NOVA IMS Information Management School

MDSAA

Master Degree Program in

Data Science and Advanced Analytics

Rural Fire Detection: A close-range approach for classification and localisation

Filipe de Correia Pedro Marçal Dias

Master Thesis

presented as partial requirement for obtaining a Master's Degree in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School

Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

Rural Fire Detection: A close-range approach for classification and localisation

by

Filipe de Correia Pedro Marçal Dias

Master Thesis presented as partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics, with a specialisation in Data Science.

Supervised by

Mauro Castelli, PhD, Nova Information Management School

July, 2024

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Filipe de Correia Pedro Marçal Dias

Lisbon, July 15th,2024

ACKNOWLEDGEMENTS

Obrigado!

Obrigado aos meus pais pela força e amor. Obrigado á minha companheira pela ajuda e paciência. Obrigado aos meus amigos pelas conversas e conselhos. Obrigado aos meus professores pelo conhecimento e supervisão. Obrigado ao meu orientador pelo apoio e disponibilidade.

ABSTRACT

Rural fires present a significant danger to both human lives and the environment. This research aims to improve fire management and firefighting strategies by utilizing computer vision (CV) and deep learning to develop a close-range rural fire detection framework. The focus is on using low computational cost devices capable of localizing and classifying fires. The study also explores the impact of near-infrared (NIR) images on model performance and compares the effectiveness of two key techniques: Convolutional Neural Networks (CNN) and Transformers. The proposed framework comprises two object detector models, YoloV8 and EfficientDet, followed by a classification module. Two CNN-based models and three Transformer-based models were used to evaluate this module. The input data for the detectors consists of a newly created dataset containing "Fire" and "No Fire" images, compiled from five other datasets. The outputs from the detector models are then used as inputs for the classification module. The results indicate that the MobileNet, EfficientNet, and MaxViT models exhibited the highest performance. Specifically, mobilenet_v3_small_pt and efficientnet_v2_s_pt achieved a 99% F1 score, while maxvit_t_pt reached a 98% F1 score, showcasing their strong suitability for this task. In conclusion, the developed framework demonstrated robust performance, with NIR images significantly enhancing model accuracy. The findings suggest that smaller networks are adequate for this specific task, making the framework suitable for deployment on low-computation resource devices. This adaptability enables the use of the framework, including the two detectors and one classifier, in microprocessors and robotic systems, considering the overall floating-point operations per second required.

KEYWORDS

Computer Vision; Deep Learning; Fire detection; Rural Fire; Object detection; Classification

Sustainable Development Goals (SDG):

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ANN	Artificial Neural Networks
BiFPN	Bidirectional Feature Pyramid Network
CV	Computer Vision
CNN	Convolutional Neural Networks
DL	Deep Learning
FPS	Frames per Second
GFLOPS	Giga Floating-point Operations per Second
GPU	Graphics Processing Unit
GT	Ground Truth
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IoU	Intersection over Union
MaxViT	Multi-Axis Vision Transformer
MLP	Multi-layer Perceptron
NIR	Near-Infrared
PCA	Principal Component Analysis
RGB	Red, Green, Blue
RELU	Rectified Linear Unit
R-CNN	Region-Based Convolutional Neural Networks
SE	Squeeze-and-Excitation
Swin Transformer	Shifted Windows Transformer
UAV	Unnamed Aerial Vehicle
ViT	Visual Transformer
YOLO	You only look once

1. INTRODUCTION

1.1. RURAL FIRES

Rural fires pose a significant threat to both human lives and the environment. This threat extends beyond the borders of Portugal, with numerous examples occurring worldwide. According to the Joint Research Centre of the European Commission, 2022 was the second-worst wildfire season in the European Union since 2000, when the Copernicus' European Forest Fire Information System records began. Furthermore, the total burnt surface in the protected areas in 2022 reached 365,308 ha, the highest amount mapped in the last ten years. The damage was mainly concentrated in three countries: Spain, Romania, and Portugal. Together, they accounted for over 75% of the total area burnt in protected areas (Joint Research Centre, 2023).

The severity of rural fires is not a coincidence or an act of isolated natural occurrences. It is a distressing manifestation of broader systemic changes, primarily driven by human activities. These activities have led to climatic shifts, setting the stage for drier conditions, longer droughts, and unpredictable weather patterns, all of which are catalysts for fires. The situation worsens when forests burn, as they release stored carbon into the atmosphere, accelerating global warming and destroying species' habitats.

In Portugal, 2022 witnessed a dry season with 10,389 rural fires, marking a continued decrease in fire incidents since 2018, with a 46% drop compared to the average from 2010 to 2019. However, there was a 26% increase in fires compared to the previous year of 2021. 83% (8,600) of these fires affected less than 1 hectare each. Notably, only 1% of the fires consumed more than 100 hectares, responsible for 85% of the total burned area. Fires over 500 hectares accounted for just 0.28% (29 fires) but were responsible for 70% of the burned area. Furthermore, 17 fires spread over 1,000 hectares each (0.16% of total fires), with one colossal fire consuming over 24,000 hectares, representing a fifth of the total area burned in 2022. Regarding fire suppression, an impressive 91% of fires were extinguished during the initial attack, with an average response time of 16 minutes to reach the operation theatre since the alert was received.

Analysing the last sentence, it is easy to understand that many rural fires are in small hectares, and a fast response to fire incidents is necessary. The research made in this thesis considers the lack of combat in such a specific environment; this is why, with the power of artificial intelligence, fire management and firefighting strategies can be enhanced due to the integration of CV and DL techniques. These techniques will equip robots with advanced vision systems that enable them to navigate hazardous terrains and assess the behaviour of the fire in real-time. With intelligent systems, one can analyse the fire's spread and intensity, providing insights into firefighting. From early detection to active suppression, these advancements empower a response more effectively and mitigate the devastating impact of fires on our communities and ecosystems.

Nevertheless, it is also critical to promote improvements that impact the governance of rural areas and define economic and fiscal incentives that mobilise owners and companies towards sustainable management. Taking care of these spaces involves a broad political and technical commitment to promote mechanisms and procedural improvements in the stakeholders to increase the scale of the operational fire combat.

1.2. COMPUTER VISION

Using computers to process and analyse images began in the 1970s. The so-called CV was viewed as the visual perception component of an ambitious agenda to mimic human intelligence and endow robots with intelligent behaviour (Szeliski, 2022). Over half a century, this domain has undergone transformative changes, with specific innovations leaving an indelible mark on the field. As the discipline evolved, it became evident that some concepts and methodologies were more resilient and influential than their counterparts. Reflecting this progression, numerous tasks have emerged as prevalent in CV, with image classification being the most common. However, other significant tasks include semantic segmentation, classification with localisation, object detection, and instance segmentation.

My research will specifically focus on image classification and localisation in line with these developments. This decision is encouraged by the remarkable pace of advancements in the field, which has been accelerated by the confluence of expansive datasets and powerful GPU architectures. The democratisation of neural network models and the swift exchange of knowledge through digital platforms have catalysed unprecedented growth in technological capabilities and an increase in the number of researchers. This era also witnessed the extension of static image analysis to dynamic video interpretation, paralleled by a surge in developing specialised hardware tailored for vision tasks.

Furthermore, with the rise of DL, CNN, which first emerged in 1998 (Lecun et al., 1998), and Vision Transformers (ViT), introduced by Google in 2020 (Dosovitskiy et al., 2021), have dramatically improved the accuracy and reliability of CV-based fire detection systems. When trained on large sets of real-world data, these systems become capable of high-accuracy real-time detection, making them viable for practical applications in various settings, from forests to industrial facilities.

Currently, there are various ways to utilise these technologies, which include local or cloud-based approaches. Many researchers opt for the cloud approach to process images, often using an UAV for data capture. However, real-time cloud connectivity is impractical due to communication interferences in closer-range fire detection and combat scenarios. Additionally, UAVs are not well-suited for fighting fires at close range primarily because of the intense heat and turbulent air conditions generated by the flames. The heat makes the air less dense, reducing the lift generated by the drone's propellers and compromising its ability to maintain altitude. Furthermore, the turbulent and unpredictable air currents can destabilise the UAV, making it difficult to control. As an alternative, devices operating from the ground at

close range can process images using microprocessors, but these impose limitations on the capacity to process large amounts of image data. Therefore, it is essential to consider not only the performance of the developed models but also the processing capacity required for their effective operation.

In addition, it is pivotal to comprehend the nature and scope of data this thesis aims to assess and the methodologies employed in its collection. In fire detection systems, three imaging technologies play critical roles: RGB images, Middle-Far Infrared (Thermal Infrared) Imaging, and NIR or Night Vision images. Each type has distinct advantages and limitations, making them suitable for different aspects of fire detection.

RGB images, derived from standard cameras, are cost-effective and widely accessible. Their primary advantage is the detailed visual information they provide under natural light conditions, which aids in detecting fires based on colour and texture. However, their effectiveness is limited in low light, smoke, and other visual obstructions, potentially reducing accuracy.

Thermal Infrared Imaging, used predominantly in UAVs, is a pivotal technology in identifying heat signatures and ongoing fires, especially in rural areas. It can detect hotspots even through smoke or foliage, which is critical where fires might start and spread undetected.

Near-infrared or Night Vision images are crucial, especially under low-light conditions or at night. While similar to thermal imaging in detecting heat signatures, they are generally less sensitive to high temperatures and used for close range. Their main advantage is their effectiveness in darkness, enhancing fire detection capabilities during the night.

Regarding cost and integration, RGB cameras are more affordable and accessible to integrate into existing systems. NIR imaging systems typically use standard camera sensors with infrared filters or sensors sensitive to the NIR spectrum. These sensors are more common and less expensive than those used in thermal infrared imaging. The technology behind NIR Imaging is also more straightforward and established, contributing to lower manufacturing costs. Therefore, NIR systems are often preferred in cost-sensitive while thermal imaging systems, are chosen in scenarios where their unique capabilities justify the higher expense.

1.3. DEEP LEARNING

Deep learning, a subset of machine learning, focuses on deriving increasingly significant representations from data through layered learning. The term 'deep' refers to the multiple layers of representations since the number of layers in a model determines its depth and the complexity of the data representation. Due to its structured approach to learning, this field could also have been aptly named 'layered representations learning' or 'hierarchical representations learning' (Chollet, 2018).

ANNs, the cornerstone models of DL, draw loose inspiration from our understanding of the human brain's architecture. However, it is crucial to clarify that these computational models do not function in the same way as biological brains. The complexities and nuances of brain functionality remain a topic of ongoing research and debate in fields like neuroscience and cognitive science, and the analogy to ANN is, at best, a conceptual one. In essence, neural networks should be considered sophisticated mathematical frameworks designed to identify patterns and learn from data; they leverage a layered structure where each layer hierarchically transforms the input data, enabling the system to learn increasingly complex representations.

Despite the biological metaphor, the processes involved are purely computational, involving algorithms, statistical processing, and a significant amount of data to 'train' these networks to perform a wide array of tasks with remarkable efficiency.

The first modern ANN, the perceptron, was developed by Rosenblatt in 1958, marking a foundational step in AI. It works as a binary classifier with a function that decides whether an input belongs to some specific class. If this decision is incorrect, it adjusts itself for a more accurate prediction in subsequent iterations, refining its accuracy over thousands or millions of these adjustments. An evolution of this algorithm is the multilayer perceptron (MLP), which is more potent than a single perceptron due to the existence of more layers of weights.

In general, neural networks have several parameters and properties crucial for their function and performance. These include the number of layers and neurons, which determine the network's depth and width; the learning rate, which affects how quickly the network updates its knowledge; activation functions, which introduce non-linearity to enable complex pattern recognition; and weights and biases, which are adjusted during training to optimise performance. Additionally, they utilise dropout rates and regularisation to prevent overfitting and ensure generalisation to new data; they also use a loss function to measure and guide the correction of prediction errors, being this process visualised through the error surface, which depicts errors across parameter spaces. Epochs refer to complete dataset cycles through the network, also known as iterations of backpropagation to be performed.

Technically speaking, it comprises multiple layers of neurons (nodes), where each neuron processes input data, applying non-linear transformations through activation functions. The network's architecture includes input, hidden, and output layers and incorporates many such hidden layers, enabling high-level features from raw data abstraction. Training these networks involves backpropagation and optimisation algorithms like stochastic gradient descent to adjust neuron weights, thereby minimising errors between the model’s output and actual data. This process enhances the network’s classification, regression, and pattern recognition accuracy.

1.4. THESIS OBJECTIVES AND RESEARCH QUESTIONS

Currently, the most effective systems are typically based on CNN. However, these networks are computationally expensive and consume much memory, usually requiring GPU to operate

correctly in emergencies (De Venâncio et al., 2022). This induces the necessity of researching approaches to apply CNN in a low computation cost device and other methods that the normal evolution of technologies brings. Therefore, my thesis proposes to answer the following questions:

1. RQ: “How to detect rural fire using CV in close range for low computational cost devices?”
    a. RO: To develop a CV framework for close-range rural fire detection for low computational cost devices.
        i. SRO1: To develop a classification and localisation framework for rural fire detection.
        ii. SRO2: To apply rural fire detection at close range.
        iii. SRO3: To evaluate the usage for low computational cost devices.
2. RQ: “How can infrared images increase rural fire detection model performance?”
    a. RO: To assess if infrared images can increase rural fire detection model performance.
3. RQ: “How can CNNs identify a fire in rural environments?”
    a. RO: To analyse how CNNs can identify fire in rural environments.
4. RQ: “How can ViT outperform CNN in rural fire detection?”
    a. RO: To compare the performance of ViT and CNN in rural fire detection.

A systematic approach will be undertaken based on the following methods to accomplish the research objectives. The primary goal is to address a specific problem within mobile networks and image processing. The first step involves a comprehensive review of existing literature to identify gaps or areas where previous research has left room for improvement. This critical literature analysis will be the foundation for the subsequent research activities.

Once the research gaps have been identified, the next step is to delve into the fundamental principles that distinguish various types of mobile networks. Understanding these underlying principles is crucial for developing practical solutions tailored to the specific challenges posed by mobile networks. This phase will thoroughly explore the core concepts and technologies that form the basis of mobile network architectures.

With a solid understanding of the research landscape and the principles governing mobile networks, my study will investigate commonly used model settings documented in scientific literature. By leveraging insights from established research, the aim is to select optimal configurations to enhance model effectiveness.

Following the initial investigation, the research will transition into a practical phase involving creating and assessing a small modelling framework explicitly designed to address the identified problem.

To further explore the problem space, the study will leverage the available mobile architecture implementations in the PyTorch framework. These pre-existing architectures will be applied to the problem, and their performance will be compared. Additionally, the research will extend its scope to include ViT models, assessing their effectiveness in solving the problem and contrasting their performance with previously tested models.

An essential aspect of the research process involves evaluating the performance of all models developed and assessing their computational cost. This evaluation will provide valuable insights into the trade-offs between model effectiveness and resource utilisation, aiding in selecting the most suitable approach for addressing the research problem.

In summary, this research follows a structured progression, beginning with a thorough literature review and a deep dive into mobile network principles, manual framework creation, automated model comparisons, ViT integration, and comprehensive model evaluation. This systematic approach aims to develop a well-informed and effective solution in the context of mobile networks to improve the real-time adaptability and robustness of CV and DL systems for rural fire detection.

1.5. THESIS STRUCTURE

The thesis comprises five chapters. Chapter 2 introduces concepts related to ANNs, CNNs, and Transformers and provides various examples of their use in fire detection. Chapter 3 outlines the research methodology used to develop the framework for fire detection. Chapter 4 presents the thesis results and compares them to the literature reviewed. Finally, Chapter 5 contains the conclusions, limitations, and potential future research directions.

2. LITERATURE REVIEW

2.1. TRADITIONAL FIRE DETECTION METHODS

Before the advent of CV, traditional fire detection methods involved physical sensors, such as smoke detectors and thermal detectors (De Venâncio et al., 2023). These detectors were widely used and relied on detecting the presence of smoke or abnormal heat levels to alert individuals of a fire. Additionally, manual human inspection played a crucial role in early fire detection; for example, security guards or designated personnel were responsible for monitoring and watching for possible fire occurrences. Their job would be to visually inspect the area for signs of smoke or flames and take appropriate action if a fire was detected. However, these traditional fire detection methods had limitations (Luo, 2022); they were prone to false alarms and missed detections and were unsuitable for detecting outdoor fires in environments like forests and farms, restricting their usefulness.

Through the years, the limitations of traditional fire detection methods became increasingly apparent, making researchers seek more sophisticated techniques to improve early fire detection. The shortcomings of human monitoring and traditional detectors led to the exploration of new technologies, such as gas sensors and image fire detection, which aimed to overcome the limitations of traditional methods by leveraging advancements in CV and sensor technology.

Based on the article "Gas Sensor Technologies for Fire Detection" by Gutmacher et al. (2012), gas sensor technology showed the ability to detect fires four minutes before the presence of smoke and marked a significant step forward in early fire detection. This provided an early alarm system that could mitigate fire-related damages and losses. In addition to this innovation, image fire detection technology advancements emerged as a promising early detection approach (P. Li & Zhao, 2020).

Early fire detection is crucial for the timely response and mitigation of fire incidents, so after the initial approaches, researchers kept developing new technologies and, more recently, fire detection relied on temperature sensors, smoke detectors, and thermal cameras. However, these methods also showed limitations in detecting fires in large spaces, complex building structures, and in the presence of disturbances. Additionally, these technologies were susceptible to false alarms or delays in detection, risking potential damage to life and property. Finally, the appearance of CV revolutionised fire detection by offering a more accurate and efficient method.

Nowadays, CV applications for fire detection in rural areas have gained significant importance. This trend is propelled by technological advancements that have enhanced the reliability and effectiveness of CV in identifying and localising fire incidents. A key driver in this domain is the rapid evolution of digital camera technology, which equips various systems with high-resolution capabilities enabling the capture of clear and detailed images crucial for the task.

The research shift towards image-based fire detection technology utilising algorithmic analysis marks a significant transformation since CV algorithms can analyse image features to detect fire, identifying elements like flames, smoke, and heat patterns. The system rapidly and accurately interprets these features, thanks to integrating DL and CNN, enhancing the detection process's accuracy and reliability.

One of the significant advantages of this technology, as outlined by (Sousa et al., 2020), is its high sensitivity and accuracy. CV algorithms can analyse images with great precision, detecting subtle visual cues of fire, such as flame colour and smoke patterns, offering a higher accuracy level than traditional methods. Furthermore, the installation flexibility is another significant benefit since image fire detection technology can seamlessly integrate with existing camera systems, making it adaptable to various spaces. Furthermore, the ability for real-time detection, leveraged by computer processing power, allows image fire detection algorithms to analyse images and provide instant alerts, facilitating a response to fire emergencies.

Lastly, its capability for continuous area monitoring with minimal human involvement ensures prompt fire detection, thereby mitigating potential extensive damage. Such technological advancements have positioned CV as an indispensable tool in rural fire detection, leveraging DL techniques for heightened accuracy and reliability.

2.2. ARTIFICIAL NEURAL NETWORKS

Neural networks have evolved significantly since Frank Rosenblatt introduced the perceptron in the 1950s (Rosenblatt, 1958). This early model was foundational for developing more advanced ANN.

The primary function of perceptron is binary classification, which lays the groundwork for complex neural architectures and algorithms. It works as a binary classifier, receiving inputs, applying weights, and processing these through an activation function to generate an output. In its structure, data features are inputs for learning; then adjustable weights represent learned knowledge; subsequently, a summation function calculates weighted input sums, and an activation function works as a function for binary outcomes. Furthermore, the learning phase involves feeding the model with training data and adjusting the weights based on the prediction accuracy until a certain level is achieved.

Compared to models like Adaline or the Hopfield network, the perceptron stands out for its simplicity and effectiveness in linear classification tasks. Its influence extends to modern Neural networks, with backpropagation, still being a pivotal technique for optimising performance through iterative weight and bias adjustments.

However, it also faced challenges, such as the inability to solve non-linear problems like the Exclusive Or function, highlighting the need for more sophisticated models. This led to the development of MLPs. MLPs work similarly to perceptrons but have hidden layers that allow them to handle non-linear classification and represent more complex data patterns.

Nevertheless, nowadays, most models with neural network architectures owe much to the perceptron's principles. These principles include using error analysis techniques, comparing the network's output with expected results to minimise errors, and using various activation functions such as Sigmoid or RELU.

The perceptron and MLP are critical components of ANN, revolutionising pattern classification and learning.

2.2.1. Convolutional Neural Networks

Convolutional Neural Networks were introduced in 1998 as a specialised neural network architecture for image processing and pattern recognition. The early experiments started with LeNet-5, a convolutional neural network developed by (Lecun et al., 1998).

The foundations of LeNet-5 are a set of several elements applied to all CNNs, beginning with the input layer, which is typically an image composed of Width, Height and Depth. From a numerical perspective, it is simple to visualise an input as an array of pixels with elements starting from 0 to a defined pixel number. Furthermore, “Width” and “Height” are easy to identify on an image since they are related to the vertical and horizontal size; however, the "Depth" attribute is not so straightforward but denotes the colour channels of the image: 3 for RGB or 1 for grayscale.

The subsequent layers in a CNN are the convolutional layers, which play a vital role in the network; they enable the integration of local data from input images of different dimensions (1D, 2D, 3D) to capture significant patterns and characteristics. The convolution process involves moving a small matrix known as a filter or kernel over the input image in horizontal and vertical directions based on the specified stride, conducting element-wise multiplication, and then summing the outcomes to generate a feature map. In this stage, using diverse kernels allows more complex and diverse features to be captured from the input image.

Another essential element is the feature map, extracted from the convolutional layers and defined by the number of filters used. These filters are influenced by the kernel pixels that slide over the input matrix. The padding parameter is used to input 0s around this matrix's border and handle the kernel's dimensions in the convolutional network.

Furthermore, CNNs have pooling layers that help reduce the feature maps' spatial dimensions. They are typically used between successive convolutional layers to reduce the parameters or computational complexity and prevent overfitting. Some examples of types of pooling operations are max pooling and average pooling. These operations reduce the size of feature maps by selecting the maximum or average value from a specified pool of pixels and transferring it to the corresponding cell in the feature map.

