الجمهورية الجزَائِرية الديمقرَاطية الشعبية
PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
وزَارة التعليم العَالي و البحث العلمي
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
جَامعة عمَار ثليجي بِالأَغوَاط
UNIVERSITY OF LAGHOUAT

FACULTY OF SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

Field : Mathematics and Computer Science
Department : Computer Sciences
Option : Distributed Networks, Systems, and Applications.

Submitted by: Cheknane Maroua

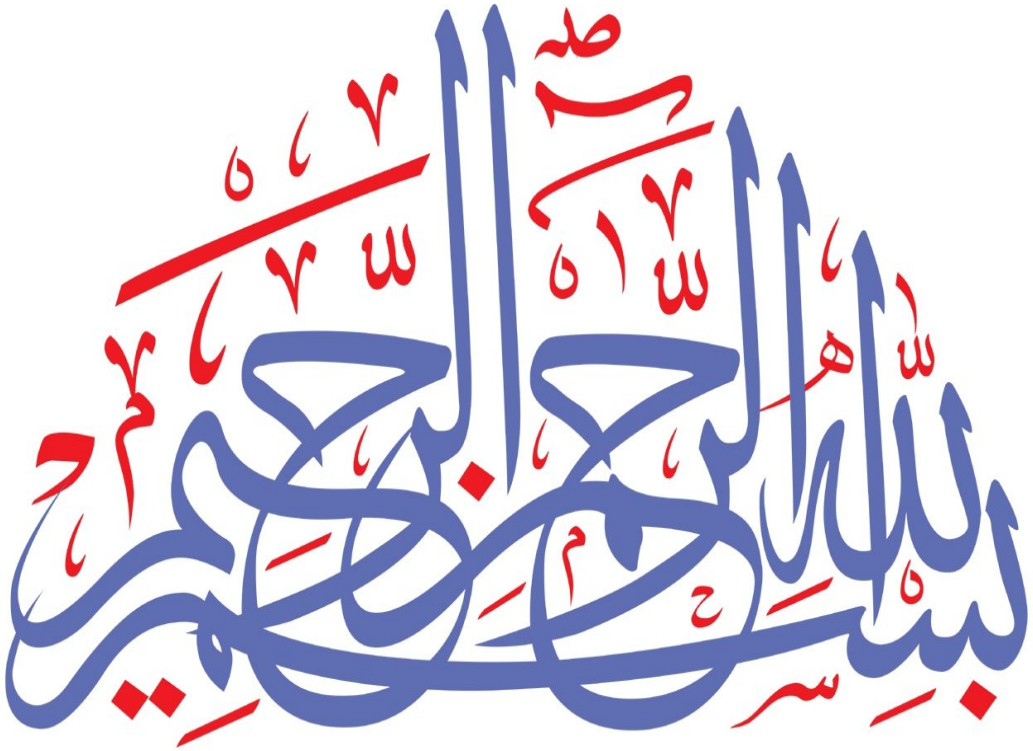
THEME

Artificial Intelligence-Based Fire Detection System

Jury members:

Dr Taher Allaoui	(University of Laghouat)	President
Dr Younes Guellouma	(University of Laghouat)	Examiner
Dr Abdelmadjid Benarfa	(University of Laghouat)	Examiner
Dr Taher Bendouma	(University of Laghouat)	Advisor
Miss Saida Sarra Boudouh	(University of Laghouat)	Co-Advisor

2023


بسم الله الرحمن الرحيم

Acknowledgments

First, I thank Allah for giving me the courage and the will to accomplish this humble job.

I am deeply grateful to my advisor, Dr. Bendouma Taher, and my co-adviser Miss Sarra Saida Boudouh, for their unwavering support and guidance throughout my master's program.

Their expertise and patience have been invaluable to me and have played a crucial role in the success of this thesis.

I am deeply thankful to my friends and family for their love and support during this process. Without their encouragement and motivation, I would not have been able to complete this journey.

I also thank all members of the jury for giving us the honor of evaluating this modest work.

And do not forget my gratitude to all the professors of the Computer Science Department and every professor pass in my academic career, and thanks to everyone who contributed to the development of this work from near or far.

Dedications

First and foremost, I would like to praise Allah the Almighty, the Most Gracious, and the Most Merciful for His blessing given to me during my study and in completing this thesis.

May Allah's blessing goes to His final Prophet Muhammad (peace be upon him), his family, and his companions.

I dedicate my dissertation work to my family. A special feeling of gratitude to my loving parents, whose words of encouragement and prayers have led me to this moment. Both of you have been my best cheerleaders.

You have taught me to be unique, determined, to believe in myself, and to always persevere.

To my loving sisters Amina and Naima. To my precious brothers Moussa, Yacine, and Khalil, thank you for your unwavering love and support. I am truly thankful and honored to have you as my siblings. You are all very special and dear to my heart.

To my sisters-in-law Safia and Meriem. I will always appreciate all they have done.

A very special and big thanks to Sarra for the many hours of helping me and being with me in every step, for encouraging me, and for being my friend.

I dedicate this work to my best friend Ahlam. I'm so grateful to have you as my friend.

And to my wonderful nephews and nieces for cheering me up.

ملخص

تتسبب الحرائق في أضرار كبيرة ، وغالبا ما تكون لها أثار تدميرية جسيمة على البيئة والمحيط. أكثر الطرق فعالية للحد من الأضرار هو الاكتشاف المبكر للحرائق قبل أن تتسع رقعتها. يبحث هذا العمل في قدرة التعلم العميق على تحديد وتمييز الحريق، وكذلك الكشف المبكر لها من خلال تطبيق اكتشاف الأشياء على تدفق الصورة. على مدى السنوات الماضية، تقدم اكتشاف الأشياء بشكل كبير من حيث السرعة والدقة.
في هذا العمل، اقترحنا حلاً يعتمد على التعلم العميق للتعامل مع مثل هذه الظواهر من جمع مجموعات البيانات المختلفة إلى تدريبهم باستخدام الكاشف أحادي المرحلة YOLOv8 و YOLOv5 والكاشف ذي المرحلتين Faster RCNN مع VGG16 / 19 و Xception و Inception و MobileNet ، ونموذجنا الهجين المقترح Xception-VGG19 الذي يتمثل في ضم كل من VGG19 و Xception كأعمدة أساسية. قمنا بتجميع 6 مجموعات بيانات مختلفة من النار والدخان. كانت النتائج التي تم الحصول عليها مرضية باستخدام YOLOv8 ، مع D4 الذي يحتوي على صور دخان فقط بنسبة 99٪(mAP@0.5) تليها D6 بنسبة 93٪ ، و D3 بنسبة 92٪ ، و D5 بنسبة 70٪. مع 43 D2 YOLOv5٪ و 34 D1٪. الكشف على مرحلتين الأفضل هو Xception-VGG19 بنسبة 44٪ يليه VGG16 بنسبة 40٪ ثم VGG19 بنسبة 35٪ ثم MobileNet بنسبة 33٪ ثم Xception بنسبة 23٪. اقترحنا أيضًا سيناريو محاكاة باستخدام الطائرات بدون طيار لإطفاءالنيران بمجرد اكتشافها بواسطة الكاميرات كامتداد لعملنا.

الكلمات المفتاحية:

YOLO, Faster Rcnn, التعلم العميق, شبكة عصبية الالتفافية, الطائرات بدون Mobilenet , Xception, Inception, VGG19/16 , طيار, كشف الكائنات

Abstract

Fires cause great damage when they burst, and often have great destructive effects on the environment and surroundings. The most effective way to limit the damage is the early detection of fire before it spreads.

This work investigates the ability of Deep Learning to identify and distinguish fire, as well as reduce detection time, by applying object detection on a video or image stream. Over the previous years, object detection has advanced gradually in terms of speed and accuracy. In this work, we proposed a solution based on Deep Learning to deal with such phenomena from collecting various datasets to training them using the one-stage detector YOLOv8 and YOLOv5 and the two-stage detector Faster RCNN with VGG16/19, Xception, Inceptionv3, MobileNet, and our proposed hybrid model Xception-VGG19 which is a concatenation of both VGG19 and Xception as backbones. We gathered 6 different datasets of fire and smoke. The obtained results were satisfying using YOLOv8, with D4 which contains smoke-only images with 99% mAP@0.5 followed by D6 with 93%, D3 with 92%, and D5 with 70%. With YOLOv5 D2 43% and D1 34%. Moving forward to the two-stage detection the best outcomes were obtained by Xception-VGG19 with 44% followed by Inceptionv3 with 43%, VGG16 with 40%, VGG19 with 35%, MobileNet with 33%, and Xception with 23%. We also proposed a simulation scenario using UAVS to take down fire once detected by cameras as an extension of our work.

Keywords : **AI, ML, DL, TL, Object detection, CNN, YOLO, Faster Rcnn, VGG19/16, Xception, Inception, Mobilenet, UAVs**

Résumé

Les incendies causent de gros dégâts lorsqu'ils s'enflamment et ont souvent de grands effets destructeurs sur l'environnement et les alentours. Le moyen le plus efficace de réduire les dommages est la détection précoce des incendies avant qu'ils ne se propagent.

Ce travail étudie la capacité de l'apprentissage profond à identifier et caractériser le feu, ainsi que sa détection précoce en appliquant la détection d'objet à un flux d'images. Au cours des dernières années, la détection d'objets a considérablement progressé en termes de vitesse et de précision. Dans ce travail, nous avons proposé une solution basée sur l'apprentissage en profondeur pour gérer de tels phénomènes, de la collecte de différents ensembles de données à leur apprentissage à l'aide du détecteur à un étage YOLOv8 et YOLOv5, du détecteur à deux étages Faster RCNN avec VGG16/19, Xception, Inceptionv3 et MobileNet, et notre modèle hybride proposé Xeption-VGG19 qui est une série de VGG19 et Xception comme épines dorsales. Nous avons collecté 6 ensembles de données différents sur les incendies et la fumée. Les résultats obtenus avec YOLOv8 ont été satisfaisants, D4 n'ayant que des images de fumée à 99% suivi de D6 à 93%, D3 à 92% et D5 à 70%. avec YOLOv5 D2 43% et D1 34%. La meilleure détection en deux étapes est Xception-VGG19 à 44%, suivie de Inceptionv3 à 43%, VGG16 à 40%, VGG19 à 35%, MobileNet à 33% et Xception à 23%. Nous avons également proposé un scénario simulé utilisant des drones pour abattre des flammes une fois détectées par des caméras dans le prolongement de notre travail.

Mots clés : **apprentissage profond,apprentissage par transfert,intelligence artificielle,détection d'objets,NN, YOLO, Faster Rcnn, VGG19/16, Xception, Inception, Mobilenet,UAVs: Véhicules Aériens Sans Pilote (VAPS)**

Contents

List of Figures

List of Tables

1 Introduction

Contents

1.1 Context

When a fire breaks out, there is little time to get safe. In a few moments, a fire can spread out and sometimes get bigger by doubling or tripling in size. A fire can burn out of control in less than half a minute, filling the area with heat and thick, toxic smoke. The heat emitted by a fire is deadly. Breathing in super-heated air can do damage to the lungs. The toxic chemicals, thick smoke, and lack of oxygen is the reason that kills the majority of people who die in fires. Therefore, early fire detection and alarm with high sensitivity and accuracy are essential to reduce fire losses. There are various sorts of detectors, each with its own set of advantages and disadvantages. The maximum appropriate ceiling height for a point smoke detector, carbon monoxide detector, and aspirating detection system is 10.5 meters. The maximum ceiling height for optical beam smoke detectors is 25 meters, and 40 meters if equipped in extra sensitive mode. However, these fire detection technologies, are not suitable for large spaces, complex buildings, or spaces with many disturbances. Due to the limitations of the above detection technologies, missed detections, false alarms, detection delays, and other problems often occur, making it even more difficult to achieve early fire warnings. Thus a deep learning-based model associated with the above detectors can reduce the damage and increase the fire's survival rate. The primary challenge for computer vision fire detection systems is caused by many natural events, such as sunlight reflecting on buildings, car lights at night, and working with different lighting conditions, which can cause a false alarm. Significant research, methods, and realizations have been made to detect or, at the very least, achieve a rather good accuracy rate. Our goal in this work is to train a neural network to detect fire and smoke from a camera in real-time using both one-stage detectors such as YOLOv8n and YOLOv5n and two stages detectors such as Faster RCNN with different backbones architectures such as VGG16/19, Xception, Inceptionv3 and MobileNet when the smoke detectors and fire alarms are less efficient. This AI might potentially function as an additional layer of protection integrated with a smoke detector for increased precision in circumstances. In addition to detecting fire,

we made a scenario to fight such phenomena using UAVs in forests or wide areas. once the fire is detected via cameras, the nearest UAV with a sufficient battery and water tank will be called to extinguish the fire, however, if the fire can not be extinguished using one UAV then multiple UAVS shall be called, if even with the assistance of those UAVs the fire is still present, then the last resort is to call the fire department.

1.2 Organization of the thesis

In the first chapter, we will give an overview of Artificial Intelligence (AI), Machine Learning (ML), and Deep Learning (DL), as well as explain the ideas of neural networks and their types, including how object identification works and its two types of detectors and their metrics.

Next in the second chapter, we attempted to concentrate on recent related work created in the same context as our study.,and provide a classification of these studies. Then we compared them.

The third chapter is devoted to explaining our approach, the techniques, and the CNN models we used. Then finally we discussed the obtained results as well as we proposed a simulation scenario using UAVs as an extension of our work

2

Background

Contents

2.1 Introduction

In these past decades, artificial intelligence has been increasingly becoming a dominant branch of computer science in various applications, it is being applied in almost every industry from self-driving cars to being used in finances, health care, stock management, gaming, and many other fields. In this chapter, we are going to define AI and mention a brief history of its appearance, its sub-fields, ML types, and DL.

2.2 Artificial intelligence :

It is well known that the phrase "Artificial Intelligence (AI)" means different things to different people. Defined by John McCarthy AI is the science and engineering of making intelligent

machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable [12].
Today, many AI researchers define AI as the study of intelligent agents. For example, Stuart Russell and Peter Norvig said that artificial intelligence means an intelligent agent and an agent means a software system that perceives its environment through sensors and acts upon that environment through actuators [13].
Roger C.Shank said that AI is the exercise in the search for the proper formalisms to use in representing knowledge[14]. He also discussed whether AI should be considered as software engineering to which he said that it is difficult to determine .he also questioned if AI is a psychology, and if it is linguistic to which he concluded that the answer to the question "What is AI" does not have one single answer. Wang gave 5 definitions of AI the first was structure-AI which mainly contributes to the study of the human brain, then behavior-AI that mainly contributes to the study of human psychology, capability-AI which mainly contributes to various application domains, by solving practical problems there, also function-AI that mainly contributes to computer science, by producing new software (sometimes also hardware) that can carry out a certain type of computation, and lastly principle-AI that mainly contributes to the study of information processing in different situations, by exploring the implications of different assumptions. [2].

2.2.1 History of AI and how it has progressed over the years:

After world war two, a number of people independently started to work on intelligent machines. The concept of AI had been known since the 1950s. Alan Turing In 1950 published his seminal article "Computing Machinery and Intelligence"[15]. where he described how to create intelligent machines and in particular how to test their intelligence. This Turing Test is still considered today as a benchmark to identify the intelligence of an artificial system: if a human is interacting with another human and a machine and is unable to distinguish the machine from the human, then the machine is said to be intelligent. The first time in 1956 the term "AI" was used in Dartmouth College in the US by Marvin Minsky and John McCarthy (a computer scientist at Stanford).they hosted a workshop founded by the Rockefeller Foundation. An early example is the famous ELIZA computer program, created between 1964 and 1966 by Joseph Weizenbaum at MIT. ELIZA was a natural language processing tool able to simulate a conversation with a human and one of the first programs capable of attempting to pass the aforementioned Turing Test [16]. the Japanese government began to heavily fund AI research in the 1980s, to which the U.S. DARPA responded with a funding increase as well. In 1997 IBM's deep blue chess playing program was able to beat the world champion at the time, it was able to process and determine the next move using a method called tree search[17]. The creation of research on Artificial Neural Networks was in the 1940s when a Canadian psychologist developed a theory known as Hebbian learning [18]that was similar to the neurons in the brain. Jumping forward to 2015 when the artificial neural networks made a comeback as Deep learning through the program AlphaGo developed by Google which was able to beat the world champion in the board game named "Go". Today artificial neural networks and Deep Learning form the basis of most applications we know under the label of AI. They are the basis of image recognition algorithms used by Facebook, and speech recognition algorithms that fuel smart speakers and self-driving cars[16].

2.2.2 How AI works :

AI systems work by combining large amounts of data records and extracting that data into compatible class objects technologies with fast processing algorithms, allowing the software to learn automatically from features or patterns in the data. AI focuses on three cognitive skills

- **Learning processes:** This area of AI programming focuses on collecting data and developing algorithms to transform the data into usable knowledge. These algorithms give computing devices detailed instructions on how to carry out a task.
- **Reasoning processes:**This area of AI programming is concerned with selecting the appropriate algorithm to achieve a desired result.
- **Self-correction processes:** This feature of AI programming is to continuously improve algorithms and make sure they deliver the most precise results.

2.2.3 Key components of AI:

There are many different fields within AI, each one having its specific algorithms. Therefore, it's essential to know that AI is not a single field but a combination of various fields. AI can be broken down into two major fields, Machine Learning (ML) and Neural Networks (NN). Both are sub-fields under Artificial Intelligence, and each one has its methods and algorithms to help solve problems. [19]

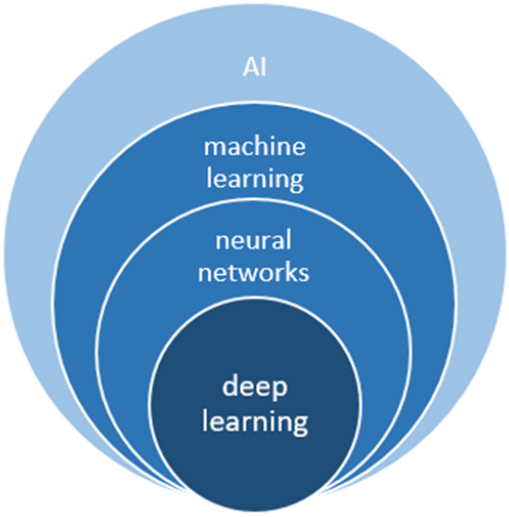


Figure 2.1: Components of AI.

2.3 Machine learning :

ML is the practice of programming computers to learn from data. In the following example, the program will easily be able to determine whether a given text (email) is spam or not spam. In ML, data is referred to as called training sets, With ML, as an AI application, comes the importance of focusing on creating systems and models that can access Big Data

sets available today, and the system automatically modifies, learns, and enhance predictability and performance through its settings to boost and enhance the experience and ensure the effectiveness of predictive analyses [20] [21] Besides, all ML are elements of AI, but not all AI is considered ML, which make ML a sub-branch of the AI domain. ML refers to the segment of computer science whose purpose is to build and operate existing algorithms to establish generalized models that give accurate patterns and predictions. Using historical data, ML algorithms are based on statistical and mathematical models to help machines mimic human behavior[20]

2.3.1 Types of machine learning:

Types of Machine Learning can be classified into three categories which are based on different algorithms to solve problems. The type of algorithm used depends on the type of problem that you want to solve, the number of variables, and the type of model that gives it would suit best. The following figure shows a hierarchy of ML :

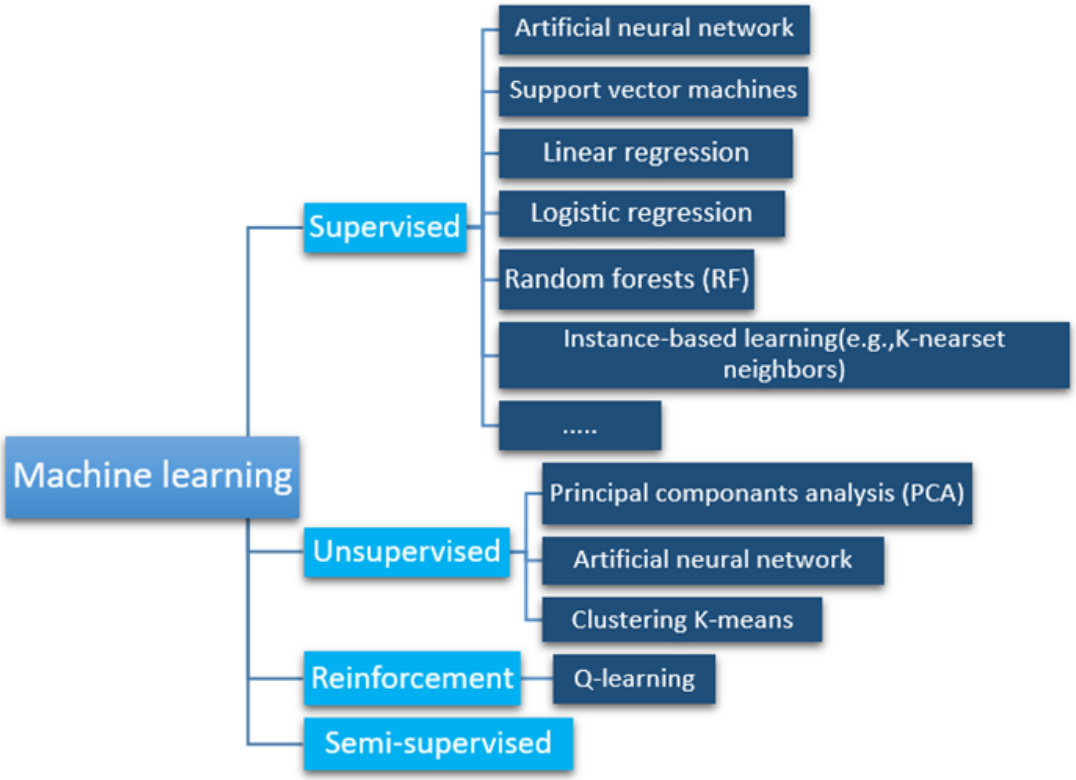


Figure 2.2: Types of ML.

