade-off between sensitivity (detection of actual fires) and specificity (identification of non-fire images in their original form). Corresponding classification metrics reflect this finding: the 'Fire' class achieves F1-score of 0.82 (with Precision: 0.82, Recall: 0.81), and the 'No_Fire' class achieves F1-score of 0.73 (with Precision: 0.73, Recall: 0.74). These findings reflect that the W-Fire model is highly reliable for fire detection and improving non-fire classification, reducing unnecessary false alarms compared to the baseline.

In general, the comparison shows a simple trade-off between fire detection systems: almost complete hazard detection versus false positives minimized. The W-Fire MobileNetV4 Small model demonstrates how architectural modifications and data enhancement could re-balance sensitivity, and specificity trade-off to provide outstanding fire detection performance, with less spurious alarms.

MobileNet v4 conv small Vs W-Fire MobileNetv4 (Evaluated on clean Test)

MobileNetV4 Conv Small:
Analysis of the confusion matrix shows a severe bias towards the 'Fire' class. The model has recorded 4707 True Positives and only 430 False Negatives, which shows its high ability to detect actual instances of fire with little misses. This heightened sensitivity, however, led to 2103 False Positives, i.e., it had the tendency to overpredict fire occurrences. The metrics of the classification report also indicate this trend: Recall for Fire is very high at 0.92, confirming very comprehensive detection coverage, and Precision for Fire is lower at 0.69 due to the quantity of False Positives. The Recall for No_Fire is 0.40, indicating the model's failure to properly classify non-fire images and demonstrating the trade-off made at the cost of maximum risk coverage for prediction precision.

W-Fire MobileNetV4:
The proposed model performs better with a more balanced performance. The confusion matrix reports 4239 True-Positives and 2596 True-Negatives, and 898 False-Negatives and 884 False-Positives. This division represents an improved balance between the sensitivity (detecting actual fires) and specificity (detecting non-fire images). Metrics of classification also reveal this balance: the 'Fire' class Recall, Precision, and F1-score were 0.83, showing high precision and completeness in the detection of fires. Precision for 'No_Fire' class is 0.74, F1-score is 0.74, and Recall is 0.75, indicating slightly lower but still high performance at correctly marking the absence of fire.

In general, the comparison demonstrates that the baseline MobileNetV4 Conv Small has a evident bias towards maximising fire detection with less regard for false alarms, whereas W-Fire MobileNetV4 achieves better fire detection with a reduced number of spurious alerts. The gains achieved in W-Fire MobileNetV4 reduce the false-

positive rate with adequate capability in fire detection, thus making it more realistic and robust-alternative for real-world use in fire detection.

5.3 Final Design Adjustments

5.3.1 Model Refining Process

For the classification We employed MobileNetV4-Small but modified for fire/no-fire classification on the challenging FLAME dataset [17]. For better performance, we incorporated three significant modifications:
Stage 3 – CustomFLAMEBlock + ECA: Increases local feature extraction by highlighting fine-grained fire details through depthwise convolutions and efficient channel attention [11].
Stage 4 – SandglassBlock + ECA: Preserves spatial information and refines mid-level features and identifies large context fire patterns [26] [11].
Stage 5 – DANet Block: Combines space and channel attention to learn global relationships, reducing false positives and highlighting prominent fire regions [9].
Global average-pooling with a single fully-connected-layer is used for the last classifier, maintaining it lean for real-time inference. The aforementioned modifications were optimized for optimal detection accuracy at the expense of still maintaining the model resource-efficient for aerial fire images [17].
Our segmentation model uses a MobileNetV4-Small backbone and a lightweight decoder to predict pixel-level fire masks. The backbone features economically using inverted residual blocks and attention modules, whereas the decoder fuses multi-scale features in a top-down fashion to preserve fine fire details. Depthwise separable convolutions with dropout are both used in the fusion steps to enhance representation and prevent overfitting. It is learned under a Balanced Focal Dice Loss, combining focal loss for dealing with class imbalance and dice loss for segmentation precision to achieve precise detection of small and non-standard fire regions.