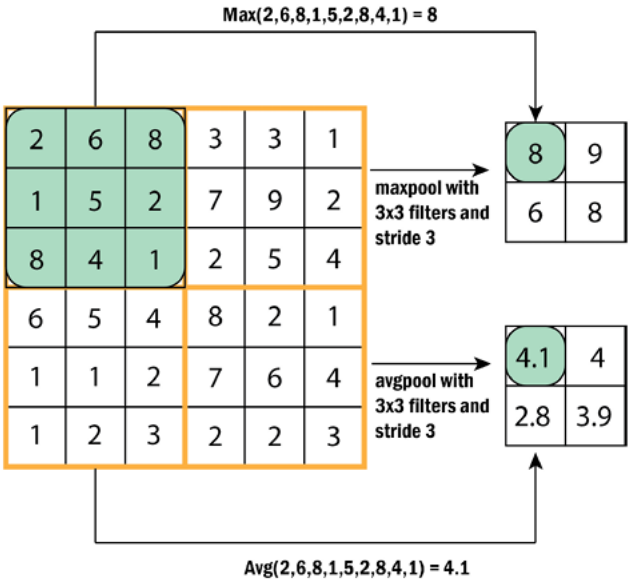


Figure 1: Illustrates how max pooling and average pooling work (Albuquerque, C, 2018)

Lastly, the fully connected layers are used in CNNs to categorise the characteristics extracted from the convolutional and pooling layers. They aim to use features obtained from earlier layers to classify objects based on training data and facilitate the learning process of non-linear combinations of features. The output volume dimensions $[1\ x\ N]$, where N represents the number of output classes being predicted, with each value signifying the probability of an object being included in a specific class. This probability is determined by examining which high-level features correlate more with a particular class and then utilising weights from current and previous layers to compute probabilities for each class. Regarding object detection problems, the output size depends on the number of classes that must be detected. The final output for each bounding box is in the form of $[pi, xi, yi, wi, hi]$, where pi corresponds to the probability value that the detected object belongs to a certain class, and the other parameters represent the coordinates of the bounding box.

Activation functions enable the nonlinear output transformation within a neural network from one layer to another. There are several types of these functions, but one of the most used is the RELU, which replaces all input negative values in the feature map with zero and leaves positive values dependent on a defined threshold.

RELU activation function has the mathematical form:

$$f(x) = max(0, x)$$

Another well-known activation function is the Softmax, which is typically used in the output layer of a CNN for multi-class classification tasks. This function normalises the network output into a probability distribution, allowing the model to predict by selecting the class with the highest probability. It has the following formula:

$$f(x) = e^x / sum(e^x)$$

In addition to RELU and Softmax functions, other activation functions are vital in DL, such as the Sigmoid function, which transforms any value into probabilities and is applied in binary classification tasks. When used, if the inputs are very small, the outputs tend towards zero; conversely, if it is too large, it approaches one. Another function that should be mentioned is the Tanh, which has a normalised range from -1 to 1. Its principal benefit lies in the ability to handle negative numbers effectively.

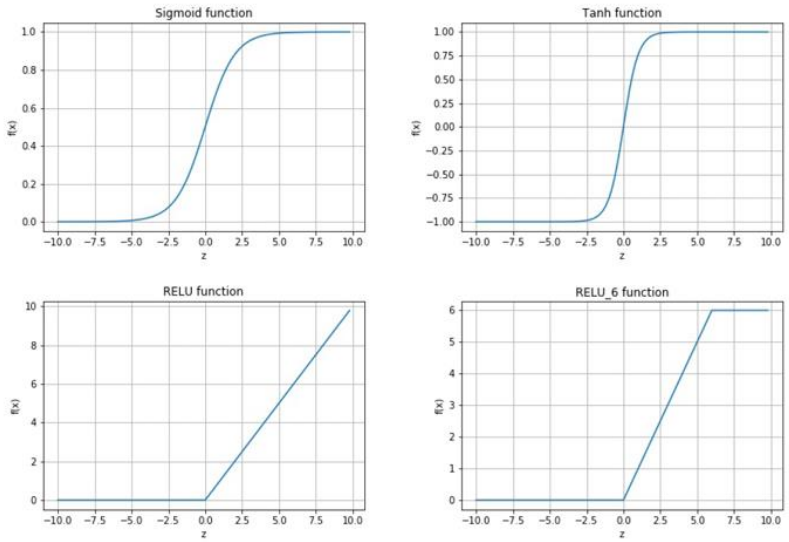


Figure 2: Activation functions used in ANN (Albuquerque, C, 2018)

Batch normalization is a technique that helps control overfitting better by introducing some level of noise. It works by normalizing the activation functions in each batch, ensuring that the mean is close to 0 and the standard deviation is near 1. This approach accelerates training by allowing the use of higher learning rates and acting as a regulator for the initial parameters.

2.2.1.1. LeNet-5

As previously stated, LeNet-5 by Yann LeCun in 1998, was one of the first CNN developed with the idea of solving CV problems, specifically handwritten digit recognition and consisted of multiple layers, as Figure 3 suggests:

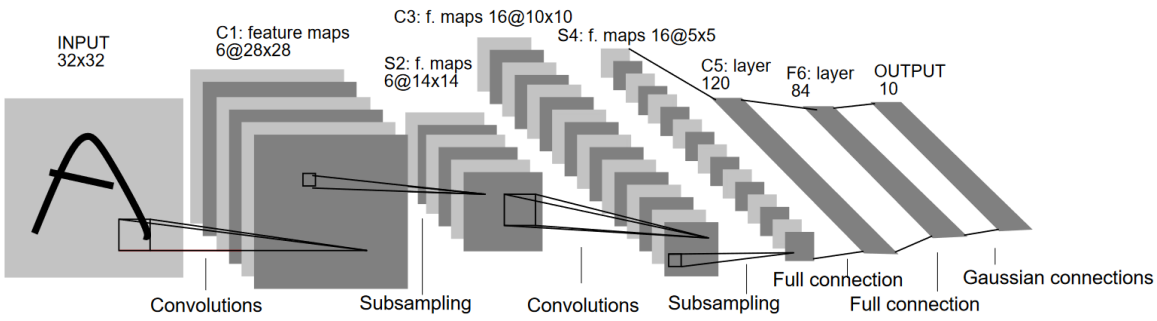


Figure 3: LeNet-5 architecture

The input layer is designed for *32x32* grayscale images. Then, the initial convolutional layer, *C1*, consists of 6 feature maps using a *5x5* kernel and produces outputs of size *28x28*. Subsequently, the subsampling layer *S2*, known as a pooling layer, downscales the spatial dimensions to generate feature maps of size *14x14*. Moving on to the next convolutional layer, *C3* comprises 16 feature maps connected to multiple neighbourhood regions in *S2* feature maps with an output dimension of *10x10*. Throughout the Convolutional layers (C1, C3) operations, the network uses convolution operations where the kernel is slipped over the input to compute the dot product. Like any other CNN, these processes help the network learn specific features like edges and textures. (Lecun et al., 1998)

Following this, a subsampling layer *S4* reduces the feature map size to *5x5*. Both pooling layers *(S2, S4)* decrease the input's spatial dimensions, resulting in a smaller and more controllable representation. Then comes the Convolutional Layer *C5*, which contains 120 feature maps, and each unit is linked to a specific region across all 16 of the *S4* feature maps. In this layer, the size of the feature maps matches the kernel size, making it function as a fully connected layer. The next component is a fully connected layer containing 84 units designed to further process the data from *C5*. Finally, the output layer contains ten output units corresponding to digits from 0 to 9 and uses a basic network function like Softmax for classification. Activation functions such as Sigmoid or Tanh throughout the network introduce non-linearity as they are re-used within each convoluted fully connected layer.

After the discovery of LeNet-5, there was a significant breakthrough in developing CNN for image classification tasks. However, image classification and object detection remained a challenge due to the complexities and variations in images, which made researchers explore architectures and techniques to enhance performance in specific problems. The key question is not which model is the best but which model and its different configurations can provide the better balance between speed, accuracy, and processing capability required for the problem.

2.2.1.2. AlexNet

One major milestone in CNN's history was the introduction of AlexNet in 2012. It was the first deep convolutional neural network with its “modest” 60 million parameters to achieve a top-5 test error rate of 15.3% in the ILSVRC, which is one of the most renowned object recognition challenges (Krizhevsky et al., 2017).

AlexNet and LeNet-5 are fundamental CNN with similar structures and capabilities. AlexNet is more complex and has a structure composed of eight layers, the first five convolutional ones. These layers filter the input image by gradually increasing the number of kernels while decreasing their size. One of his unique characteristics is using RELU activation functions with max-pooling layers connected to the convolutional layers. This setup is followed by three fully connected layers, culminating in a Softmax output layer for classifying among 1000 different classes.

One feature of AlexNet that is absent in LeNet-5 is local response normalisation, which is applied after certain RELU layers to improve generalisation. This is accomplished by introducing competition between kernel outputs, emphasising more prominent features while reducing the impact of less significant ones.

Furthermore, to mitigate overfitting AlexNet incorporates overlapping pooling which helps in error reduction and challenges the model to memorise noise patterns within the training data then, it also incorporates data augmentation techniques like image translations, horizontal reflections, and adjustments on RGB channels. Additionally, dropout techniques are used in the first two fully-connected layers to prevent co-adaptation of neurons during training and, in terms of training methodology, AlexNet uses stochastic gradient descent with a batch size of 128, momentum set at 0.9 and weight decay at 0.0005.

The success of the AlexNet architecture renewed interest in CNN and triggered another wave of research and advancements in the field, laying the foundation for many of the subsequent developments.

2.2.1.3. Region-Based Convolutional Neural Networks

R-CNN was a major development in the field of object detection when proposed in 2014 with the motto of "features matter". It focused on using deep and rich feature hierarchies for object detection and semantic segmentation tasks (Girshick et al., 2014).

The R-CNN process systematically generates approximately 2000 region proposals for each image and calculates features for each of those regions, demonstrating a comprehensive understanding of both image segmentation and feature extraction. Likewise AlexNet, the researched model, was pre-trained on the ILSVRC dataset, enabling it to acquire a diverse representation of visual features applicable across different image domains.

Following pre-training, the network was fine-tuned (a method to allow the network to adapt) on the PASCAL VOC dataset, tailored to the specific domain of detection tasks. This two-stage training methodology showed how to train deep neural networks in data-constrained environments.

After being processed by CNN layers, R-CNN classifies each region using a Support Vector Machine per class. These models are designed for computational efficiency as they only involve a matrix-vector product and the use of non-maximum suppression, which is a crucial consideration for practical applications.

The authors extended the R-CNN framework to semantic segmentation by modifying the method to generate pixel-level segmentation, demonstrating the flexibility and effectiveness of this network. Meanwhile, critics of R-CNN expressed concerns about its computational inefficiency, especially with generating region proposals for each image, the absence of end-to-end training and using three classifiers since it significantly increases the number of

parameters required. These concerns prompted further research and improvements in subsequent years, leading to the development of more efficient and streamlined object detection algorithms such as Fast R-CNN and Faster R-CNN.

In Fast R-CNN, the authors tackled R-CNN's computational inefficiency by removing the requirement to generate region proposals for each image. Instead, inference is applied only once, and its output is utilized to suggest bounding boxes. This introduced a new concept called Region of Interest Pooling, which involves pooling features from the CNN to identify regions of interest. As a result, this improvement enables end-to-end training and eliminates the need for multiple classifiers used in the previous version.

Lastly, Faster R-CNN introduces a Region Proposal Network (RPN), allowing it to generate feature maps directly from the CNN to propose regions of interest, eliminating the need for external region proposal methods like selective search or edge boxes. The RPNs are built by adding two convolutional layers: one encodes feature map positions into vectors, and the other predicts object presence and region bounds for multiple proposals. A small network, connected to every position in the feature map, transforms each area into a vector. Also, using shared features and end-to-end training, Faster R-CNN significantly improved detection accuracy and incorporated a simplified pipeline in object detection tasks (Ren et al., 2017).

Various alternatives of Faster R-CNN have been used regarding the specific use for fire detection. In 2018, a paper titled "Using Popular Object Detection Methods for Real-Time Forest Fire Detection" (Wu & Zhang, 2018) explored and compared the effectiveness of Faster R-CNN for real-time forest fire detection, which demonstrated short training times and high detection accuracy but a low frame rate. The authors adjusted the number of iterations for each training phase while maintaining the default settings for all other parameters.

Furthermore, (Barmpoutis et al., 2019) combined the power of Faster R-CNN with multi-dimensional texture analysis based on linear dynamical systems to validate fire regions. Their methodology involved identifying fire regions with a Faster R-CNN network trained for fire detection and tested with three different architectures: AlexNet, VGG16, and Resnet101. The candidate fire regions are then analysed using multidimensional texture analysis that divides each region into rectangular patches, modelled using linear dynamical systems and projected into a Grassmann manifold, representing each region as a cloud of points. For classification, a vector of locally aggregated descriptors encoding is applied to aggregate these Grassmannian points based on a local criterion on the manifold.

Another exciting approach from (Kim & Lee, 2019) related to videos involved using Faster R-CNN to detect Suspected Regions of Fire (SRoFs). After identifying SRoFs, Long Short-Term Memory accumulates summarised features within bounding boxes in successive frames over time. This allows the method to consider the longer-term dynamic behaviour of fire by simulating a human observation, where a person continuously monitors a suspected fire and accumulates temporal behaviours to decide if it's an actual fire. At last, a voting scheme is

applied to make the final decision based on the series of accumulated decisions over a long period.

"A Deep Learning Framework for Autonomous Flame Detection" by (Z. Li et al., 2021) presents a methodology where it is recognised that flames exhibit significant diversity in terms of colour, texture, and shape. This diversity made the authors design a network to account for flames' dynamic and colour characteristics by using Online Robust Principal Component Analysis (OR-PCA). This advanced technique effectively isolates the dynamic features of flames from video backgrounds and Dirichlet Process Gaussian Mixture Model (DPGMM) to capture the colour properties of flames. The integration of DPGMM allows for a more accurate representation of flame colours, which is essential given the high variability in flame appearances. The network is trained on a set of images and is subsequently tested on videos. A crucial aspect is using ResNet-50 as a backbone for fine-tuning the R-CNN. This optimises the network's ability to identify and confirm flame regions, leveraging ResNet-50's powerful feature extraction capabilities. In sum, the combined use of OR-PCA for dynamic feature isolation and DPGMM for advanced colour profiling, all integrated into a specially adapted R-CNN, exemplifies another innovative approach to the complex challenge of flame detection.

The paper titled "Forest Fire Segmentation from Aerial Imagery Data Using an Improved Instance Segmentation Model" by (Guan et al., 2022) focuses on the segmentation of forest fires from aerial imagery data. The authors try to solve both classification tasks using the Dual Semantic Attention (DSA) module in DSA-ResNet and instance segmentation tasks using MaskSU R-CNN. The DSA module incorporated in ResNet dynamically selects and fuses feature maps from different scales of convolution kernels. This module is implemented via three operators: Separate, Fuse, and Select. The Separate operator divides channel attention, the Fuse combines weights from different branches, and Select integrates this attention into the network. This approach helps focus on label-related regions, enhancing the network's ability to discern fire areas from complex backgrounds. The new MaskSU R-CNN incorporates a MaskIoU branch reconstructed by a U-shaped network to reduce segmentation errors, particularly in edge pixel correction. The structure of this model comprises the previous DSA-ResNet50 as its backbone, augmented by a Feature Pyramid Network for multi-scale feature extraction. The multi-branch prediction network within MaskSU R-CNN includes branches for classification, bounding box regression, mask generation, and segmentation evaluation, all focusing on improving segmentation quality. The U-shaped network in the MaskIoU branch helps in better feature fusion and reduces feature loss, which is crucial for segmenting edge pixels of fire. The novel combination proposed by these authors leads to higher results in forest fire segmentation but also requires a more computationally intensive model.

2.2.1.4. YOLO

YOLO presented by Redmon in 2016, revolutionised object detection by offering a real-time solution with high accuracy for that time, this was a novel approach to object detection by

rethinking it as a regression problem. Unlike the traditional methods of repurposing classifiers for detection, YOLO divides the input image into an *S × S* grid, and each grid cell is responsible for detecting an object by predicting several bounding boxes composed of five values. Those values are the *x* and *y* coordinates representing the centre of the box concerning the bounds of the grid cell, the width (*w*) and height (*h*) that are calculated based on the input and a confidence score known as the difference between the GT and the IOU metric. The significance of this confidence score $Pr(Object) * IOU_{pred}^{truth}$ lies in its dual nature since, not only is the probability that a box contains an object but also indicates how accurately this prediction has been made, which enables YOLO to balance between detecting objects and minimizing false positives (Redmon et al., 2016).

Each grid cell predicts the conditional class probabilities $Pr(Class_i|Object)$ for each class when detecting an object in the cell. These probabilities consider the presence of an object, allowing the network to be specialised in predicting certain classes based on their spatial location within the image and to learn contextual information about objects and their typical surroundings. The class-specific confidence scores for each box are given by:

$$Pr(Class_i|Object) * Pr(Object) * IOU_{pred}^{truth} = Pr(Class_i) * IOU_{pred}^{truth}$$

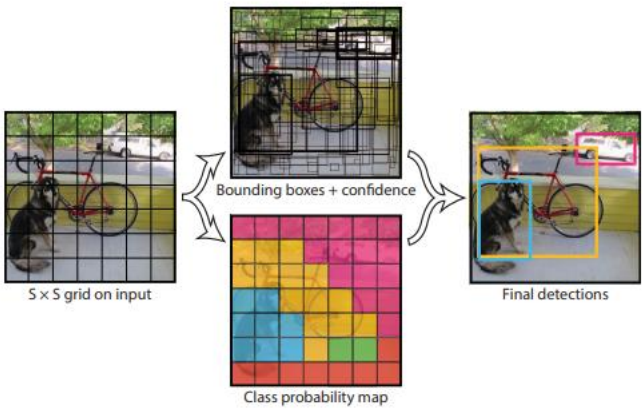


Figure 4: YOLO system models detection process (Redmon et al., 2016)

Apart from the fundamental elements of YOLO, such as its unified architecture and the efficient use of a single CNN, the system also includes 24 convolutional layers that alternate between *1x1* reduction layers and *3x3* convolutional layers before ending with two fully connected layers. This architecture is illustrated in the figure 5.

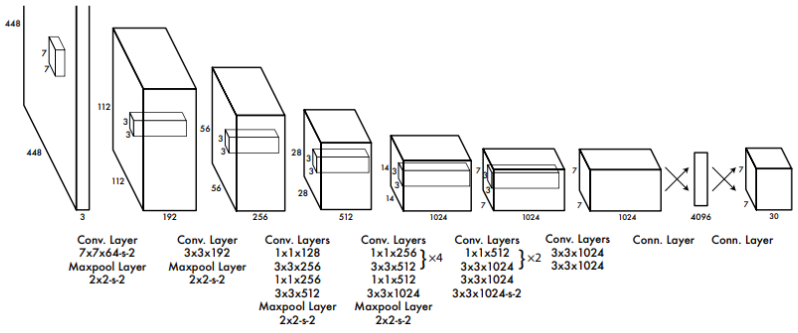


Figure 5: YOLO architecture (Redmon et al., 2016)

Like in the previous models, YOLO's training process involved using the ILSVRC dataset, where a top-5 accuracy of 88% was achieved. After testing this classification problem, the authors converted it to perform detection by adding convolutional and connected layers and changing the input resolution. Its unified training approach, characterised by a designed loss function and specialised learning rate, effectively balances the model's focus between precise bounding box localisation and reliable object classification. This is achieved through the strategic use of parameters, which increase the loss from bounding box coordinate predictions and reduce the loss from confidence predictions in non-object-containing boxes. Additionally, the model's adeptness in different object sizes is increased by predicting the square root of the bounding box dimensions rather than their direct measurements. Using the bounding box predictor figure, specialised in certain object characteristics and classes, ensures an improved recall.

Furthermore, YOLO was also trained on the PASCAL VOC datasets, with a gradual learning rate increase to stabilise gradients and a phase-wise reduction. This is combined with implementing dropout and data augmentation techniques, including random scaling, translations, and HSV colour space adjustments.

Despite its strengths, this model has limitations, such as a fixed grid structure that only allows each cell to predict two boxes and assign one class, reducing effectiveness in detecting groups of small objects or objects with unusual aspect ratios. Also, the focus on speed leads to a higher rate of localisation errors than the previous complex systems like R-CNN.

Until now, YOLO's network structure has continually evolved, with various versions, such as YOLOv1 to v8, and adaptations designed for improved speed and reduced computational resource requirements, like Fast YOLO and YOLOv4-tiny.

Fire detection using the YOLO model has various applications. In the research paper (Wu & Zhang, 2018), a proposed modification of the tiny-yolo-voc structure, trained on a dataset comprising 1000 images of fire and smoke, enhanced detection accuracy for both classes. However, additional layers, like one convolutional layer with an 8-filter and one max pooling layer, were added to create the modified version called tiny-yolo-voc1, which outperformed

the original structure and yolo-voc2.0. Ultimately YOLOv3 was identified as the superior configuration due to its effective balance between accuracy and real-time detection capabilities.

Throughout the papers, the combination of PCA with object detection algorithms such as R-CNN and YOLOv3 is recurrent. PCA is a statistical procedure utilised to transform a set of possibly correlated variables into a smaller set of uncorrelated variables known as principal components. This transformation is accomplished through the Singular Value Decomposition method, which identifies and retains the most significant features of a dataset. The authors effectively reduced computational complexity and noise by extracting and using only 300 principal components from the original high-dimensional image data. This dimensionality reduction streamlines the dataset and enhances the detection accuracy of YOLOv3. This model has an architectural design using dimensional clusters to generate anchor boxes for bounding box prediction. It incorporates continuous 3 × 3 convolutional layers with 1 × 1 convolutional layers, the standard configuration followed (Shixiao Wu et al., 2020).

In the paper "Image Fire Detection Algorithms Based on Convolutional Neural Networks" (P. Li & Zhao, 2020), the authors do a comparative analysis of Faster R-CNN and YOLOv3, examining their accuracy and robustness in fire detection. The proposed YOLOv3 uses a specialised variant of the Darknet architecture known as Darknet-53. The Darknet-53 is an open-source neural network framework that works as the backbone of YOLOv3 and comprises 53 convolutional layers, offering a balance between computational efficiency and depth for feature extraction. This adaptation for fire detection involved the removal of the last three layers of the standard architecture, resulting in a smaller-scale feature map. The YOLOv3 in the study generates three different scales of feature maps by up-sampling smaller-scale feature maps and concatenating them with earlier layer feature maps. This methodology ensures that the final feature maps possess a rich blend of location and complex feature information, crucial for accurate fire detection. Furthermore, YOLOv3 doesn’t use the traditional Softmax classification and instead uses an independent logistic classifier for each bounding box. This change is particularly beneficial for detecting overlapping instances of fire and smoke. For the specific purpose of this paper, the model is fine-tuned on two categories (fire and smoke) and highlights the performance in fire detection, with an average precision of 83.7% and a speed of 28 FPS.