2.3.1.1 Supervised learning:

Supervised learning entails learning a mapping between a set of input variables X and an output variable Y and applying this mapping to predict the outputs for unseen data. The name invokes the idea of a 'supervisor' that instructs the learning system on the labels to associate with training examples. Typically these labels are class labels in classification problems. Supervised learning algorithms induce models from these training data and these models can be

used to classify other unlabeled data[22]. Input data can be numeric, alphanumeric values, or images. Output data (variable to be predicted) can be either a Discrete variable or a continuous variable. The data set is usually divided into two sets, "Training Set" and "Test Set". The training set is used to learn the relationship between the input and the output, the test set is used to validate the model by measuring its precision.

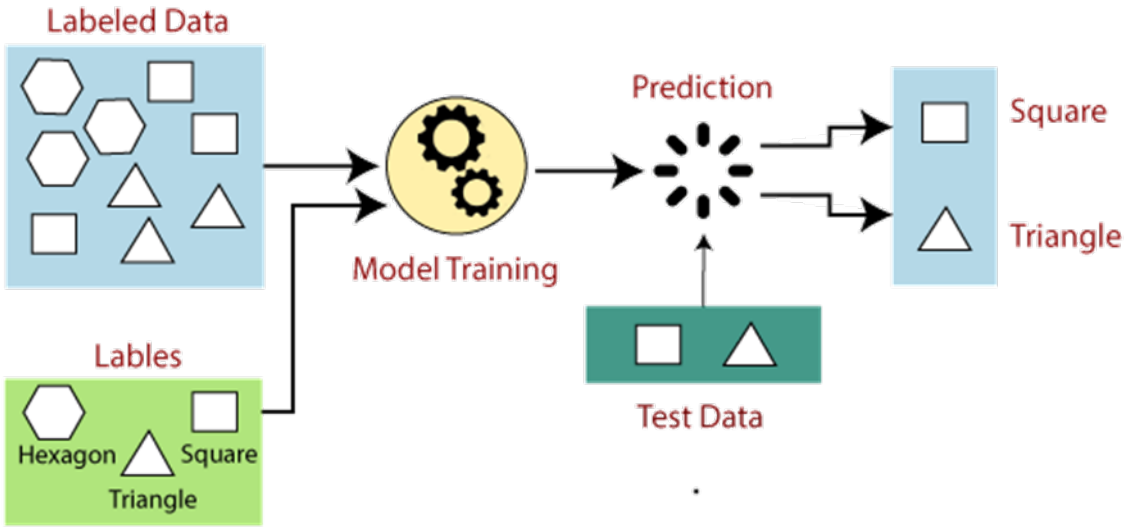


Figure 2.3: supervised learning structure.

- **The purpose of supervised learning:** Supervised learning is mainly used for two purposes:

  1. **classification** When the variable to be predicted takes a discrete value, we speak of a classification problem. When all the classes exceed two classes, we speak of multi-class classification.
  2. **Regression:** When the variable to be predicted is a specific value, we are talking about a regression problem, a regression algorithm makes it possible to find a model (a mathematical function) according to the training data (input data). The calculated model will make it possible to give an estimate on new data not yet seen by the algorithm (which was not part of the training data)

2.3.1.2 Unsupervised learning:

Unsupervised learning involves the computer learning on its own while using unlabeled data. In the unlabeled data, the machine looks for patterns and responds. By grouping data along similar features or analyzing datasets for underlying patterns, unsupervised learning is a powerful tool used to gain insight from this data. In contrast, supervised machine learning can be resource intensive because of the need for labeled data.
Unsupervised machine learning is mostly utilized to:

- Cluster datasets based on shared features or data segments.
- Recognize relationships between various data points, such as automated music suggestions.

- Execute preliminary data analysis

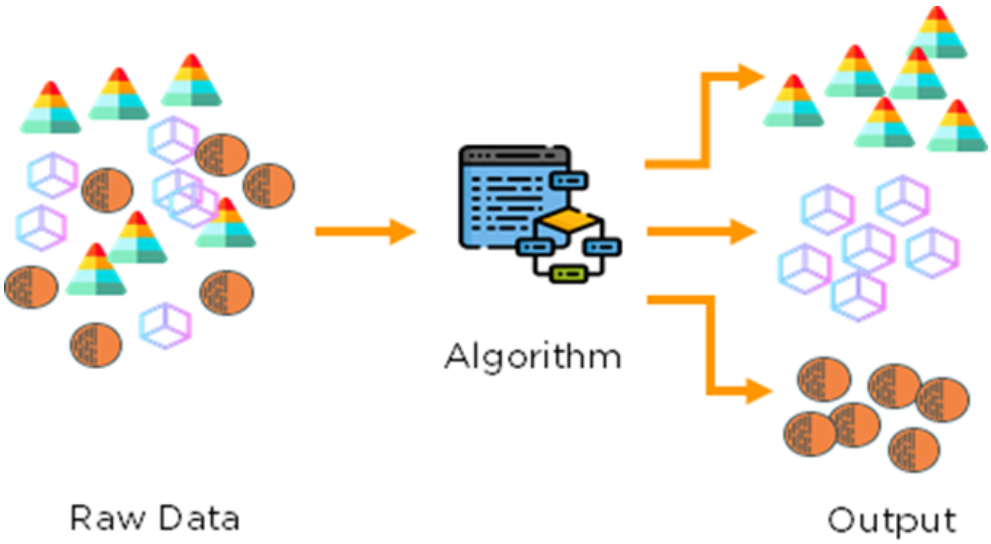


Figure 2.4: Unsupervised learning structure.

Unsupervised learning can be further grouped into types

- **Association:**

  Association is a rule-based machine learning technique to ascertain the likelihood of the co-occurrence of items in a collection.

- **Clustering:** Clustering is a technique for grouping objects into clusters that are similar among themselves and are dissimilar from the objects belonging to another cluster.

2.3.1.3 Difference between supervised and unsupervised learning:

Supervised learning techniques are more powerful than unsupervised techniques because the availability of labeled training data provides clear criteria for model optimization. [22]

Table 2.1: Difference between supervised and unsupervised learning.

Supervised	Unsupervised
all data is labeled	all data is unlabeled
It has feedback mechanism	It has no feedback mechanism
Can be divided into : classification ,regression	Can be divided into: clustering ,association
Real-life applications: Image classification, visual recognition	Real-life applications: Semantic clustering, identifying accident-prone areas

2.3.1.4 Reinforcement learning:

Unlike the other two learning frameworks, which operate using a set of static data, RL works with data from a dynamic or unknown environment. RL has no aim to regroup the data or to label the data, but to find the best sequence of actions that generate the optimal outcome. RL solves this problem by allowing a component called an agent to explore, interact with and learn from the environment.

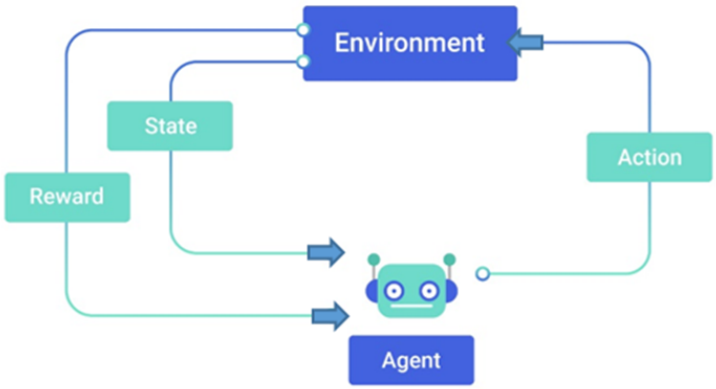


Figure 2.5: Reinforcement learning structure.

2.3.1.5 Semi-supervised learning:

It is a subset of machine learning, it trains its models using a lot of unlabeled data and little labeled data. Refers to methods that attempt to take advantage of unlabeled data for supervised learning (semi-supervised classification), or to incorporate prior information such as class labels, pairwise constraint,s or cluster membership in the context of unsupervised learning (semi-supervised clustering)[23].

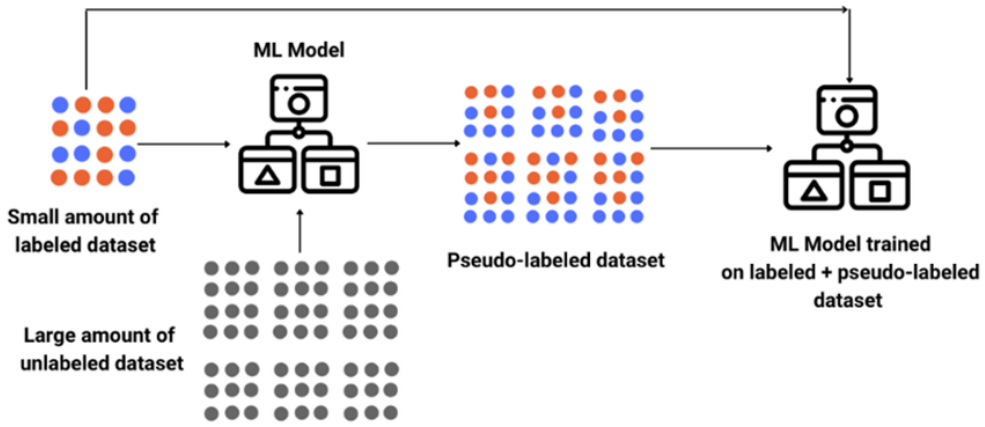


Figure 2.6: Semi-supervised learning structure.

2.4 Neural networks :

An artificial neural network (ANN) can be thought of as a greatly reduced representation of the biological neural network's structure. An ANN consists of interconnected processing units. A processing unit's overall model is composed of a summing part followed by an output part. The summing part receives N input values, weights each value, and computes a weighted sum. The weighted sum is called the activation value. The output part produces a signal from the activation value. Each input's weight sign decides whether it is an excitatory (positive weight) or an inhibitory input (negative weight). Both the input and the output have the potential to be either discrete or continuous data values. In an ANN, a topology is used to connect several processing units[24]. Input, hidden (middle), and output layers are the three types of layers that make up a NN. Hidden layers are used by NNs to process and send data to output layers. As today draws closer, a brand-new age of NN known as DL(Deep Learning) has emerged. The third ascent of NN has started generally in 2005 with the conjunction of several discoveries over a significant time span by late researchers Hinton, Andrew Ng, LeCun, Bengio, and others [25].

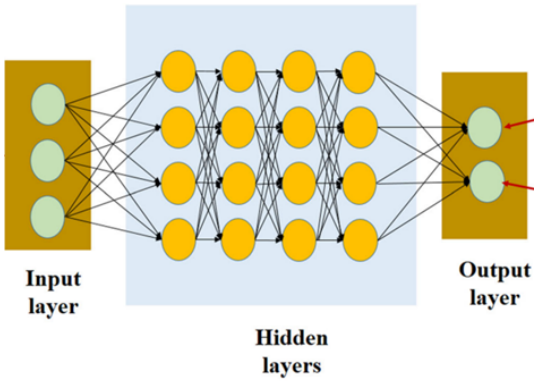


Figure 2.7: Neural network layers.

2.4.1 Types of neural networks:

2.4.1.1 Convolutional Neural Networks (CNN):

CNNs are NNs created specifically to process images and videos.The Convolution Neural Network is made up of four primary layers: convolution layers, pooling layers, fully connected layers, and classification layers; each layer performs a specific function. Layers of a CNN turn the input volume to a collection of neuron activation and provide output, which is an input volume for subsequent layers, eventually leading to completely linked layers, a mapping learned for classification, as seen in the picture below. CNNs have proven to be extremely effective in vision applications such as object identification and other applications such as face recognition, robotics, self-driving automobiles, and so on.[1]

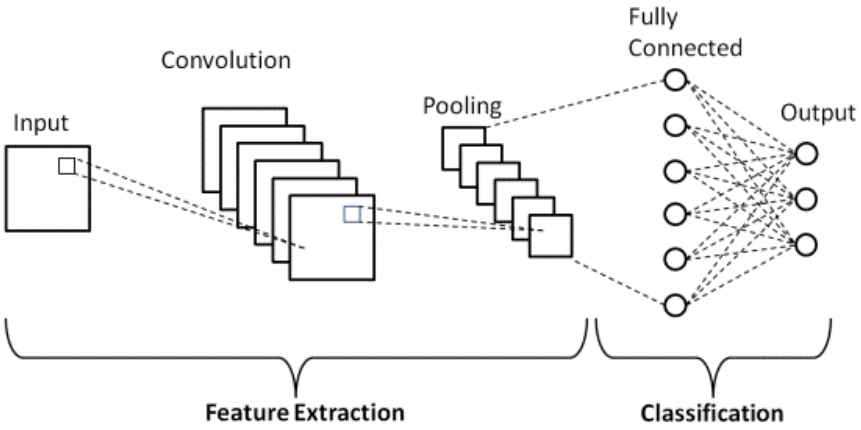


Figure 2.8: CNN layers.[1]

- **Convolutional Layer:**is a set of learnable filters (kernels). The filter sizes may be 3x3 to 5x5 to 7x7. The number of channels in the input is represented by the filter's third dimension [26]. A filter is a collection of multipliers that enables us to extract features that are essential for classifying an image. Several filters may be randomly started and trained to do tasks like identifying an object in an image.
- **Pooling Layer:** After filtering, the amount of the input typically increases, hence CNN frequently uses the pooling layer, also known as sub-sampling or downsampling, to reduce the size of the data. The filter size and strides are the two-layer parameters with the most commonly applied values. Max pooling and Average Pooling are two types of pooling layers where the maximum and average values are taken, respectively.
- **Fully Connected Layer:** is the final layer in The CNN. The dropout regularization approach can be used to reduce overfitting. The output layer, the last fully connected layer, has the same number of neurons as the classes[26].

2.4.1.2 Recurrent Neural Networks

RNNs were created specifically to handle sequential data. Due to the internal feedback loops provided by its architecture, sequential pattern learning may simulate temporal dependencies by creating a memory, they are extensively utilized in Natural Language Processing (NLP), including Neural Machine Translation, language, synthesis, and time series analysis, with applications in many disciplines including physics, medicine, and climatology. RNNs are used in combination with CNNs when the sequential data are images (videos) [27].

2.5 Deep Learning :

Deep learning is a subset of machine learning. It is a neural network with a large number of layers and parameters. Most deep learning methods use neural network architectures. Therefore it is also referred to as deep neural networks [25]. DL technology uses multiple layers to represent the abstractions of data to build computational models. While deep learning takes a long time to train a model due to a large number of parameters, it takes a short amount of time to run during testing as compared to other machine learning algorithms [28].DL approaches have

multiple abstractions of levels by using a non-linear model that transforms the original data into higher abstract levels for decision making. This simplifies finding the solution of complex and non-linear functions [27].Deep Learning has high data requirements yet requires little human intervention to perform successfully .The problem of having huge training datasets can be solved through transfer learning.

2.5.1 Deep Transfer Learning

Transfer learning is considered a valuable tool for problems where the dataset is insignificant. The concept of transfer learning is based on using a trained model with a large dataset in case studies where the dataset is cooperatively smaller. Transfer learning has recently been applied successfully in various fields (including medicine). It eliminates the need for an extensive dataset and shortens the training time that a deep learning model requires when built from scratch [26].

Figure 2.9: Deep transfer learning.[2]

2.6 Difference between machine learning and deep learning

Deep learning has made significant contribution to the recent progress in artificial intelligence. In comparison to traditional machine learning methods such as decision trees and support vector machines, deep learning methods have achieved substantial improvement in various prediction tasks. However, deep neural networks (DNNs) are comparably weak in explaining their inference processes and final results [29] Algorithms for machine learning include deep learning algorithms. Consequently, it might be wiser to consider what, specifically, in machine learning makes deep learning unique. The structure of the ANN algorithm, the reduced need for human interaction, and the higher data requirements are the answers. Deep Learning is based on an artificial neural network, as opposed to typical Machine Learning methods, which have a structure that is more straightforward, such as linear regression or a decision tree. Deep Learning algorithms want far less input from humans. With a conventional machine learning technique, a software engineer would manually select features and a classifier to classify photos, check to see if the result is what was expected, and modify the algorithm if necessary. But as a deep learning algorithm, the features are automatically extracted, and the algorithm learns from its own errors.

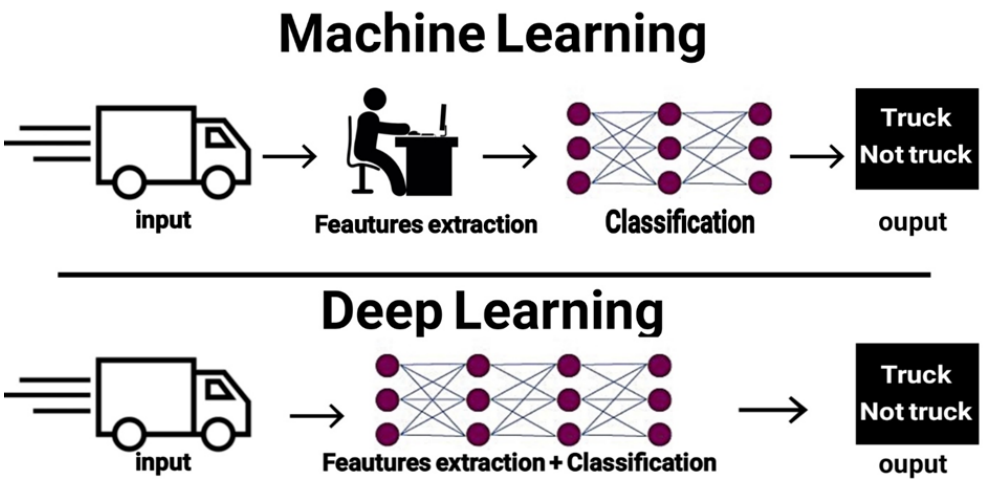


Figure 2.10: Difference between ML and DL.

2.7 Object detection

Object detection is a computer vision task that detects instances of objects of a specific class (e.g., 'car' 'truck' etc.) in images. Object detection algorithms are broadly classified into two categories.

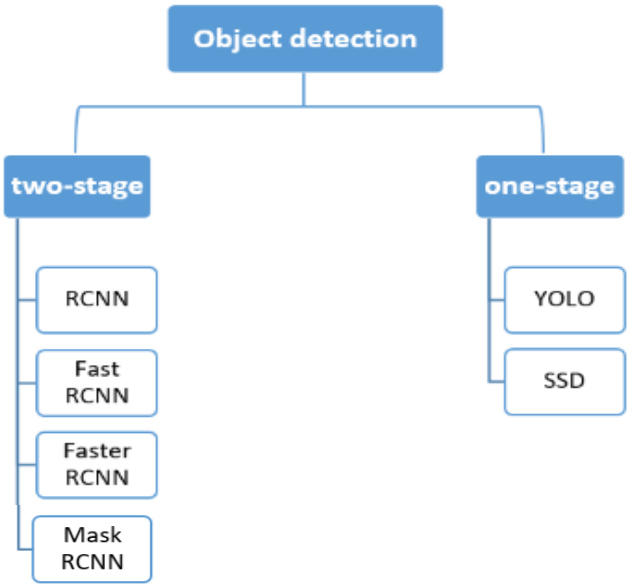


Figure 2.11: object detection categories

2.7.1 One-stage detectors

One-stage Object Detection models refer to a class of object detection models that makes predictions about the presence and placement of objects in an image using a single pass of the input image. from those models, we mention :
YOLO: For the first time, yolo was presented as a single-stage method for object detection that converts raw image pixels to bounding box coordinates and class probabilities and can be optimized end-to-end directly. This enabled the detector to predict boxes directly in a single feed-forward run without recycling any neural network components or creating proposals of any type, resulting in a faster detector. They began by splitting the image into S S grids with B boundary boxes per grid. Each cell holding the center of an object instance is in charge of detecting that object. Each enclosing box forecasts four coordinates as well as objectness and class probabilities. The object detection problem was reframed as a regression problem as a result of this.

To have a receptive field cover that covers the whole image they included a fully connected layer in their design towards the end of the network.