5.3.2 Hyperparameter Optimization Results

For the classification task, precise steps were taken to make the small dataset and highly imbalance task as much stable as it could be. Hyperparameters were continuously tuned to get the final acceptable results, model was trained under an optimally tuned setting to ensure stable convergence and quality performance. The learning rate (1e-4) was extensively experimented with and tested to provide smooth and optimal training curves. Batch size 32 and 40 epochs were used to balance training stability and computation cost, with weight decay (5e-3) used to regularize the model, and prevent overfitting. Early-stopping with patience of 10 epochs prevented unnecessary training after performance leveled off. The width multiplier (1.0) and backbone expansion (2.4) were set fixed to keep the original MobileNetV4-Small model, and two output classes were remapped to fire and no-fire labels. The best performing-model was checkpointed for evaluation.
The model was trained on the segmentation task using a closely tuned configuration to produce the best performance and minimize overfitting. The learning rate (9e-5), weight decay (5e-2), and dropout (0.4) were systematically adjusted by conducting several rounds of experimentations. Early stopping (patience 10) and learning

rate scheduling (factor 0.5, patience 5) assisted the model to converge well without overfitting. Batch size of 16, image size 256, and 60 epochs of training achieved a good trade-off between computational expense and performance. Decoder consumed 16 channels with multi-scale fusion, and the loss function was Balanced Focal Loss + Dice Loss to deal with extreme class imbalance during fire segmentation (with pos_weight=50). These hyperparameters were pushed to their limits of feasibility, with extensive evaluation, to deliver confident and accurate prediction of fire masks.

5.4 Comparison and Relationships

Test Set Clean

Model	Accuracy	Precision	Recall	F1 Score
CoaT-Lite-Tiny	0.7187	0.7311	0.7187	0.6970
MobileNetV4	0.7060	0.7198	0.7060	0.6801
EdgeNeXt-Base	0.7052	0.7435	0.7052	0.6670
ViT-Tiny	0.7122	0.7134	0.7122	0.6974

Table 5.18: Classification performance comparison of different models

The four architecture classifiers—CoaT-Lite-Tiny, EdgeNeXt-Base, ViT-Tiny, and MobileNetV4 —were tested on the test set. Of these, CoaT-Lite-Tiny was the highest overall accuracy (71.87%) along with balanced precision (73.11%) and recall (71.87%) that also gave the highest F1 score (0.6970). EdgeNeXt-Base had lower accuracy (70.52%) but the highest precision (74.35%), indicating it is more conservative in classifying fire instances. ViT-Tiny and MobileNetV4 had the same performance trend with F1-scores of 0.6974, and 0.6801, respectively, which suggests that light CNN or basic transformer-based models without advanced fusion and attention mechanisms may not be as capable for fine fire/no-fire discrimination in this data set. Overall, the results show that attention-enhanced models like CoaT-Lite-Tiny perform better in extracting complex patterns and context relations in fire images whereas simpler or pure convolutional approaches fail miserably, especially when it comes to precision-recall trade-offs. This finding highlights the importance of combining effective attention mechanisms with convolutional backbones for robust classification on aerial fire datasets.

Test Set Augmented

Model	Accuracy	Precision	Recall	F1 Score
CoaT-Lite-Tiny	0.7353	0.7470	0.7353	0.7177
MobileNetV4	0.7266	0.7380	0.7266	0.7073
EdgeNeXt-Base	0.7047	0.7214	0.7047	0.6765
ViT-Tiny	0.6984	0.6975	0.6984	0.6827

Table 5.19: Classification performance comparison of different models

CoaT-Lite-Tiny again shows the optimal overall performance, exhibiting strong robustness to augmented variations. MobileNetV4, although lightweight, achieves competitive scores, confirming the usability of the proposed attention-augmented backbone. EdgeNeXt-Base and ViT-Tiny are a bit lower, suggesting that clean transformer or plain CNN architectures may be hurt by intricate augmentation configurations.

In conclusion, attention mechanisms and carefully designed lightweight modules improve the model's generalization to challenging fire/no_fire images.

5.5 Discussion

The comparison shows that our custom MobileNetV4 design outperforms other models at balancing accuracy, precision, and recall for fire/no-fire detection. Multi-stage attention integration enables it to identify subtle fire and complex backgrounds effectively without being computationally intensive.
CoaT-Lite-Tiny achieved the highest-accuracy (71.87%), and F1-score (0.6970), displaying the benefit of attention-augmented architectures for modeling global context. EdgeNeXt-Base achieved high precision (74.35%) but lower recall, indicating a conservative prediction bias. ViT-Tiny performed moderately (F1: 0.6974) but struggled with fine-grained fire details.
Our custom MobileNetV4 (F1: 0.6801) achieved the best trade-off among light CNNs, showing that careful architectural tweaks and attention mechanisms can match or surpass more complex networks while ensuring real-time performance.