Furthermore, similar to R-CNN, the paper "Fire Detection Based on a Two-Dimensional Convolutional Neural Network and Temporal Analysis" (De Venancio et al., 2021) introduces a fire detection system that integrates YOLOv4 and Tiny YOLOv4 with temporal analysis. The methodology comprises two main stages: detection and tracking. In the detection stage, the YOLOv4 model with a complex architecture including CSPDarknet53, Spatial Pyramid Pooling, Path Aggregation Network, and YOLOv3, was a powerful but computationally intensive architecture which led to the analysis of a lighter version known as Tiny YOLOv4. The tracking stage involves analysing the temporal behaviour of detected objects over time, using the

centroids of detected fire and smoke objects. This multi-step process involves assigning unique identifiers to new detections, calculating euclidean distances between centroids across frames, and monitoring the temporal behaviour of detected objects. The system's novelty lies in its ability to confirm fire or smoke detections by analysing the growth patterns of the bounding boxes over time, thus effectively reducing false positives. Including the coefficient of variation as a measure of temporal growth rate highlights the method's sophistication in distinguishing fire and smoke occurrences from others like solar reflections or car lights.

In 2022 (Luo, 2022) proposed using YOLOv5 for detecting fires using a methodology that closely aligns with the approach intended to be replicated in this thesis. The YOLOv5 implementation involves a two-stage process. Initially, the model undergoes pre-training with an extensive image dataset, demanding substantial storage and computational resources. Once trained, the model is deployed on edge devices at fire detection terminals with limited memory and computing capabilities. However, this system incorporates diverse transmission networks like 5G, which are beyond the scope. The architecture of YOLOv5 varies from earlier versions described in this thesis by not using the Darknet framework and adopting the PyTorch framework. This resulted in a compilation of models rather than just a single model.

The writers of "An Automatic Fire Detection System Based on Deep Convolutional Neural Networks for Low-Power Resource-Constrained Devices" (De Venâncio et al., 2022) have previously explored this field, and they are now introduced a customised YOLOv4 designed for low-power devices. The methodology involves training YOLOv4, which incorporates 110 convolutional layers with 33 thousand convolutional filters and 64 million parameters, followed by applying filter pruning techniques to reduce computational costs while maintaining performance. This includes:

- Importance Criteria-Based Pruning, where filters are pruned based on metrics like '1-norm' or '2-norm' values.
- Multiple Projections, which project filter outputs to lower dimensions for evaluation.
- Clustering-based pruning, grouping similar filters and reducing redundancy.
- Random Pruning, a method of randomly removing filters to serve as a baseline for comparison with other structured pruning approaches.

The results of this study demonstrate that the pruned YOLOv4 model maintains a high level of accuracy in detecting fire and smoke while operating on a Raspberry Pi 4. This balance between performance and efficiency addresses a critical challenge in deploying advanced CNNs in real-world, resource-constrained environments.

"A Flame Detection Algorithm Based on Improved YOLOv7" by (Yan et al., 2023) uses one of the latest versions of YOLO, the YOLOv7, which has undergone improvements in both the backbone and head structure. The backbone structure of YOLOv7 has been improved by incorporating a Simple Attention Module (SimAM) into one of the modules. This enhancement

aims to improve the detection of small-scale flames, which is a challenging aspect of flame detection algorithms. The SimAM structure does not have parameters, reducing computational complexity while effectively improving the model's ability to focus on relevant features. This improvement is especially valuable when background elements can easily hide small flames. Regarding the head structure of YOLOv7, the authors incorporated a CNeB module build based on ConvNeXt into the ELAN-W module. Initially, ELAN-W undergoes a *1×1* convolution to adjust the channels. Subsequently, four *3x3* convolutions are used to extract image features before being processed by the CNeB module, which enhances feature extraction within the network model. The enhanced ELAN-W module contributes to improving the detection accuracy and addressing the false detections—this enhanced feature extraction and classification for better flame detection in complex backgrounds. The integration of CNeB is a significant advancement in enhancing YOLOv7's performance across diverse environments.

2.2.1.5. SqueezeNet

SqueezeNet is a CNN architecture that started to be known by achieving an AlexNet accuracy level with significantly fewer parameters and a smaller model size (Iandola et al., 2016). This architecture is known for its compact and efficient design, consisting of a single convolutional layer, followed by eight Fire modules, allowing additional feature extraction and concluding with a final convolution layer. The authors implemented a strategy divided into three parts:

1. Replacing *3x3* filters with *1x1* filters;
2. Reducing the number of input channels to *3x3* filters;
3. Incorporating down sampling later in the network.

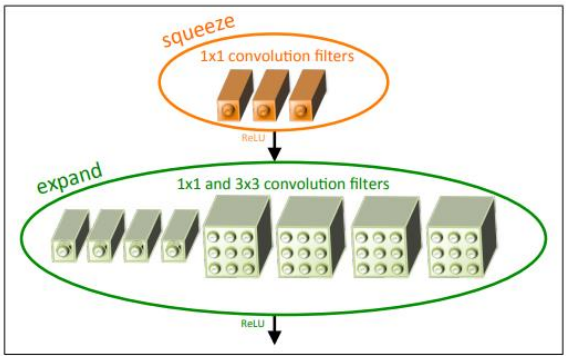


Figure 6: Organization of convolution filters in the Fire module (Iandola et al., 2016)

The highlight of SqueezeNet is the Fire module, which uses a *1x1* convolution to decrease the input depth, referred to as "squeeze", followed by a combination of *1x1* and *3x3* convolutions known as "expands". Figure 6 shows that the number of filters in the squeeze layer, denoted

by $s_{1x1} = 3$, while $e_{1x1} = 4$ represents the quantity of *1x1* filters and $e_{3x3} = 4$ indicates the number of *3x3* filters in the expand layer. The following condition is applied:

$$s_{1x1} < (e_{1x1} + e_{3x3})$$

The squeeze layer restricts the number of input channels to the 3x3 filters. When the authors mention reducing the number of input channels to the 3x3 filters, they are referring to the number of these feature maps being sent into the next layer of the network.

"Efficient Deep CNN-Based Fire Detection and Localization in Video Surveillance Applications" by (Muhammad, Ahmad, et al., 2019) uses the SqueezeNet model for fire detection, resulting in a significant reduction in both model size (from 238 MB in AlexNet to 3 MB in SqueezeNet) and GFlops/image (from 2 in Alexnet to 0.72 in SqueezeNet). This adaptation enhances its practicality and effectiveness for closed-circuit television surveillance networks. The SqueezeNet model was fine-tuned, increasing classification accuracy from 89.8% with the ILSVRC dataset to 94.50% over ten epochs. After a series of fire modules, the convolution layer is modified by reducing the number of classes to two (fire and normal), and the output is passed to the average pooling layer, feeding directly into the Softmax classifier.

In 2021, researchers proposed the ATT Squeeze U-Net architecture for forest fire detection and recognition (Zhang et al., 2021). This architecture incorporates the SqueezeNet and the U-Net architecture, enhancing local feature expression and improving the performance of fire detection. In SqueezeNet, the authors modified the Fire module to reduce parameters and enhance learning capability. It retains the *1×1* kernel in the "squeeze" layer but replaces the original *3x3* filters in the "expand" layer with a 3×3 depthwise separable convolution and a channel shuffle operation. The channel shuffle operation improves information exchange and feature description by dividing and reorganising channel groups. The standard attention U-Net is modified by integrating the SqueezeNet as an encoder, which involves altering the channel numbers in the first convolution layer of SqueezeNet from 96 to 64 and removing the average pooling layer and Softmax classifier. To enhance extraction accuracy, a 3x3 convolution layer is added between the modified SqueezeNet and the decoder. A corresponding DeFire module and an attention gate connection are introduced. The DeFire module, used in the decoding path, includes an expand layer and a squeeze layer for efficient feature map expansion. Finally, the output of the last DeFire module passes through convolutional layers, culminating in a final *1×1* convolution that maps features to desired classes for pixel prediction, which is passed to a final classification architecture.

2.2.1.6. MobileNet

MobileNet is one efficient convolutional neural network architecture designed for mobile vision applications (A. G. Howard et al., 2017). It is specifically optimised for resource-constrained devices with limited computation resources and achieves its efficiency by using, along with other techniques, a depthwise separable convolution, which breaks down a

standard convolution into two separate operations: a depthwise convolution and a pointwise convolution.

In a standard convolution, filters are applied across all input channels simultaneously, creating a computational bottleneck. However, in a depthwise convolution, a single filter is applied per input channel, which then processes the spatial information. In this case, it is followed by a pointwise convolution that uses a *1x1* convolution to combine the outputs of the depthwise convolution across channels, making this separation of spatial and depth reduce the computational cost and the model size.

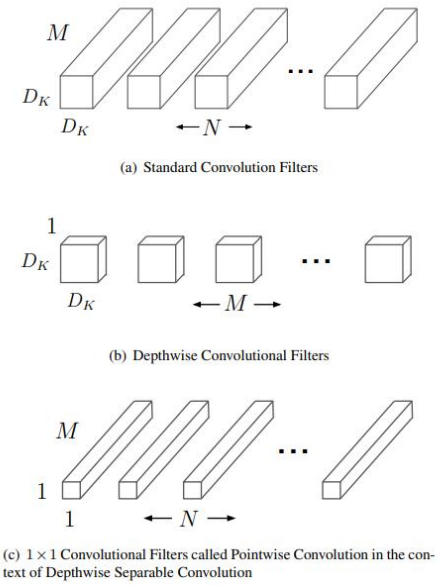


Figure 7: The standard convolutional filters in (a) are replaced by two layers: depthwise convolution in (b) and pointwise convolution in (c) to build a depthwise separable filter (A. G. Howard et al., 2017)

To further adapt to various computational and resource constraints, the authors introduced two global hyperparameters known as:

- The width multiplier (α) which allows the model to thin itself uniformly at each layer.
- The resolution multiplier (ρ) that reduces the input resolution of the image.

These hyperparameters balance latency and accuracy, allowing a dynamic scale for different devices and applications. As shown in Figure 8, each of the previous layers is followed by batch normalisation and a RELU nonlinearity, apart from the first and final fully connected layers.

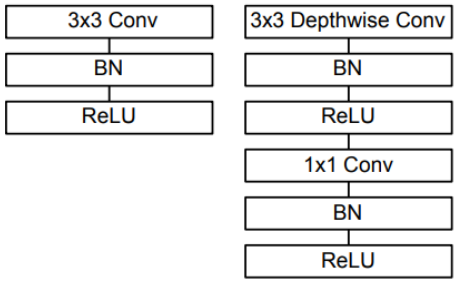


Figure 8: Left: Standard convolutional layer with batch normalisation and RELU. Right: depthwise Separable convolutions with depthwise and Pointwise layers followed by batch normalisation and RELU (A. G. Howard et al., 2017)

Furthermore, about 95% of the network's computations occur in dense *1x1* convolutions, which can be efficiently executed using optimised matrix multiply functions, which work like highly efficient calculators.

MobileNets discussed in this paper, utilise less regularisation and data augmentation compared to larger models due to their reduced tendency to overfit and are often trained using RMSprop optimiser with asynchronous gradient descent. This architecture focuses on reducing computational cost and model size while maintaining accuracy.

In fire detection systems, there is a growing need for efficient and lightweight models that can be deployed on low-computationally-intensive and mobile hardware. (Muhammad, Khan, et al., 2019) proposed a methodology considering uncertain environments like fog, smoke, and snow. Their research experiment with various CNNs and parameter settings, eventually finding MobileNetV2 more effective than other models like AlexNet, GoogleNet, and SqueezeNet. They chose to use a pipeline consisting of a 17-block followed by a *1x1* convolution MobileNet architecture and then a classification module comprising global pooling, followed by a *1x1* convolutional layer and the Softmax activation function. Finally, slight adjustments were implemented to customise the MobileNetV2 framework for fire detection by including an expansion layer.

Another research proposes a system with transfer learning and employs a window-based analysis strategy to enhance fire detection. The authors utilise Fourier analysis to calculate the frequency response of kernels in convolutional and dense layers. This enables them to identify and eliminate filters with low responses and compare convolutional kernels in the frequency domain using cosine similarity measures. The approach uses MobileNet-V2 as the base model, with its dense layers replaced by a new dense layer consisting of only two neurons with a Softmax activation function to perform classification tasks, which are subsequently converted into probabilities. Lastly, a window-based analysis is used to detect wildfire in overlapping windows, allowing for the detection of smoke even if it exists near the edge of a frame. This method divides the frame into blocks and detects smoke block-by-block, simplifying the

detection process by marking only the blocks containing fire instead of using multiple bounding boxes to mark fire and smoke regions within a frame (Pan et al., 2020).

In (Zheng et al., 2023), a combination of MobileNetV3-large and YOLOv4 is proposed for real-time fire detection on small embedded devices. The authors replaced the backbone network of YOLOv4 with MobileNetV3, which reduced computational complexity while extracting valid information. Afterwards, a Path Aggregation Network (PANet) structure is utilised in four feature layers to enhance the detection accuracy for small-scale flames and smoke by fusing multi-scale features. ViT and Spatial Pyramid Pooling (SPP) modules are incorporated into MobileNetV3-large and PANet. The SPP modules are introduced after medium- and large-scale feature layers to prevent the loss of small objects that might occur during feature propagation. They consist of three max-pooling layers and one connection layer. The ViT is known for its multi-head attention mechanism, which is adept at processing image features by simultaneously focusing on different parts of the image, thereby capturing a richer set of features than traditional convolutional approaches. Then, the path fusion concept from a BiFPN includes cross-layer connections on PANet and merges adjacent layers in series, providing bidirectional cross-scale connections and weighted feature fusion to enhance feature extraction capabilities. Finally, a module is incorporated to focus on useful information and improve the network's overall recognition performance. It’s positioned as an improved version of the SE module and is used to train the weights on the channel dimensions of the head network's layers.

The research "Forest Fire Detection Based on Light-Weight Deep Neural Networks" involves experimenting with various MobileNet and ShuffleNet configurations for fire detection tasks. These models are first trained on the ILSVRC dataset, and then their architecture is modified by removing the last fully connected layer and replacing it with a new output layer tailored for binary classification (fire or no-fire). The models remain frozen during the initial training phase except for the new output layer. Subsequently, selective layers are unfrozen, the final depthwise convolutional layer in ShuffleNetV2 and the last two building blocks in MobileNet, to avoid overfitting and enhance generalisation. A "superpixel segmentation-centered" method allows for precise fire localisation within an image by dividing the image into small, perceptually meaningful clusters of pixels and analysing each superpixel to determine whether it contains fire. Furthermore, the research utilises a method that incorporates a Bayesian neural network model and real-time weather data from a weather station to continuously update information to better understand damage assessment during a forest fire (Palanisamy & Easwaran, 2023).

Finally, aligned with other referred studies, (Tsalera et al., 2023) use a methodology with transfer learning, using SqueezeNet and MobileNetV2, which are pre-trained on the ILSVRC dataset. These networks are then fine-tuned using the Forest-Fire and Fire-Flame datasets, where specific training parameters such as learning rates, epoch numbers, and batch sizes are optimised for accurate fire detection. Finally, it highlights the incorporation of Gaussian and

Salt & Pepper noise in the datasets, along with a detailed account of cross-dataset evaluations demonstrating the adaptability and accuracy of these models in diverse environmental scenarios.

2.2.1.7. EfficientNet

EfficientNet is a convolutional neural network architecture proposed by Google that has gained significant attention in DL since it was developed to achieve high accuracy and efficiency (Tan & Le, 2020). Until this time, previous approaches to scaling CNN generally focused on increasing the depth (number of layers), width (number of channels), or image resolution. The intuition was that deeper CNN can capture richer and more complex features and generalise well on new tasks. However, deeper networks are also more difficult to train due to the vanishing gradient problem.

Despite the widespread use of various techniques, such as skip connections and batch normalisation, to address training issues, it is evident that the accuracy gain of very deep networks diminishes. For instance, ResNet-1000 exhibits similar accuracy to ResNet-101 despite having significantly more layers. Additionally, extremely wide but shallow networks face challenges in capturing higher-level features.

The empirical results demonstrated by the authors illustrate that the accuracy levels rapidly saturate when networks have higher coefficients.

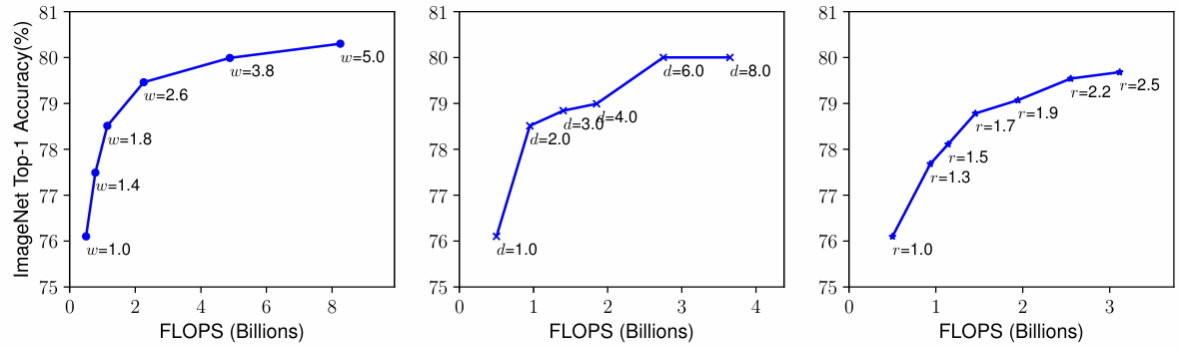


Figure 9: Scaling Up a Baseline Model with Different Network Width (w), Depth (d), and Resolution (r) Coefficients (Tan & Le, 2020)

Based on the previous arguments, the authors made two key observations:

Observation 1 – Scaling up any network width, depth, or resolution dimension improves accuracy, but the accuracy gain diminishes for bigger models.

Observation 2 – To pursue better accuracy and efficiency, it is critical to balance all network width, depth, and resolution dimensions during CNN scaling.

Those two observations were the starting point for introducing a systematic method of scaling all three dimensions - depth, width, and resolution - in a balanced manner. The core

innovation lies in its compound scaling method, which uses a compound coefficient to uniformly scale these three dimensions using a set of constants α^{φ}, β^{φ}, and $\gamma^{\varphi}$for depth, width, and resolution, respectively. This method ensures that when the network is scaled up to utilize more computational resources, these dimensions grow in a balanced way such that $\alpha * \beta^2 * \gamma^2 \approx 2 \; where \; \alpha \geq 1, \; \beta \geq 1, \; \gamma \geq 1.$

EfficientNet's architecture incorporates a hierarchical structure with multiple building blocks, including mobile inverted bottleneck convolution layers and SE modules. These building blocks help EfficientNet to capture and represent features at various levels of abstraction, enabling better generalisation and accuracy. This means that the network is scaled balanced, ensuring that losses in another do not offset the performance gains from scaling one dimension. The scaling is done in two steps:

- First, a small grid search is conducted to determine the best values for α, β, and γ, based on the formula to maximise the model accuracy and in the compound scaling method;
- Second, these values are fixed and used to scale up the baseline network using different values of φ only with the compound scaling method.

EfficientNet-B7 achieved state-of-the-art 84.3% top-1 accuracy on a well-known dataset in this research, the ILSVRC, while being significantly smaller and faster on inference than the best existing CNN. This level of performance and low computational cost make it particularly appealing for a wide range of applications, highlighting its adaptability as a key advantage.

Similarly to other referenced studies, EfficientNet has also been utilised in fire detection systems. (Xu et al., 2021) "A Forest Fire Detection System Based on Ensemble Learning" proposes an ensemble learning that incorporates the models Yolov5 (first learner), EfficientDet (second learner), and EfficientNet (third learner). Yolov5 and EfficientDet are used for object detection to identify fire locations, while EfficientNet is used as a classifier and evaluates the global information of the image to reduce false positives. The final detection result is determined based on the decisions of the three learners that are proven to perform better than individual ones.

In 2022, (Ghali et al., 2022) proposed another ensemble learning approach, using EfficientNet-B5 and DenseNet-201 models for classification tasks, along with the EfficientSeg, TransUNet and TransFire models for segmentation tasks. The authors made the choice of EfficientNet-B5 due to its ability to effectively reduce parameters and GFLOPS through an innovative scaling method that employs the compound coefficient. For the classification architecture, RGB aerial images are fed into these models to extract feature maps, which are then concatenated. This concatenated map undergoes average pooling, and a dropout layer is included to prevent overfitting before a Sigmoid function classifies the images into Fire or Non-Fire categories.

Lastly, (Z. A. Khan et al., 2022) elaborated an approach for fire detection using EfficientNet-B3 combined with a densely connected autoencoder, which the authors named Stacked Encoded-

EfficientNet. EfficientNet-B3 is the backbone architecture, consisting of 24 convolutional layers: 2 stem layers, one head layer, and 21 intermediate layers. The stem layers process the input image, while the intermediate layers extract features to produce a final output with dimensions of 1536 by the head layer. This output then passes to an autoencoder with three encoding layers that gradually reduce dimensions. The weights are randomly initialised to address vanishing gradient problems and promote faster convergence speed. All these interconnected layers enable each to receive input from all preceding ones for optimising feature representation. The model concludes with a Softmax classifier whose parameters have been optimised through experimental trials by the authors, who also tested several lightweight CNN models to select the optimal one for fire detection based on accuracy and false alarm rates.

2.2.2. Transformers Networks

There is another model category in ANN known as Transformers. These have demonstrated great effectiveness in tasks related to natural language processing and have established themselves as the standard architecture in this field. Built on a self-attention mechanism, these models have transformed how sequential data like text is processed since introduced in the paper "Attention is All You Need". This architecture allows models to capture long-range dependencies and enables efficient parallelisation, proving to be more effective overall than recurrent neural networks and CNN.

The Transformer architecture consists of an encoder-decoder structure, with both components consisting of layers of self-attention and feed-forward neural networks. Its key innovation is the self-attention mechanism, which enables the model to determine the significance of various elements within the sequence regardless of their distance from each other.

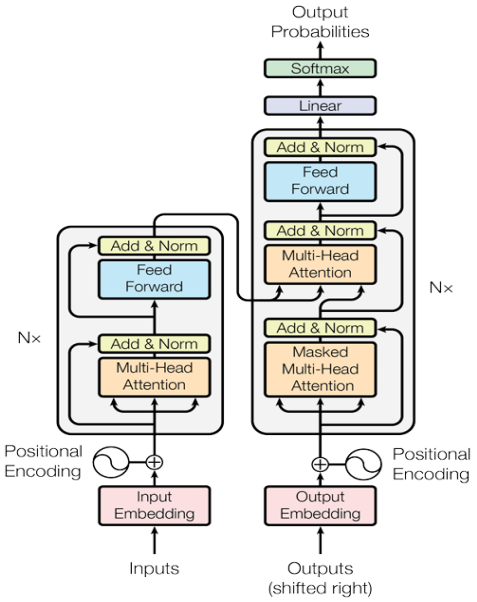


Figure 10: The Transformer - model architecture (Vaswani et al., 2023)

Figure 10 shows an encoder-decoder, where the encoder and decoder comprise a stack of N = 6 identical layers. In the encoder, each layer consists of a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. The decoder also includes an extra sub-layer for multi-head attention over the encoder's output and a masking mechanism to maintain the auto-regressive property during sequence generation. These sub-layers are linked using residual connections followed by layer normalisation.