The various advantages of the YOLO strategy are that it is extremely fast, with 45 to 150 frames per second. It sees the entire image as opposed to region proposal-based strategies which is helpful for encoding contextual information and it learns generalizable representations of objects.[30]

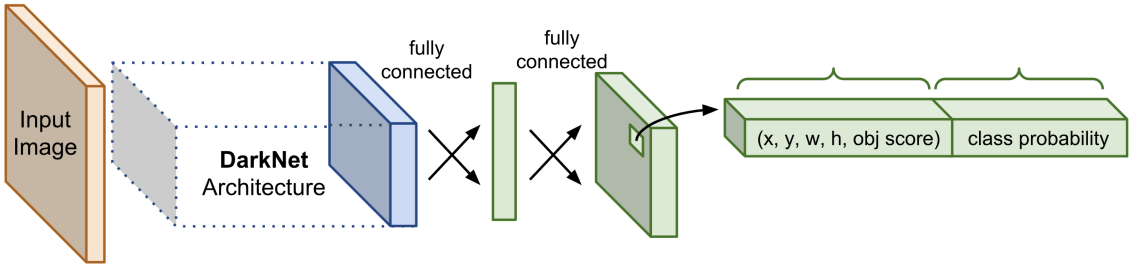


Figure 2.12: YOLO architecture

2.7.2 Two-stage detectors

The process of detecting objects can be split into two parts, proposing regions and classifying and regressing bounding boxes. The purpose of the proposal generator is to present the classifier with class-agnostic rectangular boxes which try to locate the ground-truth instances. The classifier, then, tries to assign a class to each of the proposals and further fine-tune the coordinates of the boxes.[30]

From its most known approaches, we mention :
R-CNN : The model first uses the Selective Search to extract approximately 2000 region proposals of each image to be detected. Then the size of each extracted proposals is uniformly scaled to a fixed-length feature vector and these extracted image features are input into the SVM classifier for classification. Finally, a linear regression model is trained to perform the regression operation of the bounding box. Compared with the traditional detection method, the accuracy of the R-CNN does improve a lot, but the amount of calculation is very large, and the calculation efficiency is too low. Secondly, directly scaling the region proposal to a fixed-length feature vector may cause object distortion.[1]

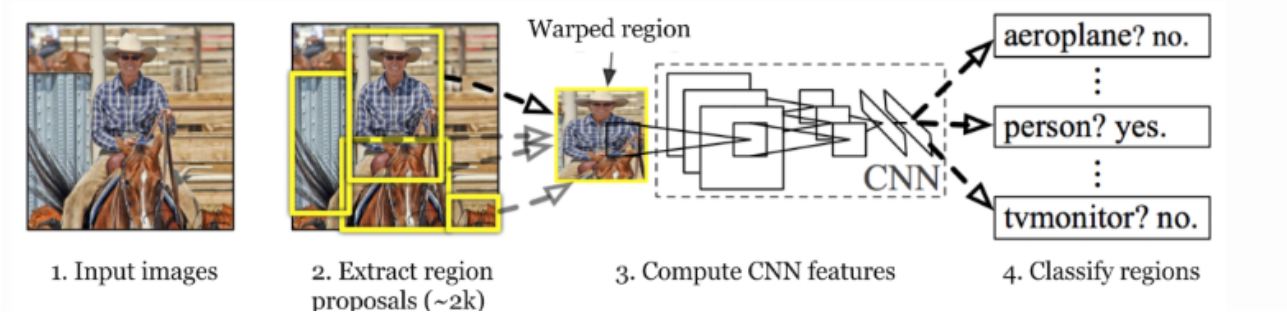


Figure 2.13: rcnn.

Fast RCNN : It takes the whole image and region proposals as input in its CNN architecture in one forward propagation. It also combines different parts of the architecture. Compared with R-CNN, Fast R-CNN has made changes. First, it replaced the SVM used in R-CNN with a softmax function for classification. Secondly, the last softmax classification layer of the CNN network is replaced by two parallel fully connected layers.[1]

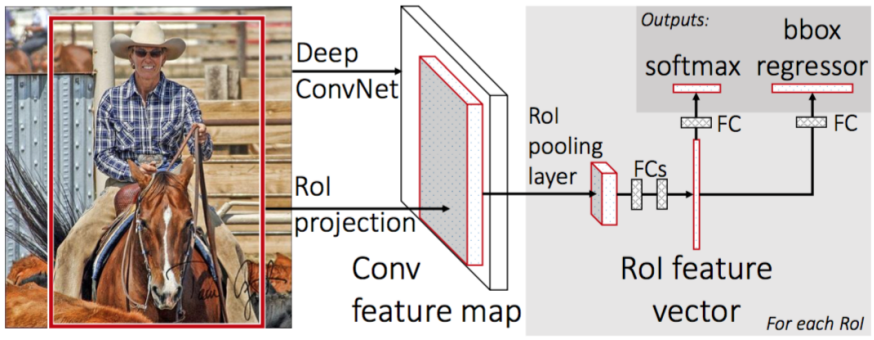


Figure 2.14: fast rcnn.

Faster RCNN : To create region proposals, the model replaces the old selected Search method with region proposal networks. The model is separated into two modules: a fully convolutional neural network used to create all region recommendations, and the Fast R-CNN detection algorithm. These two modules share a set of convolutional layers. The input image is sent through the CNN network until it reaches the final Shared convolutional layer. The feature map for the RPN network's input is obtained on the one hand, and the picture is transmitted forward to the appropriate convolutional layer to build a higher-dimensional feature map on the other. .[1]

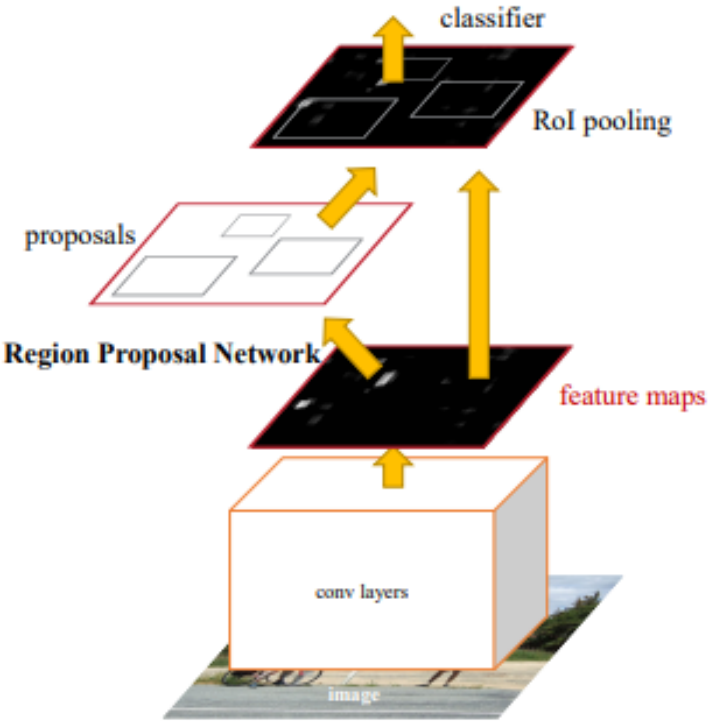


Figure 2.15: faster rcnn.

RPN works with anchor boxes, according to the figure below, the feature map of the last shared convolution layer is passed through a rectangular sliding window of size n*n, and k region proposals are created for each window. Each proposal is parameterized based on a reference box known as an anchor box. **anchor box**. The two parameters of the anchor boxes are:

1. scale
2. aspect ratio

In general, there are three scales and three aspect ratios, for a total of k=9 anchor boxes, though k may be less than nine. In other words, k regions are generated from each area proposal, with each k region differing in either scale or aspect ratio. .Some of the anchor variations are shown in the next figure[3].

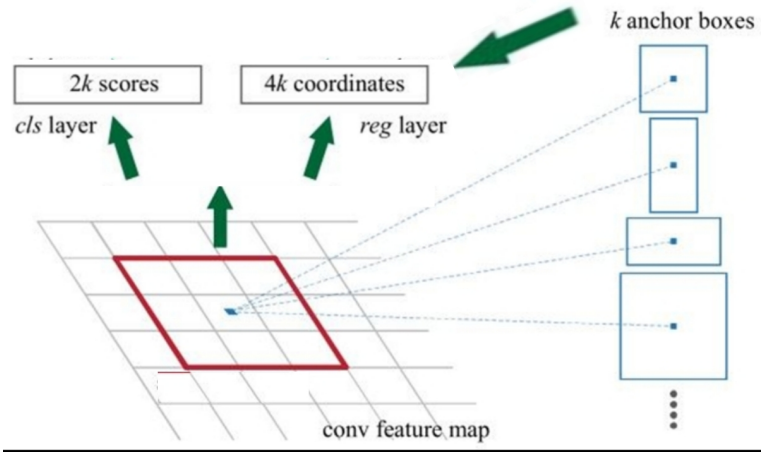


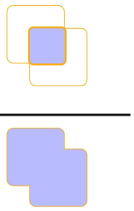
Figure 2.16: anchor boxes[3].

2.7.3 Object detection models performance evaluation metrics

- **True Positive (TP)** : The model made a correct detection .
- **False Positive (FP)** : The model made an incorrect detection.
- **False Negative (FN)**:the object detector missed a Ground-truth(not detected).
- **True Negative (TN)**: This is the background region correctly not detected by the model. This metric is not used in object detection because such regions are not explicitly annotated when preparing the annotations.

Intersection over Union (IoU) It is a metric that measures the amount of overlap between the detected object and the ground truth object. This is a good metric to make sure that the detected object is a good match for the ground truth object.[31] we calculate the union of the two bounding boxes and the intersection between them, with two rectangles, their coordinates are (x1, y1, x2, y2) and (x3, y3, x4, y4),the IoU is :

$$IoU = \frac{\text{Area of Overlap}}{\text{Area of Union}} =$$



$$Areaofintersection = (min(x2, x4) - max(x1, x3)) * (min(y2, y4) - max(y1, y3))[31] \quad (2.1)$$

$$Areaofunion = (x2 - x1) * (y2 - y1) + (x4 - x3) * (y4 - y3) - Areaofintersection[31] \quad (2.2)$$

IoU values vary from 0 to 1, with 0 indicating no overlap and 1 indicating perfect prediction and overlap.

The IoU measure is useful because of thresholding, which means we need a threshold**a** to verify whether the detection is correct.

Figure 2.17: Intersection Over Union.

Considering the IoU threshold,**a** = 0.5.
TP : IoU $\geq$ 0.5
FP : IoU $<$ 0.5

Precision :
Is the model's accuracy in identifying only relevant objects.

Recall :
It evaluates the model's capacity to discover all ground truths.

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{TP}{\text{all detections}}$$
$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{TP}{\text{all ground-truths}}$$

Precision x Recall curve :
Both confidence scoring and IoU use the threshold. Raising the confidence score threshold causes the model to miss more objects (more FNs, and hence poor recall and high precision), whereas lowering it causes it to miss more objects (low precision and high recall). This implies that we must devise some precision versus recall trade-offs.

The precision-recall (PR) curve illustrates precision and recall at various confidence levels. Even when confidence levels fluctuate, a strong model's precision and recall remain high.

Average precision
AP@**a** denotes the Area Under the Precision-Recall Curve (AUC-PR) at the IoU threshold. It is foformallyefined as follows:

$$AP@\alpha = \int_0^1 p(r)\, dr$$

Mean Average Precision (mAP) :

more advanced metric that takes into account the precision of the detector in each class. This is an excellent metric for obtaining a more complete picture of how effectively the detector is operating. It is, however, more difficult to comprehend than AP. The mAP measure is typically presented at various confidence levels (e.g., 0.5, 0.95) [31]. The formula for mean average precision (mAP) is:

$$\text{mAP@}\alpha = \frac{1}{n} \sum_{i=1}^{n} \text{AP}_i \quad \text{for n classes.}$$

The mean Average Precision (mAP) is the average of AP values over all classes, the higher the mAP the better the model's detection is.

2.8 Conclusion

In this chapter we saw the different definitions of AI, how it works its components, and a brief history of its creation. we also saw machine learning and its types, neural networks and what deep learning and deep transfer learning, and the difference between ML and DL, in another part we went through the object detection task, how it works, and its metrics. In the next chapter, we will see how these approaches are used in fire detection systems.

3

: Related Work

Contents

3.1 Introduction

Most of fire detection systems use intelligent video surveillance which is one of the fastest-growing fields of computer vision research for automatic detection and it has proven its efficiency toward this hazard. This chapter presents multiple detection systems that are made to reduce the risks of such phenomena. Current systems are either combining smoke detection as it is an initial sign of a fire with flame detection or just the flame detection by itself.

3.2 Research Problem and Research Question:

Algeria has lost nearly 50,000 hectares to wildfires as of October 20223.1 and between August 17-21, in forest and urban areas in the east of Algeria; at least 43 people were killed and 200 people injured. The year before, the country had experienced some of the worst wildfires in years. The most intense wildfires happened in mid-August when more than 130,000 hectares were burned in a single week [32]. Flame appearance in videos and images is the primary challenge for researchers in computer vision fire detection systems because many natural events, such as sunlight reflecting on buildings, the light of cars at night, and working with different lighting conditions, can trigger a false alarm as shown in the figure below. Another challenge was that some objects have similar colors to fire also if the fire is too small for the system to notice.

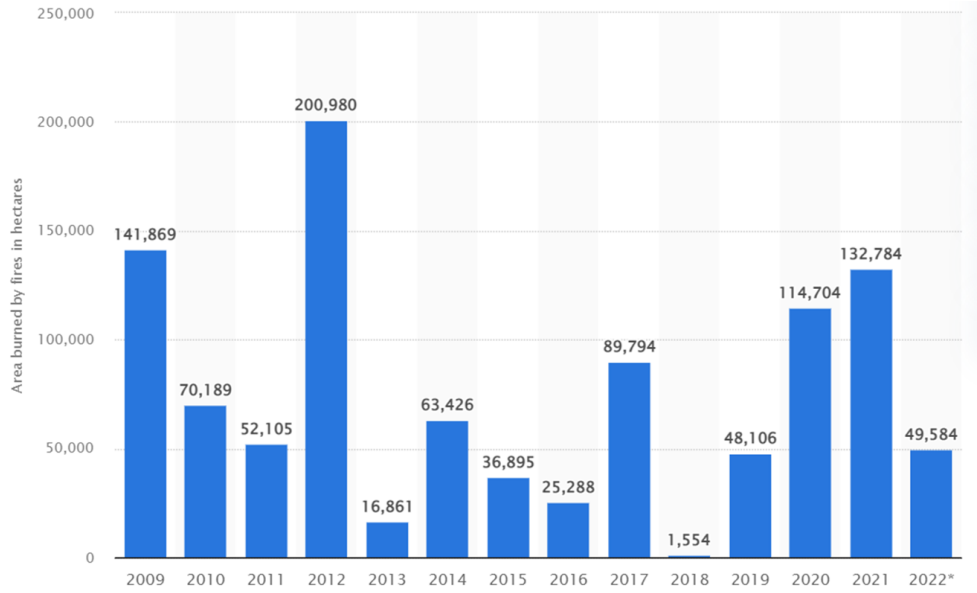


Figure 3.1: Area burned by wildfires in Algeria from 2009 to 2022(in hectares)

Figure 3.2: false detection problems

Research questions:

- How to detect fire while it's still small and has not grown too large?
- How to increase the accuracy of the system and take into consideration false alarms?

3.3 Overview on fire detection systems :

These technologies detect fire using computer vision and artificial intelligence. While object localization entails drawing a bounding box around one or more objects in an image, image classification entails labeling a picture with a class. The more challenging object detection task combines these two tasks, creates bounding boxes around each object of interest in the image, and labels each thing according to its class. The term "object recognition" refers to all of these issues collectively.

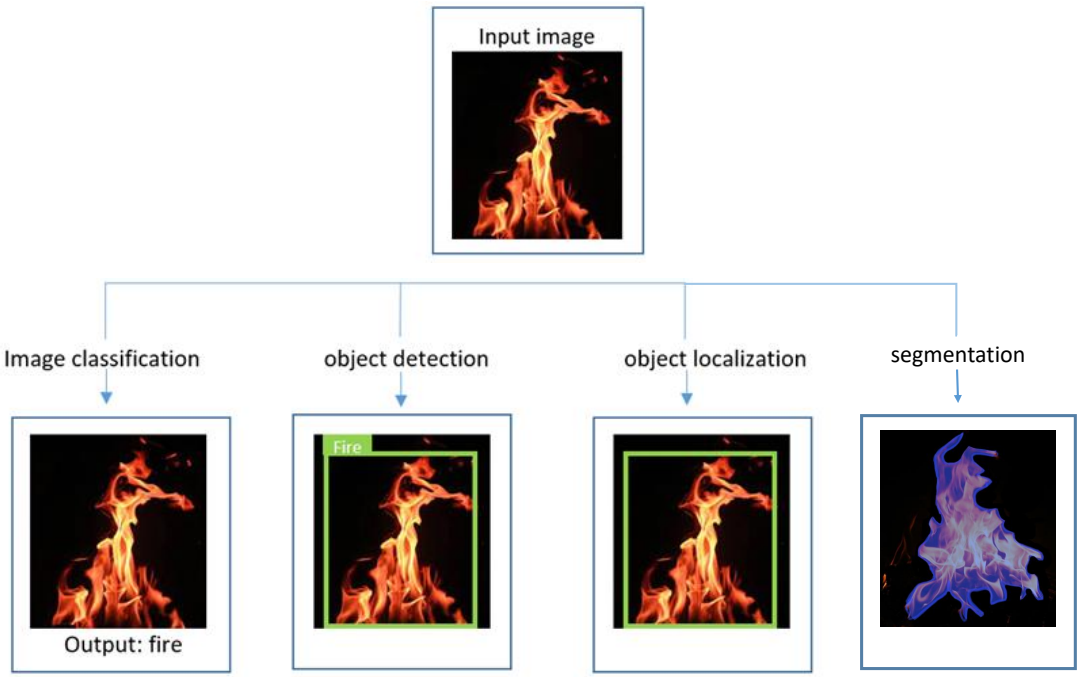


Figure 3.3: difference between object detection,object localization and image classification

Fire detection systems can be summered in three categories smoke detection,flame detection and both smoke and flame detection.

3.3.1 Smoke detection systems:

Smoke appears before flames, so it is a better for early detection of fire, but compared to the flame it is difficult to detect because gray or black color of smoke is common for non-smoke pixels of other objects. So many systems were made to detect smoke we mention:

1. **Khan et al 2019(DeepSmoke: Deep Learning Model for Smoke Detection and Segmentation in Outdoor Environments)[4] :** A CNN based smoke detection and segmentation framework for both hazy and clear environments .for a better accuracy rate this work used a CNN architecture called EfficientNet, for the smoke segmentation it used DeepLav3+(a semantic segmentation architecture)[4]. It can be divided into two modules:

   (a) A module that performs smoke detection using a lightweight CNN model [4]

(b) A module addressing semantic smoke segmentation [4]

Dataset : it contained 3 datasets

- **DS1 :** contained images in four classes "smoke", "non- smoke", "smoke with fog", "non-smoke with fog"[4].

Figure 3.4: example images from DS1 [4].

Table 3.1: DS1 dataset.

	Smoke Smoke with fog	Non-smoke and Non smoke with fog	Total	Training	Validation	Test
DS1	18,532 for each class	17,474 for each class	72,012	20%	30%	50%

- **DS2 :**it contained 7 different videos used for test purpose only

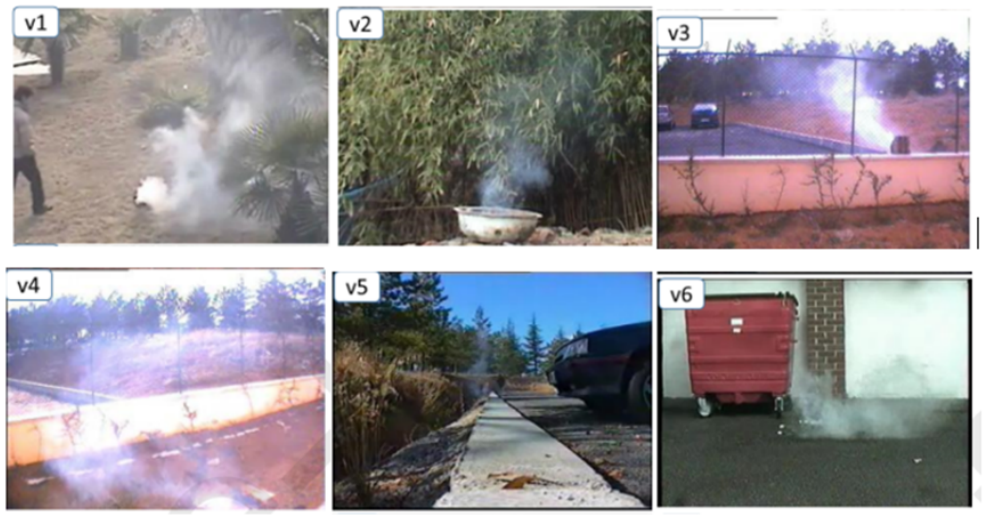


Figure 3.5: example images from videos in DS2 [4].

- **DS3 :**This is their own proprietary dataset and is used for the evaluation of the semantic segmentation module. It comprises 252 images annotated from DS1 by considering only two classes: "smoke" and "smoke with fog". This dataset is divided into 60% and 40% for training and testing, respectively [4]

Figure 3.6: example images in DS3 [4].

Results : the figure below shows the results of this method after the segmentation

Figure 3.7: results of the Deepsmoke method [4].

Table 3.2: Deepsmoke results

	Average MET	Average P	Average R	Average F
Deepsmoke	39,43	0,97	0,98	0,98

2. **Dewangan, A et all 2022(FIgLib and SmokeyNet: Dataset and Deep Learning Model for Real-Time Wildland Fire Smoke Detection)[5] :** This work present the Fire Ignition Library (FIgLib), a publicly available dataset of nearly 25,000 labeled wildfire smoke images as seen from fixed-view cameras. Also introduce SmokeyNet, a novel deep learning architecture using spatiotemporal information from camera imagery for real-time wildfire smoke detection[5].

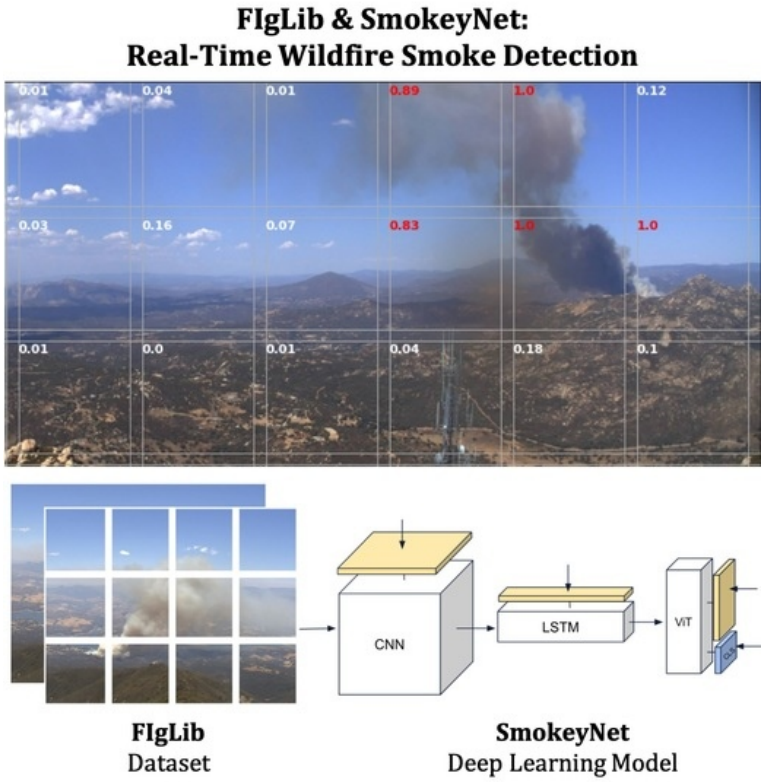


Figure 3.8: Graphicalabstract [5].