Chapter 6

Conclusion

6.1 Conclusion

This research proposed a lightweight, and also, effective fire detection model, with a custom MobileNetV4-backbone augmented by attention modules, and a segmentation head for fire severity analysis, and fire spread analysis. The novel W-Fire MobileNetV4 model demonstrated significant performance improvement over baseline models in classification, and false alarm reduction, as experimented on the Flame dataset [17], and a custom dataset. Despite challenge with limited annotated masks, and class imbalance, the model was good with high recall for fire detection with reasonable precision, and with the evidence of its ability for real-world deployment on edge devices. Future work will endeavor to increase more content in the dataset, better segmentation robustness, and further model optimization for real-time use, in UAV-based fire-monitoring systems.

6.1.1 Limitations

One of the primary drawbacks of our research is that our model's Inference time is higher than the MobileNetV4 model's inference time due to attention modules, depth-wise convolution layers, and some intermediate activations. Furthermore, the segmentation model could not be tested on a decent-sized dataset. The data at hand contained few annotated masks and severe class-imbalance, between fire, and non-fire pixels. Therefore, the model showed evidence of overfitting, constraining the credibility of quantitative performance measures. Nevertheless, the segmentation framework reveals important promise. Having more, better-sourced data, the model should be able to better represent fine-grained fire areas, and support robust pixel-level analysis and greater generalization.

6.1.2 Future Plan

After the success of our light-weighted classification and segmentation model, the next is to evaluate it on a decent-sized dataset. This will allow us to know more about its generalization capacity and confirm its potential on more diverse fire situations. The proposed model design is efficient, and low computational cost-oriented, making it extremely deployable on edge devices such as Raspberry Pi, NVIDIA Jetson, etc. Our long-term goal is to develop a comprehensive real-time aerial fire-

detection system. The model should not just classify cases of fire, but also perform pixel-level segmentation, severity estimation, and spread prediction, all in real-time. Furthermore, through the integration of the model with edge AI hardware-powered drones, our goal is to provide real-time and actionable fire surveillance that can lead to timely intervention, and ecological conservation. The promising results on the FLAME dataset [17] inspire us to further enhance the technique and generalize it for viable real-world application, eventually resulting in efficient, and autonomous wildfire monitoring solutions.

Bibliography

[1] J. Davis and M. Goadrich, "The relationship between precision-recall and roc curves," in *Proceedings of the 23rd International Conference on Machine Learning (ICML 2006)*, 2006, pp. 233–240.

[2] Z. Reitermanova, "Data splitting," in *WDS*, vol. 10, Prague: Matfyzpress, 2010, pp. 31–36.

[3] S. W. Taylor, D. G. Woolford, C. B. Dean, and D. L. Martell, "Wildfire prediction to inform fire management: Statistical science challenges," *Statistical science*, vol. 28, no. 4, pp. 586–615, 2013. DOI: 10.1214/13-STS451

[4] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[5] F. Chollet, "Xception: Deep learning with depthwise separable convolutions," *arXiv preprint arXiv:1610.02357*, 2017. DOI: 10.48550/arXiv.1610.02357

[6] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," *arXiv preprint arXiv:1612.03144*, 2017. DOI: 10.48550/arXiv.1612.03144

[7] T. Toulouse, L. Rossi, A. Campana, T. Celik, and M. A. Akhloufi, "Computer vision for wildfire research: An evolving image dataset for processing and analysis," *Fire Safety Journal*, vol. 92, pp. 188–194, 2017. DOI: 10.1016/j.firesaf.2017.06.012

[8] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *arXiv preprint arXiv:1708.02002*, 2018. DOI: 10.48550/arXiv.1708.02002

[9] J. Fu et al., "Dual attention network for scene segmentation," *arXiv preprint arXiv:1809.02983*, 2019. DOI: 10.48550/arXiv.1809.02983

[10] R. P. K. Poudel, S. Liwicki, and R. Cipolla, "Fast-scnn: Fast semantic segmentation network," *arXiv preprint arXiv:1902.04502*, 2019. DOI: 10.48550/arXiv.1902.04502

[11] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, "Eca-net: Efficient channel attention for deep convolutional neural networks," *arXiv preprint arXiv:1910.03151*, 2019. DOI: 10.48550/arXiv.1910.03151

[12] K. Govil, M. L. Welch, J. T. Ball, and C. R. Pennypacker, "Preliminary results from a wildfire detection system using deep learning on remote camera images," *Remote Sensing*, vol. 12, no. 1, p. 166, 2020. DOI: 10.3390/rs12010166