The core of the Transformer's design is based on scaled dot-product attention, which calculates attention scores by multiplying queries and keys, adjusting them in scale, and then applying a softmax function. This concept is expanded to multi-head attention, where several attention heads work simultaneously, enabling the model to concentrate on various sections of the input sequence at once. Positional encodings maintain the sequence order without recurrence and convolution using known functions with different frequencies allowing the model to learn positional information effectively.

Inspired by the success of Transformers in NLP, researchers began exploring their application in CV tasks, allowing the introduction of the ViT, which has competitive performance on image classification tasks.

2.2.2.1. Vision Transformer

The Vision Transformer pioneered the adaptation of Transformers for image classification tasks. Unlike traditional Transformers that process sequential data, the ViT model handles two-dimensional image data by splitting images into fixed-size patches and treating these patches as tokens. Each patch is linearly embedded through patch embedding, and positional embeddings are added to retain spatial information. The embedded patches are then fed into an encoder, which uses multi-headed self-attention and MLP to capture long-range dependencies across the image, as suggested in Figure 11.

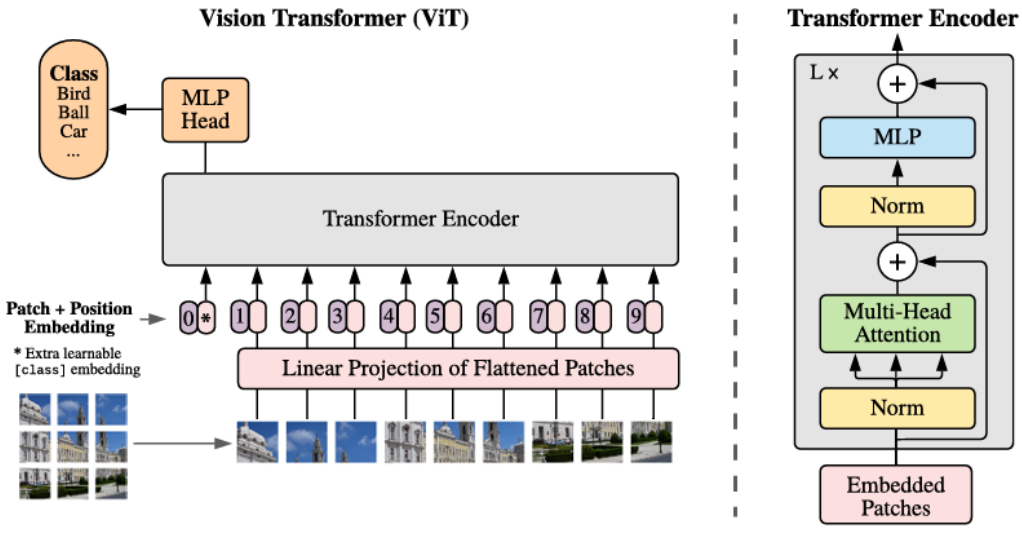


Figure 11: Model overview (Dosovitskiy et al., 2021)

The number of layers in the encoder, denoted as *Lx*, varies based on the model variant. For instance, ViT-base includes 12 layers, ViT-large comprises 24 layers, and ViT-Huge consists of 32 layers. Additionally, "+" denotes the presence of a residual connection followed by layer normalisation.

The authors of ViT have demonstrated that with sufficient pre-training on large datasets, it can outperform state-of-the-art CNN on various benchmarks. This allows ViT to leverage the scalability and efficiency of Transformers while overcoming the lack of inherent inductive biases, such as locality and translational invariance. Additionally, ViT achieves high performance with fewer computational resources for pre-training, highlighting its efficiency for large-scale image recognition tasks. For example, ViT achieves 88.55% accuracy on ImageNet, 90.72% on ImageNet-ReaL, 94.55% on CIFAR-100, and 77.63% on the VTAB suite of tasks.

The paper "Transformer-Based Fire Detection in Videos" by (Mardani et al., 2023) introduces a methodology with video surveillance systems and a transformer-based network. The encoder processes input video frames to generate attention scores that highlight regions relevant for fire detection, while the decoder refines these attention scores to produce final outputs: segmentation masks for fire localisation and binary predictions for full-frame classification. The model's backbone is a pre-trained ResNet-101 network that extracts initial features from the frames. A critical enhancement involves creating a detailed segmentation mask dataset, using a method to divide frames into superpixels labelled as fire or non-fire based on their enclosure within ground truth bounding boxes. The model training employs a combination of dice and focal loss for fire localisation, addressing accuracy and class imbalance, respectively, and cross-entropy loss for classification. Rigorous evaluation of the enhanced dataset demonstrates the model's exceptional performance, achieving 97% accuracy, 20.4 frames per second processing time, a 0.02 false positive rate for fire localisation, and a 97% f-score. These results indicate the model's superior capability in detecting and localising fire with high precision and speed, outperforming existing state-of-the-art methods.

2.2.2.2. Shifted Windows Transformer

Swin Transformer introduces a hierarchical architecture with shifted windows to improve efficiency by restricting self-attention to non-overlapping local windows while maintaining cross-window connections. It addresses the challenges of adapting Transformers for vision tasks by partitioning feature maps into non-overlapping windows and utilises multi-head self-attention layers to aggregate information within each window.

The authors proposed an architecture based on hierarchical feature maps, starting from small patches and gradually merging into deeper layers. Unlike traditional self-attention, which computes relationships between all token pairs leading to $O(N^2)$ quadratic complexity, the Swin Transformer addresses this by computing self-attention within local, reducing complexity

to O(N) linear terms relative to image size. Then, to maintain connections between windows, the window partitioning shifts between consecutive layers. For example, if the window size is 4×4, the partitioning in the next layer is shifted by 2 pixels, introducing cross-window connections that enhance modelling power without significantly increasing computational cost and deliver notable speed improvements.

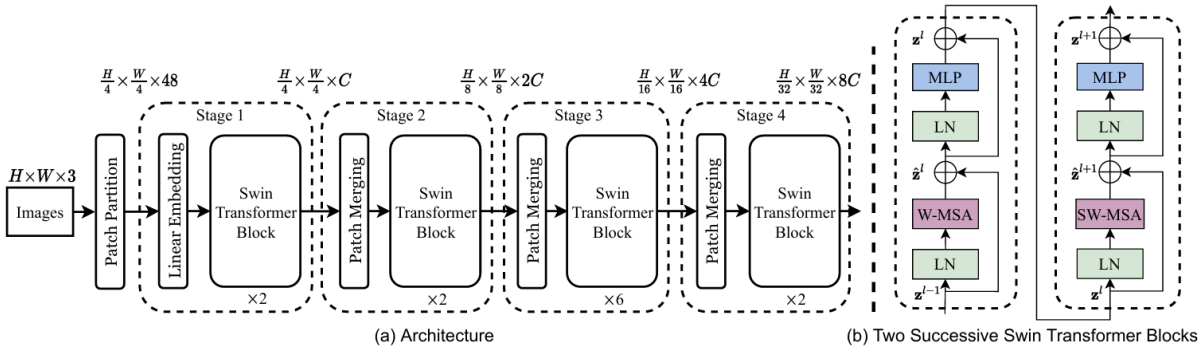


Figure 12: (a) The architecture of a Swin Transformer; (b) two successive Swin Transformer Blocks (Liu et al., 2021)

Using a patch partition module, the Swin Transformer process divides an input RGB image into non-overlapping patches. Each patch is considered a token and contains raw pixel values, which are then projected into a higher-dimensional space by a linear embedding layer. There are four stages in total: the first stage involves linear embeddings and a Swin Transformer Block, while the remaining three stages each incorporate patch merging to reduce the number of tokens and utilise a Swin Transformer Block. These blocks apply windows-based attention and Shifted windows-based attention, along with a two-layer MLP, LayerNorms, and residual connections.

Swin Transformer provides different versions, including Tiny, Small, Base, and Large. The variations are based on the number of channels C in the initial layer and the quantity of Swin Transformer blocks in stage three. These variations balance model size and computational complexity to meet specific application needs.

2.2.2.3. Multi-Axis Vision Transformer

MaxViT introduces a multi-axis self-attention mechanism that efficiently handles local and global spatial interactions within images. This approach combines the strengths of CNN's and Transformers, leveraging the strong inductive biases of convolutions and the global interaction capabilities of self-attention. The model utilises local attention to partition input feature maps into non-overlapping blocks for self-attention and dilated global attention over sparse, non-overlapping windows for long-range dependencies. With its hierarchical design incorporating attention mechanisms and convolutional layers across multiple stages, MAX-ViT progressively refines feature representations at different levels while capturing fine-grained details and high-level semantics.

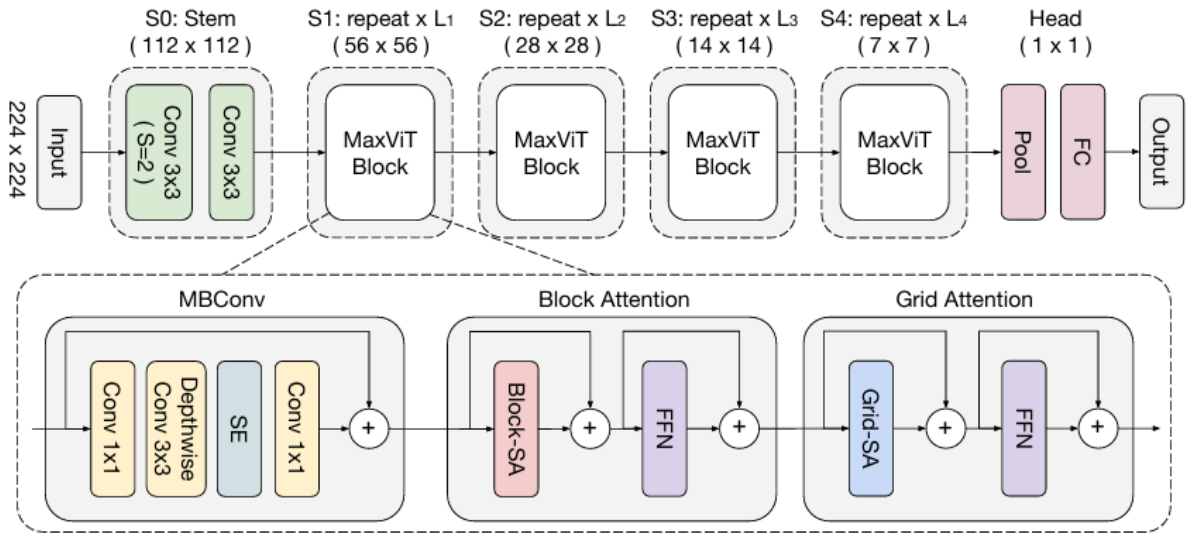


Figure 13: MaxViT architecture (Tu et al., 2022)

The stages include the Stem Stage (S0), which comprises initial convolutional layers that reduce the input image size and increase the channel dimension. The main stages (S1 to S4) consist of recurring MaxViT blocks, each operating at varied resolutions and channel dimensions to enable a multi-scale feature representation. Finally, there is the classification head, where, after the final stage, features are aggregated using a global average pooling layer, followed by a fully connected layer for classification.

Each MaxViT block consists of a sequence of layers, from self-attention layers operating in multiple axes to convolutional layers with SE modules. Inside each block, an MBConv layer utilizes depthwise separable convolutions for expansion and contraction, incorporating SE modules to enhance feature recalibration. Block attention achieves self-attention within localised windows for effective local feature aggregation. Meanwhile, grid attention conducts self-attention across a global grid to ensure efficient interactions among global features.

MaxViT, with its multiple model options (T, S, B, L, XL), demonstrates a remarkable balance between computational efficiency and performance. It exhibits leading performance in image classification, attaining an impressive 86.5% top-1 accuracy on ImageNet-1K without extra data and 88.7% with ImageNet-21K pre-training. This performance surpasses other models in terms of accuracy versus GFLOPs and accuracy versus parameter trade-offs, providing a solid foundation for its reliability and effectiveness.

3. METHODOLOGY

The methodology section is crucial as it outlines the process and techniques used to collect, organise, and analyse data. This section serves as a roadmap to describe the experimental setup used to overcome the research objectives:

1. To develop a CV framework for close-range rural fire detection for low computational cost devices.
2. To assess if infrared images can increase rural fire detection model performance.
3. To analyse how CNNs can identify fire in rural environments.
4. To compare the performance of ViT and CNN in rural fire classification.

Ensuring the methodology is based on current and relevant academic sources is essential. That is why Chapter 2 analyses the existing literature to extract valuable insights.

The pipeline's development occurred on a cloud computing interface, encompassing training and inference processes. To address the limitations of processing power and memory in low-resource devices, this research utilises two advanced object detection models – YOLO (Redmon et al., 2016) and EfficientDet (Tan et al., 2020) - known for their combination of speed, accuracy, and computational efficiency. Additionally, it involves incorporating classification models that complement these detection models to ensure high accuracy while remaining feasible on hardware with limited resources.

Choosing lightweight models does not necessarily guarantee good results in less training time. When training these models, the aim is to achieve good performance that enables further research of the proposed methodology.

Implementing the pipeline on mobile devices would require additional investigation and the design of robotic systems to integrate the advancements made in this research and evaluate them in a real-world setting. This aspect falls beyond the scope since it depends on other fields of study, such as robotics.

The methodology is organised to address several sections: Data Collection, Data Exploration, Data Preprocessing, Modelling, Evaluation, Hardware and Software configuration and Experimental Design. Each phase is pivotal in constructing a reliable and efficient fire detection system. The data collection phase focuses on acquiring relevant and diverse data, as outlined in 3.1. To comprehend their attributes, the datasets are explored in the data exploration stage (3.2). The data preprocessing stage (3.3) involves cleaning and transforming the data for input into the models. The modelling section (3.4) elaborates on selecting, modifying, and implementing detection and classification models, while the evaluation (3.5) outlines the metrics and procedures used to evaluate model effectiveness. Hardware and Software

configuration, involving setting up the necessary infrastructure and software tools, is discussed in 3.6, and Experimental Design is outlined in section 3.7.

3.1. DATA COLLECTION

The selection of datasets for the thesis was guided by the requirement to obtain a diverse range of images labelled as Fire and No Fire. This diversity was achieved by including five datasets: Corsican Fire Database, BoWFireDataset, Forest_fire, D-Fire, and Forest FireImages.

The Corsican Fire Database contains only images of fire and is divided into 1135 RGB images, 640 NIR images, and 1135 GT images. It was developed at the laboratory “Sciences Pour l’Environnement” - University of Corsica. The database includes wildfire pictures and image sequences obtained in visible and near-infrared areas under various shooting conditions, types of burning vegetation, climatic conditions, brightness levels, and distances to the fire (Rossi, L, 2024).

The BoWFireDataset consists of 226 images, 119 of which feature fire and 107 that do not (Chino et al., 2015).

The Forest_fire dataset is created for a binary classification task to detect the presence or absence of fires in forest landscapes. It contains 1900 images, with 950 allocated to each class (A. Khan & Hassan, 2020).

The D-Fire dataset consists of 21524 images along with their respective annotations. It has been categorised into four groups: Fire, Smoke, Fire and Smoke, and None (De Venâncio et al., 2022).

The Kaggle dataset "Forest Fire Images" includes 2525 images of fires and an equal number of images without fires (Prasad, M, 2022).

While dealing with these datasets, challenges arose concerning sequential images taken from the same camera at different times and images captured using a camera spectrum for long-distance and UAV applications. Addressing this issue required manual filtering.

The combined dataset comprised 2711 images with fires and 2204 without fires for 1255 GT images. Figure 14 shows an overview of the dataset.

Figure 14: Image example for the combined dataset. Top left “No Fire”, Top right “Fire”, Bottom left “NIR Fire”, Bottom right “GT Fire”

Most of the dataset was not labelled for object detection tasks, so Roboflow (Dwyer, B. et al., 2024) was employed to label it. 4835 annotations were created, with an average of 1.8 per image. Once labelled in the platform, the dataset was exported with a simple train/valid/test split allocated 80% for training, 10% for validation, and 10% for testing. This export could be done in various formats including Pascal, Tfrecord or Yolo. The decision was made to use the Yolo format.

After importing the datasets into the notebook, it was necessary to merge the portion of the dataset sent to Roboflow for labelling with the section lacking fire-bounding boxes labelled "No fire." The dataset containing non-fire images was split into training, validation, and testing sets using the same approach as the dataset with fire labels. Subsequently, a text file was generated for each image; however, in this scenario, the text files are empty since the images do not contain any bounding boxes to be annotated. Lastly, the merged dataset was ready for further analysis and model training.

3.2. DATA EXPLORATION

The data exploration process involved analysing the characteristics and structure of the collected datasets, including examining the distribution of classes, exploring the image formats and sizes, and checking anomalies in the data.

The first analysis concluded that the dataset was balanced regarding classes since it is a custom dataset, but the annotations for detection tasks were missing. It was visible that the images had different resolutions and that an approach would be necessary to consider this. Also, a careful validation regarding the environment represented in the images was made since the objective was to use outdoor images with different lighting and intensity conditions.

Regarding the diversity in fire instances, the size, intensity, and backgrounds were examined to create images to help the models learn to generalise well across all real-world scenarios. Also, after the labelling process, a distribution of the size of the bounding boxes within the images was analysed.

In the context of data quality, specific flawed images were detected and deleted, such as gif images and those with bounding boxes exceeding the possible image size. Even after labelling the dataset, instances occurred where bounding boxes extended beyond the edges of the images, potentially stemming from errors in the labelling platform or movements of boxes after creation. The noise and smoke in an image did not reach a level significant enough to warrant removal. However, smoke in these situations can sometimes render specific images unusable.

Most of these evaluations took place while labelling the data, serving as a manual review. This approach facilitated a deeper comprehension of the dataset and helped identify possible obstacles and trends that could impact algorithm training and performance, as well as determine when preprocessing should occur.

3.3. DATA PREPROCESSING

In preprocessing, two approaches were tested. The Roboflow platform provides a variety of features for data preprocessing, including resizing images to a uniform size, standardising pixel values, and enhancing the dataset with methods like 90º rotation and brightness adjustments. This could be seen as the primary approach that resulted in 11275 images categorised under the "Fire" label. However, this posed challenges when attempting to balance the class distribution of images that were not processed through Roboflow and belonged to the "No Fire" label, resulting in an unbalanced dataset.

The second approach involved using Roboflow only to annotate the dataset with "Fire" labels and then creating training, validation, and test sets to import the images into the notebook. The "No Fire" labels dataset underwent various transformations in the notebook before being combined, including creating Yolo format files and splitting them into train, validation, and test subsets. This approach aimed to address the class imbalance issue and ensure that both "Fire" and "No Fire" images were adequately represented in the dataset.

Upon considering the investigation conducted in chapter 3.2, initial preprocessing involved resizing the dataset in the train, validation, and test splits to dimensions of 250x250 and 500x500, a close value to the mean image size of the dataset, but since the thesis concentrates

on working with detection, the resizing also had to adjust the coordinates of the bounding boxes. Consequently, any transformation applied to the images had to be reflected in the corresponding files containing information about the localisation of the bounding boxes. Over 200 images were discarded during this preprocessing phase due to incorrect labelling identification. As mentioned earlier, these labels contained values beyond their respective image boundaries. This seemed unusual since it was impossible to position them outside the images using the application where they were created. However, it is possible that this occurred because of moving the boxes after placing them within an image. An additional modification to the image was carried out through augmentation exclusively in the training dataset, where the brightness was adjusted by 0.4 with a probability set at 1. As a result, the augmentations applied led to doubling the initial dataset size.

Different techniques may be employed during preprocessing, such as image rotations and additional colour modifications. However, considering the aim of training the model to comprehend real-world scenarios, it would not be reasonable to rotate an image so that fire is positioned at the top. Conventional images depict the ground at the bottom and the sky at the top; therefore, it would not make sense for ground elements to appear in place of sky elements or vice versa. Furthermore, it would also be unconventional for the model to recognise fire using a colour other than its hue.

At this point, the dataset was prepared for utilisation in the Yolo model. Nonetheless, further adjustments were necessary to make it suitable for implementation with the EfficientDet model. In contrast to Yolo, the EfficientDet model needs Tfrecords as its input format. Therefore, it incorporated pre-processing steps that involved converting the data from Yolo's txt format to a Tfrecord format compatible with EfficientDet.

In conclusion, the training set contained 7102 images, the validation set had 434 images, and the test set had 434 images.

3.4. MODELLING

The modelling phase of this research contributed to creating a successful fire detection system for devices with limited resources. Let's begin by examining Figure 15 to better understand its underlying logic.

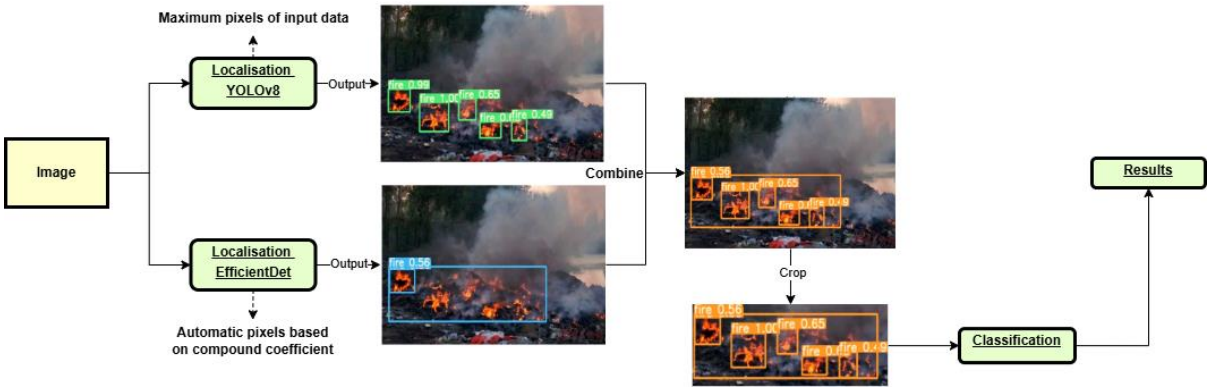


Figure 15: Modelling framework

The modelling is divided into 2 phases. The first phase involved two object detection algorithms: YOLO and EfficientDet. They worked simultaneously to detect bounding boxes in the images and then combined their results. This combination served as an additional validation, enabling the cropping of the bounding boxes for storage. The second phase happens after the training of the object detection models, and the cropped bounding boxes from their inference are used to train a classification model.

At this point, specific research goals are being evaluated, such as determining the impact of NIR images and image size on model performance and understanding whether vision transformers can outperform CNN in classification tasks.

3.4.1. Object detection

Previous studies have shown that object detection algorithms such as YOLO and EfficientDet have been well-documented for producing positive results (Xu et al., 2021).

For the research, YOLOv8 is selected for its speed and performance in real-time object detection. As Figure 16 suggests, it showcases significant improvements in all the model sizes (Nano, Small, Medium, Large, X-large) compared with other versions.