Dataset:
The dataset consists of 315 fire sequences from 101 cameras across 30 stations occurring between June 2016 and July 2021. Each sequence typically contains images from 40 min prior to and 40 min following the start of the fire, serving as binary smoke/no-smoke labels for each image, and are spaced approximately 60 s apart for a total of 81 images per fire sequence. However, 114 fires are missing an average of 6.6 images each; missing images are either at the beginning, end or randomly dispersed throughout the sequence. In total, the dataset contains 24,800 high-resolution images. To avoid out-of-distribution sequences, removed fires with black and white images, night fires, and fires with questionable presence of smoke, including one fire with 180 images and no labels, from the dataset (3700 images)[5]

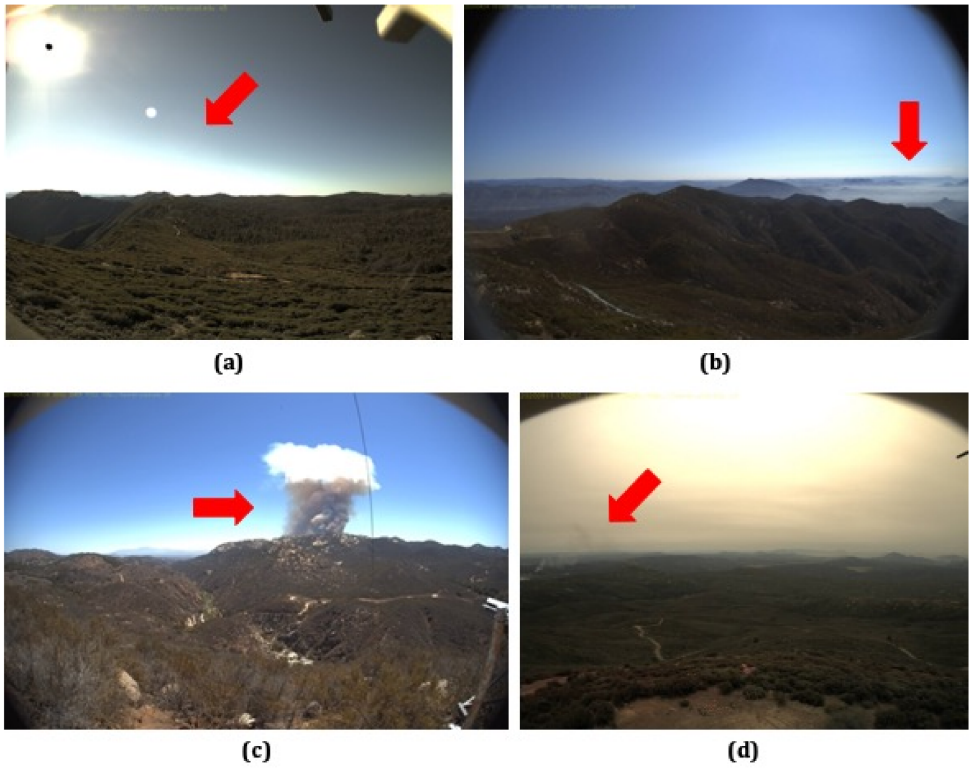


Figure 3.9: Images from the FIgLib dataset: (a) no smoke with strong glare, (b) no smoke with misleading haze, (c) very apparent wildfire smoke, and (d) faint wildfire smoke,[5].

Table 3.3: Dataset

model	#images
train	11.3k
validation	4.9k
test	4.9k
omitted	3.7k
total	24.8k

Results:
The SmokeyNet architecture with a ResNet34 backbone and two frames of input achieves an image accuracy of 83.49% and F1-score of 82.59% while delivering on the objectives of high precision (89.84%), high recall (76.45%), fast performance (51.6ms/image), and low average time to detection (3.12 min)[5].

3.3.2 Flame detection systems :

There are many colors in a flame. In addition to having a dynamic texture and changing shape over time, it also flickers and moves.Flame detection is more accurate than smoke de-

tection, especially at night when the smoke is invisible to the unaided eye.Several systems are designed to detect flame, such as:

1. **Dai P et al 2022 (Multi-Scale Video Flame Detection for Early Fire Warning Based on Deep Learning)[6]:** Based on Yolov3, the parallel convolution structure of Inception is used to obtain multi-size image information. In addition, the receptive field of the convolution kernel is increased with the dilated convolution so that each convolution output contains a range of information to avoid information omission of tiny flames. The model accuracy has improved by introducing a Feature Pyramid Network in the feature extraction stage that has enhanced the feature fusion capability of the model[6].

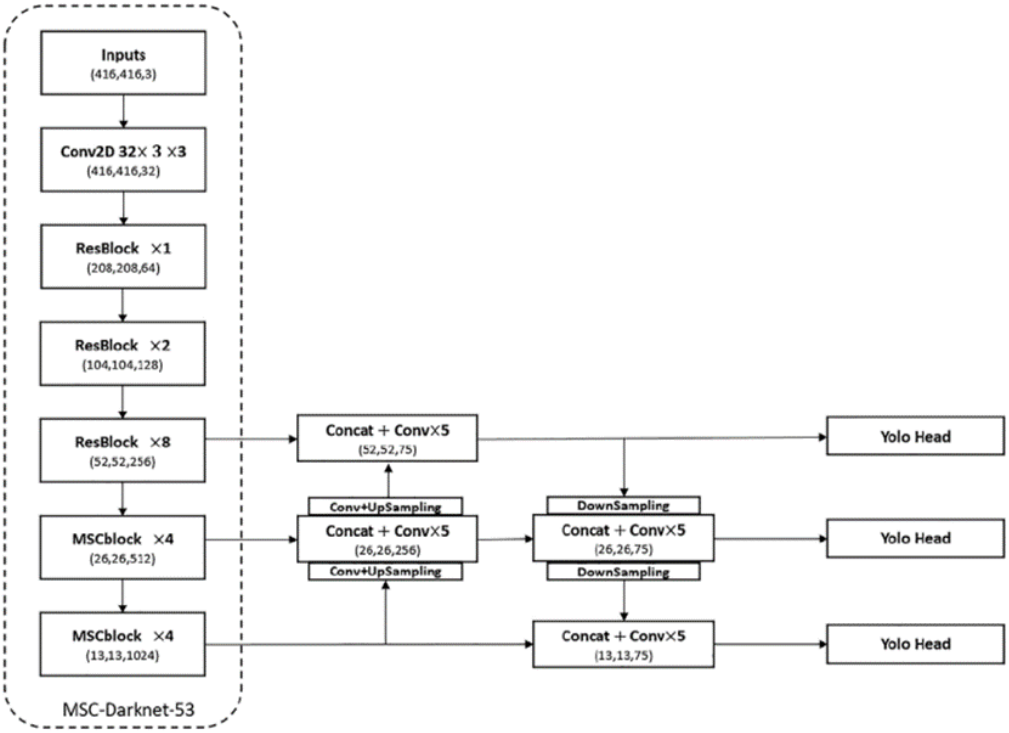


Figure 3.10: Structure of the method [6].

Dataset : Flame data sets are mainly divided into two types, indoor and outdoor. In the indoor scene. The outdoor scene selects the common parking spots in the campus. Trees, buildings, cars, and other objects are used as the background to obtain the flame data set. For the indoor scene the warehouse scene is made in the standard combustion room to obtain a similar background. a variety of combustibles and oil plates of different sizes were used to photograph the flames. Sunlight, light, personnel, and other interference items were added into the shooting background. A total of 7,254 images were selected from the filmed videos and used as the training dataset[6].

Figure 3.11: example images from the used dataset [6].

Results

Table 3.4: The proposed method's results.

	PR (%)	RR (%)	FAR (%)	AR (%)
proposed method	98.7	93.7	1.2	96.3

Figure 3.12: Partial results identified by the proposed model [6].

2. **Avazov et al 2022 (Fire Detection Method in Smart City Environments Using a Deep-Learning-Based Approach)[7]:** This work uses the YOLO4 network to detect and classify suspected regions of fire without delayed final decisions[7].

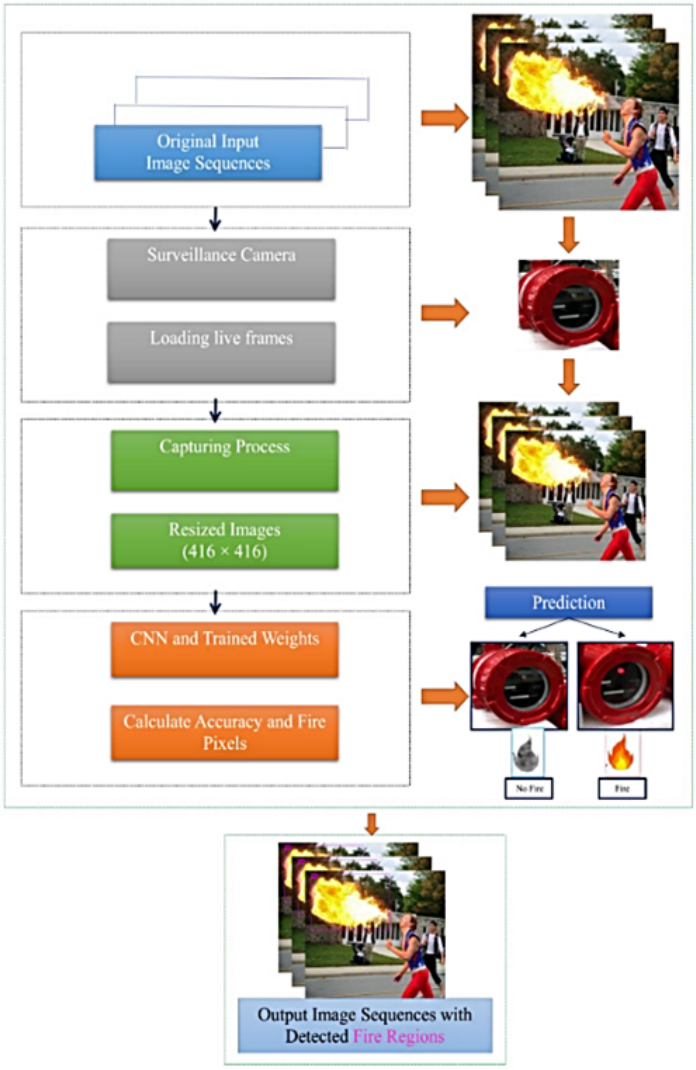


Figure 3.13: Flowchart outlining the proposed method [7].

Dataset :
the dataset 9200 flame pictures are used. After rotating all fire pictures by 90◦, 180◦, and 270◦, as shown in figure and labeling them the dataset increased by three times to 27600 images ,and over 10000 flame-like pictures are added to prevent false-positive outcomes[7]
.
Then after removing low-quality and low-resolution frames from the final database, a total of 20,100 good images are ready to use in implementation. After modifying the luminance and chromaticity of the flame frames, the total number of learning pictures increased to 80,400 [7].

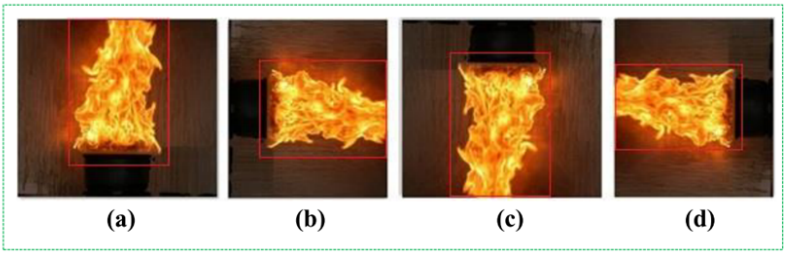


Figure 3.14: **(a)**original image,**(b)** 90 rotation,**(c)** 180 rotation,**(d)** 270 rotation [7].

Dataset	Training set	Test set	Total
Flame pictures	24385	3215	27600
Flame-like pictures	10000	0	10000

3.3.3 Deep learning approach for flame and smoke detection systems:

1. **Mimoun YANDOUZI et al 2023(Investigation of Combining Deep Learning Object Recognition with Drones for Forest Fire Detection and Monitoring):**This work uses UAVs for real-time monitoring and detection of forest fire using YOLOv6, YOLOv7, YOLOv8, and Faster RCNN with Resnet50, VGG16/19 as backbones.

   Dataset After data augmentation, the dataset reaches a total of 4236 images with the labels Fire and Smoke. These photos were taken with both ground- level cameras and aerial drones. The ground cameras were used to capture detailed images of forest fires in real time, providing accurate representations of the fires. Some images were man-made and supervised fire images []

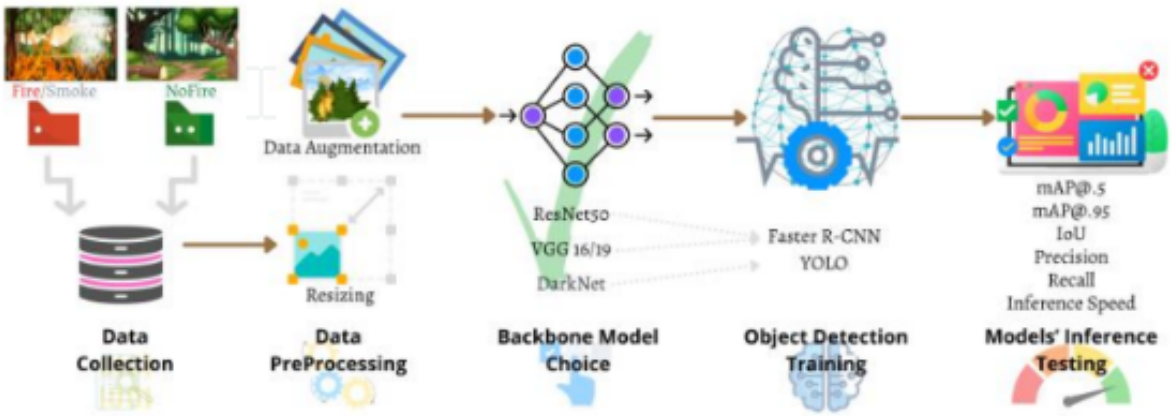


Figure 3.15: The proposed method[].

Results To obtain these test results the training phase had 500K epochs for Faster RCNN, and 1k epochs for YOLO.

Figure 3.16: Forest fire and smoke detection by drone - examples using YOLOv8n.[].

Model Name	mAP@0.5 %	mAP@0.95 %	IoU %	Recall %	Precision %	Inference time (s/image)
Faster R-CNN (RS50) C4	89.32	79.12	89.36	90.31	89.17	~0.0553
Faster R-CNN (RS50) DC5	89.16	78.96	88.15	89.74	88.96	~0.1374
Faster R-CNN (RS50) FPN	90.57	80.34	91.02	90.83	90.61	~0.0281
Faster R-CNN (VGG19)	89.75	79.65	90.44	89.74	89.41	~0.0753
Faster R-CNN (VGG16)	89.62	79.52	89.23	89.74	89.21	~0.0675
YOLOv6 n	89.12	78.96	88.06	89.04	88.82	~0.0009
YOLOv7	89.29	79.12	89.17	89.24	89.02	~0.0027
YOLOv8 n	89.45	79.28	89.36	89.61	89.44	~0.0011
YOLOv6 s	88.98	78.82	88.03	88.57	88.42	~0.0022
YOLOv8 s	89.31	79.16	89.25	89.40	89.24	~0.0015
YOLOv6 l	88.84	78.68	88.06	88.19	88.01	~0.0086
YOLOv7 x	89.01	78.85	89.12	87.79	87.62	~0.0051
YOLOv8 l	89.17	79.01	89.24	89.19	89.02	~0.0025

Figure 3.17: ACHIEVED RESULTS FOR THE IMPLEMENTED MODELS (ON THE TESTING SET)[].

2. **Raghad K. Mohammed 2022 (A real-time forest fire and smoke detection system using deep learning)[8]:** This work used a pre-trained Inception-ResNet-v2 network on the ImageNet dataset to be trained on our dataset, we used the Open CV library to read the camera stream frame by frame and predict the probability of fire or smoke[8].

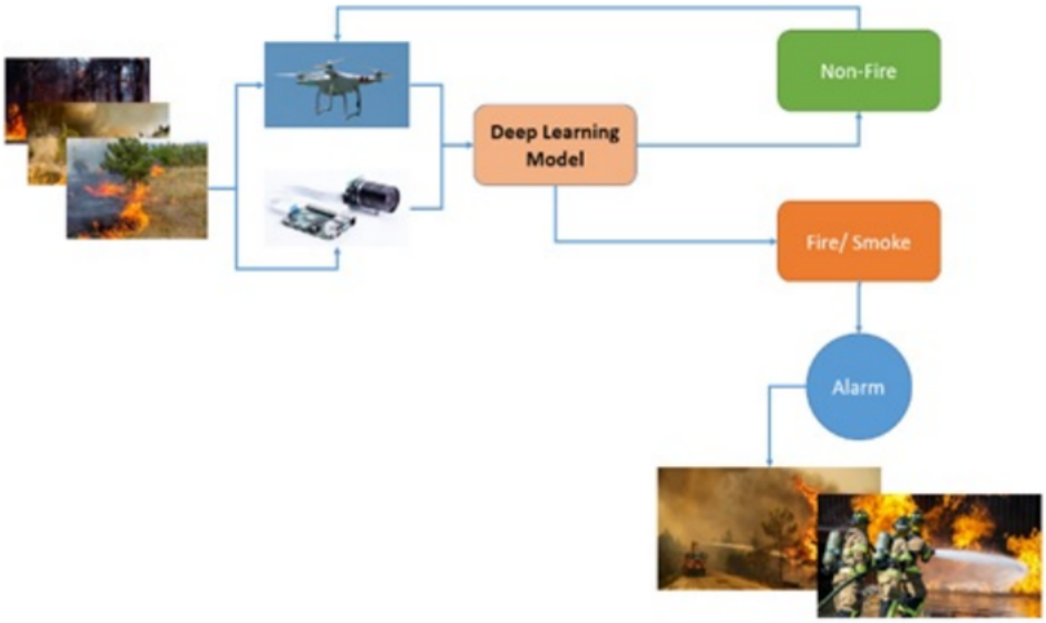


Figure 3.18: flow chart of the proposed method[8].

Dataset:
In this study, a forest images dataset of fire and smoke is used, which consists of 1,102 images with fire and 1,102 images with smoke. This dataset was collected from various internet sites including a dataset of forest fire on Kaggle[8].

Figure 3.19: Example images from the used dataset[8].

Results :

Table 3.5: Results

Performance matrices	Value
Accuracy	99.09
Precision	100
Sensitivity	98.08
F1-score	99.09
Specificity	98.3

3.3.4 Comparison

Work	Year	Used methods	Equipments used	Indoor	Outdoor	Flame	Smoke	Results
Khan et al	2021	Image segmentation	Nvidia GetForce RTX 2080 GPU with 12 GB on-board memory		×			AC=98.18%
Dewangan, A et al	2022	Image classification	NVIDIA 2080Ti GPU		×		×	AC=83.49%
Dai P et al	2022	Object detection YOLOv3	GPU is GeForce GTX 1080, CPU is Intel(R) Core(TM) I7-3960X, 32G memory	×	×	×		mAP@0.5=96.3%
Avazov et al	2022	Object detection YOLOv4	3.20 GHz CPU, 32 GB RAM, and two Nvidia GeForce 1080Ti GPUs (Nvidia, Santa Clara, CA, USA)	×	×	×		mAP@0.5=99.8%
Raghad K. Mohammed	2022	Image classification	i7-10750H CPU @ 2.60GHz with 2.59 GHz, 32.0 Gigabyte of RAM,and a display adapter from NVIDIA GeForce GTX 1650 Ti		×	×	×	AC=99.09%
Mimoun YANDOUZI et al	2023	-YOLOv8n -Faster RCNN (resnet50 fpn)	Two Intel Gold 6148 (2.4GHz/ 20-core) processors.Two NVIDIA Tesla V100 graphics cards, each having 32GB of RAM training with 500k epochs for Faster RCNN, and 1k epochs for YOLO		×	×	×	mAP@0.5=89.45% mAP@0.5=90.57%

3.4 Conclusion :

In this chapter, we showed the most recent works regarding fire detection systems using video surveillance and image recognition methods. Object detection methods are more useful in the case of the fire is still small and not noticeable for the human to determine. So in the next chapter, we will try to implement a method using deep learning and object detection on a collected dataset, and try to obtain better results.

4 : Our contribution

Contents

4.1 Introduction

In the previous chapter, we discussed fire, its threat to our society, and some potential prevention strategies. In this chapter, we illustrate our DL object detection approach where we trained and tested our detector on the preprocessed dataset. We used one-stage detection models YOLOv5n and YOLOv8n and two stages detection models using VGG16, VGG19, and Xception as backbones.This chapter illustrates the process of developing the fire and smoke detector, it will be organized as follows:

4.2 Our approach

In order to achieve our approach we worked with:

Google colab: Because we did not have a GPU we worked with google colab's free GPU Nvidia T4 online.

Tenserflow : TensorFlow is an end-to-end open source platform for machine learning, that enables building and implementing ML models simple for both beginner and expert users [33].

Keras : The open-source software program known as Keras offers a Python interface for ANN. The high-level Keras API makes it simple to build and train models while using TensorFlow and ML by offering the TensorFlow library interface [34].