[13] M. Arif et al., "Role of machine learning algorithms in forest fire management: A literature review," *Journal of Robotics and Automation*, vol. 5, pp. 212–226, 2021. DOI: 10.36959/673/372

[14] J. R. Bergado, C. Persello, K. Reinke, and A. Stein, "Predicting wildfire burns from big geodata using deep learning," *Safety Science*, vol. 140, p. 105 276, 2021. DOI: 10.1016/j.ssci.2021.105276

[15] Z. Dai, H. Liu, Q. V. Le, and M. Tan, "Coatnet: Marrying convolution and attention for all data sizes," *arXiv preprint arXiv:2106.04803*, 2021. DOI: 10.48550/arXiv.2106.04803

[16] Y. Liu, E. Sangineto, W. Bi, N. Sebe, B. Lepri, and M. D. Nadai, "Efficient training of visual transformers with small datasets," in *Proceedings of the 35th International Conference on Neural Information Processing Systems (NeurIPS 2021)*, 2021, pp. 1824–1836.

[17] A. Shamsoshoara, F. Afghah, A. Razi, L. Zheng, P. Z. Fulé, and E. Blasch, "Aerial imagery pile burn detection using deep learning: The flame dataset," *Computer Networks*, vol. 193, p. 108 001, 2021. DOI: 10.1016/j.comnet.2021.108001

[18] H. Wang, C. Xiao, J. Kossaifi, Z. Yu, A. Anandkumar, and Z. Wang, "Augmax: Adversarial composition of random augmentations for robust training," *arXiv preprint arXiv:2110.13771*, 2021. DOI: 10.48550/arXiv.2110.13771

[19] K. A. Ali and W. J. Moses, "Application of a pls-augmented ann model for retrieving chlorophyll-a from hyperspectral data in case 2 waters of the western basin of lake erie," *Remote Sensing*, vol. 14, no. 15, p. 3729, 2022. DOI: 10.3390/rs14153729

[20] Q. Ansel, D. Sugny, and B. Bellomo, "Exploring the limits of the generation of nonclassical states of spins coupled to a cavity by optimal control," *Physical Review A*, vol. 105, no. 4, p. 042 618, 2022. DOI: 10.1103/PhysRevA.105.042618

[21] M. Maaz et al., "Edgenext: Efficiently amalgamated cnn-transformer architecture for mobile vision applications," *arXiv preprint arXiv:2206.10589*, 2022. DOI: 10.48550/arXiv.2206.10589

[22] R. Ghali and M. A. Akhloufi, "Deep learning approaches for wildland fires using satellite remote sensing data: Detection, mapping, and prediction," *Fire*, vol. 6, no. 5, p. 192, 2023. DOI: 10.3390/fire6050192

[23] K. Shah and M. Pantoja, "Wildfire spread prediction using attention mechanisms in u-net," in *2023 3rd International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, 2023, pp. 1–6. DOI: 10.1109/ICECCME57830.2023.10252734

[24] A. Shmuel and E. Heifetz, "A machine-learning approach to predicting daily wildfire expansion rate," *Fire*, vol. 6, no. 8, p. 319, 2023. DOI: 10.3390/fire6080319

[25] L. Wang, H. Zhang, Y. Zhang, K. Hu, and K. An, "A deep learning-based experiment on forest wildfire detection in machine vision course," *IEEE Access*, vol. 11, pp. 32 671–32 681, 2023. DOI: 10.1109/ACCESS.2023.3262701

[26] D. Qin et al., "Mobilenetv4 – universal models for the mobile ecosystem," *arXiv preprint arXiv:2404.10518*, 2024. DOI: 10.48550/arXiv.2404.10518

[27] L. Zhang, C. Shi, and F. Zhang, "Predicting forest fire area growth rate using an ensemble algorithm," *Forests*, vol. 15, no. 9, p. 1493, 2024. DOI: 10.3390/f15091493

[28] J. Zhou, W. Jiang, F. Wang, Y. Qiao, and Q. Meng, "Comparing accuracy of wildfire spread prediction models under different data deficiency conditions," *Fire*, vol. 7, no. 4, p. 141, 2024. DOI: 10.3390/fire7040141 [Online]. Available: https://doi.org/10.3390/fire7040141

[29] A. X. Wang, V.-T. Le, H. N. Trung, and B. P. Nguyen, "Addressing imbalance in health data: Synthetic minority oversampling using deep learning," *Computers in Biology and Medicine*, vol. 188, p. 109 830, 2025. DOI: 10.1016/j.compbiomed.2025.109830