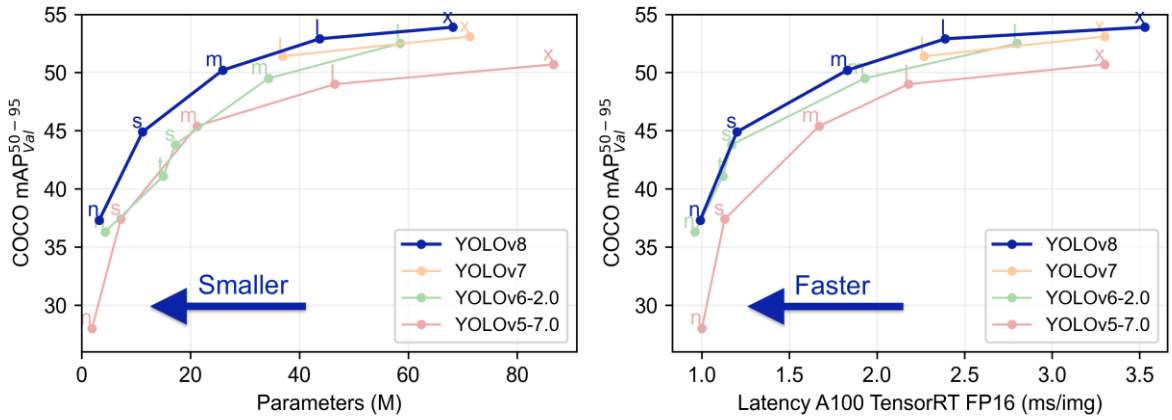


Figure 16: Yolov8 parameters/performance and latency/performance benchmark

It was released by Ultralytics in January 2023 (Jocher, G et al., 2023) and is known for its user-friendly CLI, documentation, and availability on the Github repository, making it the preferred choice.

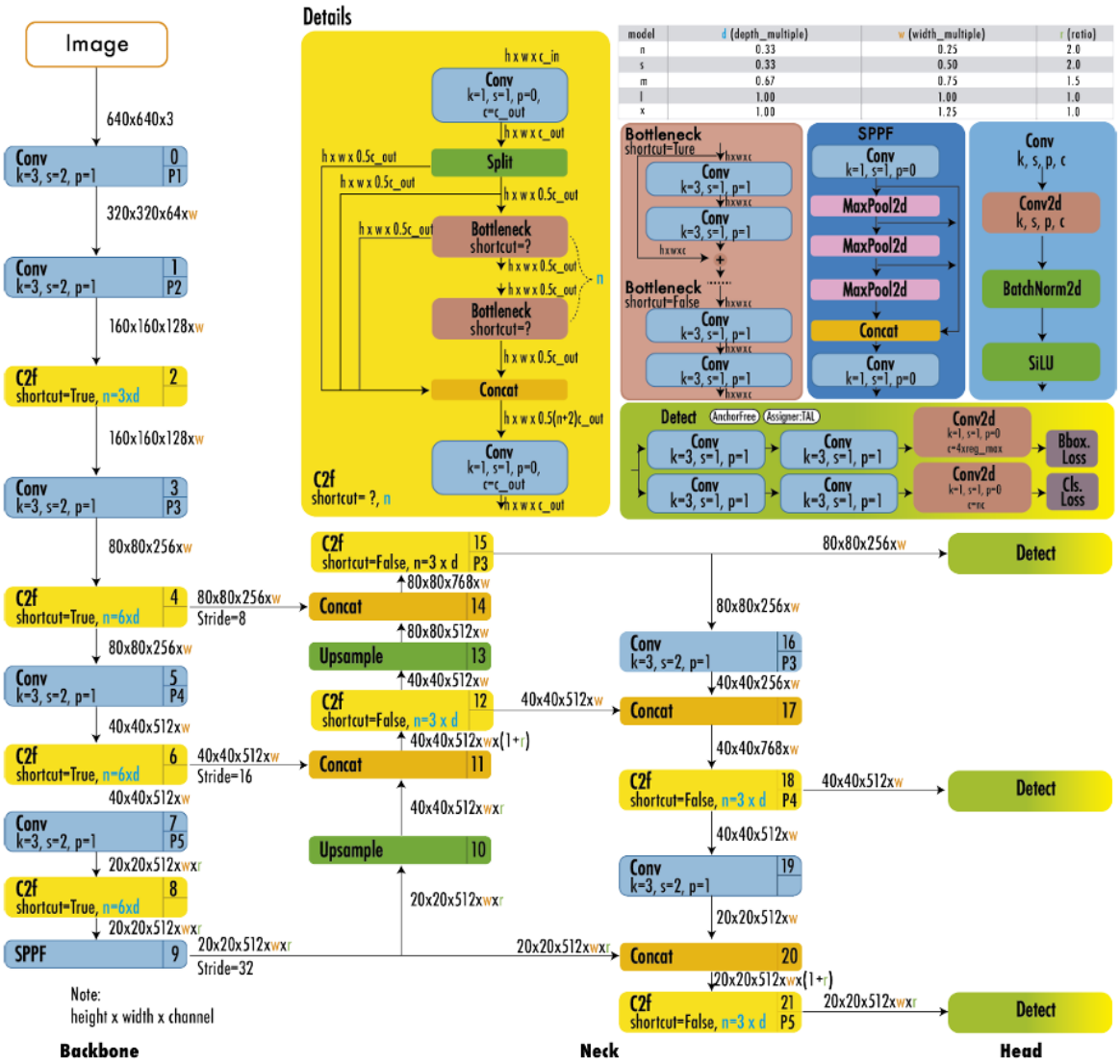


Figure 17: “YOLOv8 Architecture - The architecture uses a modified CSPDarknet53 backbone. The C2f module replaces the CSPLayer used in YOLOv5. A spatial pyramid pooling fast layer accelerates computation by pooling features into a fixed-size map. Each convolution has batch normalisation and SiLU activation. The head is decoupled to process objectness, classification, and regression tasks independently” (Terven et al., 2023)

Integrating the cross-stage partial bottleneck with two convolutions (C2f) modules in YOLOv8's architecture is crucial since this module combines high-level features, capturing abstract aspects of the input, with contextual information that provides a broader understanding of the scene. By merging these two types of information, the C2f module enhances the model's capability to discern and accurately locate objects even in complex environments.

Yolov8 is state-of-the-art in object detection, and it is an anchor-free model with a decoupled head. In the anchor-free approach, the model directly predicts the bounding boxes without relying on predefined anchor boxes, simplifying the training process and enhancing flexibility in detecting objects of various shapes and sizes. The decoupled head design refines this process by independently processing objectness, classification, and regression tasks. Each part of the head is specialised:

- The objectness branch determines the likelihood of a bounding box containing an object using the sigmoid activation function, which has output values between 0 and 1. Thus, it indicates the presence or absence of an object within a bounding box.
- The softmax function is employed for classification to ascertain the probabilities of the detected object belonging to the Fire class, providing a clear probabilistic distribution across Fire and background categories.
- Additionally, it employs the complete intersection over union (CIoU) and distribution-focal loss (DFL) functions for bounding box regression. CIoU loss enhances the accuracy of bounding box predictions by considering aspects like overlap, distance, and aspect ratio. DFL provides a distribution-based approach to handle the scale variance in object sizes, which is especially beneficial for small object detection. Combined with binary cross-entropy for classification loss, these loss functions substantially improve the model's performance across various object sizes and complexities.

EfficientDet's design achieved a balanced scaling of network dimensions, including depth, width and image resolution simultaneously. This is enabled through the compound scaling method and using EfficientNet as the backbone. The BiFPN fed by the backbone enhances feature fusion by facilitating effective information flow in both top-down and bottom-up directions to capture multi-scale information. Meanwhile, different feature fusion strategies bring different semantic information, resulting in different detection results. Furthermore, it improves efficiency through depthwise separable convolutions for feature fusion and introduces a weighted feature fusion mechanism to assign varying levels of importance to different input features. EfficientDet offers scalability in its architecture, with various model sizes (D0 to D7) and the ability to handle different input image resolutions while maintaining efficiency.

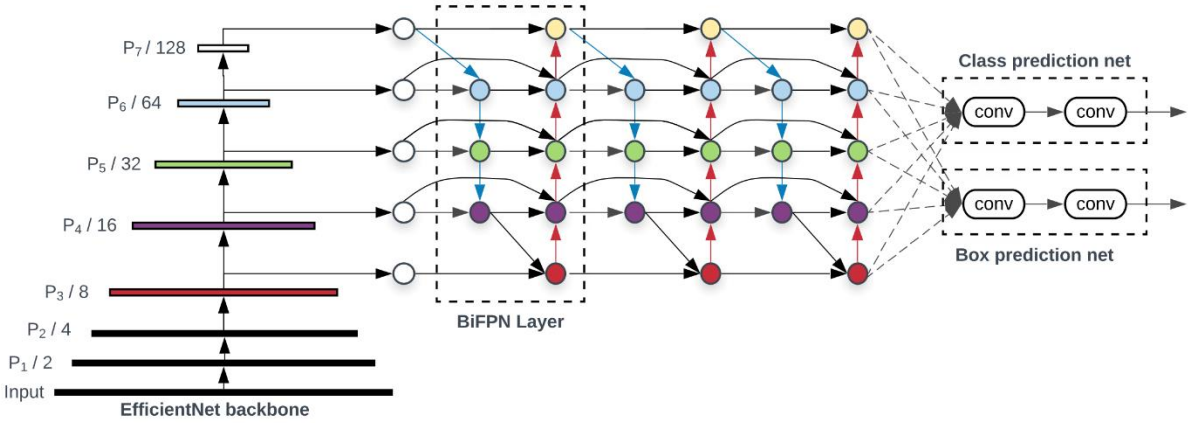


Figure 18: "EfficientDet architecture – It employs EfficientNet as the backbone network, BiFPN as the feature network, and share class/box prediction network. Both BiFPN layers and class/box net layers are repeated multiple times based on different resource constraints" (Tan et al., 2020)

Combining YOLOv8 and EfficientDet combines the strengths of both models for fire detection. YOLOv8's real-time processing and precise object localisation are enhanced by EfficientDet's multi-scale feature integration and computational efficiency. While YOLOv8 rapidly identifies and localises fire instances, particularly in large areas, EfficientDet provides robustness in diverse environmental conditions and perspectives as a more careful detector, ensuring comprehensive fire detection coverage.

The use of object detection has some limitations since it only learns the fire region, which is just a local pattern of the whole image and ignores other information like background. As a result, the object detector may treat fire-like objects as fires, making false alarms.

3.4.2. Classification

Therefore, the second phase is related to the classification model, which is trained based on a dataset of the previous detection model inference outputs. Due to the use of the crop combined outputs images, the classification model aims to differentiate between actual fire instances and fire-like objects, reducing false alarms and being specifically tailored to the type and quality of objects being detected, potentially leading to higher accuracy in practical scenarios. Considering the balance between accuracy and computational efficiency, the classification models are based on EfficientNet, MobileNet and lightweight ViT. The selection of these models allows to compare the CNN and the transformers in this task.

3.5. EVALUATION

The evaluation metrics used to assess the performance of the integrated classification and object detection system allow a comprehensive analysis of its effectiveness and exploration of its potential limitations. The examination evaluated the loss for each model aspect and their corresponding performance measurements, covering both the training and validation

datasets. The GFLOPS and parameters were also considered according to each model's documentation.

Loss Analysis

The evaluation considered the following losses: classification loss, bounding box loss and DFL for object detection.

Classification Loss: Indicates the model's ability to label objects correctly.

Bounding Box Loss: This loss evaluates the precision of the predicted boxes, reflecting how closely they align with the locations of the actual objects.

DFL: The DFL is designed to address the class imbalance by focusing on hard-to-classify examples and scale variance by emphasising smaller objects that are harder to detect (Tao et al., 2023).

For both the training and validation set, all loss metrics are supposed to decrease across epochs for all models. This indicates that the models are learning and improving in accuracy for their specific tasks. However, occasional spikes for the validation set can suggest potential overfitting.

Performance Metrics

Intersection over Unio: IoU is a metric used to quantify the intersection between a predicted bounding box and a GT bounding box, playing a pivotal role in assessing the precision of object localisation.

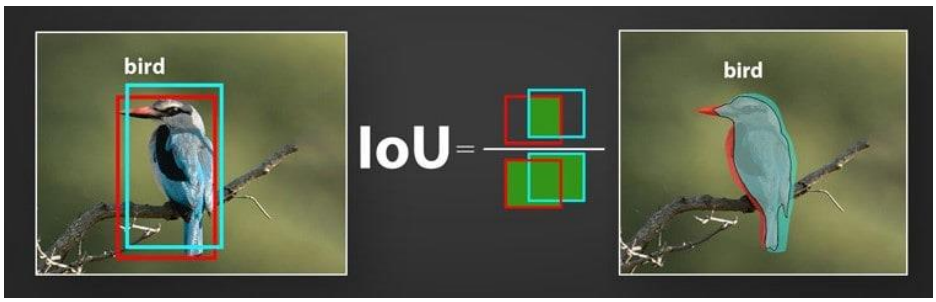


Figure 19: IoU - in red is the true label, and blue is the predicted (Kukil, 2022)

Precision(P) and Recall(R): These metrics directly indicate the accuracy of the classification model and are associated with the correctness of identified objects for the object detection model. Precision measures the true positives among all positive predictions, evaluating the model's capacity to avoid false positives. Recall calculates the true positives among all actual positives, assessing the model's ability to detect all instances of a class. They have the following formulas:

$$P = \frac{TP}{TP + FP} \qquad R = \frac{TP}{TP + FN}$$

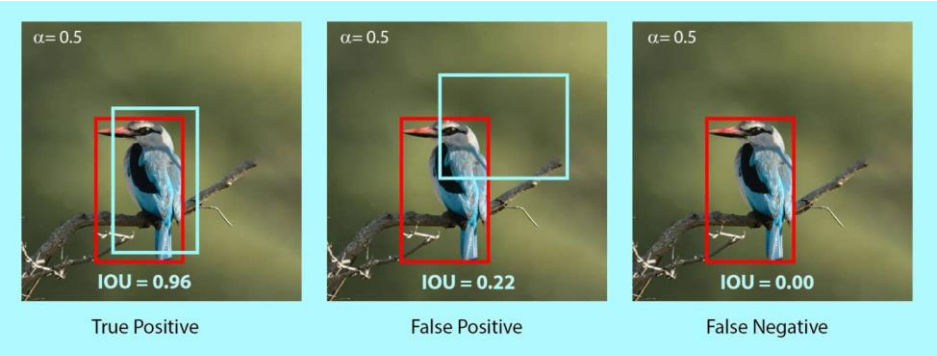


Figure 20: True Positive, False Positive and False Negative with an IoU threshold value of 0.5 (Kukil, 2022)

Average Precision (AP) computes the area under the precision-recall curve, providing a single value that encapsulates the model's precision and recall performance with the formula:

$$AP = \sum n\ (Rn - Rn - 1)\ Pn$$

where Pn and Rn are the precision and recall at the n^{th} threshold.

Mean Average Precision (mAP): The mAP measurements were utilised to assess the performance of the object detection model, encompassing IoU values from 0.5 up to 0.95. It expands on the AP concept by determining the average AP values for various object classes, making it valuable for evaluating the model's effectiveness in multi-class object detection situations, which is not the case. It has the following formula:

$$mAP = \frac{1}{N} \sum_{i=1}^{N} APi$$

F1 Score: The F1 Score is calculated from the mean of precision and recall, offering a well-rounded evaluation of a model's effectiveness by considering both false positives and false negatives has the formula:

$$F1 = 2 \cdot \frac{P \cdot R}{P + R}$$

The performance metrics are aligned with the research objectives as they directly measure necessary losses and help understand the classification and object detection models. This is extremely important for achieving the proposed research objectives.

3.6. Hardware and Software Configuration

The code utilised to investigate the research questions in this thesis was developed using Python and a variety of libraries such as:

Roboflow is a dataset management tool designed to streamline the process of building and deploying CV models with functionalities for organising and annotating datasets, versioning data, and preprocessing images for model training.

Ultralytics is behind YOLO, particularly known for its YOLOv5 and YOLOv8 implementations. Its framework offers efficient, state-of-the-art object detection algorithms widely used for real-time detection tasks.

Tensorflow, an open-source library developed by the Google Brain team, is used for exploration, preprocessing, and evaluation. It provides a comprehensive ecosystem of tools for researchers and developers to design, train, and validate DL models across various tasks, including image recognition and more (Abadi et al., 2016).

PyTorch is also an open-source library developed by Facebook's AI Research lab that offers a flexible platform for DL research and the development of classification models. It emphasises speed, agility, and ease of use, making it a favourite among researchers for experimental work and prototyping in areas such as vision (Paszke et al., 2016/2019).

Albumentations is a fast and flexible image augmentation library used in the preprocessing steps. It improves the performance of CV models by enlarging the dataset with altered versions of the input images. This library supports a wide range of augmentation techniques, which are crucial for training robust models and achieving generalisation (Buslaev et al., 2020).

Open-Source CV Library (OpenCV) is a library with functions that help in the preprocessing stage. It is now used for various applications such as facial recognition, motion tracking, and object detection. It provides powerful image and video analysis tools that effectively implement preprocessing and feature extraction tasks (Bradski, 2000).

Automl is an open-source library on GitHub that provides users access to the EfficientDet object detection model. This allows for an easier implementation of cutting-edge object detection in a more accessible and customisable manner (*Google/Automl*, 2020/2024).

Regarding the hardware, a Google Colab (Google Colaboratory, 2024) premium account was utilised for model training and validation because it has access to GPU resources (A100 GPU, V100 GPU, T4 GPU).

3.7. EXPERIMENTAL DESIGN

The experimental design involves training and evaluating the models on various configurations while comparing their performance metrics. Most of the models used in this thesis can be initialised either from scratch or with pre-trained weights.

Yolov8 and EfficientDet have the particularity of coming up with pre-trained checkpoints that initialise the weights and biases of the network, which accelerates the training process and improves convergence. EfficientDet relies on its backbone, EfficientNet, pre-trained on the ImageNet dataset. Pretraining assists the backbone network in EfficientDet with effectively extracting features from images, a crucial aspect of the object detection process. The entire EfficientDet model, including its components beyond the backbone, is pre-trained on the COCO dataset to enhance its performance.

Yolov8 uses a pre-trained model from the COCO dataset, which is then fine-tuned for specific use in detecting forest fires. The experimental setup involves using YOLOv8n to YOLOv8x and EfficientDet-D0 to EfficientDet-D7x with different batch sizes, always maintaining equality between the model's size. This seeks to evaluate the effectiveness of various setups of YOLOv8 and EfficientDet models in tasks related to object detection.

The object detector models infer from all the datasets to create a new cropped bounding box dataset. The bounding boxes will follow an interception approach, maintaining only the common detections in both models. This new dataset of cropped bounding boxes undergoes a preprocessing step where the images are resized and normalised to a standardised format before being used to train and validate a classifier to distinguish between images containing fire and those that do not have fire. This standardised format ensures consistency in the input data for the classifier, allowing for accurate and reliable classification results.

Regarding classification models, the ones that best fit to accomplish the research objectives are MobileNetV3 (A. Howard et al., 2019), EfficientNetV2 (Tan & Le, 2021), MaxVit (Tu et al., 2022), SwinTransformerV2 (Liu et al., 2022), VisionTransformer (Dosovitskiy et al., 2021), from Pytorch library. These models are tested with and without pre-trained weights and different batch sizes. The results from the experiments will provide valuable insights into the performance and effectiveness of different models in detecting and classifying forest fires while also comparing CNN and transformers.

4. RESULTS AND DISCUSSION

4.1. RESULTS

This chapter demonstrates the outcomes of implementing the methodology proposed in this study. The results presented here offer insights into the performance and effectiveness of the system in accomplishing its objectives. As previously mentioned, there is a clear distinction between the object detection and classification components. Thus, the results will be presented separately for each component: Chapter 4.1 will cover the results obtained from the object detection component. In contrast, Chapter 4.2 will focus on the results obtained from the classification component. In Chapter 4.3, a discussion and analysis of the overall performance of both components is conducted, with comparisons to evaluate their effectiveness and benchmark them against existing methods in this field.

4.1.1. Object detectors models

The initial results were obtained within the established framework using the resized 250x250 dataset. Additionally, a minor increase in brightness was applied, and the model underwent training for 50 epochs to produce these results in Table 1.

Table 1: Performance and characteristics of object detectors models

Model	Parameters	GFlops	Epochs	Input Size	Batch Size	mAP/AP	mAP/AP50
Yolov8n	3.2M	8.7	50	250x250	16	0.209	0.481
					32	0.202	0.479
					64	0.197	0.462
Yolov8x	68.2M	257.8	50	250x250	16	0.218	0.496
					32	0.213	0.504
					64	0.211	0.485
EfficientDet-d0	3.9M	2.54	50	250x250	16	0.220	0.533
					32	0.205	0.508
					64	0.207	0.508
EfficientDet-d4	20.7M	55.2	50	250x250	16	0.209	0.509
					32	0.21	0.511
EfficientDet-d7	51.9M	325	50	250x250	8	0.204	0.495

					16	0.211	0514

Beginning with Yolov8n, this represents the most compact model within the yolov8 family, featuring 3.2 million parameters and 8.7 GFlops. The aim was to assess the performance of models with reduced parameters and lower GFlops on the dataset. By testing various batch sizes in this model, it becomes evident that smaller batch sizes lead to improved overall performance of the model. Training for this particular model took approximately 40 minutes to complete from scratch, and it comes pre-trained with a COCO dataset consisting of over 330k images, which greatly contributes to its effectiveness in object detection tasks.

Yolov8x is the largest model in the Yolov8 family, containing 68.2 million parameters and operating at 257.8 GFlops. Performance evaluation across various batch sizes indicates that it yields better results with smaller batch sizes, similar to its smaller versions. The difference in performance based on batch size is not particularly significant in this instance. On average, training for this model takes approximately 1 hour and 20 minutes and comes pre-trained with the COCO dataset.

In summary, based on the comprehensive analysis presented in the table above, Yolov8x, with a batch size of 16, emerges as the most suitable and preferred model choice for this research framework.

EfficientDet models differ from Yolov8 models' approach to continuous improvement and research. They may not receive frequent updates, but they remain valuable for studying and utilising in research to develop adaptable models capable of handling diverse inputs. Due to their advanced architecture and functionality, the training process for these models presents complex challenges and demands careful analysis. Additionally, EfficientDet models come with pre-trained weights on the COCO dataset, which can provide a head start for specific applications.

EfficientDet-d0 is the most basic model with 3.9 million parameters and 2.54 GFlops. It has been evaluated against previous iterations of the Yolo model. During training, different batch sizes were used, including 16, 32, and 64. The optimal performance was achieved with a batch size of 16; each training session took approximately three hours to complete.