4.2.1 Detection

4.2.1.1 One stage detection

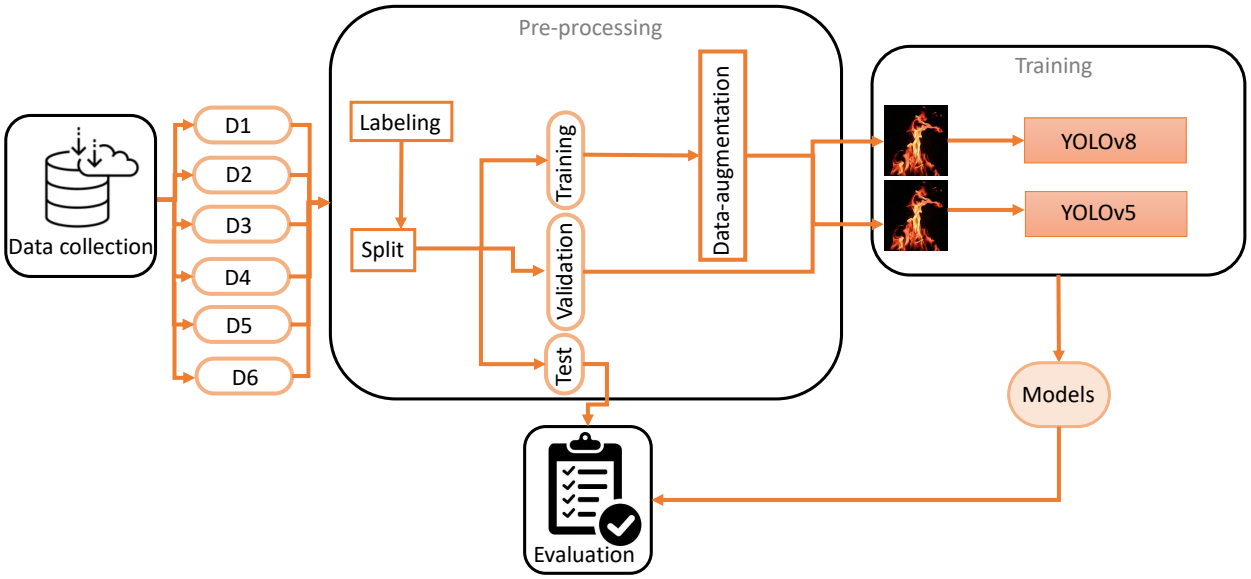


Figure 4.1: proposed one-stage detection method

1. **Dataset collection**

   We made different datasets to obtain the best results possible

   D1 The first dataset contained around 1000 indoor fire images only collected from sites like pixels[35] which contains very high-resolution images or Kaggle[36].

   D2 The second dataset contained around 500 indoor fire images only collected manually from sites.

   D3 The third dataset contained 1367 indoor and outdoor fire-only images with 1500 annotations after data augmentation.

   D4 The fourth dataset contained 737 outdoor smoke-only images with 1200 annotations

   D5 The fifth dataset contained 1000 images, both indoor and outdoor, 400 of these images are smoke and the rest are fire, collected from sites and other online datasets.

   D6 The sixth dataset with the best results contains 3022 images from D3 before augmentation and D4, then applying data augmentation.

Table 4.1: Collected datasets

Dataset	#Images	#Annotations	Indoor	Outdoor
D1	around 1000	-	x	
D2	around 500	-	x	
D3	1367	1500	x	x
D4	737	1200		x
D5	Fire:400 smoke:600	-	x	x
D6	Fire:1052 Smoke:1670	Fire:1862 Smoke:1699	x	x

Figure 4.2: example images from D6

2. **Preprocessing**

   it is used to decrease training time and increase performance by applying image transformations to all images in this dataset.in our work, we applied auto orientation to all images and resized them to 640*640.

   Labeling

   For labeling we used Roboflow which helped us convert all bounding boxes in each image to a numeric value and store it in a txt file with the same name as the image, containing the annotations(bounding boxes) for the corresponding image file, that is object class, object coordinates, height, and width. **<object-class>** **<x>** **<y>** **<width>** **<height>**

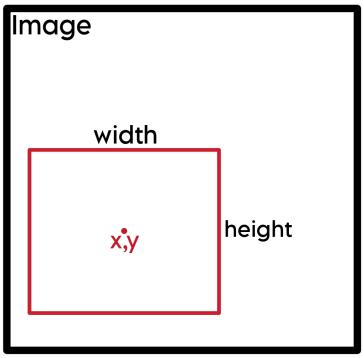
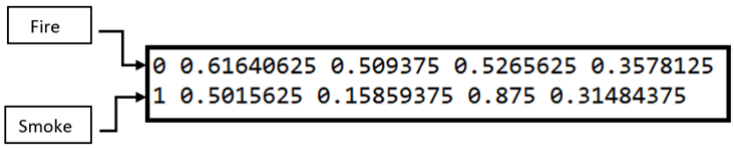


Figure 4.3: Labeling

We take an example of an image.txt from our dataset containing one fire and one smoke



Split

In our approach, we divided the datasets into 80% for the train and 13% for the test, and 7% for the validation. After that, the training set was augmented.

Data Augmentation

The purpose of augmenting train set images is to boost diversity, decrease overfitting, increase data volume, improve generalizability, and overcome various forms of variances. . In our work, we applied multiple data augmentation techniques:

Blur	Up to 0.75px
Rotation	Between -15° and +15°
Crop	0% Minimum Zoom, 20% Maximum Zoom

- **Blur** This shows its importance when:
  - A camera is stationary, but the objects it detects are frequently moving.
  - A camera is not stationary, but the objects it is detecting are.
  - A camera and the objects it is detecting are moving.
  - In our case if a camera is foggy due to the smoke
- **Rotation** Because it modifies the angles at which objects appear in the dataset during training, rotation is a particularly useful augmentation.
- **Crop** Because the items of interest that we want our models to learn are not always fully visible in the image or at the same scale in our training data, this allows our model to generalize more effectively. .

3. **Training** For D3 D„5 and D,6 we can notice that the model's training classification loss stabilized starting from the 80th, 20th, 70th, and 130th epochs respectively, while the box loss didn't stabilize. As for the validation classification loss values, the stabilization started from the 90th epoch for D3, at the 10 epoch for D4, the 50th epoch for D5, and the 70th for D6, however, the box loss didn't stabilize for D4, D3, and D5 but for D6 we can say that it did stabilize a bit starting from the 120th epoch

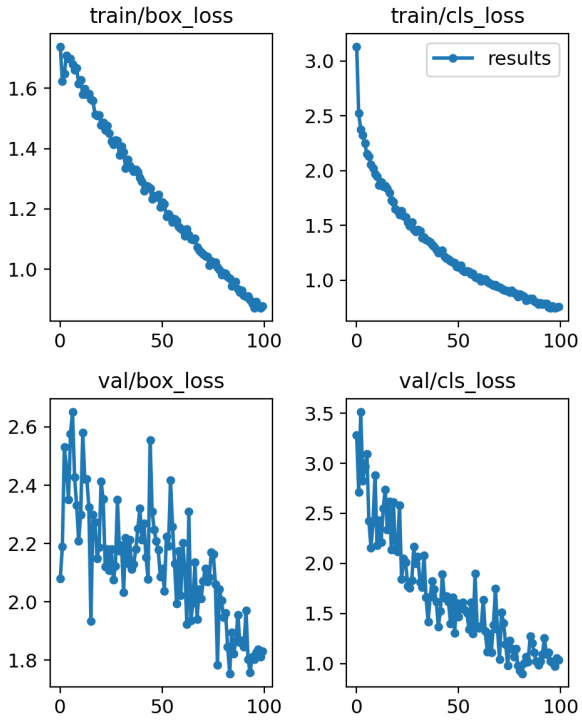


Figure 4.4: D3 training results

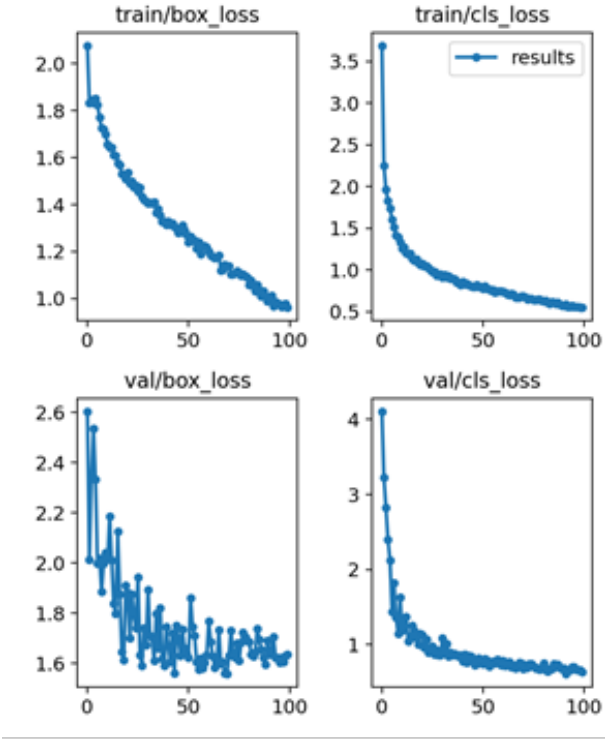


Figure 4.5: D4 training results

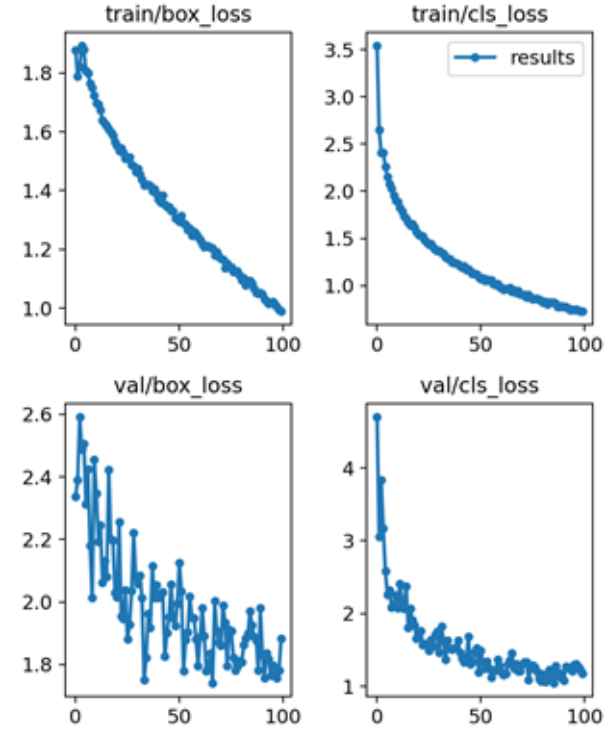


Figure 4.6: D5 training results

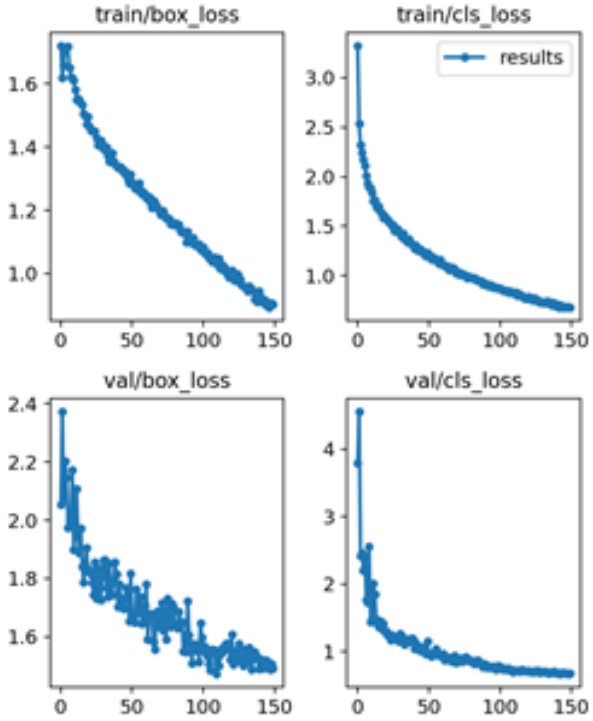
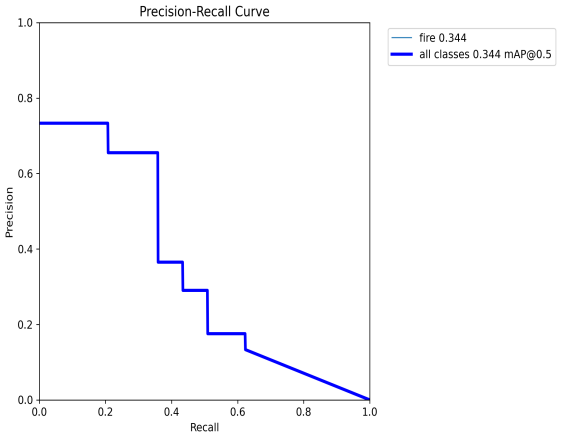


Figure 4.7: D6 training results

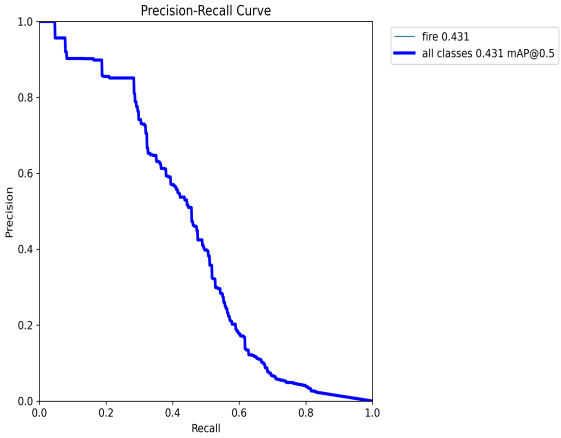
4. **Test results**

Table 4.2: Yolo test results

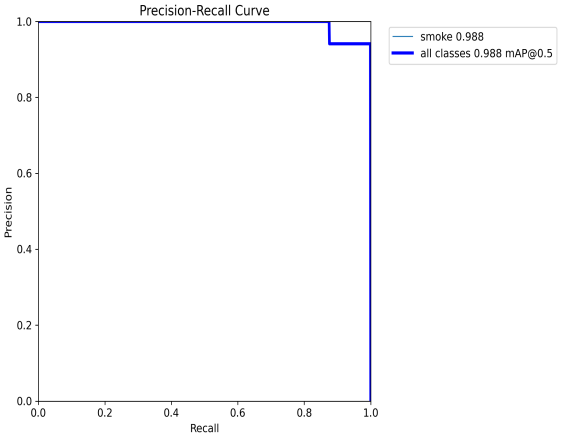
Model	Dataset	Class	Precision	Recall	**mAP@0.5**	mAP@0.5-0.95
YOLOv5n	D1	All	0.64	0.36	**0.34**	0.131
	D2	All	0.62	0.46	**0.43**	0.18
YOLOv8n	D3	All	0.81	0.86	**0.92**	0.42
	D4	All	0.94	1	**0.99**	0.56
	D5	Fire	0.65	0.73	**0.68**	
		Smoke	0.86	0.82	**0.73**	
		All	0.87	0.78	**0.70**	
	D6	Fire	0.84	0.76	**0.88**	0.44
		Smoke	0.95	0.92	**0.97**	0.56
		All	0.89	0.84	**0.93**	0.50



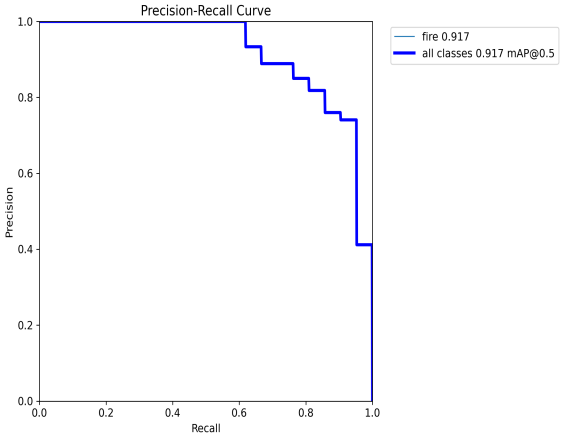
(a) Precision and recall curve for D1



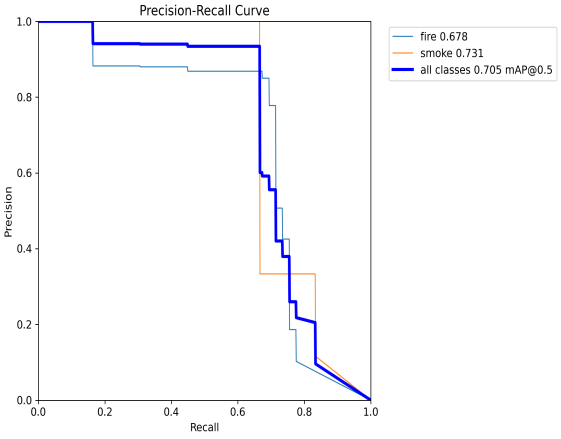
(b) Precision and recall curve for D2



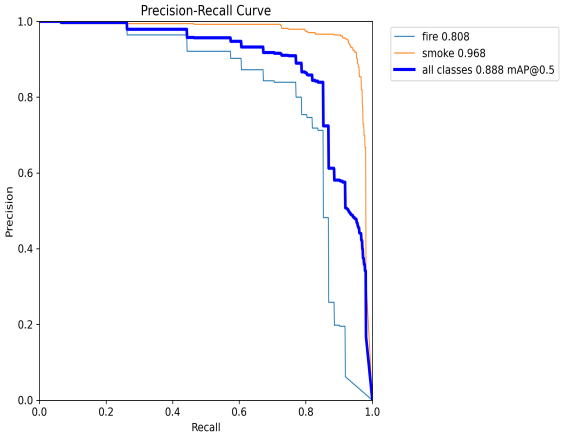
(c) Precision and recall curve for D3



(d) Precision and recall curve for D4



(e) Precision and recall curve for D5



(f) Precision and recall curve for D6

Figure 4.8: Yolo precision and recall graphs

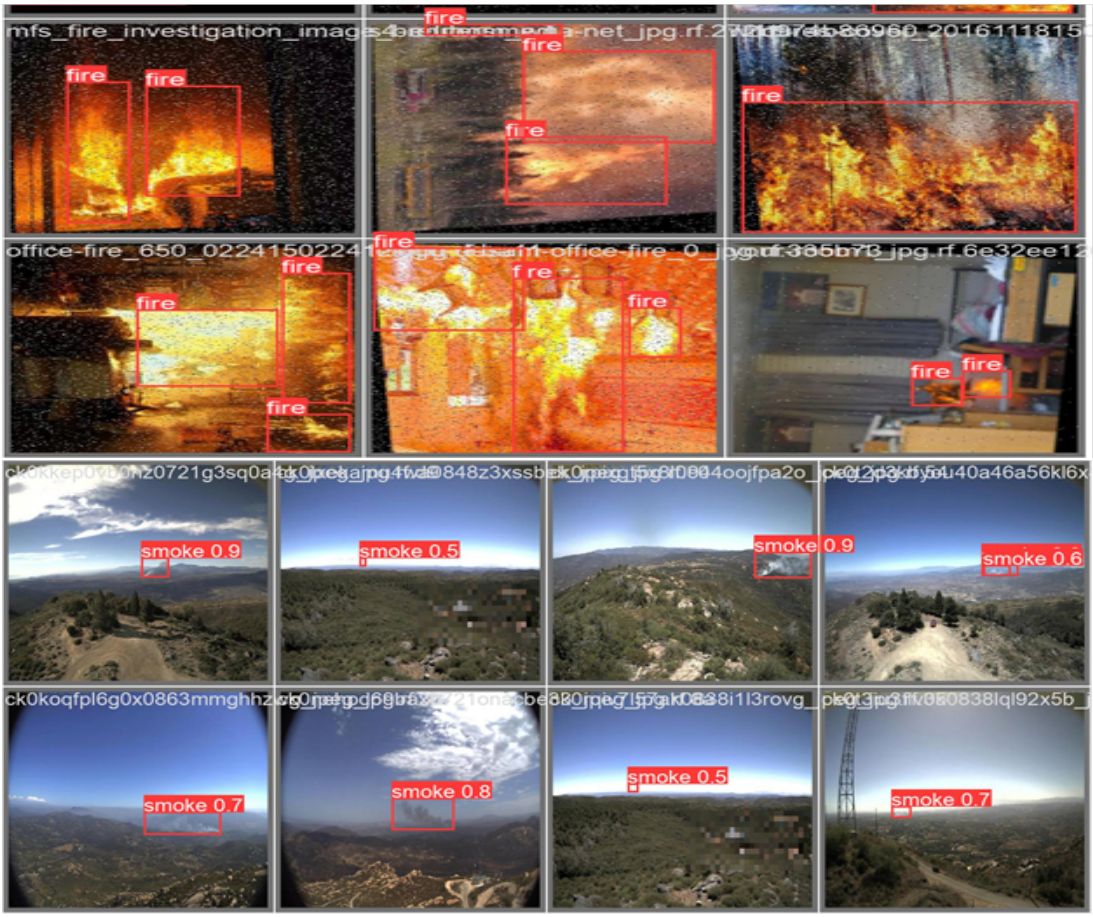


Figure 4.9: Example images from the test dataset for D6

4.2.1.2 Two stages detection

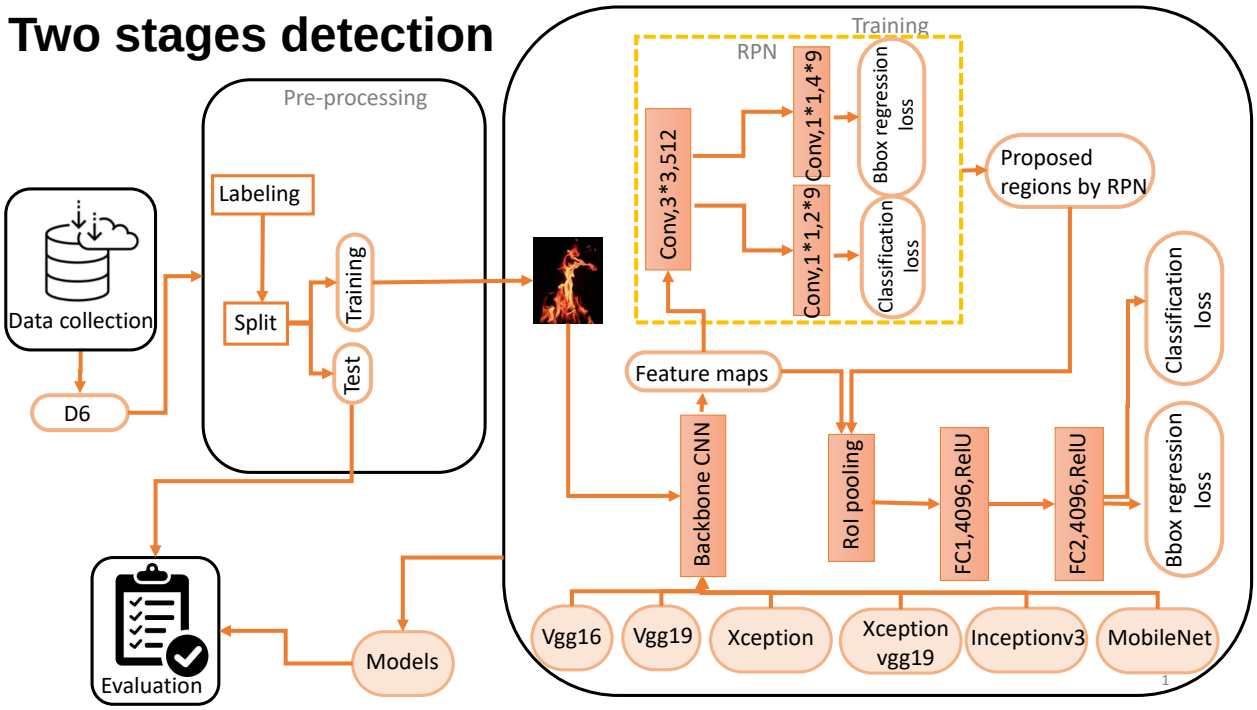


Figure 4.10: proposed two stages detection method

1. **Data collection**

   For this method, we used D6 (after applying the data augmentation to its training set)