The first model examined in this study from the EfficientDet model range was the EfficientDet-d7, which had 51.9M parameters and 325 GFlops. However, limitations in computational resources made scaling this model up to batch sizes of 32 and 64 impossible. To continue comparing two detectors from the EfficientDet family, an evaluation was conducted to determine which model could accommodate an increased batch size within the available computational capacity.

The results indicated that the Efficientdet-d4, with its lower parameter count of 20.7 million and 55.2 GFlops, has been identified as a promising choice based on performance trade-offs, making it a favourable option for further exploration. The model was trained using various batch sizes, such as 16 and 32. Upon comparison of all the outcomes from this detector, it was found that the EfficientDet-d0 with a batch size of 16 exhibited superior performance, indicating that smaller batch sizes may lead to better results.

Overall, the most effective detector in terms of performance was the EfficientDet-d0. This indicates that larger models with increased parameters may not always be the optimal choice, especially when dealing with a less complex problem such as distinguishing between "Fire" and "No Fire". This finding implies that smaller models with lower computational requirements can still perform satisfactorily in fire detection tasks. Therefore, this is another research example where it is essential to consider model size and complexity carefully depending on the specific application requirements.

Another experiment assessed the impact of varying image sizes on training the detectors. The image inputs at 500x500, which closely align with the median size of images in the dataset, were tested. Specifically, outcomes were obtained using the top-performing models from earlier analysis shown in Table 2.

Table 2: Performance of object detectors with different input sizes

Model	Epochs	Input Size	Batch Size	mAP/AP	mAP/AP50
Yolov8x	50	500x500	16	0.212	0.488
EfficientDet-d0	50	500x500	16	0.213	0.511

The best-performing models from the previous analysis, Yolov8x and EfficientDet-d0, underwent evaluation using image inputs set at a 500x500 resolution. This adjustment extended their training time to 5 and 8 hours, respectively. However, the findings did not show substantial differences, as the image resolution did not lead to performance improvements. This implies that opting for smaller image sizes could be an effective approach for fire detection tasks since it reduces computational workload without compromising performance quality.

Table 3: Performance of object detectors with different datasets

Model	Epochs	Input Size	Batch Size	mAP/AP		mAP/AP50	
Yolov8x	50	250x250	16	NIR + RGB	0.193	NIR + RGB	0.449
				RGB	0.162	RGB	0.406

EfficientDet-d0	50	250x250	16	NIR + RGB	0.212	NIR + RGB	0.545
				RGB	0.170	RGB	0.471

Furthermore, the same models were evaluated using an updated original dataset to assess whether model accuracy could be enhanced for detecting forest fires without utilising NIR images. This set includes 489 training images, 54 validation images, and 56 test images. The original unaugmented dataset included 3755 images for training, 469 for validation, and 470 for testing. This reflects a reduction of about 13%,12%, and 12% respectively. Following the augmentation process, there were 6124 training images, 380 validation images and 388 testing images. The findings presented in the previous table indicate that excluding near-infrared imagery did have a substantial impact on the models' performance. These findings emphasise the need to utilise alternative image types to accurately detect forest fires, rather than relying solely on RGB images.

4.1.2. Classification models

Moving on to the classification model, Table 4 presents a comprehensive overview of the setup of different models employed for classification. It clearly outlines each model's parameters, GFLOPs, and whether pre-trained weights are utilised, providing valuable insight into their performance characteristics.

Table 4: Classification model characteristics

Model		Parameters	GFLOPS	Pretrained
MobileNetV3	mobilenet_v3_small_st	2542856	0.06	No
	mobilenet_v3_small_pt			Yes
	mobilenet_v3_large_st	5483032	0.22	No
	mobilenet_v3_large_pt			Yes
EfficientNetV2	efficientnet_v2_s_st	21458488	8.37	No
	efficientnet_v2_s_pt			Yes
	efficientnet_v2_m_st	54139356	24.58	No
	efficientnet_v2_m_pt			Yes
MaxVit	maxvit_t_st	30919624	5.56	No

	maxvit_t_pt			Yes
SwinTransformer	swin_v2_t_st	28351570	5.94	No
	swin_v2_t_pt			Yes
	swin_v2_s_st	49737442	11.55	No
	swin_v2_s_pt			Yes
VisionTransformer	vit_b_16_st	86567656	17.56	No
	vit_b_16_pt			Yes
	vit_b_32_st	88224232	4.41	No
	vit_b_32_pt			Yes

Before employing classification models, it is crucial to generate a new dataset for input into the models. The new dataset was originated from the outputs of Yolov8x and EfficientDet-d0 without a minimal confidence score for inference. The overlap of outcomes from both models maintains a minimum IOU threshold of 0.5, indicating that bounding boxes with at least 0.5 IOU between them are selected. For this next step, 180 images originating from bounding boxes were used, which were detected by the combination of the detector inference and 180 "No Fire" labelled images from the initial dataset. Subsequently, through an augmentation pipeline comprising resize, random horizontal flips, random rotation, random affine transformation, random perspective transformation, and normalisation, it was possible to increase this number to a bigger dataset utilised in the classification models.

The initial augmentation pipeline utilised a combination of methods, starting with a random horizontal flip. The images were then subjected to random rotations of up to 10 degrees to build resilience against minor rotational variations. Following this, random affine transformations were applied, encompassing translations equivalent to 10% of the image dimensions and scaling between 90% and 110% without rotation, aiming to simulate changes in image position and size. Furthermore, random perspective distortions were incorporated using a distortion scale 0.5 and a probability of 0.5 to replicate alterations in camera angle or viewpoint orientation. After implementing these transformations, the dataset comprised an equal distribution of 1307 images across the "Fire" and "No Fire" classes. The dataset was used to perform an initial test with the classification models. Tables 5 and 6 display the specified parameters, including 25 training epochs and an input dimension of 224x224.

Table 5: Performance of classification models with a batch size of 32

Model	Train				Validation			
	Loss	Precision	Recall	F1-Score	Loss	Precision	Recall	F1-Score
mobilenet_v3_small_st	0,029	0,992	0,993	0,992	0,034	0,990	0,990	0,990
mobilenet_v3_small_pt	0,000	1,000	1,000	1,000	0,000	1,000	1,000	1,000
mobilenet_v3_large_st	0,028	0,990	0,993	0,991	0,042	0,986	0,987	0,986
mobilenet_v3_large_pt	0,001	1,000	1,000	1,000	0,008	0,992	0,994	0,993
efficientnet_v2_s_st	0,066	0,976	0,979	0,976	0,088	0,967	0,972	0,968
efficientnet_v2_s_pt	0,050	0,985	0,989	0,986	0,036	0,986	0,987	0,986
efficientnet_v2_m_st	0,144	0,950	0,953	0,949	0,196	0,936	0,936	0,930
efficientnet_v2_m_pt	0,006	0,997	0,998	0,998	0,026	0,993	0,994	0,993
maxvit_t_st	0,393	0,871	0,873	0,861	0,335	0,875	0,889	0,878
maxvit_t_pt	0,004	0,998	0,998	0,998	0,000	1,000	1,000	1,000
swin_v2_t_st	0,696	0,373	0,505	0,385	0,704	0,306	0,500	0,338
swin_v2_t_pt	0,261	0,919	0,918	0,914	0,295	0,904	0,904	0,900
swin_v2_s_st	0,273	0,920	0,920	0,917	0,419	0,844	0,840	0,834
swin_v2_s_pt	0,702	0,305	0,502	0,355	0,694	0,493	0,506	0,387
vit_b_16_st	0,280	0,900	0,890	0,888	0,265	0,899	0,897	0,894
vit_b_16_pt	0,155	0,947	0,949	0,945	0,114	0,977	0,975	0,975
vit_b_32_st	0,159	0,944	0,937	0,938	0,372	0,898	0,882	0,876
vit_b_32_pt	0,110	0,960	0,957	0,956	0,155	0,964	0,955	0,957

Table 6: Performance of classification models with a batch size of 64

Model	Train				Validation			
	Loss	Precision	Recall	F1-Score	Loss	Precision	Recall	F1-Score

mobilenet_v3_small_st	0,027	0,993	0,994	0,993	0,248	0,938	0,940	0,934
mobilenet_v3_small_pt	0,000	1,000	1,000	1,000	0,000	1,000	1,000	1,000
mobilenet_v3_large_st	0,020	0,997	0,997	0,997	0,030	0,994	0,994	0,994
mobilenet_v3_large_pt	0,000	1,000	1,000	1,000	0,002	1,000	1,000	1,000
efficientnet_v2_s_st	0,075	0,970	0,976	0,972	0,200	0,957	0,958	0,955
efficientnet_v2_s_pt	0,000	1,000	1,000	1,000	0,001	1,000	1,000	1,000
efficientnet_v2_m_st	0,080	0,970	0,973	0,971	0,169	0,930	0,938	0,930
efficientnet_v2_m_pt	0,000	1,000	1,000	1,000	0,001	1,000	1,000	1,000
maxvit_t_st	0,696	0,418	0,517	0,384	0,700	0,485	0,484	0,482
maxvit_t_pt	0,000	1,000	1,000	1,000	0,000	1,000	1,000	1,000
swin_v2_t_st	0,461	0,802	0,800	0,799	0,504	0,791	0,789	0,789
swin_v2_t_pt	0,044	0,988	0,986	0,987	0,206	0,955	0,951	0,950
swin_v2_s_st	0,764	0,564	0,558	0,505	0,695	0,567	0,555	0,537
swin_v2_s_pt	0,695	0,508	0,505	0,444	0,701	0,538	0,514	0,418
vit_b_16_st	0,284	0,879	0,876	0,875	0,266	0,880	0,882	0,880
vit_b_16_pt	0,093	0,964	0,961	0,961	0,146	0,949	0,955	0,950
vit_b_32_st	0,168	0,941	0,932	0,934	0,208	0,952	0,946	0,944
vit_b_32_pt	0,124	0,965	0,963	0,963	0,239	0,936	0,932	0,931

From these results, it is evident that the models are performing very well. The dataset used lacks diversity, which may suggest that the obtained performance indicates overfitting. However, it is important to note that models with pre-trained weights perform significantly better than those without. It's also worth noting that simple models like mobilenetv3 outperform more complex ones like EfficientNet and Transformers. This suggests that the classification problem may not be as complex and does not require sophisticated models to solve. Furthermore, using pre-trained weights greatly improved the performance of the models, indicating that transfer learning is an effective technique in this context.

An enhanced augmentation approach has been implemented with robust methods to address the overfitting issue and improve model generalisation. This includes increasing the rotation limit to 30 degrees and extending affine transformations to cover rotations of up to 15

degrees, translations by 20% of image dimensions, scaling between 80% and 120%, and shearing by 10 degrees. Additionally, intensifying colour jittering, which alters brightness, contrast, saturation, and hue by up to 50%, introduces greater variability in a colour that aids performance across varying lighting conditions. Periodically converting images to grayscale with a probability of 20% emphasises structural details over colour information. Moreover, integrating random vertical flips with a probability of 20% enriches the training data by introducing more orientation variability. The outcomes are detailed in the subsequent table:

Table 7: Performance of classification models with a batch size of 32

Model	Train				Validation			
	Loss	Precision	Recall	F1-Score	Loss	Precision	Recall	F1-Score
mobilenet_v3_small_st	0,194	0,927	0,933	0,926	0,219	0,891	0,905	0,892
mobilenet_v3_small_pt	0,019	0,995	0,995	0,995	0,072	0,972	0,972	0,972
mobilenet_v3_large_st	0,120	0,947	0,952	0,947	0,184	0,932	0,942	0,933
mobilenet_v3_large_pt	0,052	0,980	0,985	0,982	0,112	0,978	0,981	0,979
efficientnet_v2_s_st	0,194	0,917	0,928	0,915	0,242	0,887	0,885	0,875
efficientnet_v2_s_pt	0,033	0,986	0,990	0,988	0,128	0,962	0,965	0,961
efficientnet_v2_m_st	0,243	0,907	0,915	0,905	0,391	0,853	0,838	0,815
efficientnet_v2_m_pt	0,119	0,958	0,970	0,959	0,218	0,938	0,947	0,939
maxvit_t_st	0,376	0,816	0,813	0,804	0,339	0,837	0,837	0,831
maxvit_t_pt	0,055	0,983	0,985	0,983	0,067	0,964	0,970	0,965
swin_v2_t_st	0,674	0,608	0,595	0,551	0,690	0,544	0,537	0,520
swin_v2_t_pt	0,694	0,430	0,484	0,443	0,693	0,481	0,486	0,424
swin_v2_s_st	0,482	0,798	0,796	0,785	0,490	0,795	0,756	0,745
swin_v2_s_pt	0,697	0,452	0,496	0,415	0,699	0,495	0,510	0,436
vit_b_16_st	0,453	0,805	0,796	0,790	0,537	0,796	0,741	0,722
vit_b_16_pt	0,282	0,878	0,873	0,869	0,331	0,850	0,838	0,829
vit_b_32_st	0,464	0,805	0,791	0,786	0,469	0,774	0,762	0,763
vit_b_32_pt	0,316	0,865	0,859	0,852	0,279	0,888	0,872	0,869

Table 8: Performance of classification models with a batch size of 64

Model	Train				Validation			
	Loss	Precision	Recall	F1-Score	Loss	Precision	Recall	F1-Score
mobilenet_v3_small_st	0,104	0,962	0,965	0,963	0,242	0,905	0,905	0,904
mobilenet_v3_small_pt	0,007	0,997	0,997	0,997	0,021	0,990	0,992	0,990
mobilenet_v3_large_st	0,119	0,954	0,959	0,955	0,121	0,954	0,954	0,953
mobilenet_v3_large_pt	0,023	0,995	0,997	0,996	0,114	0,977	0,980	0,978
efficientnet_v2_s_st	0,201	0,920	0,926	0,920	0,227	0,900	0,918	0,897
efficientnet_v2_s_pt	0,027	0,992	0,992	0,992	0,036	0,991	0,991	0,991
efficientnet_v2_m_st	0,276	0,900	0,902	0,897	0,242	0,899	0,905	0,896
efficientnet_v2_m_pt	0,018	0,992	0,994	0,993	0,094	0,966	0,968	0,965
maxvit_t_st	0,195	0,924	0,928	0,924	0,197	0,924	0,930	0,924
maxvit_t_pt	0,023	0,993	0,994	0,993	0,045	0,986	0,983	0,984
swin_v2_t_st	0,688	0,554	0,554	0,544	0,689	0,526	0,525	0,521
swin_v2_t_pt	0,658	0,636	0,651	0,604	0,535	0,717	0,709	0,704
swin_v2_s_st	0,636	0,648	0,642	0,636	0,645	0,699	0,611	0,561
swin_v2_s_pt	0,203	0,918	0,921	0,916	0,135	0,950	0,948	0,946
vit_b_16_st	0,489	0,757	0,751	0,748	0,460	0,837	0,794	0,787
vit_b_16_pt	0,431	0,797	0,787	0,783	0,354	0,854	0,852	0,852
vit_b_32_st	0,405	0,820	0,806	0,802	0,393	0,821	0,798	0,787
vit_b_32_pt	0,314	0,851	0,845	0,840	0,227	0,913	0,911	0,912

The results from Table 8 indicate that the mobilenet_v3_small_pt and efficientnet_v2_s_pt achieved a 99% F1 score, while maxvit_t_pt reached a 98% F1 score. These models exhibited the highest performance, showcasing their strong suitability for this task.

4.2. DISCUSSION

My research centres on creating a CV model for identifying rural fires at close range for devices with limited computational resources. It includes investigating the impact of infrared images on improving model accuracy, comparing the efficiencies of ViT and CNN in this scenario, and testing a novel framework that integrates detection and classification models using an ensemble approach.

Forest fires differ from other static objects due to their dynamic nature. Typically, a forest fire begins as a small-scale blaze, starting as a ground fire, progressing into medium and large-scale fires, spreading to the trunk and finally reaching the canopy. Their diverse shapes, sizes, textures, and colours contribute to the complex evolution process and pose significant challenges for fire detection.

Furthermore, comparing performance across studies in this field requires careful consideration of the different methodologies, datasets and purposes.

(Zheng et al., 2023), combined YOLOv4 with MobileNetV3 to achieve real-time detection on resource-constrained devices. They emphasise the significance of improving computational efficiency while upholding high accuracy. This approach aligns with my research, as it involves utilising a similar family model within a different framework. Rather than replacing the backbone of YOLOv4 with MobileNetV3-large, my study applies MobileNetV3-large to the output of YOLO and other detectors. Their algorithm attained a 96.24% accuracy in identifying fire incidents using the BoWFire public dataset. The experimentation encompassed 2334 images, comprising flame and smoke visuals along with control images devoid of fire-related content, and a training process extended up to 1000 epochs.

Compared with the study of (Xu et al., 2021), there are many similarities, especially in utilising YOLO and EfficientDet as detectors. However, there has been significant evolution from YOLOv5 to YOLOv8. The approach in this research involved using a classification model alongside the detectors to capture global information from images and prevent false positives. My methodology follows a similar process, but the classifiers utilise outputs from the detectors, which makes their training tailored to these types of images. Additionally, it was chosen not to apply NMS after the detectors' output because it would result in a limited bounding boxes dataset regarding image quantity. It can be noted that their study achieved better performance in detection models but also managed to train for more epochs despite having a smaller dataset.

5. CONCLUSIONS, LIMITATIONS AND FUTURE RESEARCH

5.1. CONCLUSIONS

Many models and frameworks are being used in the field to address various environments and circumstances, including different types of detection and classification. While these technologies continue to develop rapidly, they require more specialization to address specific challenges. Achieving perfection for real-world applications that impact human lives and the environment remains a challenge.

This master's thesis presents a unique and comprehensive approach to rural fire detection using a close-range approach. This novel method, tailored to Portugal's unique geographical and environmental conditions, addresses a significant challenge that the country faces when fighting forest fires.

The framework developed in this thesis distinguishes itself from other studies by adopting a ground-based approach, using standard near-infrared and RGB cameras on vehicles. This choice of more budget-friendly systems ensures easy deployment and integration into the Portuguese infrastructure, making it a practical and cost-effective solution.

The thesis's primary objectives were to:

1. To develop a CV framework for close-range rural fire detection for low computational cost devices.
2. To assess if infrared images can increase rural fire detection model performance.
3. To analyse how CNNs can identify fire in rural environments.
4. To compare the performance of ViT and CNN in rural fire classification.

The proposed methodology harnesses the advancements in CV and DL to overcome the critical limitations of traditional fire detection systems. It demonstrates its effectiveness by accurately classifying and localising small-scale rural fires in complex and dynamic environments, bringing hope for an improved fire detection and mitigation system.

The proposed framework, a key contribution of this research, combines ensemble methods with two object detection models, Yolov8 and EfficientDet. This combination allows for a more robust and accurate fire detection, followed by a classification model that further enhances the system's performance. Importantly, this framework has demonstrated strong performance without relying on complex networks or computationally intensive processing.

Throughout this procedure, two new sets of data were generated. The first set was created by combining several datasets mentioned in Chapter 3. A second intermediate dataset was

then produced, focusing on the bounding boxes resulting from the initial combined inference process performed by the object detector models.

During the classification stage, the task was somewhat simplified by combining the output of two object detectors. This facilitated rapid and straightforward learning for the classifiers, proving the unnecessary use of complex models at the end of the framework. As a result, deployment on low computational cost devices becomes feasible.

The experimental results have demonstrated the efficiency of the proposed approach in localising and classifying fire versus non-fire scenes. Furthermore, according to Nvidia documentation, the Jetson Nano, a microprocessor used in the implementation of robotics, takes 472 GFLOPS for 16 FPS or 128 GFLOPS for 32 FPS. This means that the summation of the GFLOPS of the proposed framework for an FPS of 16 doesn't pass the maximum value of GFLOPS, considering using the two detectors, Yolov8x and EfficientDet-D0, and one of the classification models, mobilenet_v3_small_pt, efficientnet_v2_s_pt or maxvit_t_pt. However, if the objective is to go for 32 FPS, some choices regarding using the detectors should be considered. For instance, using Yolov8n isn't the best option, but it would make the GFLOPS drop considerably compared with the Yolov8x making the combination of the object detectors plus a classification model within the limited threshold of GFLOPS of the hardware benchmarked.

The key findings from this research include:

- The framework for rural fire detection achieved impressive performance in the classification stage, as shown in Chapter 4. This allowed the conclusion that better results are attainable with smaller and more efficient networks.

- Furthermore, the combined use of RGB and NIR cameras can significantly improve the performance of rural fire detection models. The specific characteristics of NIR provide complementary information that enhances object detector performance.

- In addition, it has been observed that CNN models outperform transformer models in rural fire classification when used later in the pipeline. They also offer a more efficient and scalable approach. However, small transformer models (e.g., MaxViT) remain viable for deployment on low-compute devices.

- In conclusion, the analysis of image resolution for close-range approaches indicates that having high-quality image resolution is optional. This finding has significant implications for the practical implementation of fire detection systems, suggesting that a balance between image quality and computational resources can be obtained, thereby making the system more efficient and cost-effective.

5.2. LIMITATIONS

The complexity of research tasks and the need to create a new dataset led to a compressed schedule, leaving little room for exploring alternative methodologies or conducting more extensive validations. This limitation, while challenging, also presents opportunities for future researchers to delve deeper into these areas and make additional contributions.

Another constraint faced during the research was the availability of computational resources. Using a paid Google Collab account was a good approach since it allowed for accumulating some computational units when less computational power was necessary. However, limited access to high-performance computing resources was still restricted, and the more powerful GPUs were usually unavailable. This was particularly relevant in the training of the object detector models, especially EfficientDet, which led to longer training times and the limitation of the parameters tested.

Integration of the developed framework with the robotic world is still a challenge since, to test the developed pipeline in real-world scenarios, it would be necessary to consider hardware limitations, like energy consumption and physical environmental factors. The complexity of implementing the framework into practical robotic applications means that detection and mitigation systems, like water supply and guns, would be necessary. This gap between simulation and practical deployment could lead to unforeseen issues in actual operation, where variables are far more unpredictable than those in controlled experimental settings.

5.3. FUTURE RESEARCH

Notwithstanding these limitations, the research has significantly contributed to rural fire detection, laying a foundation for future work in this critical area. Some key recommendations for future research follow the use of real-time fire combat systems with adaptive combat based on the behaviour of the flames. This would involve the integration of the pipeline with robotic systems capable of extinguishing fires based on real-time fire behaviour analysis. Besides including the pipeline to detect fire, it needs additional programmability to take action into the fire and release suppression techniques with water and others. Finally, further exploration of temporal factors such as flame development, fire spread, and colour variance in fire detection algorithms could be conducted using videos to include a temporal analysis and predict flame behaviour.