2. **Pre-processing**

   For this approach, we resized the images from 640*640 the images to 224*224 after labeling

   Labeling

   Labeling for the Faster RCNN model is different from labeling for YOLO, and we already had D6 labeled for YOLO so instead of relabeling it from scratch we had to convert our labeling from YOLO format to Faster RCNN format which is:

   <object-class> <x-min> <y-min> <x-max> <y-max>

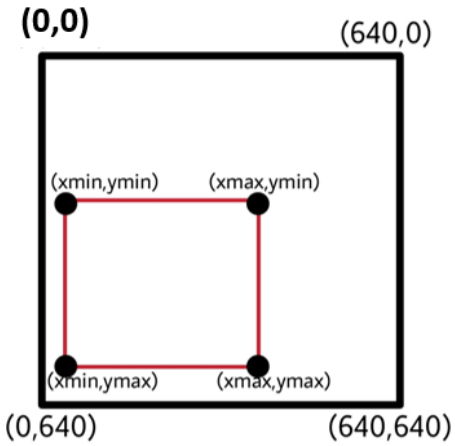


Figure 4.11: Faster RCNN labeling

to transform the labeling from YOLO format to Faster RCNN format:

$$xmax = x + \frac{w}{2} \tag{4.1}$$

$$xmin = x - \frac{w}{2} \tag{4.2}$$

$$ymax = y + \frac{h}{2} \tag{4.3}$$

$$ymin = y - \frac{h}{2} \tag{4.4}$$

where (x,y,w,h) are the labeling used in YOLO for the bounding box.

the data labeling will be stored in the form of a .csv file In order to facilitate the processing of data

filename	xmin	xmax	ymin	ymax	class
ck0nd794g	77	177	292	436	smoke
Datacluste	95	574	0	305	fire
ck0ukig53t	34	485	260	453	smoke
your-roon	449	561	368	474	fire
your-roon	348	467	405	515	fire
ck0na2cdf	464	506	347	407	smoke
ck0tsymgp	353	573	260	411	smoke
ck0oukl7v	148	196	378	417	smoke
ck0txrr4ns	400	432	237	272	smoke
32313rwe	86	627	0	416	fire
ck0nftcfgk	530	620	361	440	smoke
ck0txxd8ft	205	324	255	342	smoke
burning-ch	317	612	103	482	fire

Figure 4.12: csv file

After creating this file we had to add the whole path to each image in the filename as shown above(exp: the path for an image is /content/Drive/mydrive/dataset/training-set/images/image)

Split

For the split, we added the validation dataset to the training dataset, resulting in a training set with 90% and a test set with 10%.

3. **Training**

Backbones :

VGG :VGG means Visual Geometry Group and this architecture has 16 to 19 layers with very small 3×3 convolution filters[37].

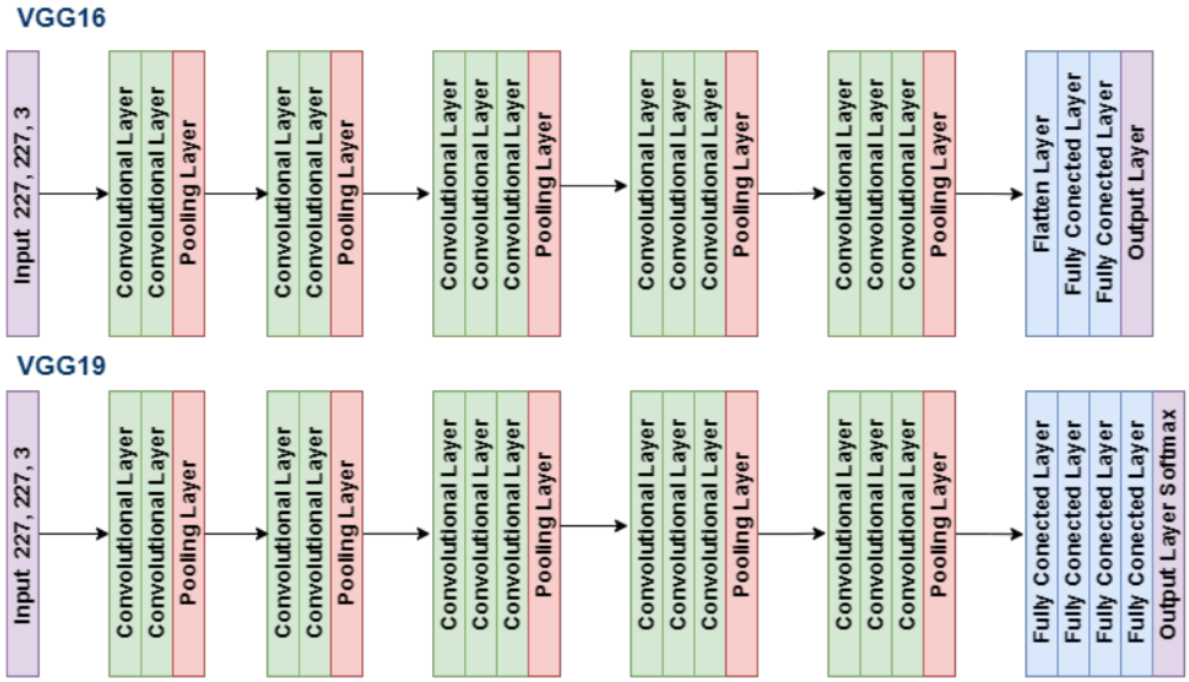


Figure 4.13: VGG16/19 architecture [9]

Xception: The Xception deep neural network which stands for extreme inception, and is inspired by Inception, was made by François Chollet [38]. Xception architecture has depth-wise separable convolutions. Xception has 36 convolutional layers to extract features [38]

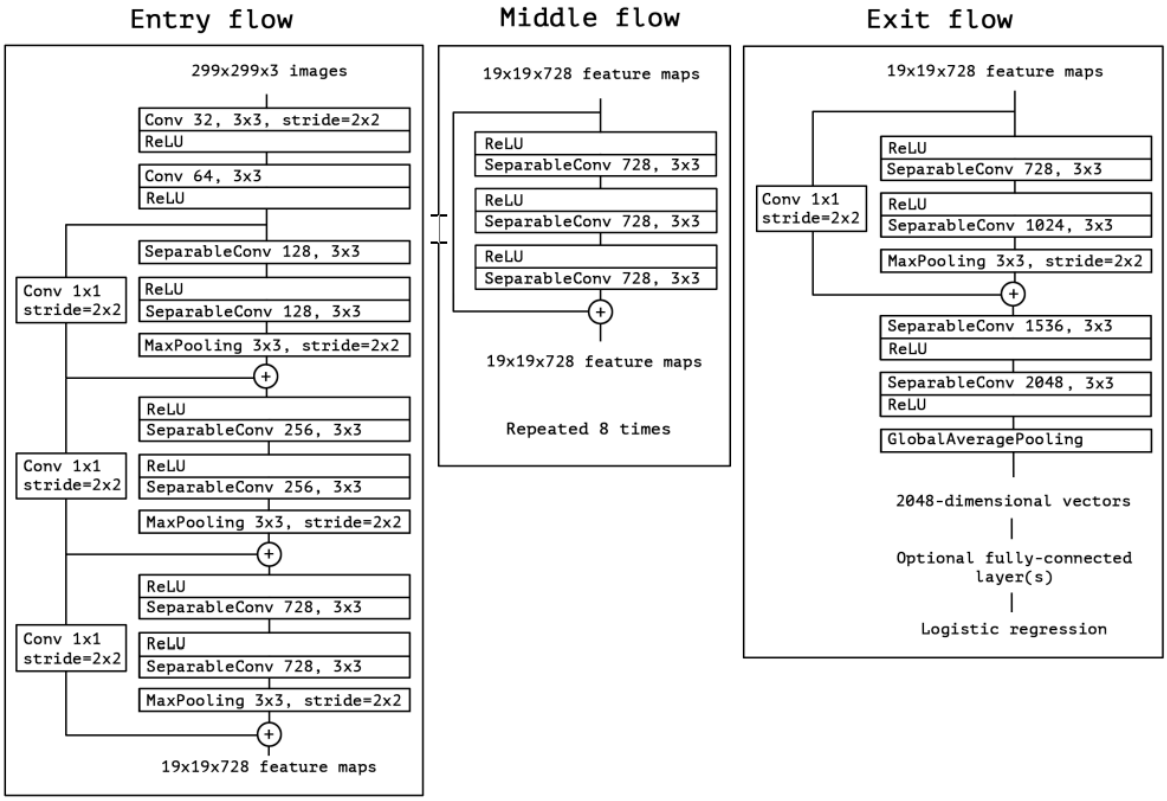


Figure 4.14: Xception architecture

Xception-VGG19 : for this model we used the Keras concatenate layer, it takes as input a list of tensors, all of the same shape except for the concatenation axis, and returns a single tensor that is the concatenation of all inputs.[39]

inputs: The layers of two models at which we want to merge these models.[40]

axis: The axis along which we want to concatenate the two layers.[40]

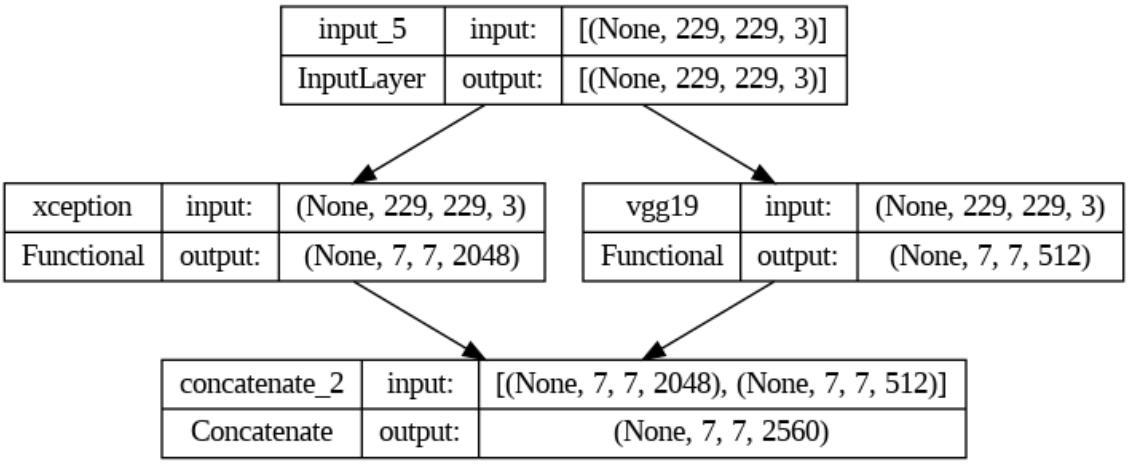


Figure 4.15: Xception-vgg19 architecture

MobileNet : MobileNet is a computer vision model open-sourced by Google and designed for training classifiers. It uses depthwise convolutions to significantly reduce the number of parameters compared to other networks, resulting in a lightweight deep neural network. [41]

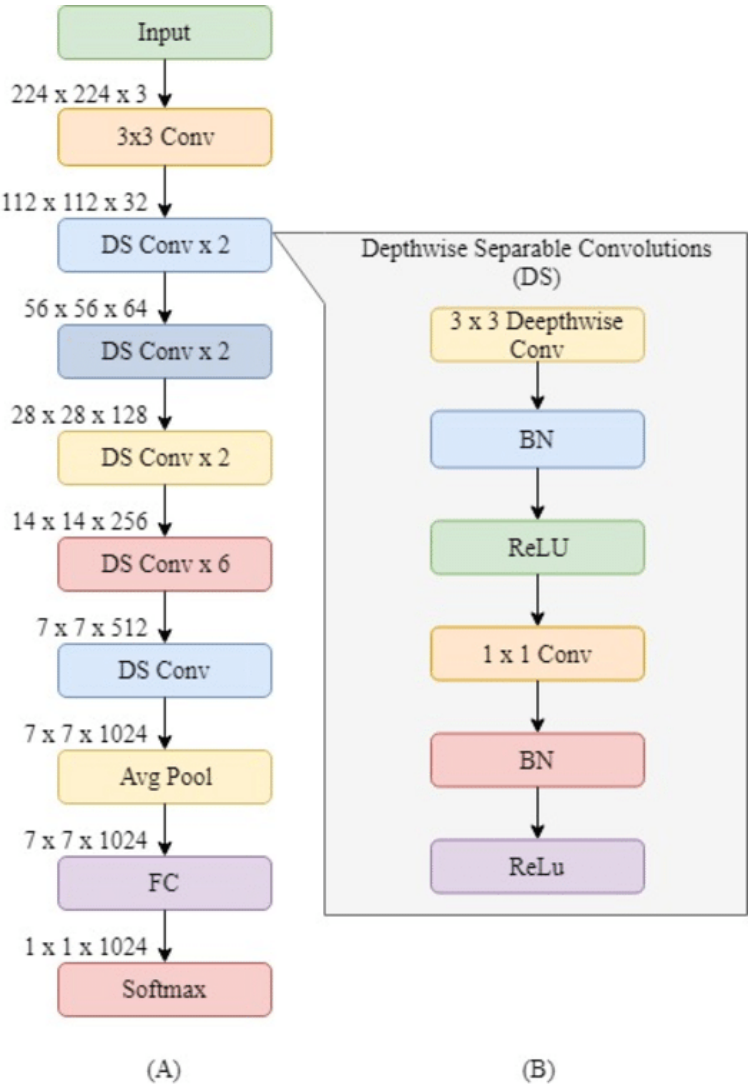


Figure 4.16: MobileNet architecture [10]

Inception v3 : Inception v3 is an image recognition model that has been shown to attain greater than 78.1% accuracy on the ImageNet dataset. The model is the culmination of many ideas developed by multiple researchers over the years[42]. It is based on the original paper: "Rethinking the Inception Architecture for Computer Vision" by Szegedy, et. al[11].

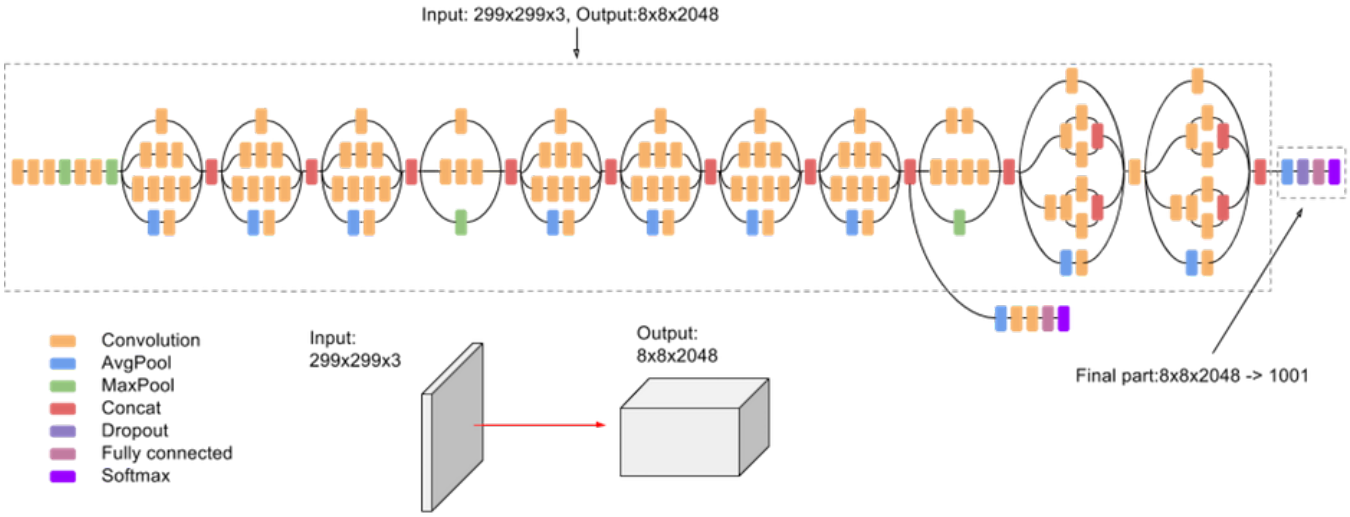


Figure 4.17: Inceptionv3 architecture[11]

When we started the training phase in Google Colab using each pre-trained model mentioned above as backbone with 100 epochs and 200 steps only to the limited usage time, it resulted in 5 losses[43]:

1. RPN classification: anchor box is object / not an object
2. RPN regression: predict transform from anchor box to proposal box
3. Object classification: classify proposals as background/object class, classification loss for the Faster R-CNN layer is defined as the multi-class loss function shown below:

$$L_{cls}\left(p_i, p_i^*\right) = -\log\left(p_i^c\right)$$

where:

p_i	Predicted probability of anchor "i" being a positive sample
p_i^*	The confidence target. It equals 1 when it is an object. Otherwise, it equals zero
p_i^c	The predicted probability for the proposed region belonging to class *c* that is the ground truth class.

4. Object regression: predict transform from proposal box to object box [43].

$$\mathcal{L}_{\text{box}}(t^u, v) = \sum_{i \in \{x,y,w,h\}} L_1^{\text{smooth}}(t_i^u - v_i)$$

$$L_1^{\text{smooth}}(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \tag{4.5}$$

where true bounding box

$$v = (v_x, v_y, v_w, v_h) \tag{4.6}$$

and The predicted bounding box correction

$$t^u = (t^u_x, t^u_y, t^u_w, t^u_h) \tag{4.7}$$

5. The total loss is the sum of the 4 losses above

RPN an anchor box is a predefined bounding box with a certain size , first, we create "anchor boxes", and calculate which object's bounding box has the highest IoU.

If the highest IoU is greater than or equal to 0.5, then the anchor box is a positive one.

If the highest IoU is less than 0.5, then the anchor box should predict that there is no object(negative anchor box).

- **Step1:** To generate region proposals, we slide a small network over the convolutional feature maps output by the last shared convolutional layer(backbone) to a 3x3 512 .
- **Step2:** Pass the step1 to two convolutional layers to replace the fully connected layer

  classification layer: number of anchors (9 in our situation) channels for 0, 1(object/not object) sigmoid activation output

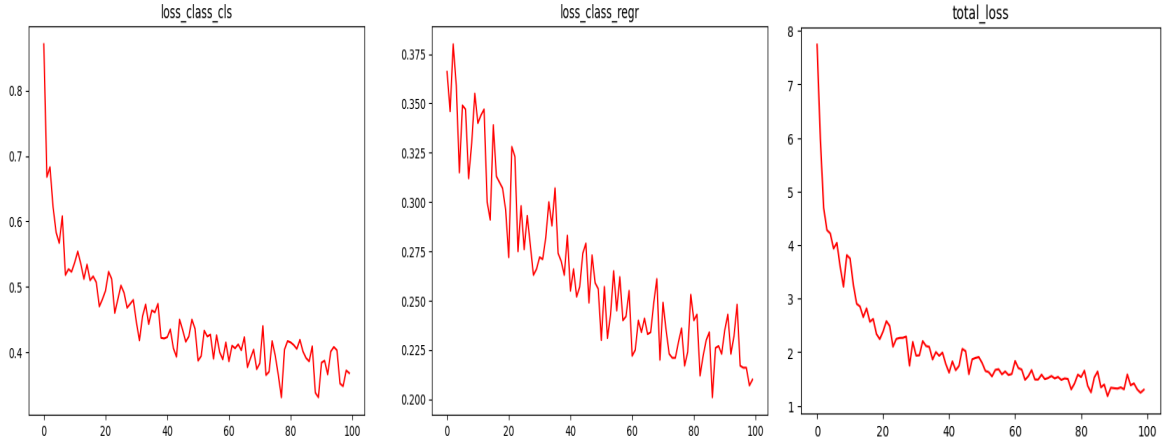
  regression layer: number of anchors*4 (36 in our situation) channels for computing the regression of bboxes with linear activation

RoI Pooling we applied the max pooling method for the feature map

classification and box regression After identifying ROIs, Faster RCNN predicts a score and bounding box regression for each ROI and for every class.