BIBLIOGRAPHICAL REFERENCES

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., … Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. *OSDI'16: Proceedings of the 12th USENIX Conference on Operating Systems Design and Implementation*. https://doi.org/10.48550/arXiv.1605.08695

Albuquerque, C. (2018). *Convolutional neural networks for cell detection and counting: A case study of human cell quantification in zebrafish xenografts using deep learning object detection techniques* [Master's thesis, NOVA Information Management School]. http://hdl.handle.net/10362/62425

Barmpoutis, P., Dimitropoulos, K., Kaza, K., & Grammalidis, N. (2019). Fire Detection from Images Using Faster R-CNN and Multidimensional Texture Analysis. *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 8301–8305. https://doi.org/10.1109/ICASSP.2019.8682647

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*. http://www.drdobbs.com/open-source/the-opencv-library/184404319

Buslaev, A., Iglovikov, V. I., Khvedchenya, E., Parinov, A., Druzhinin, M., & Kalinin, A. A. (2020). Albumentations: Fast and Flexible Image Augmentations. *Information*, *11*(2), 125. https://doi.org/10.3390/info11020125

Chino, D. Y. T., Avalhais, L. P. S., Rodrigues Jr., J. F., & Traina, A. J. M. (2015). BoWFire: Detection of Fire in Still Images by Integrating Pixel Color and Texture Analysis. *2015*

28th SIBGRAPI Conference on Graphics, Patterns and Images, 95–102. https://doi.org/10.1109/SIBGRAPI.2015.19

Chollet, F. (2018). *Deep learning with Python*. Manning Publications Co. https://www.manning.com/books/deep-learning-with-python

De Venâncio, P. V. A. B., Campos, R. J., Rezende, T. M., Lisboa, A. C., & Barbosa, A. V. (2023). A hybrid method for fire detection based on spatial and temporal patterns. *Neural Computing and Applications*, *35*(13), 9349–9361. https://doi.org/10.1007/s00521-023-08260-2

De Venâncio, P. V. A. B., Lisboa, A. C., & Barbosa, A. V. (2022). An automatic fire detection system based on deep convolutional neural networks for low-power, resource-constrained devices. *Neural Computing and Applications*, *34*(18), 15349–15368. https://doi.org/10.1007/s00521-022-07467-z

De Venancio, P. V. A. B., Rezende, T. M., Lisboa, A. C., & Barbosa, A. V. (2021). Fire Detection based on a Two-Dimensional Convolutional Neural Network and Temporal Analysis. *2021 IEEE Latin American Conference on Computational Intelligence (LA-CCI)*, 1–6. https://doi.org/10.1109/LA-CCI48322.2021.9769824

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale* (arXiv:2010.11929). arXiv. http://arxiv.org/abs/2010.11929

Dwyer, B., Nelson, J., & Hansen, T. (2024). *Roboflow: Give your software the power to see objects in images and video* (Version 1) [Computer software]. https://roboflow.com/

Ghali, R., Akhloufi, M. A., & Mseddi, W. S. (2022). Deep Learning and Transformer Approaches for UAV-Based Wildfire Detection and Segmentation. *Sensors*, *22*(5), 1977. https://doi.org/10.3390/s22051977

Girshick, R., Donahue, J., Darrell, T., & Malik, J. (2014). Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation. *2014 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/CVPR.2014.81

Google/automl. (2024). [Jupyter Notebook]. Google. https://github.com/google/automl (Original work published 2020)

Guan, Z., Miao, X., Mu, Y., Sun, Q., Ye, Q., & Gao, D. (2022). Forest Fire Segmentation from Aerial Imagery Data Using an Improved Instance Segmentation Model. *Remote Sensing*, *14*(13), 3159. https://doi.org/10.3390/rs14133159

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., & Adam, H. (2017). *MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications* (arXiv:1704.04861). arXiv. http://arxiv.org/abs/1704.04861

Howard, A., Sandler, M., Chu, G., Chen, L.-C., Chen, B., Tan, M., Wang, W., Zhu, Y., Pang, R., Vasudevan, V., Le, Q. V., & Adam, H. (2019). *Searching for MobileNetV3* (arXiv:1905.02244). arXiv. http://arxiv.org/abs/1905.02244

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J., & Keutzer, K. (2016). *SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size* (arXiv:1602.07360). arXiv. http://arxiv.org/abs/1602.07360

Jocher, G, Chaurasia, A, & Qiu, J. (2023). *Ultralytics YOLO* (Version 8) [Computer software]. https://github.com/ultralytics/ultralytics

Joint Research Centre. (2023). *The EU 2022 wildfire season was the second worst on record*. https://joint-research-centre.ec.europa.eu/jrc-news-and-updates/eu-2022-wildfire-season-was-second-worst-record-2023-05-02_en

Khan, A., & Hassan, B. (2020). *Dataset for Forest Fire Detection* (Version 1) [Dataset]. Mendeley Data. https://doi.org/10.17632/gjmr63rz2r.1

Khan, Z. A., Hussain, T., Ullah, F. U. M., Gupta, S. K., Lee, M. Y., & Baik, S. W. (2022). Randomly Initialized CNN with Densely Connected Stacked Autoencoder for Efficient Fire Detection. *Engineering Applications of Artificial Intelligence*, *116*, 105403. https://doi.org/10.1016/j.engappai.2022.105403

Kim, B., & Lee, J. (2019). A Video-Based Fire Detection Using Deep Learning Models. *Applied Sciences*, *9*(14), 2862. https://doi.org/10.3390/app9142862

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, *60*(6), 84–90. https://doi.org/10.1145/3065386

Kukil. (2022, August 9). *Mean Average Precision (mAP) in Object Detection*. https://learnopencv.com/mean-average-precision-map-object-detection-model-evaluation-metric/

Lecun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, *86*(11), 2278–2324. https://doi.org/10.1109/5.726791

Li, P., & Zhao, W. (2020). Image fire detection algorithms based on convolutional neural networks. *Case Studies in Thermal Engineering*, *19*, 100625. https://doi.org/10.1016/j.csite.2020.100625

Li, Z., Mihaylova, L., & Yang, L. (2021). A deep learning framework for autonomous flame detection. *Neurocomputing*, *448*, 205–216. https://doi.org/10.1016/j.neucom.2021.03.019

Liu, Z., Hu, H., Lin, Y., Yao, Z., Xie, Z., Wei, Y., Ning, J., Cao, Y., Zhang, Z., Dong, L., Wei, F., & Guo, B. (2022). *Swin Transformer V2: Scaling Up Capacity and Resolution* (arXiv:2111.09883v2). https://doi.org/10.48550/arXiv.2111.09883

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows* (arXiv:2103.14030). http://arxiv.org/abs/2103.14030

Luo, W. (2022). Research on fire detection based on YOLOv5. *2022 3rd International Conference on Big Data, Artificial Intelligence and Internet of Things Engineering (ICBAIE)*, 537–540. https://doi.org/10.1109/ICBAIE56435.2022.9985857

Mardani, K., Vretos, N., & Daras, P. (2023). Transformer-Based Fire Detection in Videos. *Sensors*, *23*(6), 3035. https://doi.org/10.3390/s23063035

Muhammad, K., Ahmad, J., Lv, Z., Bellavista, P., Yang, P., & Baik, S. W. (2019). Efficient Deep CNN-Based Fire Detection and Localization in Video Surveillance Applications. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, *49*(7), 1419–1434. https://doi.org/10.1109/TSMC.2018.2830099

Muhammad, K., Khan, S., Elhoseny, M., Hassan Ahmed, S., & Wook Baik, S. (2019). Efficient Fire Detection for Uncertain Surveillance Environment. *IEEE Transactions on Industrial Informatics*, *15*(5), 3113–3122. https://doi.org/10.1109/TII.2019.2897594

Palanisamy, S., & Easwaran, D. (2023). Forest Fire Detection Based on Light-Weight Deep Neural Networks. *2023 2nd International Conference on Advancements in Electrical, Electronics, Communication, Computing and Automation (ICAECA)*, 1–6. https://doi.org/10.1109/ICAECA56562.2023.10201047

Pan, H., Badawi, D., & Cetin, A. E. (2020). Computationally Efficient Wildfire Detection Method Using a Deep Convolutional Network Pruned via Fourier Analysis. *Sensors*, *20*(10), 2891. https://doi.org/10.3390/s20102891

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Garnett, R. (2019). PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32* (pp. 8024–8035) [Python]. Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf (Original work published 2016)

Prasad, M. (2022). *Forest Fire Images* [Dataset]. https://www.kaggle.com/datasets/mohnishsaiprasad/forest-fire-images

Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). You Only Look Once: Unified, Real-Time Object Detection. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 779–788. https://doi.org/10.1109/CVPR.2016.91

Ren, S., He, K., Girshick, R., & Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *39*(6), 1137–1149. https://doi.org/10.1109/TPAMI.2016.2577031

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, *65*(6), 386–408. https://doi.org/10.1037/h0042519

Rossi, L. (2024). *Corsican Fire Database* [Dataset]. https://cfdb.univ-corse.fr/index.php?menu=1

Shixiao Wu, Chengcheng Guo, & Jianfeng Yang. (2020). Using PCAand one-stage detectors for real-time forest fire detection. *The 3rd Asian Conference on Artificial Intelligence Technology (ACAIT 2019)*. https://doi.org/10.1049/joe.2019.1145

Sousa, M. J., Moutinho, A., & Almeida, M. (2020). Wildfire detection using transfer learning on augmented datasets. *Expert Systems with Applications*, *142*, 112975. https://doi.org/10.1016/j.eswa.2019.112975

Szeliski, R. (2022). *Computer Vision: Algorithms and Applications* (2nd ed.). Springer. https://doi.org/10.1007/978-3-030-34372-9

Tan, M., & Le, Q. V. (2020). *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks* (arXiv:1905.11946). arXiv. http://arxiv.org/abs/1905.11946

Tan, M., & Le, Q. V. (2021). *EfficientNetV2: Smaller Models and Faster Training* (arXiv:2104.00298). arXiv. http://arxiv.org/abs/2104.00298

Tan, M., Pang, R., & Le, Q. V. (2020). EfficientDet: Scalable and Efficient Object Detection. *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 10778–10787. https://doi.org/10.1109/CVPR42600.2020.01079

Tao, L., Dong, M., & Xu, C. (2023). *Dual Focal Loss for Calibration* (arXiv:2305.13665). arXiv. http://arxiv.org/abs/2305.13665

Terven, J., Cordova-Esparza, D., & Romero-González J. (2023). A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Machine Learning and Knowledge Extraction*, *5*(4), 1680–1716. https://doi.org/10.3390/make5040083

Tsalera, E., Papadakis, A., Voyiatzis, I., & Samarakou, M. (2023). CNN-based, contextualized, real-time fire detection in computational resource-constrained environments. *Energy Reports*, *9*, 247–257. https://doi.org/10.1016/j.egyr.2023.05.260

Tu, Z., Talebi, H., Zhang, H., Yang, F., Milanfar, P., Bovik, A., & Li, Y. (2022). *MaxViT: Multi-Axis Vision Transformer* (arXiv:2204.01697). arXiv. http://arxiv.org/abs/2204.01697

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). *Attention Is All You Need* (arXiv:1706.03762). arXiv. http://arxiv.org/abs/1706.03762

Wu, S., & Zhang, L. (2018). Using Popular Object Detection Methods for Real Time Forest Fire Detection. *2018 11th International Symposium on Computational Intelligence and Design (ISCID)*, 280–284. https://doi.org/10.1109/ISCID.2018.00070

Xu, R., Lin, H., Lu, K., Cao, L., & Liu, Y. (2021). A Forest Fire Detection System Based on Ensemble Learning. *Forests*, *12*(2), 217. https://doi.org/10.3390/f12020217

Yan, G., Guo, J., Zhu, D., Zhang, S., Xing, R., Xiao, Z., & Wang, Q. (2023). A Flame Detection Algorithm Based on Improved YOLOv7. *Applied Sciences*, *13*(16), 9236. https://doi.org/10.3390/app13169236

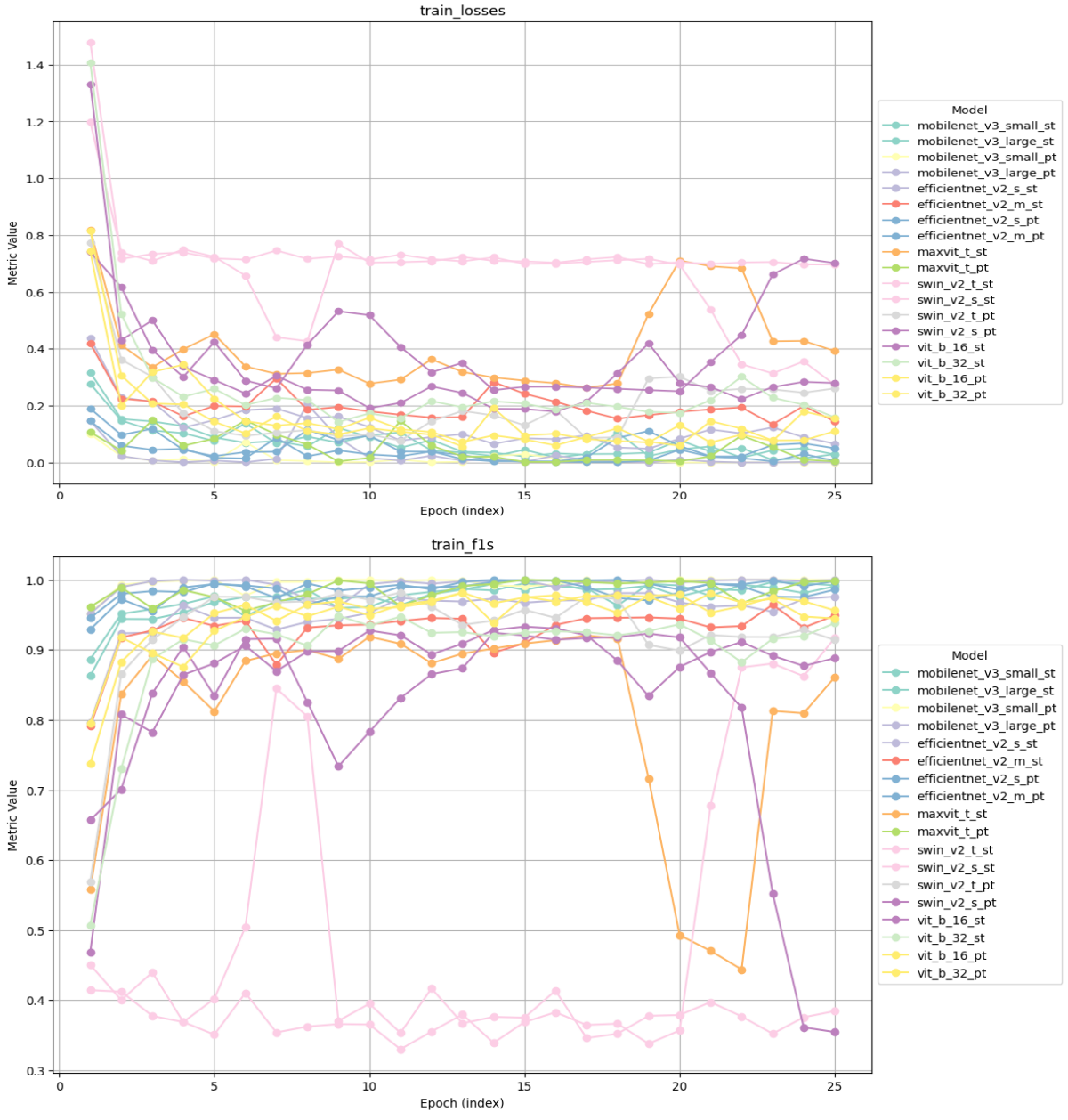
Zhang, J., Zhu, H., Wang, P., & Ling, X. (2021). ATT Squeeze U-Net: A Lightweight Network for Forest Fire Detection and Recognition. *IEEE Access*, *9*, 10858–10870. https://doi.org/10.1109/ACCESS.2021.3050628

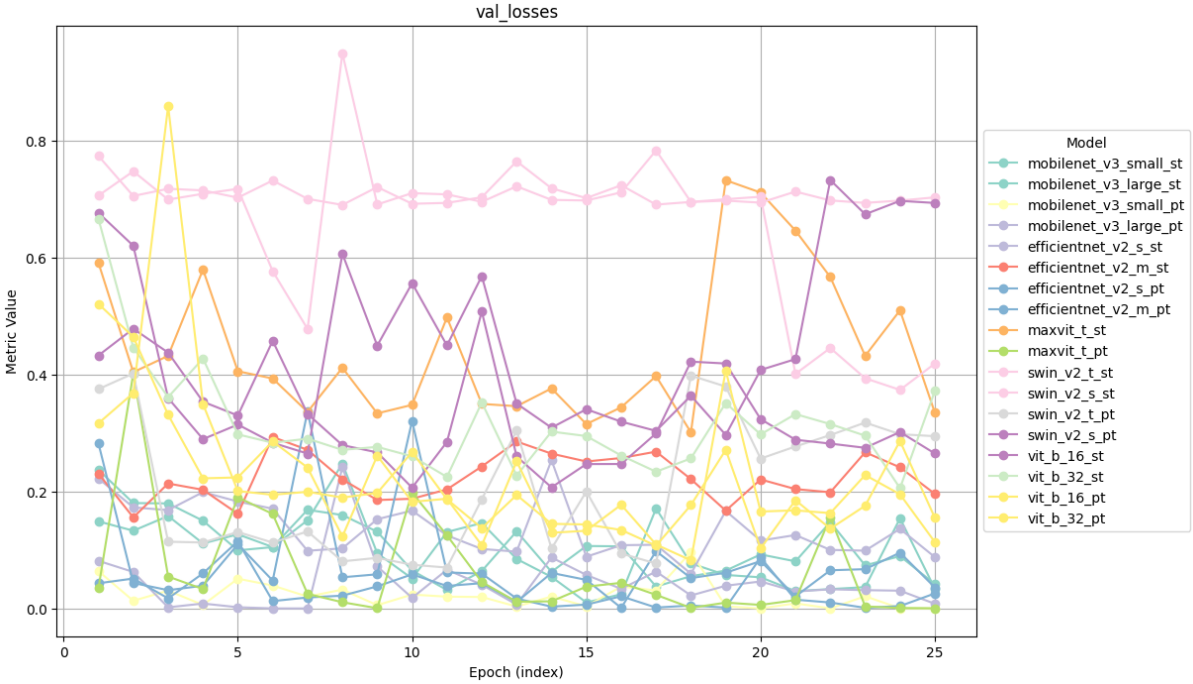
Zheng, H., Duan, J., Dong, Y., & Liu, Y. (2023). Real-time fire detection algorithms running on small embedded devices based on MobileNetV3 and YOLOv4. *Fire Ecology*, *19*(1), 31. https://doi.org/10.1186/s42408-023-00189-0

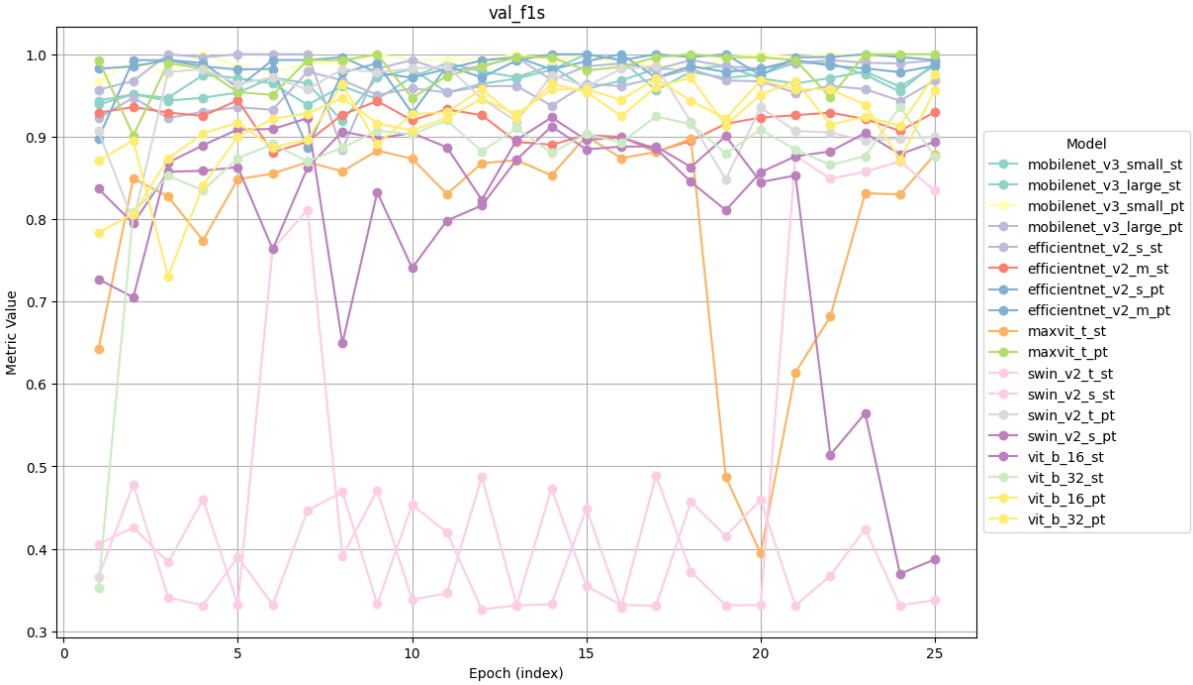
APPENDIX A

This appendix shows the evolution of the classification models in the training process. Each of them is referenced to a table in chapter 4.