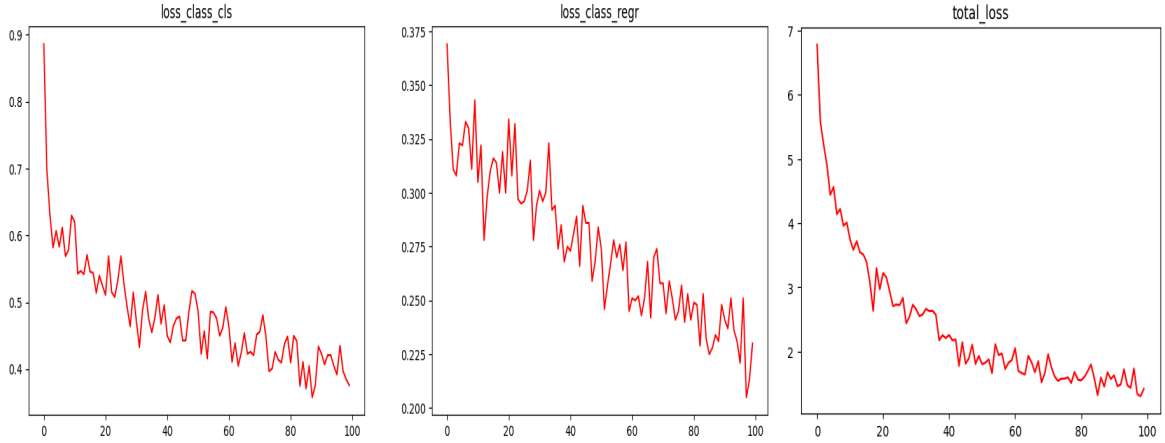
4. **Test results**

We notice that the classification loss is close to being stable for VGG19 and VGG16, but the regression loss did not stabilize. Meanwhile, the total loss settled starting from the 40th epoch for both VGG19 and VGG16 but the loss value for VGG16 was less than the VGG19. As for the Xception, Xception-VGG19, and the MobileNet, the classification and the regression losses did not become stable, the total loss somehow did at the 80th epoch but at a much higher value compared to the previous models. As for the Xception-VGG19, the total loss stabilized starting from the 50th epoch. The MobileNet's total loss had the same behavior starting from the 75th epoch. On the other hand, we observed that for the Inceptionv3, the classification loss started to stabilize from the 80th epoch, but the regression loss did not. However, the total loss became stable at the 65th epoch.



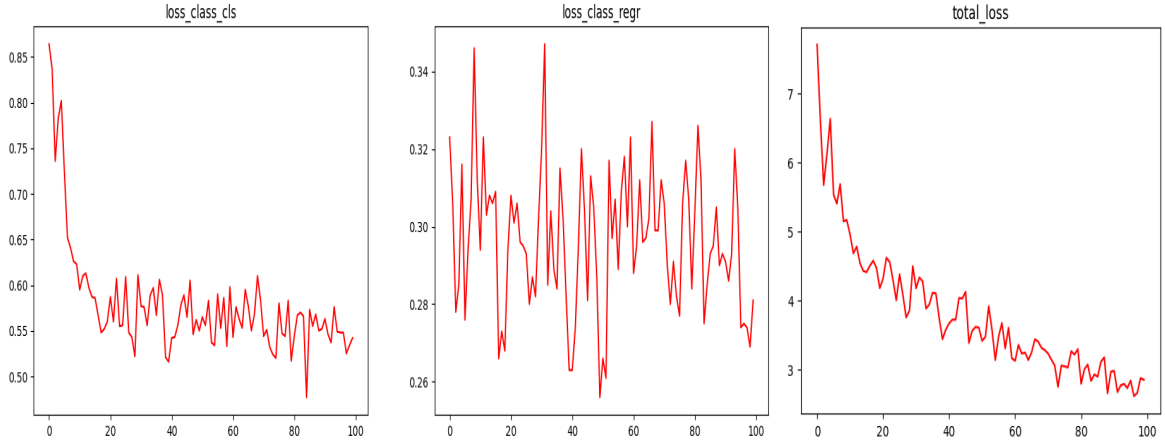
(a) Classification and regression loss(VGG16) (b) Total loss(VGG16)

Figure 4.18: VGG16 losses(loss values by the number of epochs)



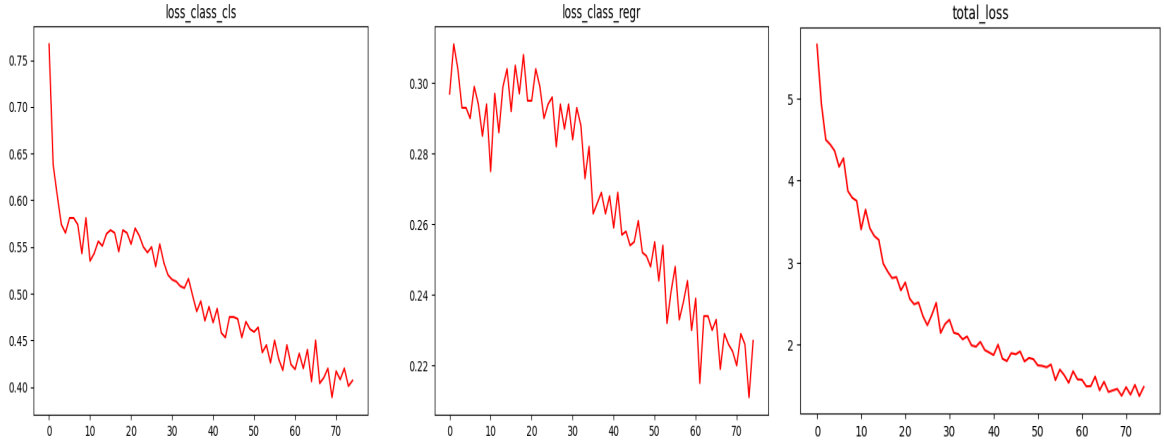
(a) classification and regression loss(VGG19) (b) Total loss (VGG19)

Figure 4.19: VGG19 losses(loss values by the number of epochs)



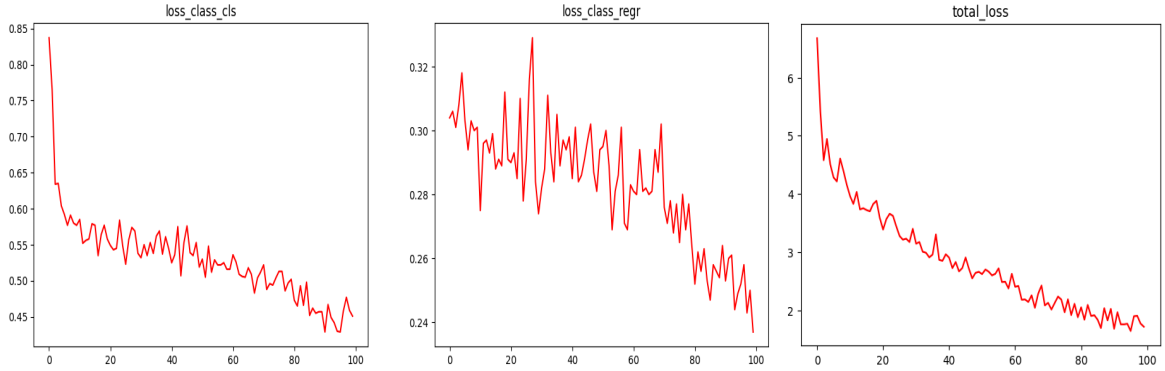
(a) classification and regression loss(Xception) (b) Total loss (Xception)

Figure 4.20: Xception losses(loss values by the number of epochs)



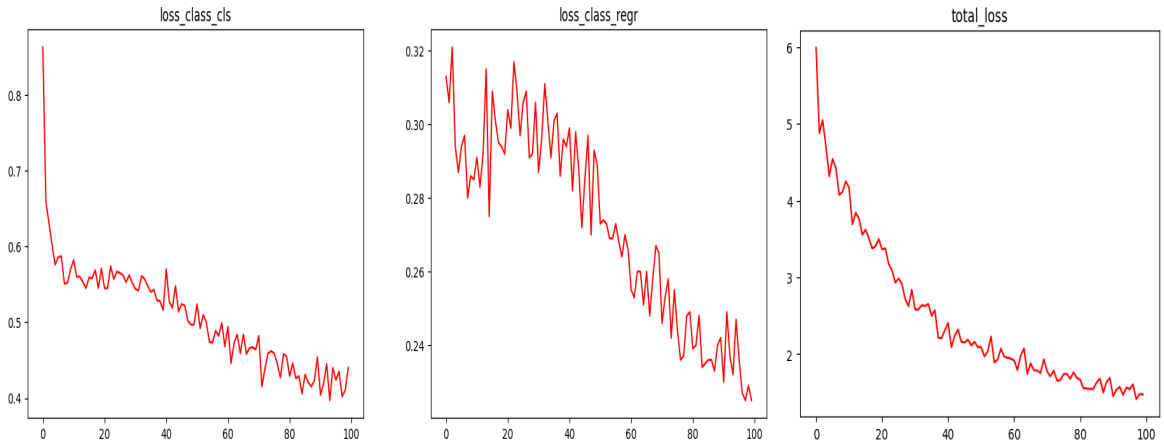
(a) classification and regression loss (b) Total loss (Xception-VGG19)

Figure 4.21: Xception-VGG19 losses(loss values by the number of epochs)



(a) classification and regression loss(MobileNet) (b) Total loss (MobileNet)

Figure 4.22: MobileNet losses(loss values by the number of epochs)



(a) classification and regression loss(Inceptionv3) (b) Total loss (Inceptionv3

Figure 4.23: Inceptionv3 losses(loss values by the number of epochs)

- **Test results**

Table 4.3: Faster RCNN test results

Backbone	Class	**mAP@0.5**
faster RCNN VGG16	Fire smoke All	0.36 0.44 0.40
faster RCNN VGG19	Fire Smoke All	0.26 0.43 0.35
fastre RCNN Xception	Fire Smoke All	0.35 0.11 0.23
faster RCNN Xception_vgg19	Fire Smoke All	0.41 0.46 0.44
faster RCNN MobileNet	Fire Smoke All	0.34 0.32 0.33
faster RCNN Inceptionv3	Fire Smoke All	0.43 0.43 0.43

VGG16 test images exemples

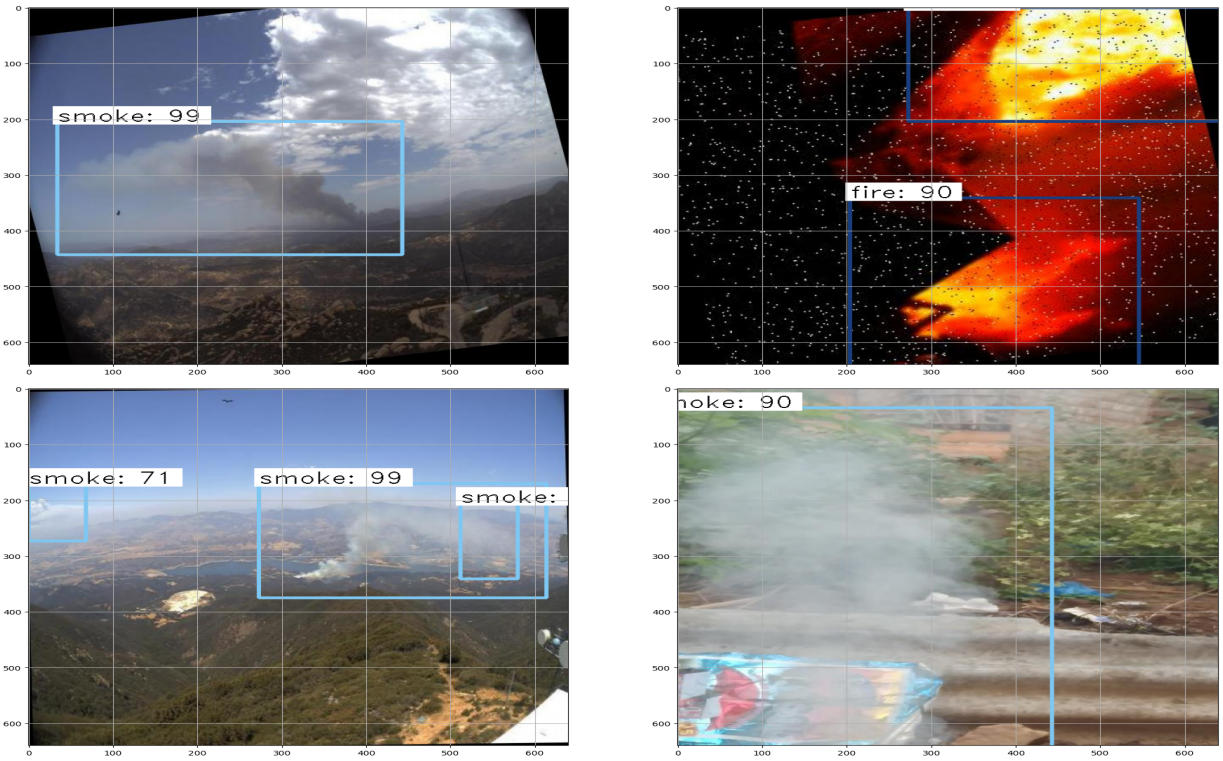


Figure 4.24: VGG16 test images

VGG19 test images exemples

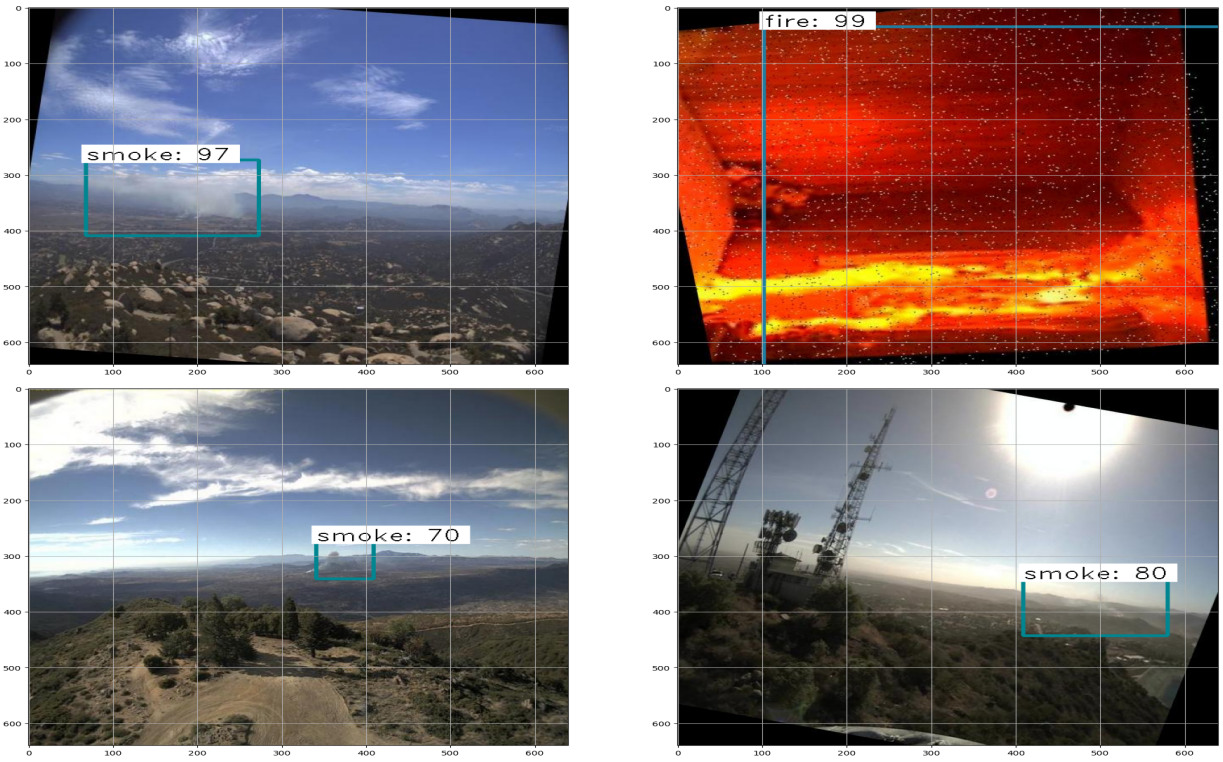


Figure 4.25: VGG19 test images

Xception test images exemples

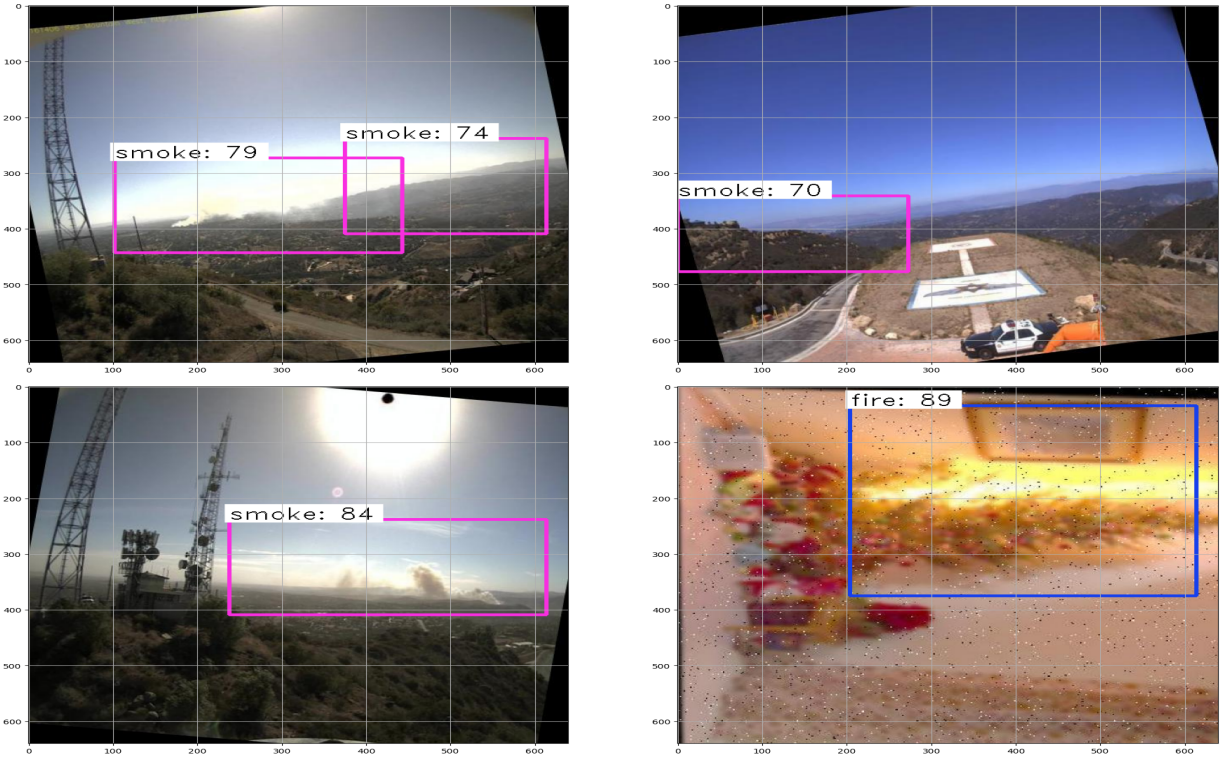


Figure 4.26: Xception test images

Xception-VGG19 test images exemples

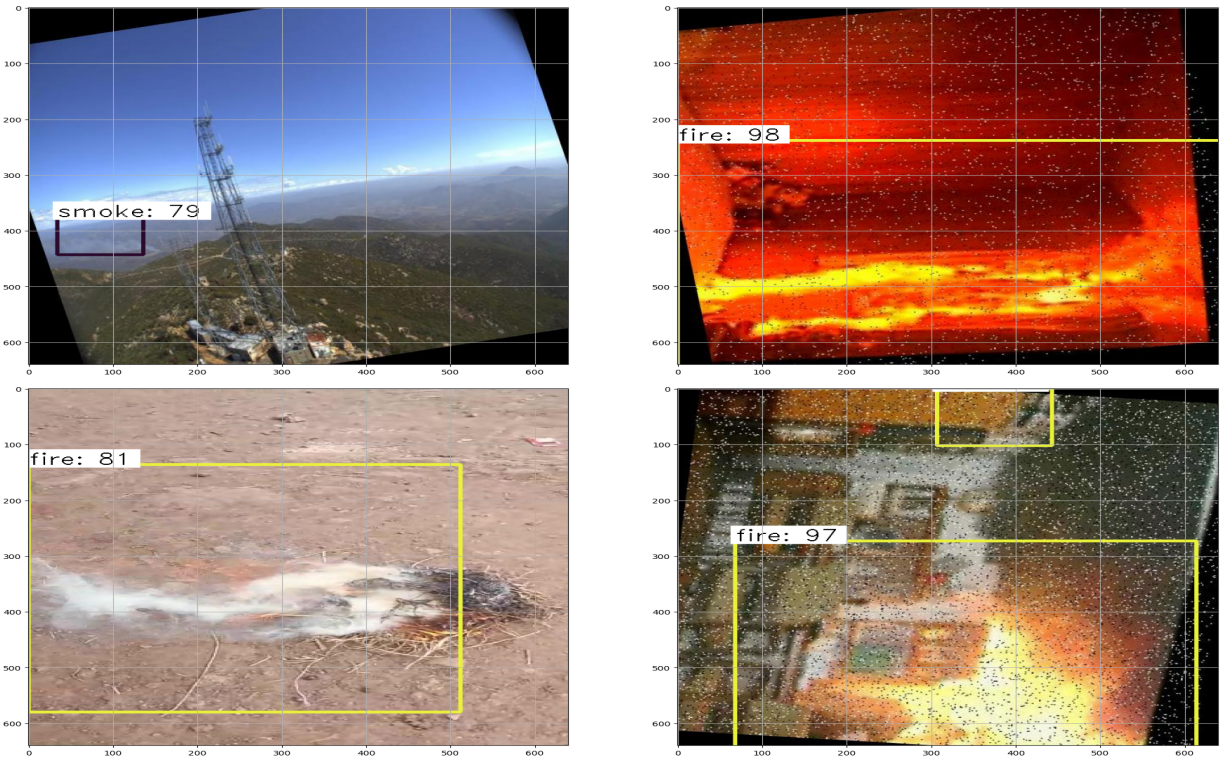


Figure 4.27: Xception-vgg19 test images

MobileNet test images exemples

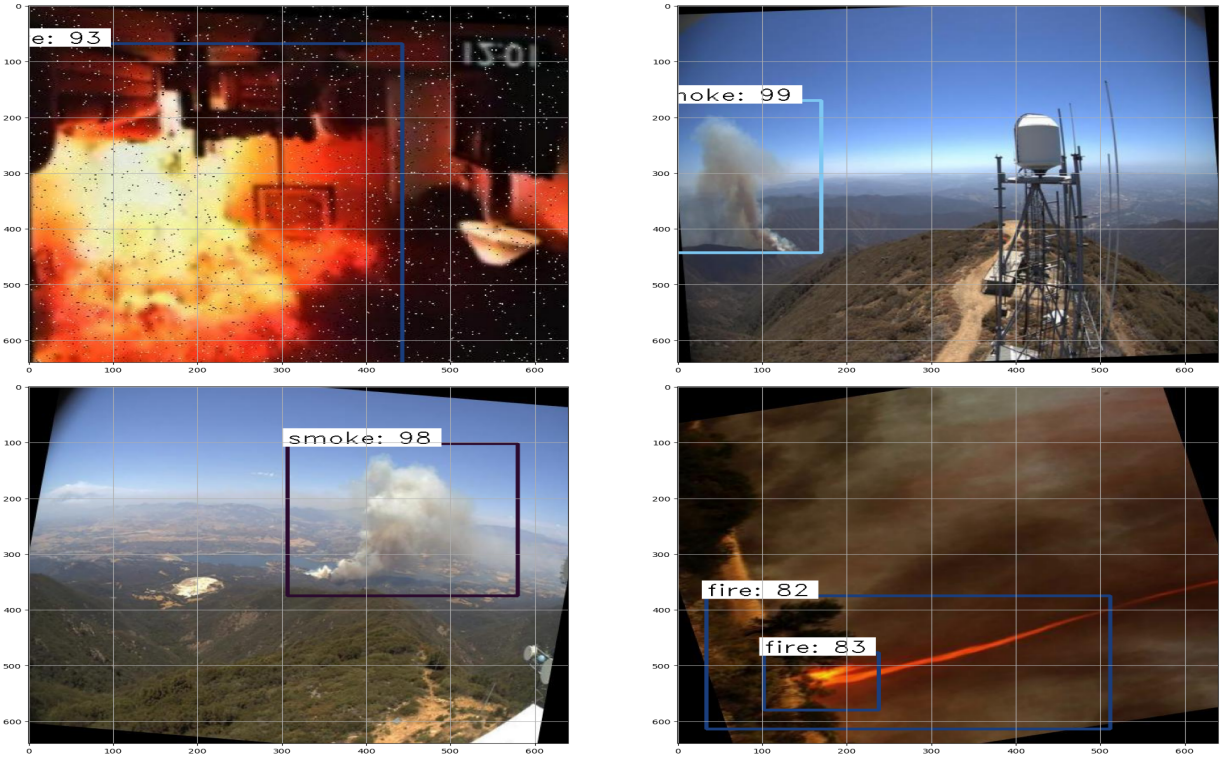


Figure 4.28: MobileNet test images

Inceptionv3 test images exemples

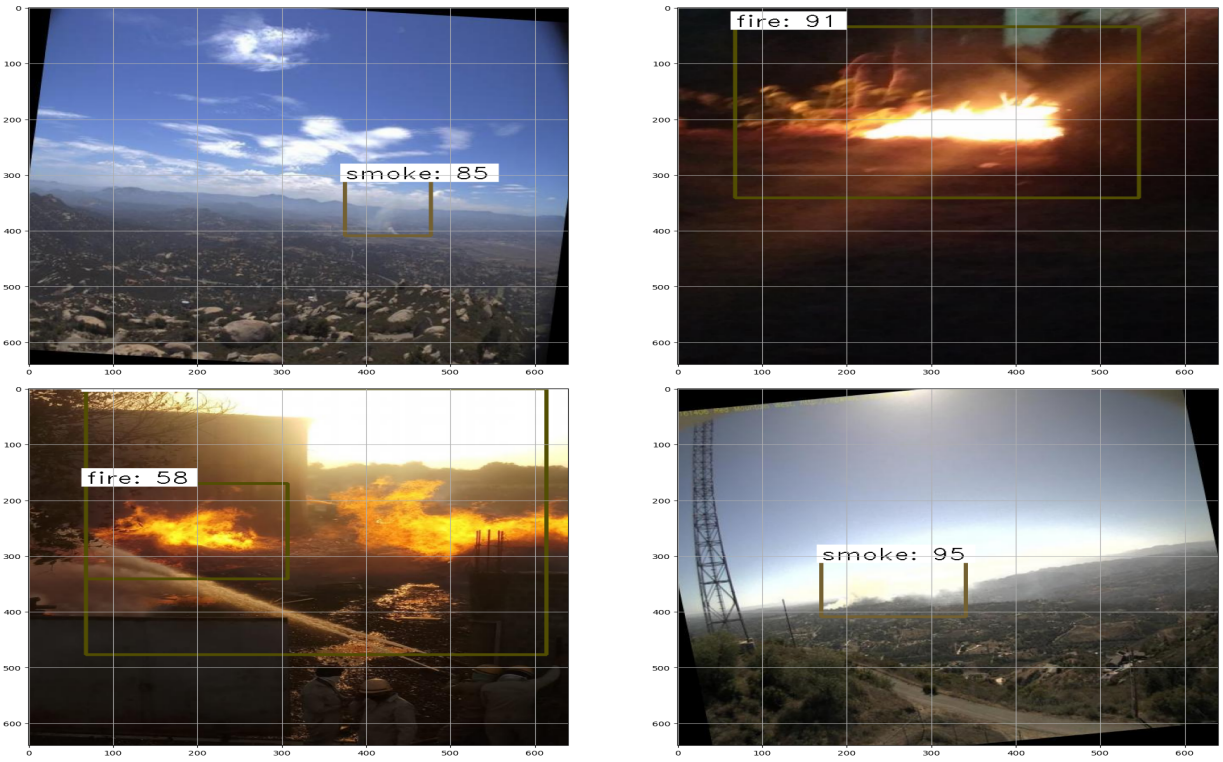


Figure 4.29: Inceptionv3 test images

4.2.1.3 Comparison between the two approaches

The two best models used on D6 were YOLOv8n and faster RCNN with the hybrid model Xception-VGG19 as a backbone.

Table 4.4: Comparison between one stage two stages results

	Method used	mAP@0.5
Our approache	YOLOv8n	92%
	Faster RCNN (Xception-vgg19)	44%

4.2.1.4 Comparison between related work and our contribution

From the previous related work that we mentioned before, we notice that they had better machines and hardware compared to our work, we also observe that our contribution is the second one of all time in fire detection to use faster RCNN and the first to use Xception, MobileNet, Inceptionv3, and the hybrid model Xception-VGG19 as backbones. Additionally, our study outperformed the work that employed the same model by utilizing YOLOv8. Furthermore, the usage of YOLO's latest version, the majority of those works focused on outdoor fire more specifically forest or wildfire, while we focused on both indoor outdoor and forest fire and smoke detection. Table ?? summarizes our study compared to related work.

Work	Used methods	Equipments used	Indoor	Outdoor	Flame	Smoke	Results
Dai P et a	Object detection YOLOv3	GPU is GeForce GTX 1080, CPU is Intel(R) Core(TM) I7-3960X, 32G memory	×	×	×		mAP@0.5=96.3%
Avazov et al	Object detection YOLOv4	3.20 GHz CPU, 32 GB RAM, nd two Nvidia GeForce 1080Ti GPUs (Nvidia, Santa Clara, CA, USA)	×	×	×		mAP@0.5=99.8%
Mimoun YANDOUZI et al	-YOLOv8n -Faster RCNN (resnet50 fpn)	Two Intel Gold 6148 (2.4GHz/ 20-core) processors.Two NVIDIA Tesla V100 graphics cards, each having 32GB of RAM training with 500k epochs for Faster RCNN, and 1k epochs for YOLO		×	×		mAP@0.5=89.45% mAP@0.5=90.57%
Our contribution	**YOLOv8n**	**Google Colab's Nvidia Tesla t4 training with 100 epochs for Faster RCNN and 150 epochs for YOLOv8**	×	×	×	×	**mAP@0.5=92%**
	Faster RCNN Xception-VGG19 (concatination)						**mAP@0.5=44%**

4.2.1.5 Results discussion

From the test results, we notice that the YOLO performed better than the Faster RCNN and that could be due to the fact that Faster RCNN is a two-staged architecture so it takes a considerably much longer time to train. Hence, outcome better results, and because our training time on Google Colab was limited we only had 5 hours a day to train two stages. However, YOLO had the whole 5 hours to train its only one stage.

When it comes to the results of the proposed backbone for the faster RCNN we find that the one we created Xception-VGG19 has the highest mAP and better results than each backbone trained separately.

4.2.2 Simulation using UAVs

We developed a scenario for combating such events with UAVs in forests or large expanses. The following scenario takes place after the detection of fire using our models via cameras. Once the fire or smoke is detected the nearest UAV with a sufficient battery and water tank will be summoned to extinguish it, however, if the fire cannot be extinguished by one UAV, multiple UAVs must be summoned; if the fire continues to exist even with the assistance of those UAVs, the last resort is to ask for the intervenience of the fire department.

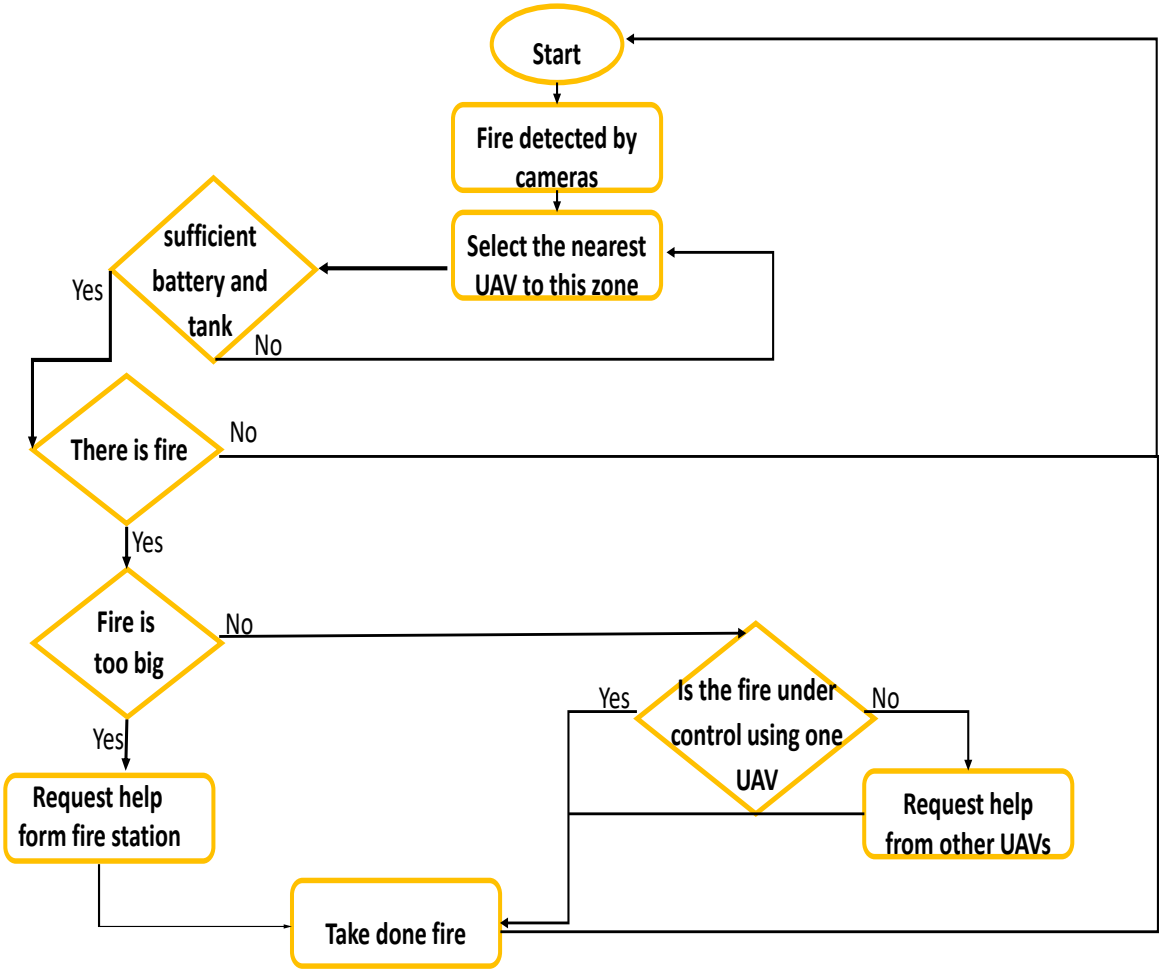


Figure 4.30: Our proposed simulation scenario

4.3 Conclusion

In our study, the detection of fire and smoke was done with two approaches of object detection YOLO and Faster RCNN. We choose the Deep Transfer Learning approach. We used VGG 16/19, Xception, a concatenation between VGG19 and Xception, MobileNet, and Inceptionv3 for the Faster RCNN approach, and for YOLO we used YOLOv5n and YOLOv8n with 6 datasets. After the training and testing, we demonstrated that D6 using YOLOV8 had the best results with an mAP@50=92% because it was the biggest dataset among the six. In addition to that, it contained both fire and smoke images . After that, we trained D6 using the Faster RCNN, we found that the Xception-VGG19 had the greatest results, with an mAP@50 of 44%. In the second part of our study, the simulation phase is supposed to be implemented using UAVs. However, due to a lack of time and documentation about tools for the UAVs simulation, we were incapable to implement our proposed scenario.

5

Conclusion and future perspectives

Contents

5.1 General conclusion

We proposed a deep learning-based approach for fire and smoke detection. Our approach is divided into two parts, the first part is for detection, it includes one stage of detection using YOLO and two stages detection using Faster RCNN. The second part is for the simulation.

The detection part starts with the preprocessing phase, where we collected several datasets and then labeled them, for the one stage detection we used YOLOv5n and YOLOv8n, meanwhile for the two stages detection we used Faster RCNN with different models as backbones, which are VGG16/19, Xception, the hybrid model Xception-VGG19, Mobilenet, and Inceptionv3, the obtained results showed that the usage of one stage detection in our case had better results than the two stages detection where YOLOv8 had an mAP@0.5 at 92% and faster RCNN with the hybrid model Xception-VGG19 had an mAP@0.5 at 44%. The reason behind the big difference between the two approaches' mAP@0.5 is the limited time we had for the training phase.

Our approach with the Faster RCNN is the only one that includes several backbone models such as Inceptionv3 and Xception and Mobilenet, additionally, our hybrid model that concatenates each of VGG19 and Xception. As far as the simulation part we made a scenario using UAVs to take down fire once detected in forests or wide areas

5.2 Limitations

Even though we reached great results for smoke and fire detection, our approach had many limitations:

- The major limitation of the project is the lack and limited time for training on Google Colab taking into consideration the size of our dataset

- The lack of hardware equipment when it comes to the GPU
- The other limitation we had was that we couldn't implement the simulation part due to a lack of time and documentation.

5.3 Future Perspectives

our future perspectives are:

- Train the faster RCNN models for a longer period in better hardware in order to achieve more accurate results.
- Implement the simulation using UAVs for outdoor and forest detection with OMNET++ or NS2
- For the indoor fire, we want to create a system with CCTVs that sends a notification attached with the live streaming or images to the user if fire or smoke is detected and if the detection is false the user indicates that its false if not he can call the fire department.
- Try different backbones for the one-stage detection.

Bibliography

[1] Martin Aruldoss Enoch Arulprakash. A study on generic object detection with emphasis on future research directions, journal of king saud university - computer and information sciences, volume 34, issue 9,. 2022.

[2] Pei Wang. What do you mean by "ai"?

[3] https://blog.paperspace.com/faster-r-cnn-explained-object-detection/.

[4] Muhammad K Hussain T Del Ser J. Cuzzolin F. Bhattacharyya S. Akhtar Z. de Albuquerque V.H.C. Khan, S. Deepsmoke: Deep learning model for smoke detection and segmentation in outdoor environments. (2021).

[5] Y.; Braun H.-W.; Vernon F.; Perez I.; Altintas I.; Cottrell G.W.; Nguyen M.H. Dewangan, A.; Pande. Figlib smokeynet: Dataset and deep learning model for real-time wildland fire smoke detection. doi:https://doi.org/10.3390/rs14041007. (2022).

[6] Lin G-Shafique MM Huo Y Tu R Dai P, Zhang Q and Zhang Y. Multiscale video flame detection for early fire warning based on deep learning. front. energy res. 10:848754. doi: 10.3389/fenrg.2022.848754. (2022).

[7] M.; Makhmudov F.; Cho Y.I. Avazov, K.; Mukhiddinov.

[8] Raghad K. Mohammed.

[9] Saida Sarra Boudouh and Mustapha Bouakkaz. Breast cancer: Toward an accurate breast tumor detection model in mammography using transfer learning techniques. 2022.

[10] https://www.researchgate.net/figure/illustration-of-the-mobilenet-architecture-a-the-overall-mobilenet-architecture-and$_fig3_340470168$.

[11] et. al Szegedy. Rethinking the inception architecture for computer vision. 2015.

[12] John McCarthy. what is artificial intelligence. 2004.

[13] Stuart Russell and Peter Norvig. Artificial intelligence: A modern approach. pearson education, 3rd edition. 2009.

[14] Roger C. Schank. What is ai, anyway? ai magazine. 1987.

[15] Alan Turing. Computing machinery and intelligence. pages 433–460.

[16] Kaplan A Haenlein, M. A brief history of artificial intelligence: On the past, present, and future of artificial intelligence. 2019.

[17] A. Joseph Hoane Jr. Murray Campbell and Feng-Hsiung Hsu. "deep blue," artificial intelligence,. pages 57–83, 2002.

[18] Donald Olding Hebb. The organization of behavior: A neuropsychological theory. 1949.

[19] https://www.red-gate.com/simple-talk/development/data-science-development/introduction-to-artificial-intelligence/.

[20] Mounime El Kabbouri Casa Bp Settat Hassan Wassima Lakhchini, Rachid Wahabi. Artificial intelligence machine learning in finance: A literature review. . *International Journal of Accounting, Finance, Auditing, Management and Economics*, 2022.

[21] Abedin et al. 2019.

[22] Cord M. Delany S. J. (n.d.) Cunningham, P. Supervised learning. cognitive technologies. page 21–49.

[23] Schwenker F. Hady, M.F.A. Semi-supervised learning. in: Bianchini, m., maggini, m., jain, l. (eds) handbook on neural information processing. intelligent systems. 2013.

[24] B.YEGNANARAYANA. Artificial neural networks. page 24.

[25] Pramila P. Shinde and Seema Shah. A review of machine learning and deep learning applications. 2018.

[26] Saida Sarra Boudouh and Mustapha Bouakkaz. Breast cancer: Breast tumor detection using deep transfer learning techniques in mammogram images. 2022.

[27] Nitin Kumar Chauhan and Krishna Singh. A review on conventional machine learning vs deep learning. 2018.

[28] Liu Z Chen Y Li Y Zhu H Gao M Hou H Wang C Xin Y, Kong L. Machine learning and deep learning methods for cybersecurity. 2018.

[29] Xu.f et al. Explainable ai: A brief survey on history, research areas, approaches and challenges.

[30] Frederic Jurie Shivang Agarwal, Jean Ogier du Terrail. Recent advances in object detection in the age of deep convolutional neural networks. 2019.

[31] Mohammed BERRAHAL Idriss IDRISSI Omar MOUSSAOUI Mostafa AZIZI Kamal GHOUMID Aissa KERKOUR ELMIAD Mimoun YANDOUZI, Mounir GRARI.

[32] https://www.statista.com/statistics/1281357/algeria-area-burned-by-wildfire/. 2023.

[33] Create production-grade machine learning models with tensorflow[online]. https://www.tensorflow.org. 2023.

[34] Keras[online]. https://keras.io. 2023.

[35] www.pexels.com.

[36] https://www.kaggle.com/datasets.

[37] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[38] Francois Chollet. Xception: Deep learning with depthwise separable convolutions,proceedings of the ieee conference on computer vision and pattern recognition (cvpr). 2017.

[39] Keras concatenate layers [online]. https://keras.io/api/layers/merging$_layers/concatenate/$.2023.

[40] How to merge two different models in Keras. https://www.educative.io/answers/how-to-merge-two-different-models-in-keras. 2023.

[41] mobilenet. https://builtin.com/machine-learning/mobilenet. 2023.

[42] inceptionv3. https://cloud.google.com/tpu/docs/inception-v3-advanced. 2023.

[43] https://towardsdatascience.com/anchor-boxes-the-key-to-quality-object-detection-ddf9d612d4f9.