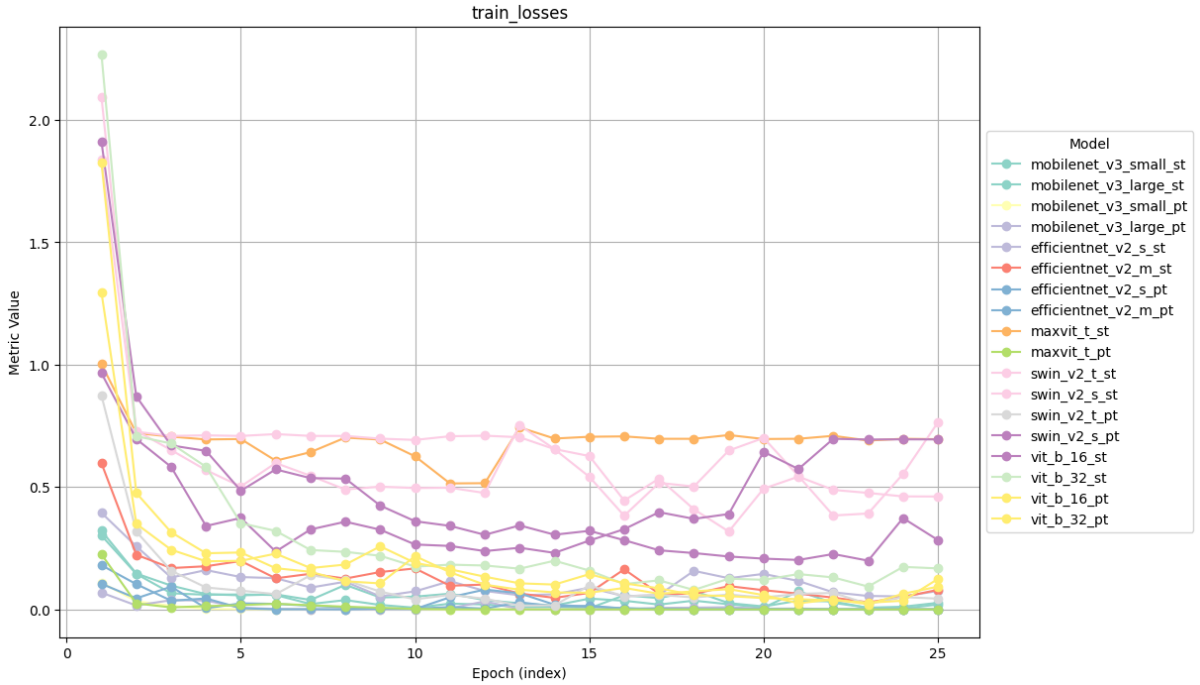
Appendix of table 5:

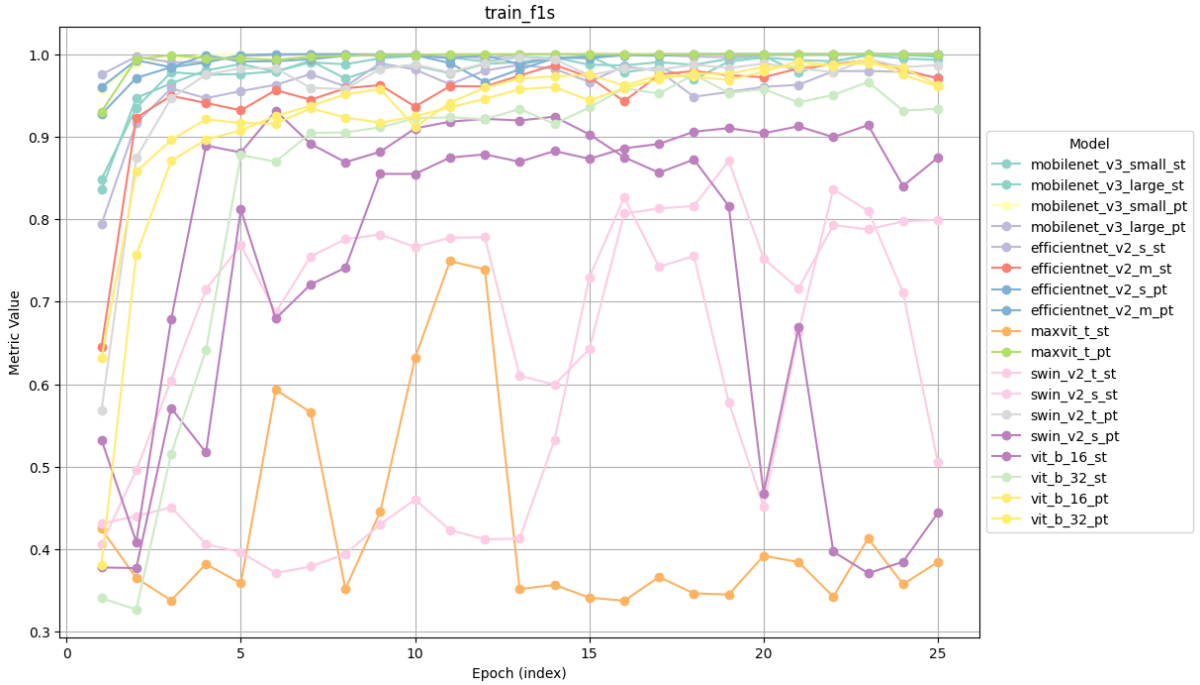


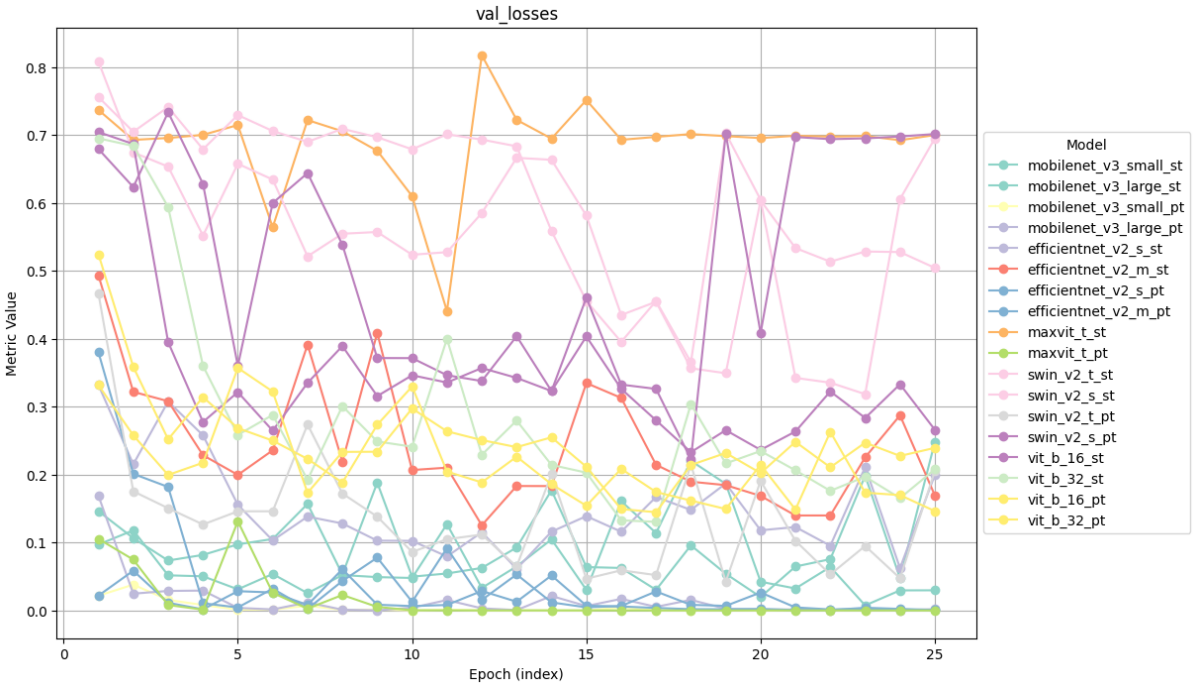

val_losses
Metric Value
Epoch (index)
Model
mobilenet_v3_small_st
mobilenet_v3_large_st
mobilenet_v3_small_pt
mobilenet_v3_large_pt
efficientnet_v2_s_st
efficientnet_v2_m_st
efficientnet_v2_s_pt
efficientnet_v2_m_pt
maxvit_t_st
maxvit_t_pt
swin_v2_t_st
swin_v2_s_st
swin_v2_t_pt
swin_v2_s_pt
vit_b_16_st
vit_b_32_st
vit_b_16_pt
vit_b_32_pt

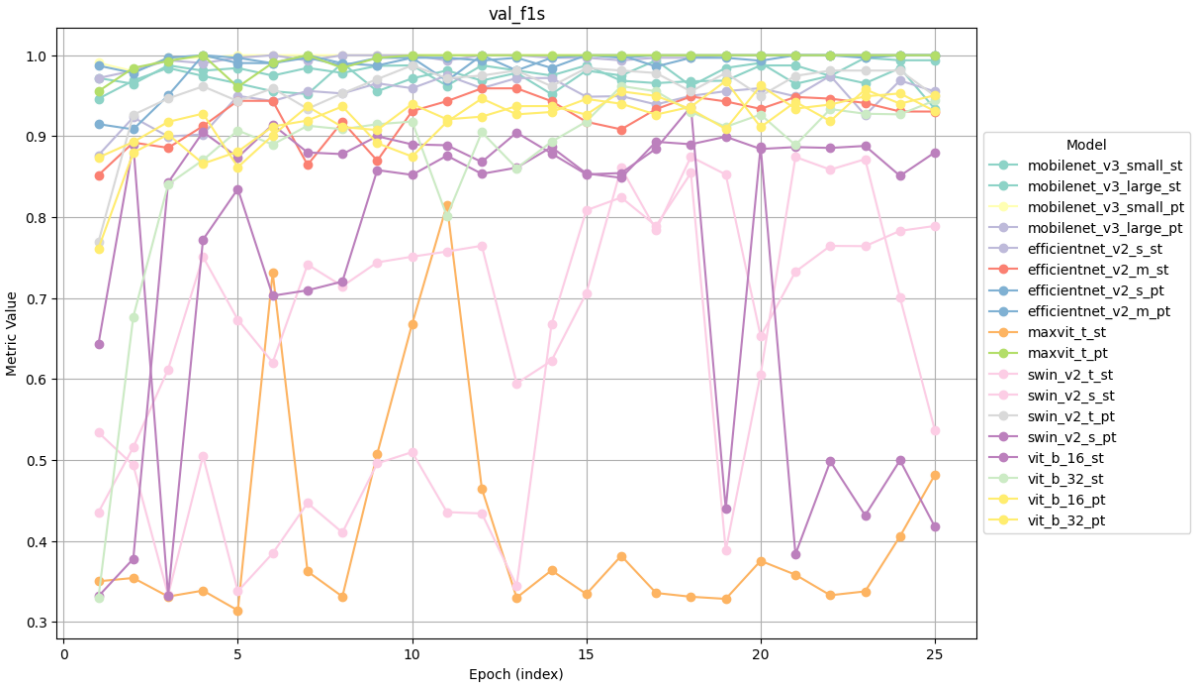

val_f1s
Metric Value
Epoch (index)
Model
mobilenet_v3_small_st
mobilenet_v3_large_st
mobilenet_v3_small_pt
mobilenet_v3_large_pt
efficientnet_v2_s_st
efficientnet_v2_m_st
efficientnet_v2_s_pt
efficientnet_v2_m_pt
maxvit_t_st
maxvit_t_pt
swin_v2_t_st
swin_v2_s_st
swin_v2_t_pt
swin_v2_s_pt
vit_b_16_st
vit_b_32_st
vit_b_16_pt
vit_b_32_pt

Appendix of table 6:

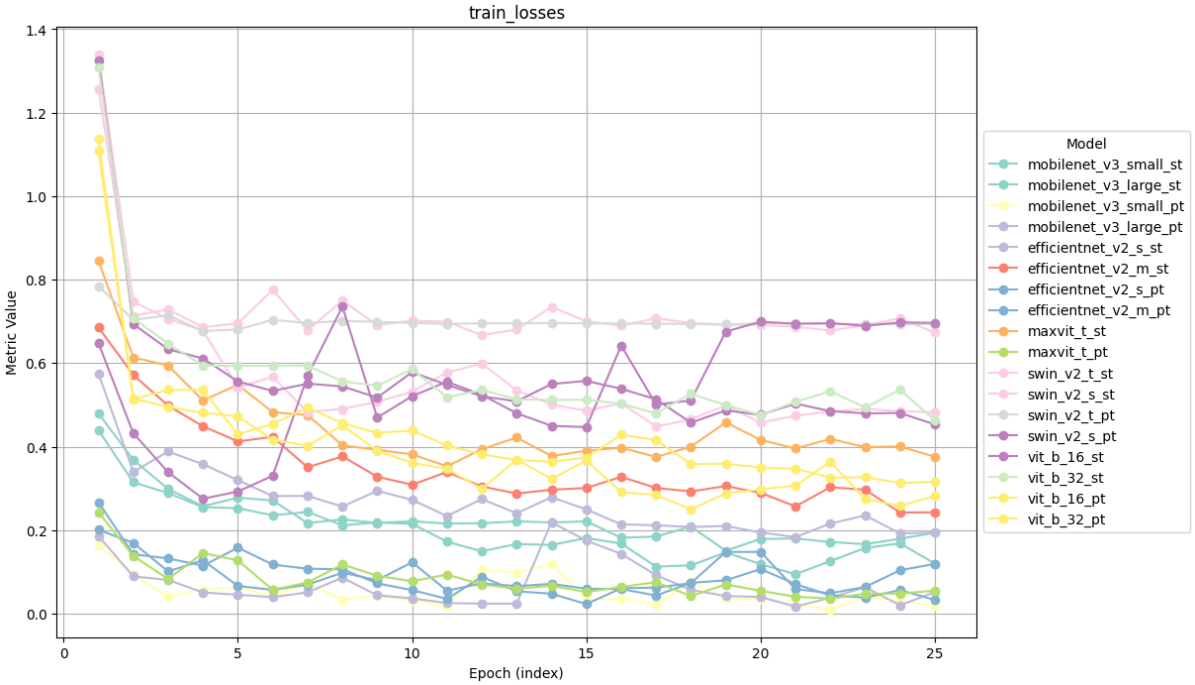


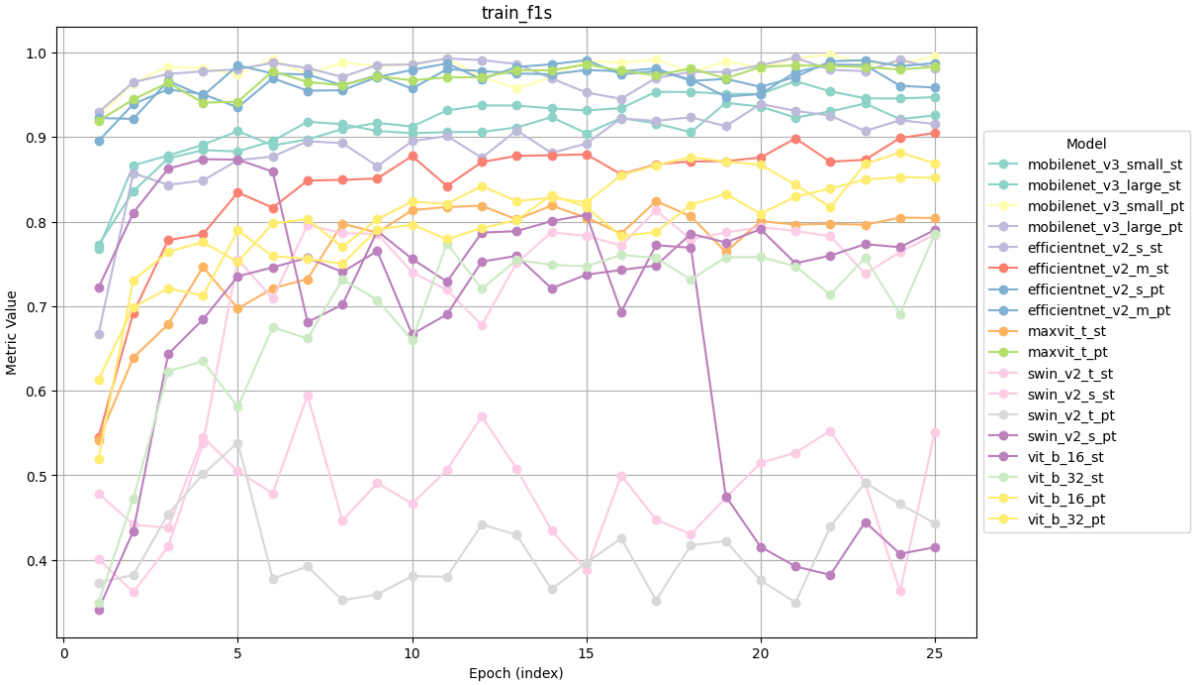


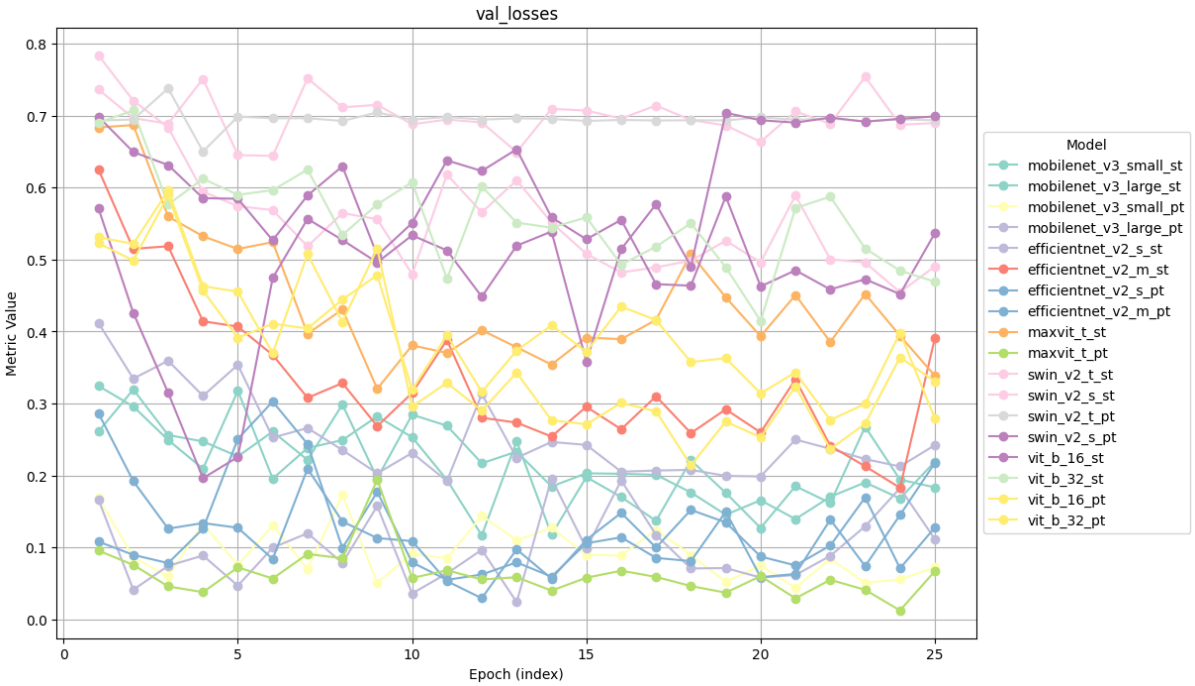

val_losses
Metric Value
Epoch (index)
Model
mobilenet_v3_small_st
mobilenet_v3_large_st
mobilenet_v3_small_pt
mobilenet_v3_large_pt
efficientnet_v2_s_st
efficientnet_v2_m_st
efficientnet_v2_s_pt
efficientnet_v2_m_pt
maxvit_t_st
maxvit_t_pt
swin_v2_t_st
swin_v2_s_st
swin_v2_t_pt
swin_v2_s_pt
vit_b_16_st
vit_b_32_st
vit_b_16_pt
vit_b_32_pt

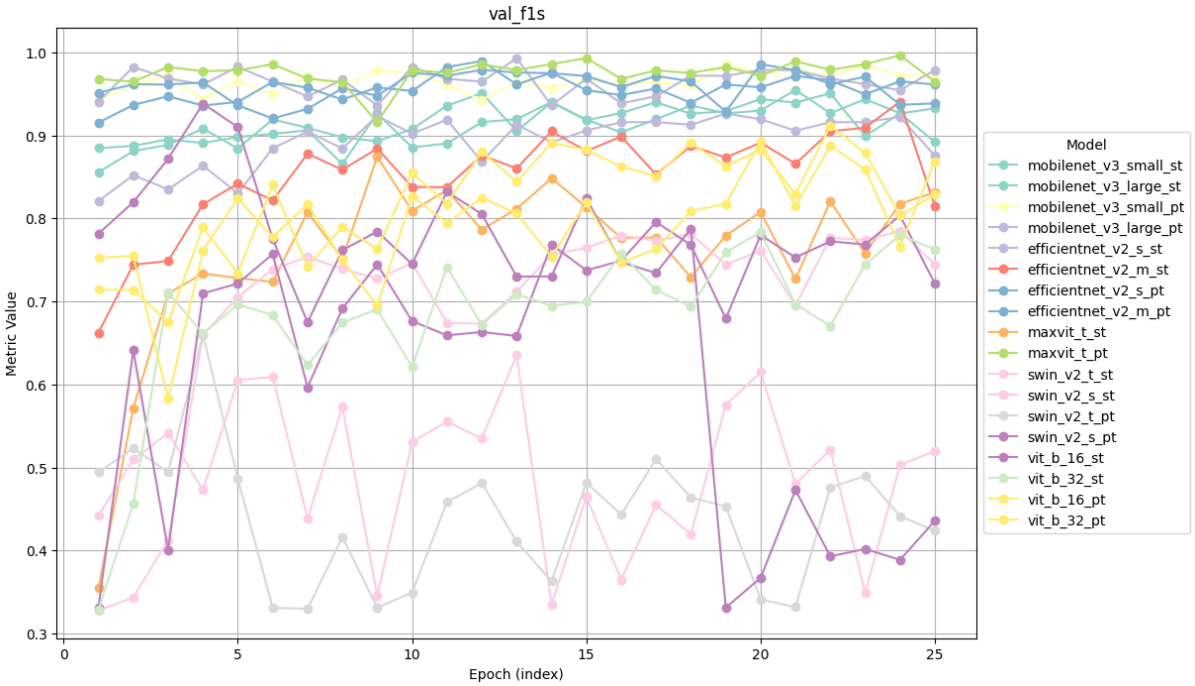

val_f1s
Metric Value
Epoch (index)
Model
mobilenet_v3_small_st
mobilenet_v3_large_st
mobilenet_v3_small_pt
mobilenet_v3_large_pt
efficientnet_v2_s_st
efficientnet_v2_m_st
efficientnet_v2_s_pt
efficientnet_v2_m_pt
maxvit_t_st
maxvit_t_pt
swin_v2_t_st
swin_v2_s_st
swin_v2_t_pt
swin_v2_s_pt
vit_b_16_st
vit_b_32_st
vit_b_16_pt
vit_b_32_pt

Appendix of table 7:






val_losses
Metric Value
Epoch (index)
Model
mobilenet_v3_small_st
mobilenet_v3_large_st
mobilenet_v3_small_pt
mobilenet_v3_large_pt
efficientnet_v2_s_st
efficientnet_v2_m_st
efficientnet_v2_s_pt
efficientnet_v2_m_pt
maxvit_t_st
maxvit_t_pt
swin_v2_t_st
swin_v2_s_st
swin_v2_t_pt
swin_v2_s_pt
vit_b_16_st
vit_b_32_st
vit_b_16_pt
vit_b_32_pt


val_f1s
Metric Value
Epoch (index)
Model
mobilenet_v3_small_st
mobilenet_v3_large_st
mobilenet_v3_small_pt
mobilenet_v3_large_pt
efficientnet_v2_s_st
efficientnet_v2_m_st
efficientnet_v2_s_pt
efficientnet_v2_m_pt
maxvit_t_st
maxvit_t_pt
swin_v2_t_st
swin_v2_s_st
swin_v2_t_pt
swin_v2_s_pt
vit_b_16_st
vit_b_32_st
vit_b_16_pt
vit_b_32_pt

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa