وزارة التعليم العالي والبحث العلمي

*Ministry of Higher Education and Scientific Research*

**University of Saad Dahlab Blida 1**

**Institute of Aeronautics and Space Studies**

**Department of Aeronautical Construction**

**Manuscript**

**Towards the attainment of the degree of**

**Master's in Aeronautics**

*Specialty:* Avionics

# UAV Aerial Image-Based Forest Fire Detection Using Artificial Intelligence

*Startup project presented within the framework of ministerial decree "1275", conducted by the Institute of Aeronautics and Space Studies (IAES).*

*Under the Supervision of:*

*Dr. Ahmed Kechida*

*Dr. Hocine Bentrad*

*Submitted by:*

*Nesrine Touahria,*

*Romaissa Bouhamam*

*Jury Members:*

| | | |
|---|---|---|
| *Dr. Saliha Benchikh* | *MCA* | *President* |
| *Dr. Boussad Azmedroub* | *MCB* | *Examiner* |
| *Dr. Amine Taberkit* | *MCB* | *Examiner* |

*Promotion: 2022 / 2023*

## ACKNOWLEDGMENT

First and foremost, we would like to express our gratitude to Allah for all the knowledge we have acquired.

I extend my sincere appreciation to my two advisors, Dr.Amine Taberkit and Dr.Ahmed Kechida, for their expert guidance, valuable insights, and constant encouragement. Their expertise and mentorship have played a pivotal role in shaping the direction of my research and enhancing the quality of this thesis.

I would like to express my gratitude to the CRTI (Centre for Research in Industrial Technologies) for providing the necessary resources and infrastructure that facilitated my research work. Their commitment to scientific excellence and their support in material-related research have been instrumental in the successful completion of this thesis.

I would also like to extend my appreciation to Mr. Djamel Touahria Forest General director and Mr. Fritas Protection director for providing the valuable data required for my research. Their cooperation and willingness to share their expertise have significantly enriched the findings of this thesis.

To my beloved family, I am indebted to your unwavering love, encouragement, and understanding throughout this challenging journey. Your constant belief in my abilities, patience during moments of stress, and words of motivation have been my driving force. Your sacrifices and steadfast support have meant the world to me, and I cannot thank you enough for being my pillars of strength.

A special mention goes to my study partner and best friend Romaissa Bouhamam, whose unwavering support, late-night discussions, and brainstorming sessions have made this academic journey enjoyable and meaningful. Your presence has been invaluable, and I am grateful for the camaraderie we shared.

I warmly thank the jury members for agreeing to preside over and examine our work. We express our heartfelt gratitude for their presence and consideration.

Last but not least, I am grateful to my friends who have been a constant source of motivation, laughter, and support throughout this academic journey.

To all those mentioned above and to anyone else who has played a part in shaping my academic and personal growth, I offer my heartfelt gratitude. Your contributions have made this thesis possible, and I am forever indebted to you.

**Nesrine Touahria**

# ACKNOWLEDGMENT

First and foremost, we would like to express our gratitude to Allah for all the knowledge we have acquired.

I would like to extend my heartfelt gratitude to my two advisors, Dr. Amine Taberkit and Dr. Ahmed Kechida. Their expert guidance, invaluable insights, and unwavering support have significantly shaped the trajectory of my work and enhanced its overall quality. I am truly grateful for their mentorship throughout this journey.

I also wish to convey my deep appreciation to the CRTI (Centre for Research in Industrial Technologies) for their provision of essential resources and state-of-the-art infrastructure. Their unwavering commitment to scientific excellence and their support in the specific area of my research have been pivotal in the successful completion of this project.

Furthermore, I would like to express my profound gratitude to Mr. Djamel Touahria Forest General Director, and Mr. Fritas for their assistance in providing the indispensable data required for my research. Their cooperation and willingness to share their expertise have greatly enriched the findings of this project.

To my beloved family, I want to express my utmost gratitude for your unwavering love, understanding, and encouragement throughout this endeavor. Your unwavering belief in my abilities, patience during challenging moments, and motivating words have been my driving force. I am truly blessed to have such a supportive and loving family by my side.

A special mention goes to my study partner and dear friend, Nesrine Touahria. Your unwavering support, collaborative efforts, and thought-provoking discussions have made this academic journey incredibly rewarding and meaningful. Your friendship has been an invaluable source of inspiration and motivation.

I warmly thank the jury members for agreeing to preside over and examine our work. We express our heartfelt gratitude for their presence and consideration.

Lastly, I am grateful to all my friends who have been there for me with their unwavering support, encouragement, and shared laughter. Your presence in my life has made this journey more enjoyable and memorable.

To all those mentioned above and to all others who have contributed to the realization of this project, I offer my heartfelt gratitude. Your contributions and support have been indispensable, and I am forever thankful for your involvement and dedication.

Thank you all

**Romaissa Bouhamam**

## ABSTRACT

In the past years, 40 831 hectares of forests have been devastated by rampant wildfires, resulting in the tragic loss of numerous lives. To address this challenge, our research focuses on developing an early wildfire detection system capable of identifying potential fire outbreaks before they escalate, as controlling them once they have spread becomes arduous. Our proposed approach utilizes unmanned aerial vehicles (UAVs) to capture aerial data, which is then processed on board to automatically detect early signs of wildfires. This enables us to promptly alert relevant emergency services and facilitate a rapid response.

The core technique employed in our approach is transfer learning, specifically applied to the YOLOv3 model for object detection through several Batch sizes and epochs. We validate the effectiveness of our model using FLAME dataset. The performance metrics we have achieved demonstrate the success of our approach, with Precision, Recall, F1-Score and Accuracy rates reaching impressive levels of 100%, 96.66667%, 98.305085%, and 96.66667% respectively.

**Keywords:** Wildfires, UAVs, detection, transfer learning, YOLOv3, early detection.

## Résumé

Ces dernières années, de vastes étendues de forêts ont été ravagées par des incendies dévastateurs, entraînant la perte tragique de nombreuses vies. Afin de relever ce défi, notre recherche se concentre sur le développement d'un système de détection précoce des incendies capable d'identifier les débuts potentiels d'incendie avant qu'ils ne s'aggravent, car il est difficile de les maîtriser une fois qu'ils se sont propagés. Notre approche proposée repose sur l'utilisation de drones pour recueillir des données aériennes, qui sont ensuite traitées à bord pour détecter automatiquement les premiers signes d'incendie. Cela nous permet d'alerter rapidement les services d'urgence concernés et de faciliter une réponse rapide.

La technique clé utilisée dans notre approche est le transfert d'apprentissage, appliqué spécifiquement au modèle YOLOv3 pour la détection d'objets à travers plusieurs tailles de lots (batch sizes) et époques (epochs). Nous validons l'efficacité de notre modèle en utilisant l’ensemble de données FLAME. Les mesures de performance que nous avons obtenues démontrent le succès de notre approche, avec des taux de Précision, rappel, de score F1 et d'exactitude très satisfaisants qui sont: 100%, 96.66667%, 98.305085%, et 96.66667% respectivement.

**Mots clés :** Incendie, drone, détection, le transfert d’apprentissage, YOLOv3, stade précoce.

## ملخص

في السنوات الأخيرة، تعرضت مساحات شاسعة من الغابات لحرائق مدمرة، مما أدى إلى فقدان العديد من الأرواح بشكل مأساوي. من أجل مواجهة هذا التحدي، يركز بحثنا على تطوير نظام مبكر لاكتشاف الحرائق قادر على تحديد بدايات الحرائق المحتملة قبل أن تتفاقم، حيث أصبح من الصعب جدًا السيطرة عليها بمجرد انتشارها. يعتمد النهج المقترح على استخدام الطائرات بدون طيار لجمع البيانات الجوية، ومن ثم معالجة تلك البيانات على متن الطائرة للكشف التلقائي المبكر عن بؤر الحرائق. وبذلك، نتمكن من إبلاغ فرق الطوارئ بشكل سريع وتسهيل التدخل السريع.

التقنية الأساسية المستخدمة في نهجنا هي نقل التعلم، وهي تُطبق بشكل خاص على نموذج YOLOv3 لاكتشاف مختلف الاغراض من خلال عدة أحجام للدُفعات وفترات زمنية. نثبت فعالية نموذجنا باستخدام مجموعة بيانات FLAME. تظهر قياسات الأداء التي حققاها نجاح نهجنا، حيث بلغت معدلات الضبط والاستدعاء ونسبة F1 والدقة مستويات مذهلة تبلغ 100٪ و96.66667٪ و98.305085%و96.66667٪ على التوالي.

**الكلمات المفتاحية:** الحرائق، الطائرات بدون طيار، نقل التعلم، الكشف التلقائي المبكر، اكتشاف، نموذج YOLOv3.

# TABLE OF CONTENTS

# LIST OF TABLES

# LIST OF FIGURES

# LIST OF ABBREVIATIONS

| | |
|---|---|
| **AI** | Artificial Intelligence |
| **ANN** | Artificial Neural Network |
| **AP** | Average Precision |
| **API** | Application Programming Interface |
| **BGD** | Batch Gradient Descent |
| **CNN** | Convolutional Neural Network |
| **COCO** | Common Objects in Context |
| **CPU** | Central Processing Unit |
| **CUDA** | Compute Unified Device Architecture |
| **DGF** | Direction Generale des Forets |
| **DL** | Deep Learning |
| **FLAME** | Fire Luminosity Airborne-based Machine learning Evaluation |
| **FN** | False Positive |
| **FP** | False Negative |
| **GD** | Gradient Descent |
| **GPS** | Global Positioning System |
| **GPU** | Graphical Processing Unit |
| **IDE** | Integrated Development Environment |
| **IMU** | Inertial Measurement Units |
| **IoU** | Intersection over Union |
| **LR** | Learning Rate |
| **mAP** | Mean Average Precision |
| **ML** | Machine Learning |
| **MLL** | Machine Learning Library |
| **NMS** | Non-Maximum Suppression |
| **NN** | Neural Network |

| | |
|---|---|
| **PASCAL VOC** | Pattern Analysis, Statistical Modelling, and Computational Learning Visual Object Classes |
| **ReLU** | Rectified Linear Unit |
| **RGB** | Red Green Blue Camera |
| **SGD** | Stochastic Gradient Descent |
| **Tanh** | Hyperbolic Tangent |
| **TN** | True Negative |
| **TP** | True Positive |
| **TPU** | Tensor Processing Unit |
| **UAV** | Unmanned Aerial Vehicle |
| **VGG** | Very Deep Convolutional Networks |
| **WSN** | Wireless Sensor Networks |
| **XML** | Extensible Markup Language |
| **YOLO** | You Only Look Once |
| **YOLOv1** | You Only Look Once Version 1 |
| **YOLOv2** | You Only Look Once Version 2 |
| **YOLOv3** | You Only Look Once Version 3 |

# GENERAL INTRODUCTION

Forests are a crucial part of our planet's ecosystem, providing a range of essential resources and services that benefit both humans and wildlife. They are often referred to as the "lungs of the planet" because they purify the air by absorbing carbon dioxide and releasing oxygen. Forests also provide a habitat for a diverse range of species and play a critical role in maintaining the planet's biodiversity. Apart from their ecological importance, forests offer valuable economic benefits by providing a range of resources, such as timber, fuel wood, fruits, nuts, and medicinal plants. In addition, they support various industries, including ecotourism, which generates revenue through activities such as camping, hiking, bird watching, and wildlife viewing. Ecotourism can create jobs and provide income for local communities, promoting sustainable development and encouraging the conservation of natural resources. To recognize the significance of forests, March 21st marks the annual celebration of the International Day of Forests, a day dedicated to raising awareness about the importance of forests, their vital role in supporting biodiversity, and their many contributions to the well-being of people and the planet. However, a significant issue that forests are currently experiencing is the occurrence of wildfires.

Wildfires have become a growing global concern due to their devastating impact on the environment, economy, and public health. These uncontrollable disasters can destroy vast areas of forests and wild lands, leading to the loss of critical resources such as timber, and the destruction of crops which can have serious implications for food security.

In recent years, the world has witnessed several catastrophic wildfires that have caused immense destruction of natural resources, loss of human life, and threatened public health and safety. The Australian bushfires of 2019-2020 burned over 47 million acres of land and killed over 30 people. The California wildfires of 2020 were one of the largest wildfire outbreaks in the state's history, burning over 4 million acres of land and destroying over 10,000 structures. In addition, the Amazon Rainforest fires of 2019, were a major environmental disaster and burned over 2.2 million acres of land. According to Vardoulakis et al (S. Vardoulakis, 2020). And Bo et al (M. Bo, 2020), climate change is the fundamental cause of the aforementioned wildfires. The vast majority of wildfires that occurred in the United States between 1992 and 2015 were caused by human activities, accounting for around 85% of incidents (Short, 2017). In contrast, natural causes such as lightning strikes and climate change only contributed to approximately 15% of wildfires during that time. This underscores the critical role that human behavior plays in wildfire occurrence. Regardless of the causes of wildfires, the problem must be taken seriously.

In Algeria, the forest faces a serious danger from wildfires, which have already caused irreversible harm to the ecology, environment, economy, and citizens wellbeing in various provinces. Additionally, the threat of forest fires is intensified by inadequate prevention measures during the summer season, as well as human negligence that includes irresponsible acts like lighting barbecues and carelessly disposing of cigarette butts in forested areas. However, criminal activities contribute to amplifying the risk of forest fires.

These devastating events highlight the urgent need for effective measures to prevent and control wildfires and mitigate their impact on the environment and human life. The importance of early wildfire detection cannot be understated, as it serves as a crucial factor in reducing the risks and losses associated with these fires, enabling firefighters to extinguish them at an early stage.

Methods based on remote sensing using drones, and the processing of aerial data acquired by deep learning techniques allow the automatic detection of fires at their early stages and reduce the rate of false alarms, to accelerate the intervention of the civil protection services. Also deploying drones is far cheaper with less risk, compared to other methods, which are extremely expensive.

In this kind of problem, the trade-off between the speed and the precision of the model used is important, to have fast detection while reducing false alarms, this is why we chose to study the effectiveness of the YOLOv3 model in the early detection of wildfire.

Our thesis is structured into three chapters. In the first chapter, we present the problem of forest fires and some statistics that highlight the danger and damages of the aforementioned fires. We present then, some prevention strategies used, to reduce the risk of these fires, which are considered a major risk.

The second chapter is devoted to presenting the generalities of deep learning as well as the most important notions on the architecture of convolutional neural networks. This chapter prepares the thesis reader to properly understand the methods and results presented in the third chapter.

In the third chapter, we will present the YOLO model. We will also present some criteria used in deep learning, which allow evaluation of the performance of the trained model. Next, we present the data Labelling method and the process of model training. Finally, we present the results obtained and some perspectives on this work.

# CHAPTER 1. EFFECTIVE STRATEGIES AND APPROACHES TO CONFRONT FOREST FIRES IN ALGERIA

## 1.1 Introduction

Forest fires have the potential to cause extensive economic and ecological losses as well as endangering human lives. Therefore, there is growing worldwide attention paid to the early detection and monitoring of forest fires, to preserve natural resources and safeguard people and property.

In order to combat the threat of forest fires, there is a need to employ more efficient and rapid fire detection techniques to prevent the fires from spreading and causing extensive damage.
Recent developments in technology, such as the use of drones and advanced machine learning algorithms, have provided new solutions for monitoring forests and detecting fires in a timely manner, enabling faster response and containment of the fires.

This opening chapter provides a broad understanding of forest fire detection, encompassing statistics on forest fires in Algeria, methods used for detecting forest fires, and the significance of drones in addressing this issue. Moreover, it offers an overview of recent studies conducted on this matter to provide a comprehensive view of the problem.

## 1.2 Presentation of the Algerian Forest

The Mediterranean forest is among the most important forest ecosystems in the world. It is a fragile and disturbed natural environment, constantly influenced by several unfavorable factors, particularly climatic conditions, and human pressure. It covers approximately 65 million hectares, while pre-forest formations extend over nearly 19 million hectares.

In Algeria, the forest plays a very important role from an ecological and socio-economic perspective. It extends over an area of 2.381 million square kilometers. However, forest cover in Algeria only occupies 4.1 million hectares, representing a woodland rate of 11% for the northern part of the country and only 1.7% if the arid Saharan regions are also taken into account. This woodland rate remains very low to maintain a physical and biological balance that allows for the conservation of natural resources (FOSA, 2001).

Currently, Algerian forest formations are dangerously exposed to several degradation factors, including deforestation, desertification, and soil erosion due to human activities such as overgrazing, extensive logging, and agriculture. In the face of this situation, enormous efforts are being made to preserve and develop these natural spaces within a framework of sustainable management.

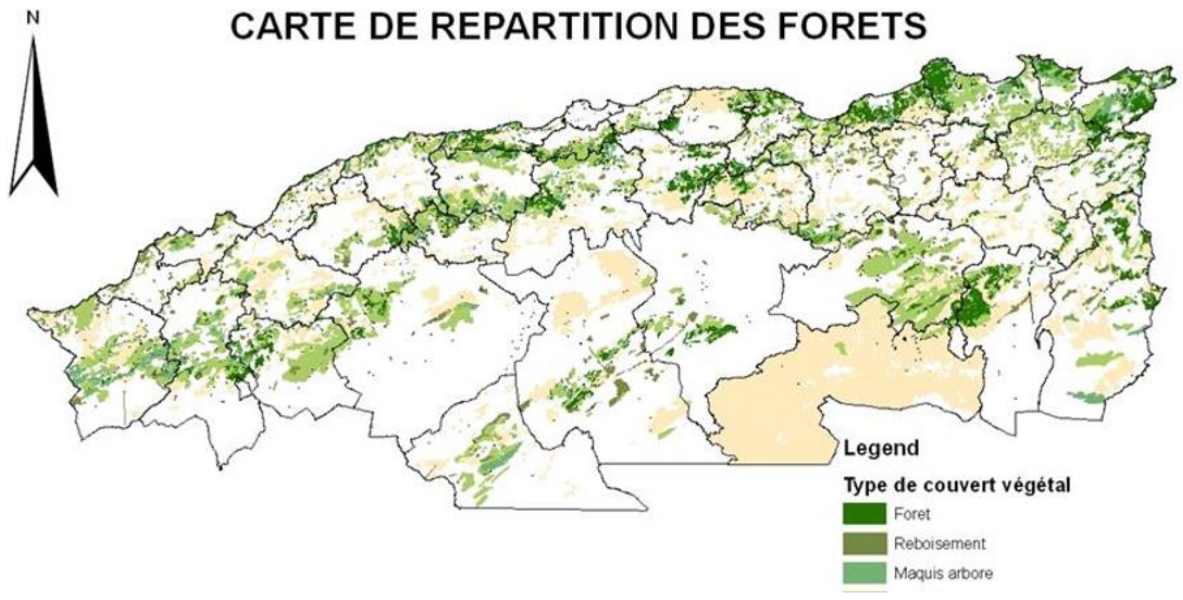


Figure 1.1. Distribution of forests in Algeria.

## 1.3 State of Vegetation Fires in Algeria

Forest fires are devastating events that occur when flames spread through a wooded area, destroying trees, wildlife, and natural habitats.

### 1.3.1 Distribution of Algerian Forests by Species

In general, the main tree species in Algeria cover 1,491,000 hectares, scattered throughout the country as shown in Figure 1.2 and Table 1.1.

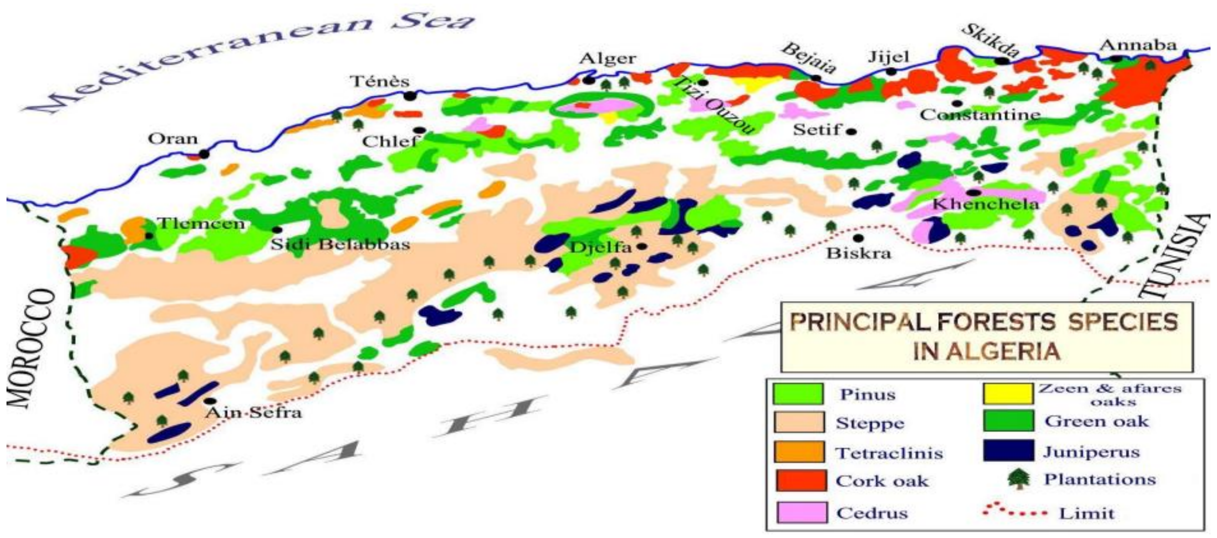


Figure 1.2. Map shows the major principal forest species in Algeria; 60% of the forest are Aleppo pine (Pinus halepensis) (Kazi Aoual N., 2010).

| | Forest tree species | | | | | | |
|---|---|---|---|---|---|---|---|
| | Aleppo pine | Cork oak | Zeen oak | Atlas cedar | Eucalyptus | Maritime pine | Various |
| Surface(HA) | 1.158.533 | 349.218 | 34.922 | 32.909 | 29.355 | 28.490 | 68.391 |

Table 1.1. The surface area in hectares (ha) of the main forest tree species in Algeria.

- **Aleppo Pine Forest:** This species extends from the humid to semi-arid zones, occupying the largest area in Algeria. It is capable of surviving with low rainfall levels, around 350 mm per year. Aleppo pines are known for their high adaptability to the challenging environmental conditions of this region.
- **Cork Oak Forest:** These forests hold great ecological and economic importance in Algeria. They typically thrive in the humid and sub-humid bioclimatic zone, mainly located in the western and central regions of the country. The cork oak forests stretch across a 450-kilometer strip from Algiers to Cap Roux.
- **Deciduous Oak Forests:** Represented by the Zeen Oak and the Afarès Oak, these forests flourish with rainfall levels of at least 800 mm. They colonize the eastern regions of the country, from Kabylie to the Tunisian border.
- **Holm Oak:** The holm oak is an evergreen tree that can reach heights of up to 20 meters. It is well adapted to the challenging environmental conditions of the Mediterranean climate, characterized by hot and dry summers and mild and humid winters.
- **Maritime Pine:** Maritime pine is naturally present in northern Algeria and tends to establish itself in moderately degraded cork oak forests (FOSA, 2001).

- **Atlas Cedar:** Thriving at altitudes ranging from 1400 to 2800 meters, the Atlas cedar is found in humid and cold zones. It can be found in the Aurès, Djurdjura, Atlas Blidien, and Teniet El Haad regions.

### 1.3.2 Forest fires in Algeria

The total mapped burnt area of the 2022 forest fire prevention and control campaign shows that 27,684.56 Ha of the area was consumed by fire, caused by 1,607 fires spot, including:

- 10,421.21 Ha of bushes (i.e. 38% of the total area covered by the fires)
- 8465.60 Ha of brushwood (i.e. 31% of the total area covered by fires)
- 6791.30 Ha of forest (i.e. 25% of the total area covered by the fires)
- 1925.58 Ha of mountain fruit trees (i.e. 7% of the total area)
- 80.87 Ha of Alfa (i.e. 0.29% of the total area covered by fires)

The vegetation formation most affected by forest fires is the bushes with 38% of the total lost. Knowing that the bushes are a representation of degraded and undeveloped forests, fire takes hold and spreads easily.

#### 1.3.2.1 Comparative study during last 10 years

- **In terms of burnt area**

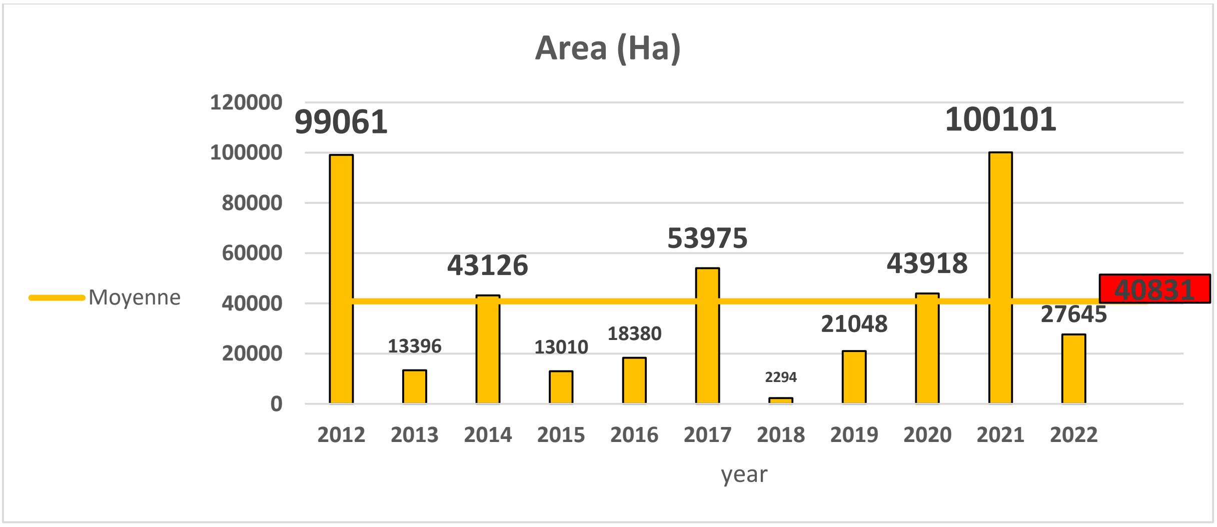


Figure 1.3. Burnt Area (Ha) (DGF, 2022).

If we compare the burned area of the 2022 campaign with those of the last ten years, we see that the total burned area is below the average, which is 40 831 Ha on average during the last decade. The graph shows also the lost area caused by forest fires and we notice that the year of 2018 was the least affected burnt area with only 2294 Ha succeeded by 2015 with 13010 Ha; 2013 with 13396 Ha and the most affected burnt area was registered in 2021 with 100101 Ha followed by 99061 Ha in 2012.

- **In terms of the number of forest fire outbreaks**

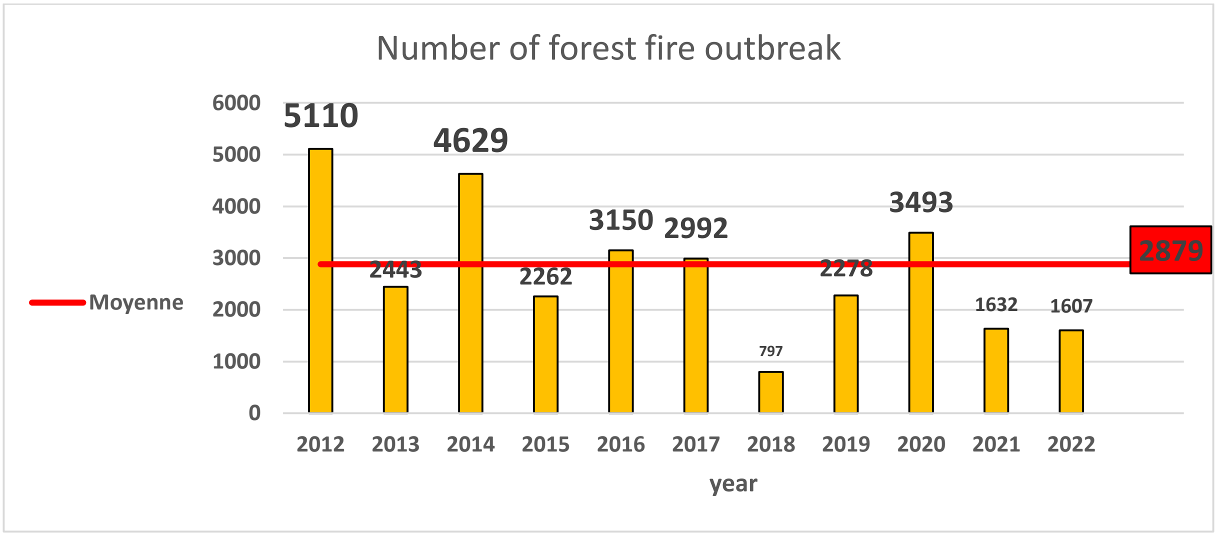


Figure 1.4. Number of forest fire outbreak (DGF, 2022).

The 2022 campaign is ranked second after 2018 in terms of the lowest number of forest fire over the last ten years with 1,607 and 797 respectively, knowing that the average is 2,879. We also note that the 2021 campaign recorded a slightly higher number of outbreaks, which is 1632. However, the highest number of forest fire was registered in 2012 with 5110 followed by 2014 with 4629.

### 1.3.2.2 Comparative between Wilayas

- **In terms of burnt area**

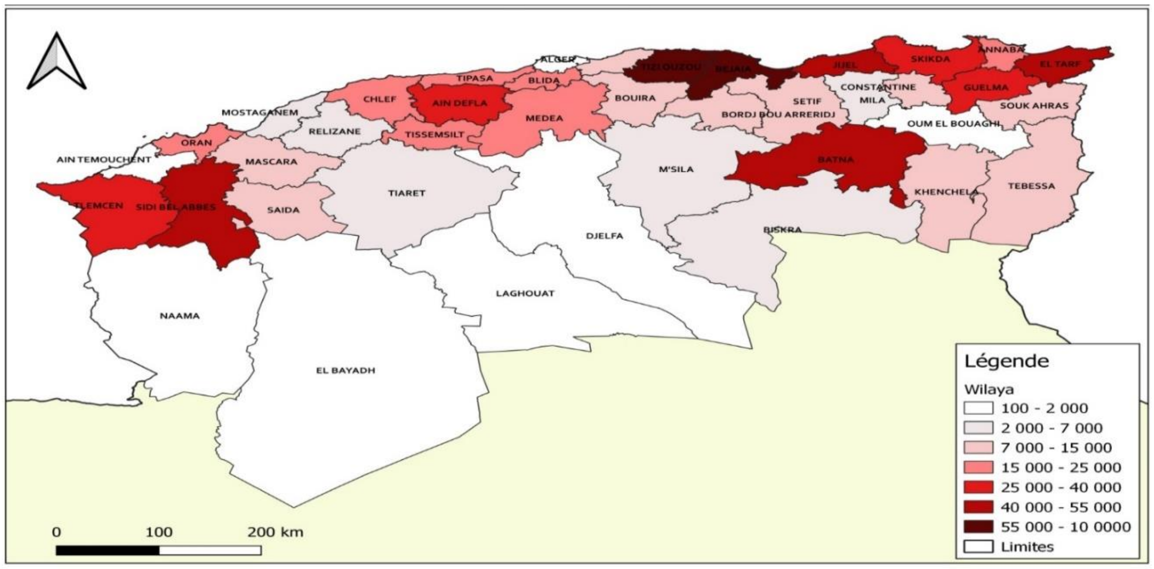


Figure 1.5. Map of cumulative burnt areas by Wilayas (2000-2022) (DGF, 2022).

This map shows that Tizi Ouzou and Bejaia are the most touched wilayas by burnt forest (cumulus between 55 000 to 100 000 Ha in 22 years), followed by El Tarf, Batna and Sidi Bel-Abbes the second row (40 000 to 55 000 Ha).

- **In terms of the number of forest fire outbreaks**

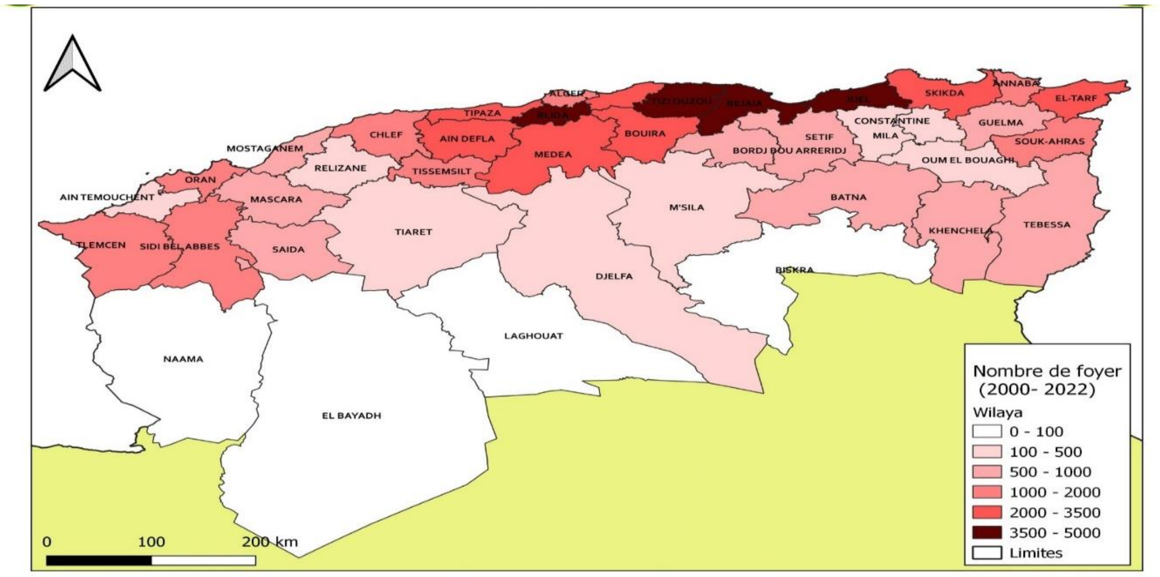


Figure 1.6. Map of cumulative number of forest fire outbreaks by Wilayas (DGF, 2022).

It shows that Tizi ouzou, Bejaia, Jijel and Blida are the most wilayas with more number of forest fire (3500 up to 5000 in 22 years), in the second row comes El Taref, Skikda, Boumerdes, Bouira, Medea, Ain Defla and Tipaza (2000 to 3500), we notice also that Batna and Sidi-Bel-Abbes have not a big number of forest fire outbreak yet we found them in the most wilaya touches by forest fire.

### 1.3.3 The origin of forest fires

The generation of forest fires can occur due to various contributory factors, including:

#### 1.3.3.1 Predisposition factors

The characteristics of vegetation formations and the presence of favorable climatic conditions are significant factors that greatly contribute to the occurrence of forest fires. Indeed, vegetation becomes highly combustible when specific meteorological conditions exceptionally reunited. It should also be noted that the intensity and frequency of fires are closely related, on one hand, to the structure and composition of forest stands and, on the other hand, to the chemical composition of plant species, which determines their flammability. Therefore, maquis and garrigue vegetation are the most vulnerable pre-forest formations (Branka A., 2001) (Moreira F., 2001) (Nunes M.C.S., 2005) (Bajocco S. and Ricotta C., 2008). This vulnerability is particularly linked to their density and the nature of the species composing them, including their moisture content. In terms of climate, drought conditions, low air humidity, the occurrences of tempestuous and dry wind events, especially in the current context of global changes, contribute to a high level of flammability, thus increasing the risk of fires.

#### 1.3.3.2 Ignition factors

The natural conditions of ignition are generally attributed to the degree of flammability of plant species, which varies depending on their content of volatile substances or resin. The moisture content influences both the inflammability and combustibility of vegetation (Margerit J., 1998).These factors can only have an impact in the presence of specific meteorological parameters that affect plant moisture content, such as temperature, wind, and sunlight.

However, human causes, overall, remain the main triggers of fire outbreaks in Mediterranean countries. In fact, there are unintentional causes resulting from carelessness and accidents, as well as intentional causes, sometimes of a criminal nature. In this context, the social, economic, political, and legislative contexts of each country must be taken into consideration in wildfire prevention plans.

#### 1.3.3.3 Propagation factors

There are several factors that contribute to the development and propagation of forest fires. Meteorological parameters, in particular, play a significant role in the intensity and spread of wildfires. The temperature heats up the fuels and the atmosphere, making the fires spread more easily. The wind is a meteorological parameter that increases combustion and propagation by drying out the fuels and promoting their heating. It also influences the direction and spread of fires by carrying sparks or other ignited materials over long distances. On a topographical level, the slope and exposure of the terrain directly affect the behavior of fires. The rate of spread, for example, doubles on a slope of 10 to 15 degrees and quadruples on a slope of 20 degrees (Arfa A.M.T., 2008). Exposure, on the other hand, impacts the amount of heat absorbed by the fuels based on solar radiation, which increases the flammability of the materials. Furthermore, it should be noted that within a single day, there are certain moments more conducive to the ignition and development of fires compared to others. Indeed, throughout the day, temperature, relative humidity, and wind vary and directly impact the burning cycle. The period between 1:00 PM and 6:00 PM is particularly challenging because the temperature reaches its highest level, fire intensity is at its maximum, and relative humidity is at its lowest. This period is critical, and firefighting conditions become more difficult.

### 1.3.4 Fires Consequence

Fires are catastrophic events that can cause considerable damage, ranging from loss of human lives to the destruction of homes, infrastructure, farmland, landscapes, natural heritage, and the sustainability of terrestrial ecosystems.

#### 1.3.4.1 Action on forest ecosystems

Fires have disastrous consequences for forest ecosystems and all forms of life they contain. The ecological impact of fires is complex and describes negative consequences on microorganisms, vegetation dynamics, fauna, landscapes, water cycle, and water quality. In terms of physiognomy, repeated fires are destructive as they represent a key factor in the depletion of biological diversity in ecosystems and can compromise forest sustainability. If the fire return interval is less than 30 years, it can lead to a progressive regression of forests, transforming them into garrigue or maquis. Additionally, fires are often followed by the colonization and infestation of insects that disrupt the ecological balance. Some native species disappear, while invasive plants proliferate.

#### 1.3.4.2 Action on the soil

The destruction of vegetation cover increases the risks of erosion and flooding due to runoff. (Aubert G., 1991) Highlights that fires lead to changes in the structure of the humus layer, reduction in water retention capacity, increased pH levels, higher limestone content due to rock fragmentation, and decreased total exchange capacity. The ashes also destroy a significant portion of the soil's nutrients.

#### 1.3.4.3 Impact of forest fires on the environment and health

Forest fires have a significant impact on the environment. They contribute to the greenhouse effect and climate change due to increased levels of carbon dioxide in the atmosphere. In terms of health, the fine particles present in the smoke generated by fires make breathing difficult and can worsen cardiovascular and respiratory diseases. Furthermore, the destruction of landscapes can have traumatizing effects on individuals' mental health.

#### 1.3.4.4 Socioeconomic impact of forest fires

On a socioeconomic level, forest fires have significant consequences resulting from the costs associated with fire suppression, damage to property (homes, infrastructure and vehicles), forest destruction, and the deterioration of ecosystem services. Indirect costs are also incurred, including the expenses of vegetation and landscape restoration, as well as the impact on the tourism and leisure economy.

## 1.4 Forest fire prevention and detection strategies

Many countries that have recognized the significant importance of forest fire monitoring have developed effective technologies, including monitoring via observation towers, cruising aircrafts, remote sensing using meteorological satellites, and sensor networks, to improve their response ability.

### 1.4.1 Manned observation Towers

Protecting wild animals, forests, and the environment from forest fires has long been a major concern in environment and natural resources management. Traditionally, detecting forest fires relied on human observation from fire lookout towers, strategically positioned on high vantage points such as hilltops and mountains, to swiftly discover and locate fires before they escalate into unmanageable disasters. Equipped with tools like the Osborne fire Finder, powerful binoculars, and telescopes, lookout towers facilitated the visual scanning of the horizon for any indications of smoke or flames. Forest rangers and firefighters stationed in these towers played a vital role in observing the surrounding areas, promptly identifying wildfires, and issuing timely alerts. The utilization of optical devices and other advanced technologies in these watchtowers not only alleviated the operator's workload but also enhanced the accuracy and reliability of fire detection and alarm systems. Consequently, watchtower-based systems remain prevalent in practice, ensuring an ongoing and valuable contribution to effective wildfire surveillance and efforts.

### 1.4.2 Satellites

The use of satellite images for detecting forest wildfires is already real. Due to the large number of satellites launched and the decrease of associated costs, research efforts are being focused on detecting forest fires using Earth observation (EO) satellites.

These satellites can be classified into different categories based on their orbit, such as:

- Geostationary orbit (GEO).
- Low Earth orbit (LEO).
- Polar sun-synchronous orbit (SSO).

Satellites in the SSO orbit are used for sun-synchronous satellites and have high spatial resolution but low temporal resolution. On the other hand, satellites in the GEO orbit have high temporal resolution but low spatial resolution.

In the realm of forest fire detection, various initiatives have been undertaken to harness the potential of satellite-based systems. Sun-synchronous satellites are particularly useful for this purpose, as they can provide a consistent and frequent coverage of the Earth's surface with high spatial and temporal resolution; therefore, there are various on-board sensors dedicated for forest fire detection in satellites (Barmpoutis P, 2020), including:

- NASA's Terra and Aqua satellite with MODIS (Moderate Resolution Imaging Spectroradiometer) sensor - launched in 1999 and 2002 respectively (Barmpoutis P, 2020).
- NOAA's Suomi National Polar-orbiting Partnership (Suomi NPP) satellite with VIIRS (Visible Infrared Imaging Radiometer Suite sensor) - launched in 2011 (Barmpoutis P, 2020).
- NOAA-20 satellite with VIIRS sensor - launched in 2017 (Barmpoutis P, 2020).

And in Geostationary Satellites:

- GOES-16 satellite with ABI (Advanced Baseline Imager) - Launched in 2016, operated by NASA and the NOAA (Barmpoutis P, 2020).

Despite the advancements in satellite-based systems for forest fire detection, their capabilities remain limited in detecting fires in their early stages. This is due to the fact that satellites follow orbits that restrict the acquisition of forest images around the clock. Moreover, the effectiveness of forest fire scanning by satellite-based systems is compromised by the quality of images, which is heavily influenced by weather conditions. Consequently, these challenges hinder the ability of satellite-based systems to provide a comprehensive and reliable solution for early forest fire detection (Yi, 2020).

### 1.4.3 Spotter plane

A spotter plane is an aircraft that is used for forest fire detection. It is equipped with a variety of sensors and equipment that allow it to detect and monitor wildfires. Spotter planes are often used in conjunction with ground-based fire crews to help coordinate firefighting efforts.

One of the key features of a spotter plane is its ability to quickly cover large areas of forested land. The plane can fly over the forest at a high altitude, using specialized sensors to detect the heat signatures of wildfires. Once a fire has been detected, the spotter plane can then relay the location and other relevant information to ground crews, allowing them to respond quickly and efficiently.

In addition to detecting fires, spotter planes can also be used to monitor the spread of existing fires. This information can be used to help predict the fire's behavior and make decisions about how best to allocate firefighting resources.

Overall, spotter planes play a critical role in forest fire detection and management, helping to prevent the spread of wildfires and protect communities and natural resources.

### 1.4.4 Camera surveillance

The utilization of optical sensors and camera networks has been widely adopted as a fire detection strategy. Two primary types of cameras used for early fire detection are optical cameras and infrared (IR) cameras, which possess the ability to capture data across a wide range of resolutions. Optical cameras provide color information, while IR cameras can measure the thermal radiation emitted by objects in the scene. Detection methods utilizing optical sensors or RGB cameras combine various features related to the physical properties of flame and smoke, including color, motion, spectral, spatial, temporal, and texture characteristics. By integrating color and motion information from images and videos, these multi-feature fire-based detection systems offer results that are more accurate. However, improvements are needed in optical systems to reduce the occurrence of false alarms caused by dynamic phenomena such as wind-tossed trees, cloud shadows, reflections, and human activity (Barmpoutis P, 2020).

However, it should be noted that this technology only offers line-of-sight vision, limiting its effectiveness when high trees, hills, or mountains obstruct the view. Additionally, capturing images of the point of ignition may be challenging, and there may be a significant delay before a detectable smoke cloud forms. Optical systems were initially designed to cover large areas with minimal camera towers, but factors such as weather conditions and night vision can impact their performance.

Camera surveillance systems with short-distance links have been attempted, but they proved ineffective for fire detection due to the need for manual installation of each camera in a suitable position, as well as issues related to line-of-sight images, night vision, bad weather, and a high likelihood of false alarms caused by factors such as daily motion of the sun, moving clouds, variation of atmospheric extinction, and vegetation (Alkhatib, 2013).

To conclude, the cost associated with implementing these systems presents a noteworthy consideration. Individually, the expense of each camera tower can exceed thirty thousand dollars, thereby contributing to the overall high cost. Moreover, the establishment and installation of communication infrastructure in remote forested regions are necessary prerequisites.

### 1.4.5 Wireless Sensor Networks (WSN)

Forest fire detection and prevention pose significant challenges in various countries. Several approaches have been proposed to monitor fire outbreaks. Initially, manned observation towers were utilized; however, this method proved inefficient and lacked effectiveness. Subsequently, attempts were made to employ camera surveillance systems and satellite imaging technologies. Unfortunately, these methods also fell short in efficiently monitoring the initial stages of surface fires (Alkhatib, 2013). For instance, camera networks positioned at different locations within forests only offer line-of-sight images and can be susceptible to weather conditions and physical obstacles, limiting their effectiveness.

Forest fire detection can be effectively achieved through the deployment of Wireless Sensor Networks (WSNs) consisting of wireless sensor nodes strategically placed in forested areas. These sensor nodes are capable of detecting various environmental and physical parameters, including temperature, pressure, humidity, as well as chemical parameters such as carbon monoxide, carbon dioxide, and nitrogen dioxide. Operating in a self-healing and self-organizing wireless networking environment, these sensors derive power from the same source batteries, unlike cell phones which have the advantage of periodic recharging. In Wireless Sensor Networks (WSNs), sensor nodes acquire physical information from the environment, converting it into electrical signals that are wirelessly transmitted to a central node or base station. This central node serves as a focal point for data processing, analysis, and the generation of alerts to notify relevant authorities in the event of a fire. Additionally, the collected data can be forwarded to a remote location for further analysis, leading to the deployment of the obtained insights across diverse applications tailored to the specific needs of the WSN (Alkhatib, 2013).

WSNs offer numerous advantages for forest fire detection. Firstly, they present a cost-effective alternative to traditional fire detection systems, as they require minimal infrastructure and wiring. Secondly, WSNs provide real-time monitoring of environmental conditions, enabling swift detection and response to potential forest fires. Moreover, the scalability of WSNs allows for effective monitoring of both large and small forested regions. Lastly, WSNs are designed with low power consumption in mind, enabling extended operational periods without frequent battery replacements. These advantages collectively position WSNs as a formidable tool in the realm of forest fire detection and prevention. Complementing this, the utilization of information fusion techniques enhances the quality of response to critical events by leveraging the integration of diverse data sources, including sensors and databases.

Notwithstanding their robustness and efficiency, WSN-based systems do have inherent limitations. These primarily include limited energy capacity, relatively low communication speed, and demanding installation and maintenance requirements. Furthermore, forest fire detection using WSNs can be hindered by factors such as the restricted range of wireless signals due to terrain, foliage, and other environmental factors, which limit the coverage of WSN nodes. Additionally, the large volume of data generated by WSNs can pose challenges in terms of real-time processing and analysis. Moreover, false alarms may arise from natural disturbances like wind, further complicating the accuracy of forest fire detection using WSNs (Yi, 2020).

## 1.5 The importance and advantage of using drones and neural network model in the prevention strategy

### 1.5.1 Unmanned aircraft vehicles history

The history of unmanned aerial vehicles (UAVs) dates back to the end of the 19th century, where the first aircraft used as a drone were inflatable balloons. However, it wasn't until 1917 when the British Royal Navy developed the first remote-controlled aircraft for use as a target drone and in the 1950s the first true UAVs were developed for military purposes.

In the 1960s, the US Air Force began using UAVs for reconnaissance missions over Vietnam. These early UAVs were primarily used for surveillance and intelligence gathering, and were operated by remote control from ground-based stations.

During the 1980s and 1990s, the development of more advanced technologies led to the creation of new generations of UAVs with improved capabilities. These included the «Predator», a long-endurance UAV designed for reconnaissance and strike missions, and the «Global Hawk», a high-altitude UAV used for reconnaissance and surveillance.

In the 2000s, the use of UAVs expanded beyond military applications to include civilian and commercial applications. Today, UAVs are used for a wide range of purposes, including aerial photography, surveying and mapping, agriculture, wildlife monitoring, search and rescue, and package delivery (Namoune A., 2021).

As the technology continues to evolve, it is likely that UAVs will become even more advanced and widely used in a variety of fields, from transportation to healthcare to disaster response.

### 1.5.2 Unmanned aircraft vehicles

A UAV is commonly denoted as a pilotless aircraft with the ability to fly and stay airborne without necessitating a human operator on board, offering more cost-effective operations than equivalent manned systems, and accomplishing cost-efficient critical missions without jeopardizing human life. UAVs can be remotely controlled from a ground-based station using a remote control, or autonomously through the utilization of autopilot and different sensors, such as global positioning system (GPS) and inertial measurement units (IMU) which ensures precise positioning.

In recent times, Unmanned Aerial Vehicles (UAVs) have gained significant popularity in various forestry applications such as forest scouting, search and rescue operations, forest resource surveying, and forest fire fighting. UAVs are emerging as one of the most promising and powerful tools to tackle these issues. The decision to use UAVs as platforms for these applications is primarily due to the fact that they are all designed to be maneuverable and flexible to some extent, providing access to difficult areas, but the degree of maneuverability and flexibility can vary depending on several factors, such as its design, construction, and intended use, plus their ability to fly at different altitudes. In addition to these advantages, UAVs are user-friendly and easy to operate compared to other available technologies. Furthermore, recent advances in both hardware and software technologies have made it possible to process heavy and complex visual data on board, significantly enhancing their capabilities for forestry applications. Despite all these advantages, UAVs have limitations in terms of coverage area, and flight time compared to other technologies. They also require more frequent data collection to cover large areas, which can be time-consuming.

The detection and prevention of forest fires is a critical task that traditionally involves human monitoring and visual inspection, which can be a time-consuming and labour-intensive process. In contrast, advancements in UAV technology enable the quick and efficient monitoring of vast forest areas using a fleet of robotic aerial vehicles. This approach enhances the coverage and reliability of the mission compared to the use of a single UAV.

The use of Unmanned Aerial Vehicles (UAVs) equipped with high-resolution cameras for detecting forest fires has become increasingly favored owing to its capability to efficiently cover extensive territories within a short period. Nonetheless, the primary obstacle faced is the need for accurate fire detection while minimizing the occurrence of false alarms. This requires the implementation of advanced image processing and analysis techniques that can differentiate between actual fires and other phenomena that might mimic the appearance of a fire in the captured images.

To address the challenge of detecting forest fires accurately while minimizing false alarms, one effective approach is to develop a neural network model which is inspired by the structure and function of the human brain. Such a model can be trained using a dataset that consists of UAV-based aerial images of forests and learn to identify specific patterns and extract features in the images that are indicative of actual fires, such as smoke, flames, and other patterns associated with forest fires, while also being able to distinguish between these patterns and other features that may appear similar but are not indicative of fires. With training, the model can achieve high accuracy in fire detection and significantly reduce false alarms, making it a valuable tool in the fight against forest fires (True alarm). Additionally, by utilizing this technology, fire emergencies can be identified in their early stages, and the necessary steps can be taken to prevent them from escalating, leads to more rapid response times and decreases the potential risk of damage to property and human lives. Thus, enabling more efficient management of forest fires, which is critical in safeguarding the environment and protecting public safety.

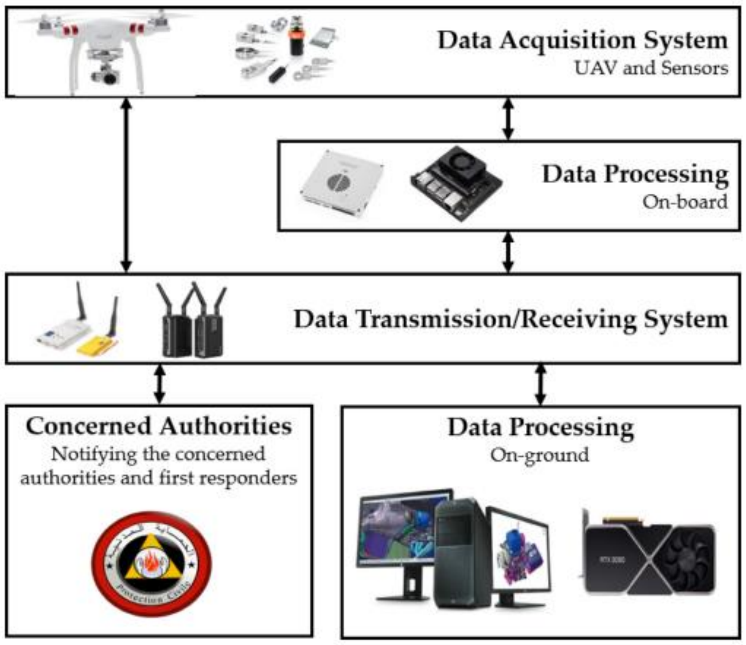


Figure 1.7. Flowchart depicting the process of using UAV-based remote sensing system for detecting forest fires and notifying the relevant authorities (Bouguettaya A., 2022).

UAV-based remote sensing systems have proven to be an effective tool not only for detecting forest fires, but also for automatically alerting the relevant authorities. These systems are equipped with sensors such as thermal cameras and RGB cameras, which capture images and data of the forest and surrounding areas. The captured data can be processed either on board the UAV or on the ground. On-board processing can reduce the amount of data that needs to be transmitted to the ground, but it requires powerful computing resources and may increase the UAV's weight and power requirements. In addition, the UAV must be equipped with a reliable data transmission/transceiver system to transmit the captured data and alerts to the ground. Ground-based data processing involves the transmission of the captured data from the UAV to a ground station, where it is processed using computer vision and machine learning algorithms. The ground station can be a fixed or mobile unit and must have sufficient processing power to handle large amounts of data in real-time. Ground-based processing can utilize powerful computing resources that are not feasible to have on board the UAV, as well as allow for easier maintenance and upgrading of the processing capabilities. Regardless of where the data is processed, a reliable data transmission system is crucial in ensuring that the concerned authorities receive timely and accurate information about the forest fire's location, size, and direction of spread, allowing them to take prompt action to contain and extinguish the fire (Bouguettaya A., 2022).

## 1.6 Conclusion

We have devoted this chapter to exploring the root causes of forest fires, as well as highlighting the techniques employed for preventing and identifying them.

As mentioned earlier, the utilization of a neural network model trained with aerial images captured by UAVs has emerged as an influential solution for forest monitoring and fire detection. This technology has gained substantial attention for its ability to enhance the accuracy of forest fire detection and reduce false alarms. Its potential to provide rapid and precise information makes it a valuable tool in mitigating the damages caused by forest fires.

The following chapter will provide an overview of deep learning techniques used in computer vision, including concepts related to object detection and convolutional neural networks (CNNs).

# CHAPTER 2. GENERAL OVERVIEW OF DEEP LEARNING

## 2.1 Introduction

In recent years, Deep Learning, which is a subfield of AI and Data Science, has made remarkable strides, and its impact can be observed across multiple domains, including computer vision, natural language processing, robotics, self-driving cars, healthcare, and others. By utilizing artificial neural networks modeled on the human brain, Deep Learning has the potential to transform industries and enhance our daily lives.

This chapter aims to provide a general understanding of Artificial Intelligence, Machine Learning, and Deep Learning by presenting an overview of each of these fields. Additionally, we will provide a comprehensive overview of our research by exploring convolutional neural networks in depth. These networks serve as the backbone of our work, and we will elucidate their various architectures, layering concepts, and operational principles. Our aim is to offer a detailed explanation of how these neural networks function and provide a solid foundation for our research.

Finally, we concluded our discussion by covering popular optimization algorithms and the concept of gradient descent.

## 2.2 Definition of Deep Learning

Artificial Intelligence (AI) is a branch of computer science that focuses on creating systems which are capable of performing tasks that normally require human intelligence. In order to achieve this, AI involves programming machines with rules and algorithms that mimic human-like intelligence. AI has two subcategories, namely Machine Learning (ML) and Deep Learning (DL) (Jakhar D., 2020).

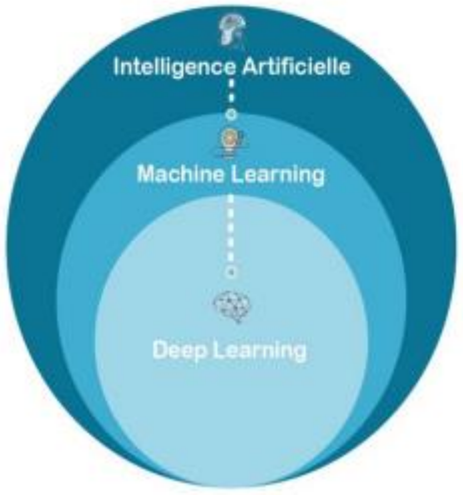


Figure 2.1. Relation between AI, ML and DL.

Machine Learning (ML) is a subset of AI that trains machines to learn from data without being explicitly programmed. This involves processing data and information in a way that enables machines to make decisions and adjust their algorithms based on new data. The main goal of ML algorithms is to minimize errors and increase the accuracy of their predictions.

Deep Learning (DL), also known as artificial neural networks (ANNs), is a subset of Machine Learning (ML) that incorporates computational models and algorithms based on the architecture of biological neural networks in the human brain. Like the brain, DL processes new information by comparing it to known data, categorizing and labelling it to make sense of it. This enables DL to extract relevant information and make accurate predictions (Jakhar D., 2020).

Artificial neural networks (ANNs) consist of interconnected nodes that perform mathematical operations on the input data. The nodes are arranged in three different layers: input, output, and hidden. The input layer receives the raw data, the output layer produces the result of data processing, and the hidden layers extract patterns and relationships from the data.

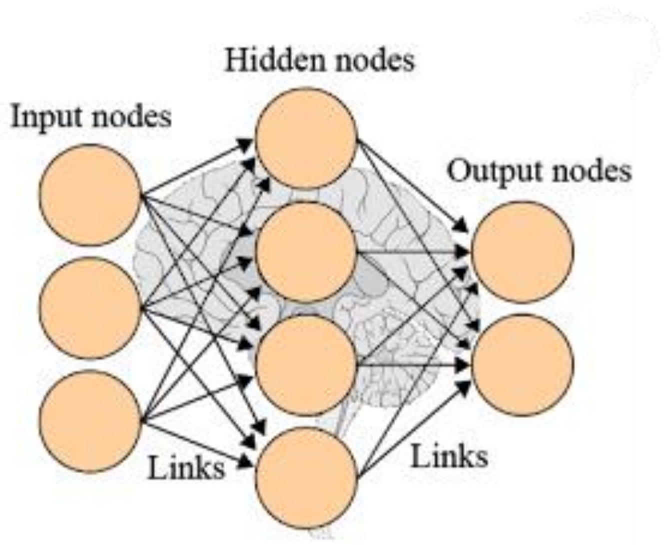


Figure 2.2. The structure of a neural network.

Deep Artificial Neural Networks (ANN) are more complex than superficial ANNs as they contain multiple hidden layers. This increased depth allows them to perform more intricate tasks by recombining and reconfiguring simpler features into more complex ones as the data moves through each layer.

In a typical machine learning workflow, the first step is to manually extract relevant features from images for building a classification model. However, deep learning automates this process by utilizing neural network architectures to automatically extract relevant features from the images. Additionally, deep learning employs "end-to-end learning" where the network learns how to complete a given task, such as classification, when presented with raw data. Deep Learning (DL) is particularly effective for unstructured data and typically provides greater accuracy than Machine Learning (ML). However, to achieve these results, DL requires a substantial amount of training data and significant hardware and software resources (Zhang W. J., 2018).

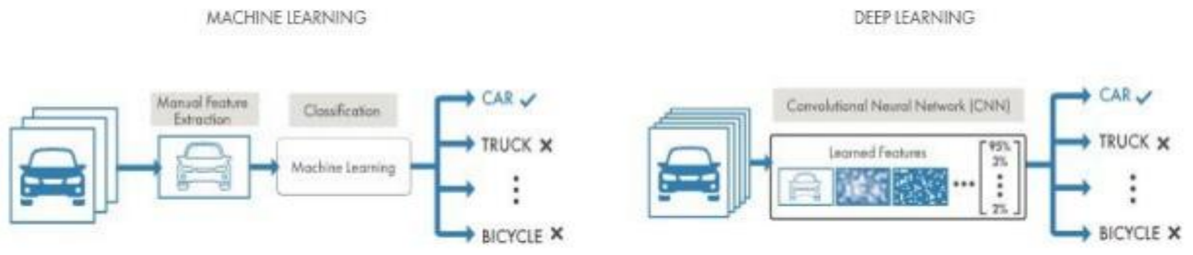


Figure 2.3. Difference between ML and DL.

## 2.3 Application domains of deep learning

Deep learning is a versatile field that finds applications in various domains such as computer vision, natural language processing, speech recognition, and robotics. It has the potential to solve existing problems and explore new domains, including medical imaging, sound creation, and art generation. In practice, deep learning has already proven its worth by making significant contributions in addressing real-world problems like automatic colorization, machine translation, and image recognition. It has also opened up new possibilities with self-driving cars and interactive robots, which can change the way we approach transportation and human-robot interaction. Deep learning is a powerful tool that can analyse vast amounts of data and generate accurate predictions, leading to informed decision-making. Its versatility allows it to streamline complex processes, provide solutions to complicated problems, and unlock new opportunities in various fields, thereby enabling progress and innovation (Shinde P. P., 2018).

Deep learning's potential is broad and varied, with applications across domains including:

### 2.3.1 Computer Vision

Deep learning algorithms have significantly improved object recognition, face recognition, image classification, and segmentation. This has enabled the development of applications such as self-driving cars, augmented reality, and security systems.

### 2.3.2 Natural language processing

Deep learning is used to improve speech recognition, machine translation, sentiment analysis, and language generation. This has enabled the development of applications such as virtual assistants, chatbots, and language translation software.

### 2.3.3 Autonomous vehicles

Deep learning plays a vital role in self-driving cars by enabling them to identify objects, anticipate the movements of vehicles and pedestrians, and determine the optimal route. This utilization holds the promise of substantially decreasing accidents and enhancing traffic efficiency.

### 2.3.4 Medical diagnostics:

Deep learning is used for medical imaging to improve diagnosis accuracy, disease detection, and drug discovery. It can also be used to predict patient outcomes and personalize treatment plans.

### 2.3.5 Finance

Deep learning is used for fraud detection, credit risk assessment, and portfolio optimization. It can analyze large amounts of data quickly and accurately, helping financial institutions make better decisions and reduce risks.

### 2.3.6 Robotics

Deep learning is used for robot vision, motion control, and grasping. This enables robots to interact with their environment and perform complex tasks.

### 2.3.7 Beating humans in games

Deep learning is used for game AI, character animation, and game simulation. It can generate realistic game environments and create more challenging opponents.

For instance, AlphaGo holds the distinction of being the initial computer program to outperform a professional human Go player, the first program to triumph over a Go world champion, and potentially the strongest Go player in recorded history.

### 2.3.8 Agriculture

Deep learning is used for crop and soil analysis, yield prediction, and disease detection. It can help farmers make better decisions and optimize their yield, leading to improved food security and sustainability.

To sum up, the bottom line is that deep learning is well-suited for newer areas of application because it requires large amounts of data, specialized hardware, and automatic feature engineering. The use of high-performance hardware, such as GPUs, has made it possible to train deep learning models much faster and more efficiently. Additionally, deep learning's automatic feature engineering capability allows it to extract high-level features from raw data, making it more effective than traditional machine learning approaches. Due to these strengths, deep learning is expected to see a multitude of new applications in the future and offer promising solutions to complex problems across various domains, as the field continues to evolve and improve.

## 2.4 Convolutional Neural Networks (CNN)

### 2.4.1 Definition of Convolutional Neural Networks (CNN)

Recently, there has been a noticeable spike in the level of interest surrounding deep learning. One particular class of artificial neural networks, known as convolutional neural networks (CNNs), has emerged as the most adopted algorithm among several deep learning models. CNN has become the dominant method used in computer vision tasks as it has achieved expert-level performances in various fields.

A CNN is designed to process and analyze visual data such as images and videos. It can learn to identify and extract important patterns and features directly from the input data, without requiring human intervention. The key feature of a CNN is its convolutional layer, which applies a set of small, optimized filters to the input data to detect features such as edges and shapes at various locations in the image. Following each convolutional layer, the output undergoes a nonlinear activation function and a subsequent pooling layer to decrease data dimensionality and extract more complex features. As the data advances through the layers of the Convolutional Neural Network (CNN), the extracted features become progressively more abstract and advanced. The final layers of the CNN typically include one or more fully connected layers, which use the extracted features to classify the input into one of several predefined categories.

CNNs are trained using a large dataset of labeled images to optimize the parameters of the filters and other layers using backpropagation and gradient descent algorithms. Through this process, the network can accurately recognize and classify images, making it a valuable tool in computer vision, robotics, and autonomous vehicles.

### 2.4.2 CNN layers

The Convolutional Neural Network (CNN) is a mathematical model characterized by three primary layer types: convolution, pooling, and fully connected layers. Generally, the architecture of a CNN entails the repetition of convolution and pooling layers in a stacked manner, concluding with one or more fully connected layers. The first two layers, convolution and pooling, are primarily involved in feature extraction from the input data, whereas the third layer, fully connected, maps these extracted features to a final output, playing a crucial role in producing it, such as classification (Yamashita, 2018).

#### 2.4.2.1 Convolutional layer

A convolution layer is a crucial building block of the CNN architecture that performs feature extraction using a combination of linear and nonlinear operations. The convolution operation is a specialized linear operation used for feature extraction, which combines an input image with a small array of numbers, called a kernel, to produce a convolved feature map. The kernel slides over the input image computing an element-wise product at each position, and the resulting products are summed together to generate a single output value at that position. The process is repeated for every possible position, resulting in an output feature map. This process is repeated using multiple kernels to create an arbitrary number of feature maps, each representing distinct characteristics of the input data.

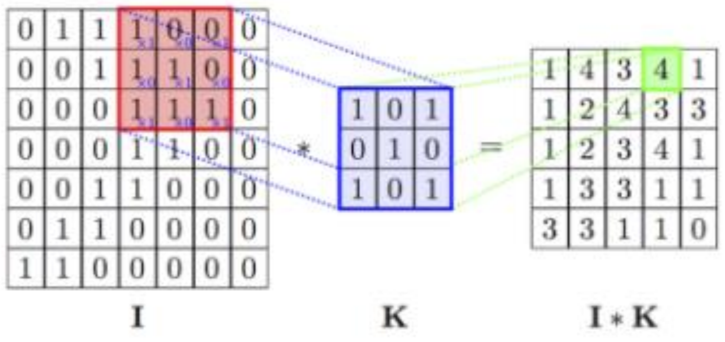


Figure 2.4. An example of a convolution operation using a 3 × 3 kernel size.

Nonlinear operations, such as activation functions, are also applied to the output of the convolution layer to introduce nonlinearity and enhance the model's ability to effectively capture complex patterns and interconnections in the data. Within the dataset.

The behavior of this layer in a NN can be modified by adjusting three critical parameters: filter size, stride, and padding. Ultimately, the size of the output feature map is determined by these parameters (Yamashita, 2018).

- **Kernels**

In CNN, a kernel is a set of automatically learnable filters that are convolved with an input image to produce a feature map. Throughout the training process of a CNN model, the identification of optimal kernels is crucial. These kernels are determined based on the specific task and training dataset. Two critical parameters that characterize the convolution operation and must be predefined prior to initiating the training process are the size and number of kernels. The kernel size determines the size of the window used to scan the input image during the convolution process, which defines the field of view of the convolution. Typically, the kernel size is $3 \times 3$, but it can be $5 \times 5$ or $7 \times 7$. The number of kernels is arbitrary, determining the depth of output feature maps (Prove P.L., 2017).

- **Padding**

A technique used to handle the borders of an input image during the convolution operation. In particular, it refers to adding extra rows and columns of zeros around the image before the convolution is performed.

A padded convolution keeps the spatial output dimensions equal to the input, the extra rows and columns of zeros ensure that the filter can slide over the edges of the input without losing information. Zero padding is a common type of padding used in modern CNN architectures.

Whereas an unpadded convolution, each successive feature map would become smaller after the convolution operation (Prove P.L., 2017).

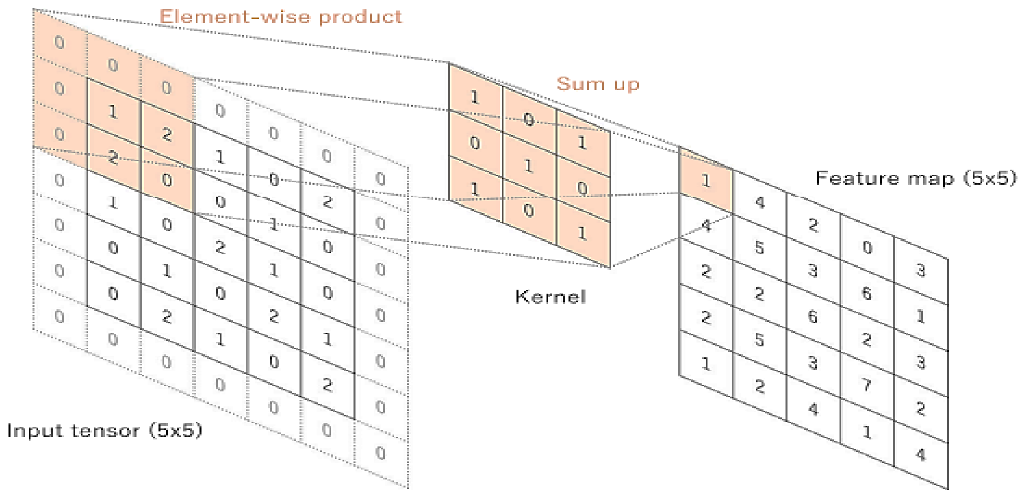


Figure 2.5. An illustration of a convolution operation with zero padding using a 3 × 3 kernel size, and a stride of 1 (Yamashita, 2018).

- **Stride**

The stride parameter determines the step size at which the kernel moves across the input image during the convolution process. When the stride is defined as 1, the filter traverses the input image pixel by pixel. In other words, the kernel moves horizontally and vertically, progressing one pixel at a time, until it covers the entire image. This leads to the generation of a higher resolution output feature map, which captures more detailed information about the input. However, when the stride is set to a value greater than 1, such as 2 or 3 which is less frequently used, the kernel skips over some pixels, resulting in a downsampled output feature map (Prove P.L., 2017).

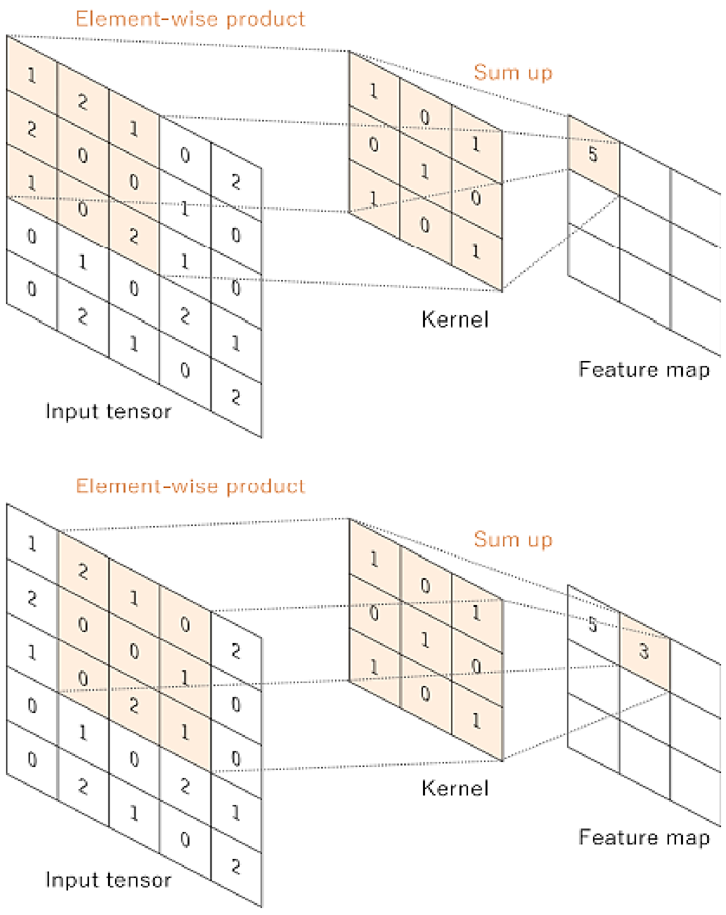


Figure 2.6. An example of a convolution operation using a 3 × 3 kernel size, without padding, and a stride of 1 (Yamashita, 2018).

### 2.4.2.2 Pooling layer

A pooling layer is a typical downsampling technique that can reduce the in-plane dimension of feature maps generated by the prior convolutional layer.

There are several types of pooling layers, including max pooling, average pooling, and sum pooling (Yamashita, 2018).

- **Max pooling:** Max pooling is the most widely used form of pooling operation, which involves extracting patches from input feature maps and outputting only the maximum value from each patch, while discarding all other values. A commonly employed approach for max pooling involves the utilization of a 2 x 2 filter with a stride of 2. This process facilitates downsampling of the in-plane dimension of feature maps by a factor of 2. It is important to note that while the height and width dimensions are reduced, the depth dimension of feature maps remains unaltered.

- **Average pooling:** Global average pooling is a downsampling technique that reduces a feature map's size of height × width to a 1 × 1 array by taking the average of each element in each feature map while keeping the depth of feature maps intact. This technique is typically employed merely once, exclusively prior to the fully connected layers, offering two distinct advantages: a reduction in the total learnable parameters and the CNN's capability to process inputs of varying sizes.
- **Sum pooling:** Sum pooling is a type of pooling operation that involves extracting patches from input feature maps and outputting the sum of all the values within each patch, while discarding all other values.

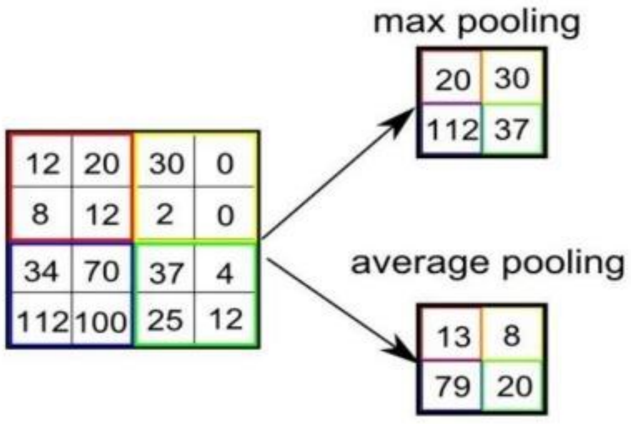


Figure 2.7. Example of Max pooling and Average pooling.

### 2.4.2.3 Fully connected layer

Once the convolution layers have extracted the relevant features and these have been downsampled by the pooling layers, the resulting output feature maps from the final convolution or pooling layer are generally transformed into a one-dimensional (1D) array or vector. This flattening process involves converting the output feature maps into a linear arrangement of numbers. Subsequently, this flattened array is connected to one or more fully connected layers, often referred to as dense layers. In a fully connected layer, each input is linked to every output node via a weight that can be adjusted through learning. The extracted features are then mapped by a subset of fully connected layers to generate the ultimate network outputs, which represent the probabilities associated with each class in classification tasks. Typically, the number of output nodes in the final fully connected layer matches the number of classes in the classification problem. Nonlinear function is applied after each fully connected layer.

During the training phase of a neural network, the model starts with random initial parameters and makes predictions based on them, which may be accurate or not. If a false prediction is made, the backpropagation and gradient descent algorithms are used to adjust the network's parameters. By iteratively adjusting these parameters, the network can learn from its mistakes and make more accurate predictions over time, ultimately improving its performance (Yamashita, 2018).

**Loss function**

A loss function is used to assess how well a neural network's output predictions align with the actual ground truth labels provided during training. It is also frequently referred to by the term “cost function”. The loss function determines the compatibility or mismatch between them.

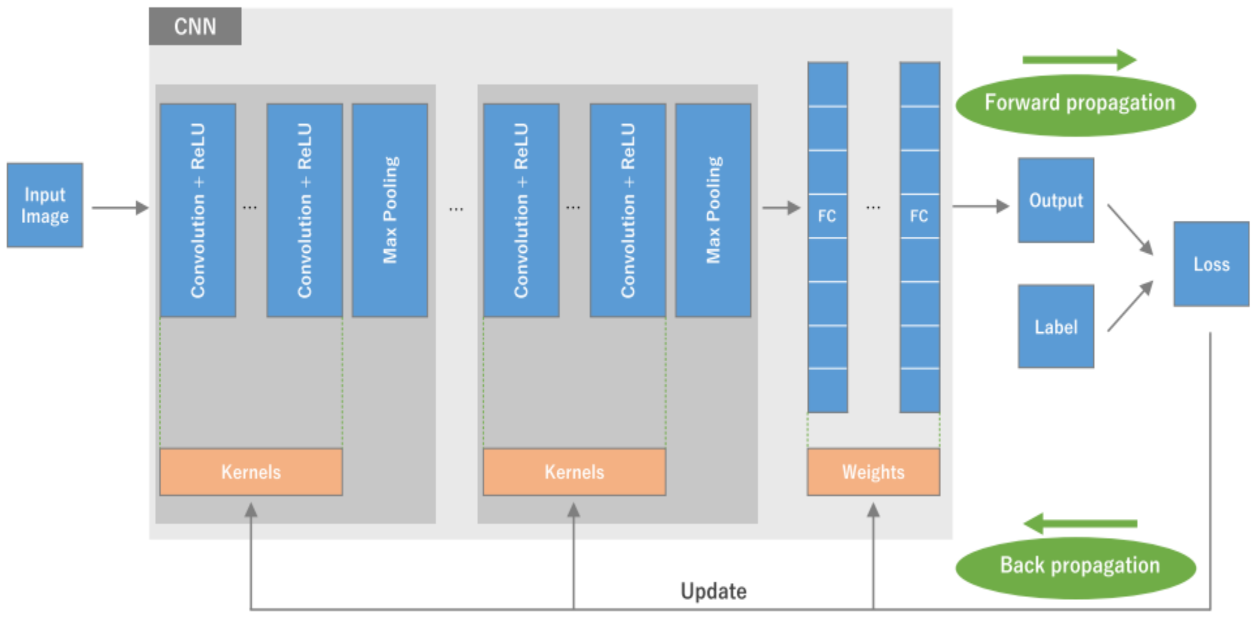


Figure 2.8. Convolutional neural network (CNN) architecture (Yamashita, 2018).

### 2.4.3 Popular convolutional neural networks

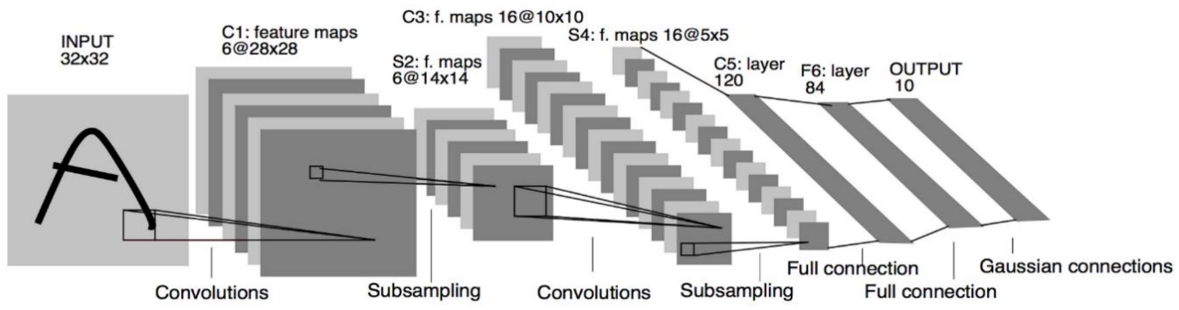
Convolutional Neural Networks (CNNs) find extensive application across diverse domains and encompass a range of distinct variations. Each variation is designed to address specific objectives and tasks. Prominent examples of CNN architectures include LeNet, AlexNet, VGG, GoogLeNet, ResNet, and MobileNet, which have been trained on extensive datasets to effectively identify patterns and extract pertinent features from images.

### 2.4.3.1 LeNet

LeNet is the first CNN architecture that was developed by Yann Le Cun, Corinna Cortes, and Christopher Burges in 1998 for handwritten digit recognition. It consists of multiple convolutional and pooling layers, followed by a fully connected layer.

The LeNet model has five convolutional layers followed by two fully connected layers. However, the LeNet model had difficulty training properly due to the problem of vanishing gradients. To overcome this issue, a shortcut connection layer called max-pooling is used between the convolutional layers to reduce the spatial size of the images.

This particular architecture has demonstrated successful application in various image recognition tasks, notably including digit recognition.



.Figure 2.9. LeNet architecture.

### 2.4.3.2 AlexNet

AlexNet is a convolutional neural network architecture that was introduced in 2012 by Alex Krizhevsky, Ilya Sutskever, and Geoffrey Hinton. It won the ImageNet Large Scale Visual Recognition Challenge in 2012, which demonstrated the effectiveness of deep neural networks for image recognition tasks. It comprises a total of 5 convolutional layers incorporating max-pooling layers, along with 3 fully connected layers and 2 dropout layers. The activation function employed across all layers is Rectified Linear Unit (ReLU). Moreover, the output layer utilizes the Softmax activation function (Alzubaidi, 2021).

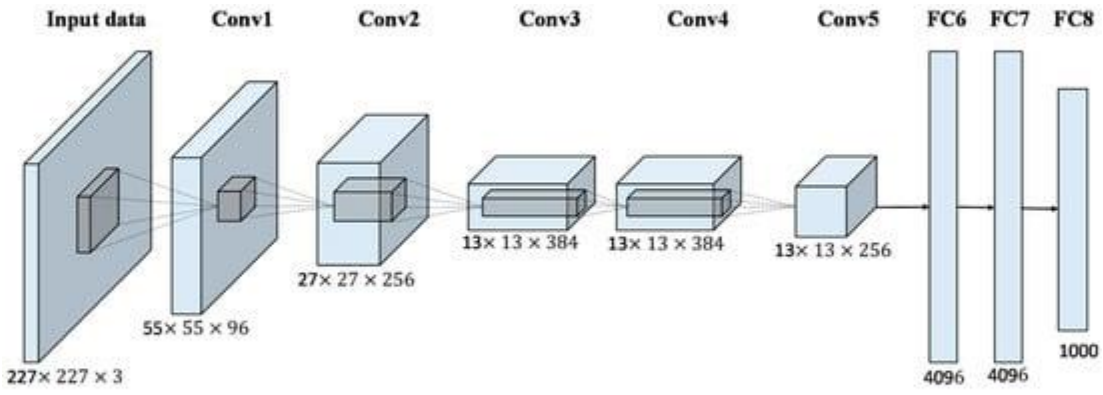


Figure 2.10. AlexNet architecture.

### 2.4.3.3 VGG16 or VGGNet

VGGNet is a 16-layer CNN architecture developed by Karen Simonyan, Andrew Zisserman and colleagues at the University of Oxford. It has up to 95 million parameters and was trained on over a billion images across 1000 classes. However, convolutional Neural Networks (CNNs) incorporating large filters incur significant training costs and necessitate abundant data, thereby leading to the superior performance of architectures like GoogLeNet and AlexNet over VGGNet. Nevertheless, VGGNet exhibits computational efficiency and serves as a robust foundation for numerous computer vision applications, owing to its adaptability across diverse tasks, including object detection. The deep feature representations generated by VGGNet find application in various neural network architectures, such as YOLO, SSD, and other CNN networks (Alzubaidi, 2021).

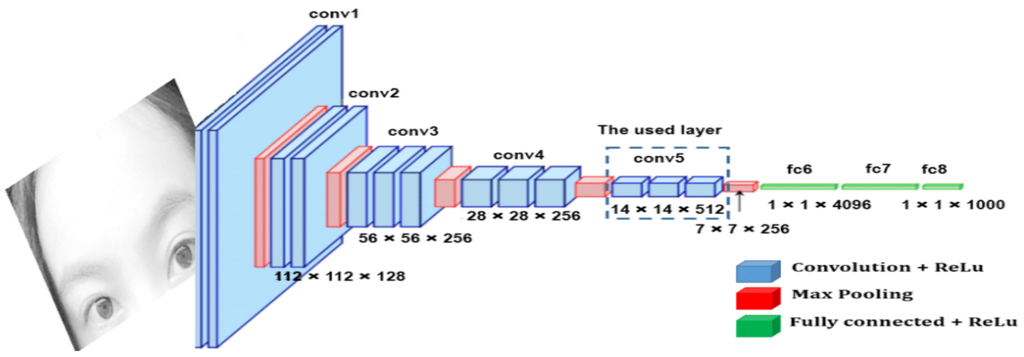


Figure 2.11. VGGNet architecture.

### 2.4.3.4 GoogleNet

GoogleNet, also known as Inception v1, is a deep convolutional neural network architecture developed by Google researchers in 2014. It was the first architecture to use multi-connections between layers in a module called "Inception" and eliminated the fully connected layers by replacing them with a softmax layer. GoogleNet consists of 9 "Inception" modules and provides multiple softmax outputs. It was designed to address the challenge of training very deep neural networks by using an efficient architecture that reduces the number of parameters while maintaining high accuracy. GoogleNet has achieved state-of-the-art performance on several computer vision tasks, including image classification, object detection, and visual question answering (Alzubaidi, 2021).

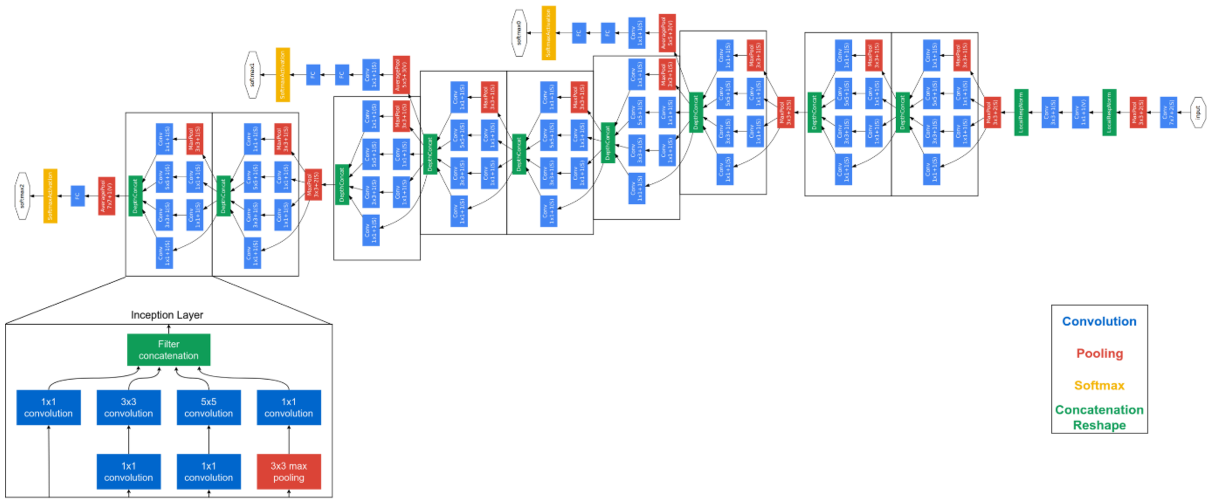


Figure 2.12. GoogleNet architecture.

### 2.4.3.5 ResNet

ResNet, short for Residual Network, represents a specific variant of convolutional neural network (CNN) architecture that was developed by Kaiming He et al. The key innovation of ResNet is the use of residual connections, also known as skip connections, that allow the network to learn identity mappings more easily, making it easier to train very deep networks with many layers. ResNet was first introduced in 2015 and won the first place in the ILSVRC 2015 image classification and detection competition. The original ResNet architecture contains 152 layers and over a million parameters. It has demonstrated superior performance on a wide range of computer vision tasks, including object recognition, segmentation, and detection (Alzubaidi, 2021).

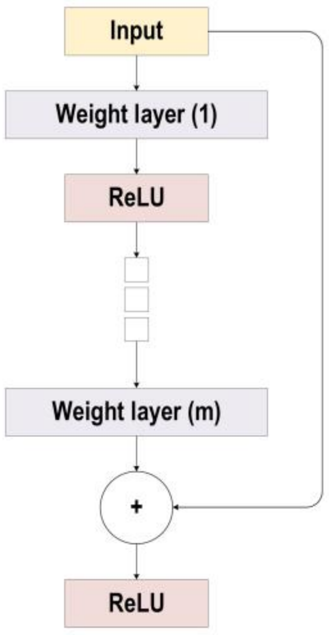


Figure 2.13. ResNet architecture.

## 2.5 Advantages and disadvantages of deep learning compared to traditional learning

### 2.5.1 Advantages of deep learning

- Deep learning has the ability to process large amounts of complex and unstructured data, which can pose challenges for traditional learning methods.
- Deep learning offers improved performance in areas such as speech recognition, image recognition, and automatic translation, by producing more precise results than traditional learning methods.
- Deep learning is able to perform unsupervised learning, enabling it to identify patterns in data without relying on classification labels. This capability makes deep learning useful in situations where large amounts of data need to be explored to uncover underlying structures or relationships.
- Deep learning is a flexible learning method due to the adaptability of deep neural networks. These networks can adjust to different types of data and tasks, making deep learning a versatile approach that can be applied to a wide range of problem domains and data sources.

### 2.5.2 Disadvantages of deep Learning

- Accurate outcomes in deep learning heavily rely on the availability of substantial volumes of data.
- Deep learning requires significant computing resources, including graphics processing units and computing clusters, which can be costly and require high computing times.

## 2.6 Gradient descent variants

In the realm of deep learning, the optimization algorithm of choice is often gradient descent, which serves the purpose of iteratively modifying the parameters of a neural network model in order to minimize the associated error or cost function. The goal is to find the set of parameters that results in the lowest cost, which is often equivalent to achieving the highest accuracy or performance on specific tasks. The fundamental concept underlying gradient descent involves the calculation of the gradient of the cost function concerning the parameters. This gradient information is subsequently utilized to adjust the parameters in a manner that diminishes the overall cost. By repeating this process over multiple iterations, the algorithm gradually converges towards the optimal set of parameters that minimizes the cost function and improves the performance of the neural network model on the given task. The Gradient Descent (GD) algorithm encompasses three primary variants that vary in terms of the data quantity employed for evaluating the objective function's gradient and the approach used to update the model parameters. These variants are Batch Gradient Descent (BGD), Stochastic Gradient Descent (SGD), and Mini-batch Gradient Descent. While BGD updates the parameters using the average of the gradients of the entire training dataset, SGD updates the parameters using the gradient of a single training sample. Mini-batch GD updates the parameters using the average of the gradients of a small subset of the training data at each iteration.

### 2.6.1 Batch gradient descent

BGD, also referred to as vanilla GD, is an iterative algorithm that computes the error for each training example and updates the model parameters after all examples have been processed in an epoch of training. BGD offers the advantage of producing a balanced error gradient and stable convergence, but can lead to suboptimal convergence due to the steady error gradient, and it requires the entire training dataset to be available in memory. While offering precise performance, BGD is computationally expensive, as it involves multiple scans of the real data to calculate the gradient of the cost function with respect to the parameters θ (Ruder, 2016), as seen in Equation where:

$$\boldsymbol{\theta\ new} = \boldsymbol{\theta\ old} - \boldsymbol{\eta}.\nabla\boldsymbol{\theta J}\ (\boldsymbol{\theta}) \qquad (2\text{-}1)$$

- $\theta_{new}$ = Next Position.
- $\theta_{old}$ = Current Position.
- η = Learning Rate (Step Size).
- $\Delta\theta_J$ (θ) = Direction of the fastest increase.

### 2.6.2 Stochastic Gradient Descent

SGD algorithms have proven to be an effective technique in optimizing large-scale deep learning models. The term "stochastic" refers to a mechanism or method that involves random chance, where only a few samples are selected randomly for each iteration, rather than using the entire dataset. The primary goal of SGD is to find the global minimum by modifying the network structure after each training stage. It approximates the gradient for a randomly chosen batch, rather than computing the gradient for the entire dataset. In practical implementation, random sampling is achieved by shuffling the dataset and iteratively moving through batches. Stochastic Gradient Descent (SGD) is recognized for its recurrent high-variance updates, which introduce significant variations to the objective function (Ruder, 2016).

### 2.6.3 Mini-batch Gradient Descent

Mini-batch GD is another variant of SGD and BGD that divides the training dataset into smaller batches and updates the model parameters for each of these batches. This approach effectively achieves a balance by combining the robustness of Stochastic Gradient Descent (SGD) with the efficiency of Batch Gradient Descent (GD) (Ruder, 2016).

While BGD computes the gradient of the objective function over the entire training dataset to update the model parameters, it is computationally expensive. On the other hand, SGD computes the gradient of the objective function using only one training sample at a time, making it computationally efficient but providing a noisy estimate of the gradient. Mini-batch GD provides a balance between the accuracy of BGD and the efficiency of SGD by computing the gradient of the objective function using a small random subset of the training data at each iteration. Therefore, the trade-off between the correctness of the update parameter and the time to execute an update depends not only on the amount of data used but also on the specific variant of GD employed. The choice of GD variant to use depends on the specific requirements of the problem being solved.

## 2.7 Gradient descent optimization algorithms

Optimization algorithms play a vital role in enhancing the performance of machines by computing gradients and minimizing cost functions. There are diverse optimization algorithms available, enabling the implementation of learning in various ways.

### 2.7.1 Adagrad

AdaGrad, which stands for Adaptive Gradient Descent, is a method that adjusts the learning rate based on the situation at hand. The actual learning rate is computed from parameters, allowing it have a tendency to adapt. When the parameter gradient is high, the learning rate decreases, and vice versa. AdaGrad measures different learning rates for various parameter elements, similar to the AdaDelta algorithm. However, AdaGrad differs from AdaDelta in its approach to gradient square aggregation by utilizing the moving average of gradient squares (Saad H.H., 2021).

### 2.7.2 AdaDelta

AdaDelta is an extension of AdaGrad that uses fixed-sized windows to monitor available gradients, rather than accumulating them like AdaGrad. Unlike traditional SGD algorithms that require manual learning rate selection, AdaDelta is an optimization that uses an adaptive learning rate (LR) algorithm. Selecting an inappropriate learning rate can lead to low prediction accuracy, but AdaDelta's automatic learning rate adjustment can prevent this issue (Saad H.H., 2021).

### 2.7.3 RMSProp

RMSProp is a popular adaptive stochastic algorithm used for Deep Neural Network (DNN) training. It is a modification of Adagrad, and works by accumulating the gradient and taking an exponentially weighted average of the accumulated gradients. Unlike Adagrad, RMSProp discards older gradients and only preserves current ones. This approach helps to optimize the algorithm's efficiency and adaptability to different types of data. RMSprop is a gradient-based optimizer that modifies the LR adaptively rather than considering it as a hyperparameter as is traditionally done (Saad H.H., 2021).

### 2.7.4 Adam optimizer

Adaptive Moment Estimation (Adam) is a common optimization method for stochastic gradient descent (SGD) that calculates adjustable learning rates for each parameter. In the realm of neural networks, Adam is one of the most popular step-size strategies. The upgrading operation takes into account the smooth gradient variant and includes a mechanism for bias correction. One of the advantages of Adam is that it requires less computational costs, has a lower execution memory, and is not affected by gradient diagonal rescaling (Saad H.H., 2021).

## 2.8 Activation function

### 2.8.1 Activation function definition

Activation functions play a crucial role in artificial neural networks as they facilitate the transformation of input signals into output signals, which are subsequently passed on to the subsequent layers in the network. The primary purpose of an activation function is to determine the activation state of a neuron, indicating whether it should be activated or not. Which makes it a critical component in the neuron's decision-making process. This decision is based on the inputs the neuron receives, and the activation function determines it by applying a non-linear function to the weighted sum of these inputs plus the neuron's bias. In the absence of an activation function, the output signal would be a simple linear function with limited performance and power, akin to a linear regression model. Prominent examples of activation functions encompass the sigmoid function and the ReLU function, both of which enable neural networks to acquire the capacity to learn and discern intricate mappings from diverse forms of data, including images, videos, audio, speech, and text (Sharma S., 2017).

### 2.8.2 Popular activation function

#### 2.8.2.1 The sigmoid function

The sigmoid activation function is commonly used due to its non-linear nature. By transforming input values into a range of 0 to 1. The function is defined as f(x) = 1/e-x.
The sigmoid function is a smooth and continuously differentiable S-shaped function, with a derivative of f'(x) = 1-sigmoid(x). However, since the function is not symmetrical around zero, all output values of neurons will have the same sign. To mitigate this issue, the sigmoid function can be scaled (Sharma S., 2017).

#### 2.8.2.2 The Tanh function

The Tanh function, also known as the Hyperbolic Tangent function, shares similarities with the sigmoid function but has a crucial difference, it is symmetric around the origin. This symmetrical behavior leads to varying signs of outputs from preceding layers, subsequently serving as input to the subsequent layer. The Tanh function can be expressed as f(x) = 2sigmoid (2x)-1, and it produces continuous and differentiable values between -1 and 1. Compared to the sigmoid function, the Tanh function has a steeper gradient and is preferred due to its non-directional gradients and zero-centered nature (Sharma S., 2017).

#### 2.8.2.3 The ReLu function

The ReLU function, also known as the Rectified Linear Unit, is a popular non-linear activation function used in neural networks. The primary benefit of utilizing the ReLU function is capability to selectively activate neurons, rather than activating all neurons simultaneously. Instead, the function sets the output of a neuron to zero if it is deactivated. This means that the neuron has no impact on the network's computation or decision-making process, as its output is effectively ignored. Mathematically, this is represented by the ReLU function's formula, f(x) = max (0, x). The efficiency of ReLU compared to other activation functions lies in its ability to activate only a limited number of neurons at any given time. This targeted activation is more computationally efficient and can improve the overall speed of the neural network. However, during the backpropagation step in the training process, a potential problem can arise if the gradient is zero.

In such cases, the weights and biases remain unchanged, which can negatively impact the network's performance (Sharma S., 2017).

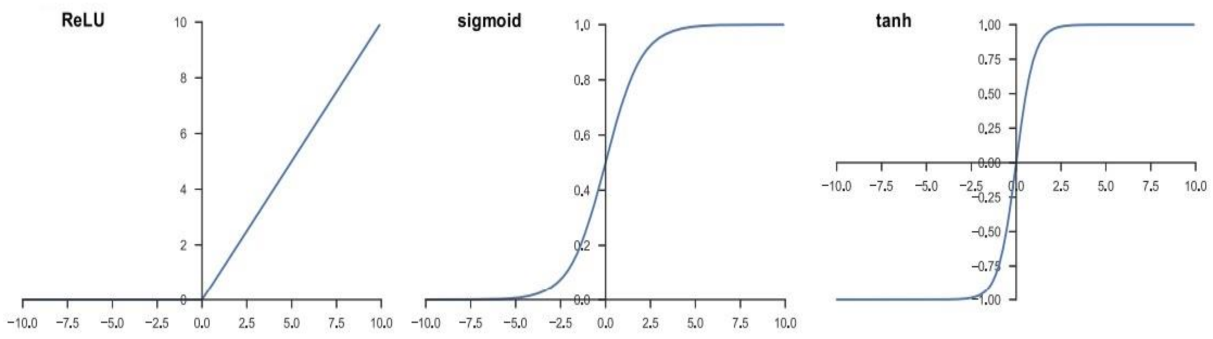


Figure 2.14. Activation functions frequently utilized in neural networks (Yamashita, 2018).

The sigmoid activation function is rarely utilized, except for the output layer in binary classification problems. Tanh, on the other hand, is considered superior because it centers the data, making learning smoother. However, both sigmoid and Tanh suffer from the drawback of nearly zero derivatives when x is extremely large or small, which can hamper gradient descent. Consequently, the most commonly employed activation functions are ReLU, as they do not face this issue.

Finally, artificial neural network techniques namely Deep Learning utilize activation functions to extract knowledge from complex, high-dimensional, and nonlinear datasets, ultimately improving accuracy and achieving better results.

## 2.9 Conclusion

This chapter was dedicated to presenting general information on Deep Learning, specifically focusing on the Convolutional Neural Network and its widely used variants. Moreover, we explored the fundamental concept of gradient descent and some famous notable algorithms associated with this domain. In the next chapter, we aim to provide an overview of the YOLO model, specifically YOLOv3, and its potential advantages in terms of performance in detecting objects, with a particular emphasis on its effectiveness in the early detection of forest fire outbreaks

# CHAPTER 3. OBJECT DETECTION USING YOLOV3 MODEL

## 3.1 Introduction

The recent progress in artificial intelligence (AI) and deep learning has brought significant advancements to image-based modeling and analysis tasks, including classification, real-time prediction, and image segmentation, across a wide range of applications. Furthermore, the introduction of nanotechnology semiconductors has given rise to a new generation of powerful computational units, such as Tensor Processing Units (TPUs) and Graphical Processing Units (GPUs), which possess exceptional processing capabilities for data-driven methods. Moreover, modern drones and unmanned aerial vehicles (UAVs) can now be equipped with small-sized edge TPU/GPU platforms, enabling real-time processing on board. This capability is particularly beneficial for early fire detection, allowing for proactive measures to be taken before a catastrophic event occurs.

Deep learning has seen extensive application and has significantly contributed to the advancement of image recognition and object detection technologies. These technologies rely on convolutional neural networks (CNNs) as a fundamental framework for extracting image features. While the history of object detection traces back two decades, substantial breakthroughs occurred in 2014. This pivotal year witnessed notable improvements in GPU computational capabilities and deep learning-based image recognition techniques. Consequently, 2014 stands as a significant milestone in the field of object detection, signifying a transformative shift from conventional approaches to deep learning-based methodologies (Zhao L., 2020).

Figure 3.1 illustrates the evolutionary progression of object detection algorithms.

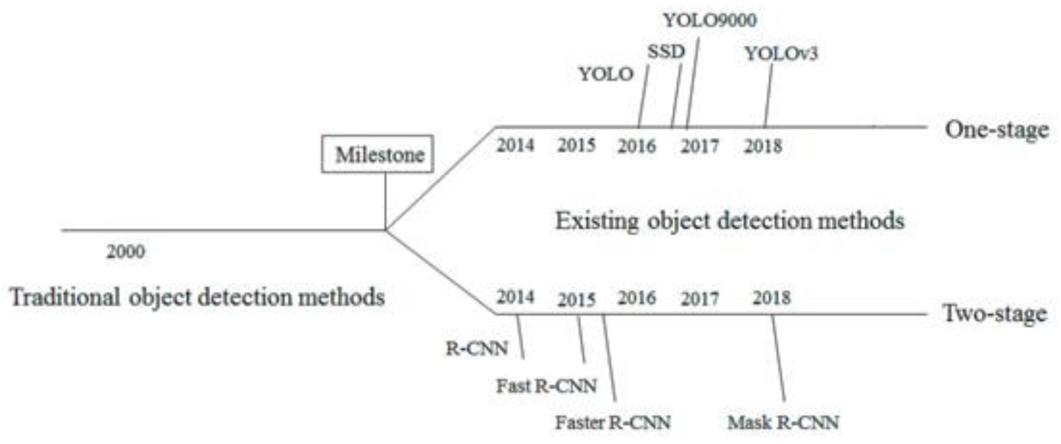


Figure 3.1. Timeline of object detection algorithms. (Chung Y.-L., 2020)

Advancements in object detection techniques have been noteworthy. The introduction of a two-stage approach-based Region-CNN (R-CNN) in 2014 brought about significant progress. This approach involves generating region proposals from the input image to identify potential objects of interest. Subsequently, a convolutional neural network is employed to extract relevant features from these proposed regions, enabling accurate object recognition and classification.

Building upon R-CNN, further improvements led to the development of Fast R-CNN and Faster R-CNN. These variations incorporated optimizations that enhanced detection accuracy and speed. More recently, the revolutionary Mask R-CNN has emerged, which not only detects the position and type of target objects but also extends into semantic segmentation, enabling the illustration of target contours (Chung Y.-L., 2020).

Despite the higher detection accuracy achieved by these algorithm models, their complex methodology and time-consuming nature pose challenges for real-time applications. To address this, one-stage approach models based on regression algorithms have been proposed. These models directly predict target locations by performing regression analyses, allowing for faster computation. Although this method exhibits lower accuracy compared to the aforementioned models due to its utilization of single-shot detection, it maintains an acceptable range of precision. Noteworthy one-stage object detection models include Single-Shot Detectors (SSD) and the You Only Look Once (YOLO) series, encompassing YOLO, YOLO9000, and YOLOv3 (Chung Y.-L., 2020).

## 3.2 YOLO Description

In 2015, Joseph Redmon et al. introduced a paper titled "You Only Look Once: Unified, Real-Time Object Detection," which drew significant attention from computer vision researchers upon its release. This marked the emergence of YOLO on the computer vision landscape.

By utilizing a fundamentally different approach to object detection, YOLO outperformed other real-time object detection algorithms by a significant margin, achieving state-of-the-art results.

The author's objective is to construct a comprehensive neural network model that integrates all stages into a unified framework. By utilizing a single neural network composed of multiple convolutional networks, the system generates predictive vectors for each object detected in the input image. The YOLO system computes the complete set of image features and performs predictions for all objects simultaneously. This concept is encapsulated by the principle of "You Only Look Once." (Redmon J. D., 2016).

### 3.2.1 YOLOv1

The primary concept behind YOLOv1 involves overlaying a grid cell of size S x S (default of 7 x 7) onto an image. If the center of an object is located within a particular grid cell, then that grid cell is solely responsible for detecting the object, as illustrated in Figure 3.2. This approach ensures that multiple detections of the same object do not occur in other cells, and thus, those cells are ignored.

To enable object detection, every grid cell within the YOLO framework predicts B bounding boxes along with their respective parameters and confidence scores, as shown in Figure 3.2. These confidence scores indicate whether an object is present or absent within a given bounding box. The definition of the confidence score is as follows:

$$confidence\ score = p(Object) * IOU_{pred}^{truth}$$

The confidence score in the YOLO framework is determined by two factors. Firstly, $(Object)$ represents the probability of an object being present within a given cell. Secondly, $IOUpred\ truth$ represents the intersection over union of the prediction box and the ground truth box. The confidence score ranges from 0 to 1, indicating the likelihood of object presence. If no object exists in the cell, the confidence score approaches 0. Conversely, if an object is present, the confidence score matches $IOUpred\ truth$.

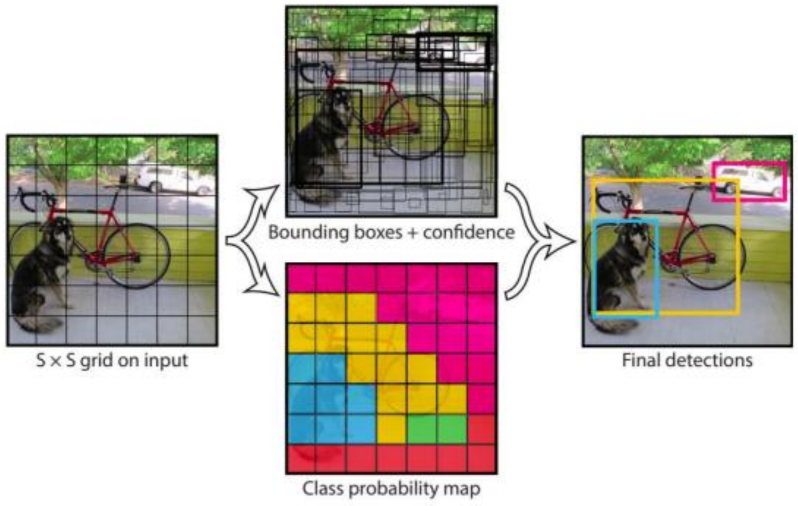


Figure 3.2. YOLO model was applied on the input image (Redmon J. D., 2016).

The YOLO model employs a grid of $S \times S$ cells for image analysis. Each cell predicts $B$ bounding boxes with 5 parameters, and these boxes share prediction probabilities for $C$ classes. The YOLO model output's total number of parameters can be computed using the formula $S \times S \times (5 * B + C)$. For instance, when evaluating the YOLO model on the COCO dataset, which includes 80 classes, and assigning 2 bounding boxes per cell, the total output parameters are $7 \times 7 \times (5 * 2 + 80)$.

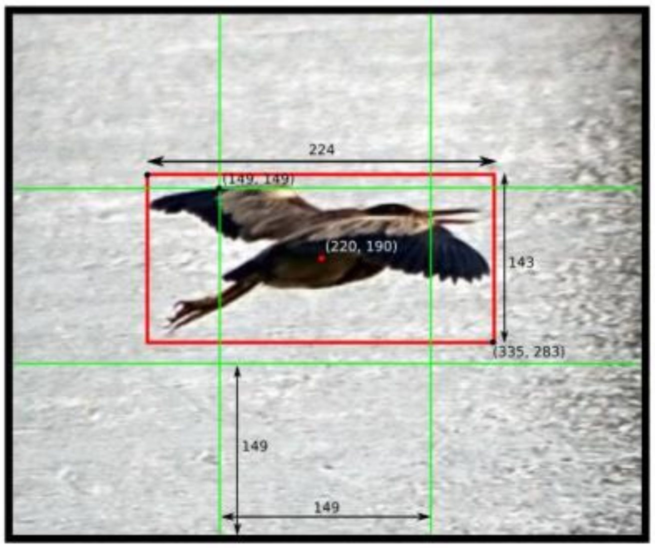


Figure 3.3. Illustration of a bounding box (Menegaz, 2018).

The purpose of the YOLO algorithm is to detect an object by precisely predicting the bounding box containing that object and localize the object based on the bounding box coordinates.

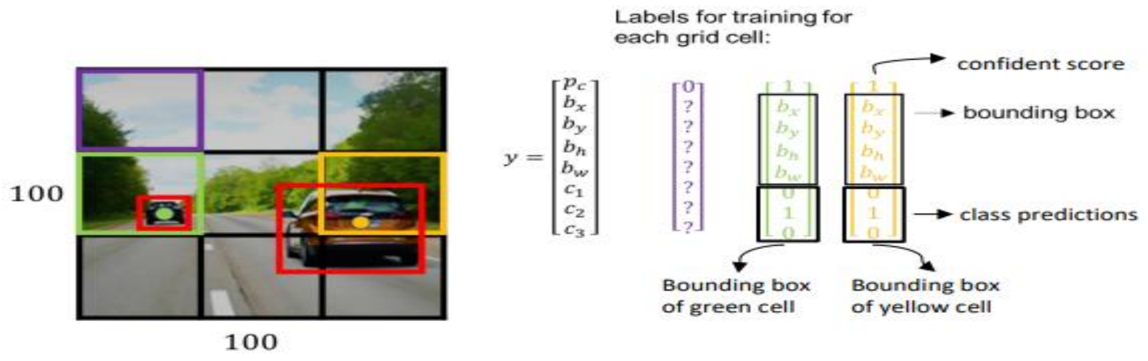

Labels for training for each grid cell:
confident score
bounding box
class predictions
Bounding box of green cell
Bounding box of yellow cell
100
100
$y = [p_c, b_x, b_y, b_h, b_w, c_1, c_2, c_3]$

Figure 3.4. Specifying label vector $y$ in a YOLO model to predict object for 3 classes (Datahacker.rs, 2018).

In this figure, the purple cell does not contain any object, the confidence score for the bounding boxes within that cell is set to 0. As a result, all other parameters associated with those bounding boxes are disregarded.

As a post-processing step, YOLO employs Non-Maximum Suppression (NMS) to eliminate all bounding boxes that either do not contain any object or overlap the same object as other bounding boxes (Science, 2018).

### 3.2.1.1 YOLOv1 architecture

The YOLO model utilizes the Darknet architecture, which includes 24 Convolutional layers, enabling it to handle complex datasets and achieve higher accuracy in processing image features. Followed by two fully connected layers that are responsible for predicting the bounding boxes of objects as shown in Figure 3.4. The Darknet framework provides the necessary structure and functionality to implement this architecture effectively.

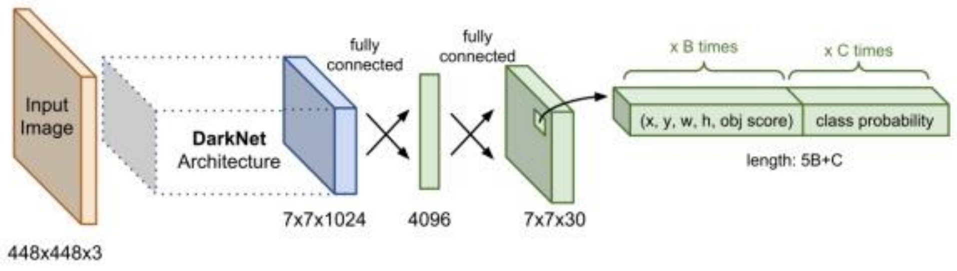


Figure 3.5. Preliminary YOLOv1 architecture.

Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating 1 × 1 convolutional layers reduces the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution (224 × 224 input image) and then double the resolution for detection.

The final layer uses a linear activation function instead of Leaky Rectified Linear Unit (leaky ReLU) activation as all other layers.

### 3.2.2 YOLOv2

YOLOv2 incorporates the Darknet-19 architecture, which consists of 19 convolutional layers, as a fundamental component for object detection and recognition tasks.

#### 3.2.2.1 The impact of batch normalization

Batch normalization, widely utilized in deep learning models, effectively stabilizes the input layer distribution during training, resulting in faster and more stable training of deep neural networks. By normalizing the features, which represent the post-activation outputs of each layer, to a zero-mean state with a standard deviation of 1, batch normalization aids in achieving convergence and acts as a regularization technique to combat overfitting. While the computational demands may lead to longer epochs, the ultimate outcome is accelerated convergence (Ag, 2018) (Ioffe, 2015).

#### 3.2.2.2 The use of anchor boxes for bounding box prediction

YOLOv1 detects only one object per grid cell so if two or more objects have the center inside the same grid cell, the prediction may be flawed. To solve this problem, the author tried to allow a grid cell to predict more than one object at the same time. In YOLOv2, a new approach was introduced by incorporating an anchor box architecture for predicting bounding boxes, in contrast to the utilization of fully connected layers in YOLOv1. The anchor box system consists of a predefined set of boxes that closely match the desired objects. This system not only relies on ground truth boxes but also incorporates a predefined set of anchor boxes. These anchor boxes serve as reference templates with specific shapes, designed to align with prototypical object configurations. Multiple anchor boxes are assigned to each grid cell, and the model predicts the coordinates and corresponding class probabilities for each anchor box, enabling more accurate object localization and classification (Redmon J. &., 2016).

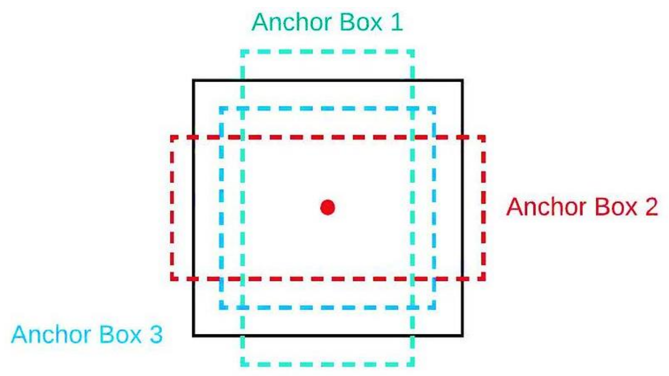


Figure 3.6. For each grid cell (black), 3 anchor boxes with different shapes are predefined.

Rather than manually selecting the best-fit anchor boxes, the k-means clustering algorithm was utilized.

### 3.2.3 YOLOv3

#### 3.2.3.1 Bigger network with ResNet

In deep neural networks, an increased number of layers usually corresponds to a higher level of accuracy. However, as the input image is downsampled in deeper layers, the network may lose crucial fine-grained features, making it difficult to detect small objects. This was a limitation of YOLOv2. To address this issue, the ResNet architecture introduced the concept of skip connections, which enabled the activations to propagate through deeper layers without experiencing gradient vanishing. ResNet uses skip connections to create shortcut paths that directly connect the input to the output within each block. This allows gradients to flow easily through the network, enabling the training of deeper architectures and improving performance. By effectively capturing both low-level and high-level features, ResNet improves the accuracy of object detection tasks, particularly in detecting small objects (He, 2015).

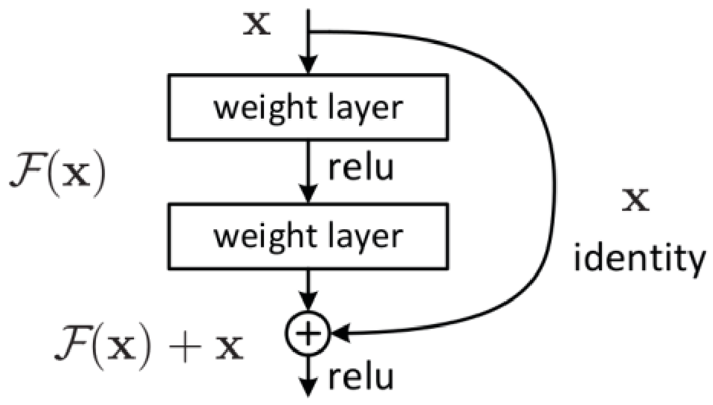


Figure 3.7.ResNet skip connection architecture.

YOLOv3 revolutionized the field by presenting an advanced architecture that employed a novel hybrid feature extractor, Darknet-53 consisting of 53 convolutional layers, and Residual networks (Redmon J., 2018). This fusion of architectures significantly enhanced the model's capabilities and performance in the domain of object detection.

The YOLOv3 model adopts the Darknet-53 architecture, which consists of a 53-layer network used as a feature extractor during training. Additionally, 53 additional layers are stacked for the detection head to train the object detector, resulting in a fully convolutional underlying architecture with a total of 106 layers.

#### 3.2.3.2 Multi-scale detector

In the two preceding versions of YOLO, the predictions were made in the last layers of the object detector after training the feature extractor with the Darknet architecture. However, YOLOv3 introduced a different approach by incorporating the prediction layers alongside the network instead of stacking them at the end layers as previously done. A notable feature of YOLOv3 is its ability to perform detections at three different scales (Redmon J., 2018).

Specifically, predictions at three scales are made in YOLOv3, corresponding to specific layers in the network. The scales are determined by the stride of each layer, which indicates the downsampling ratio applied to the input. These scales are located at the 82nd, 94th, and 106th layers of the network, corresponding to strides 32, 16, and 8, respectively.

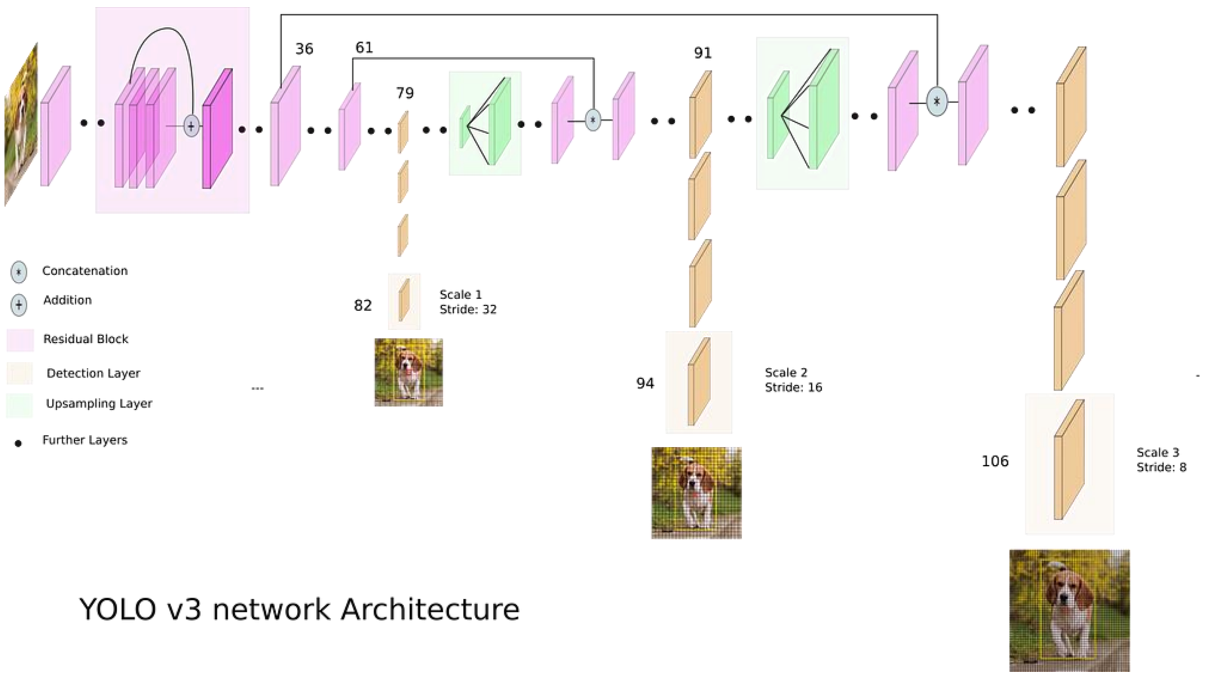


Figure 3.8.YOLOv3 network architecture combining both feature extractor and object detector (Kathuria, 2018).

The initial detection in YOLOv3 occurs within the 82nd layer, using a default input image size of 416 × 416 for better interpretability. After passing through the preceding 81 layers, the input image undergoes downsampling with a stride of 32, resulting in a 13 × 13 feature map comprising a grid of 13 × 13 cells. Within each detection layer, 1 × 1 detection kernels are applied to the feature maps, enabling the prediction of $B$ bounding boxes for each grid cell. In YOLOv3, the model is trained on the COCO dataset with $B$ = 3 (three bounding boxes per cell) and $C$ = 80 (80 classes). Consequently, the 1 × 1 kernel size is determined as 1 × 1 × (3 × (5 + 80)) = 1 × 1 × 255. At the first detection layer, the resulting feature map dimensions become 13 × 13 × 255 (Kathuria, 2018).

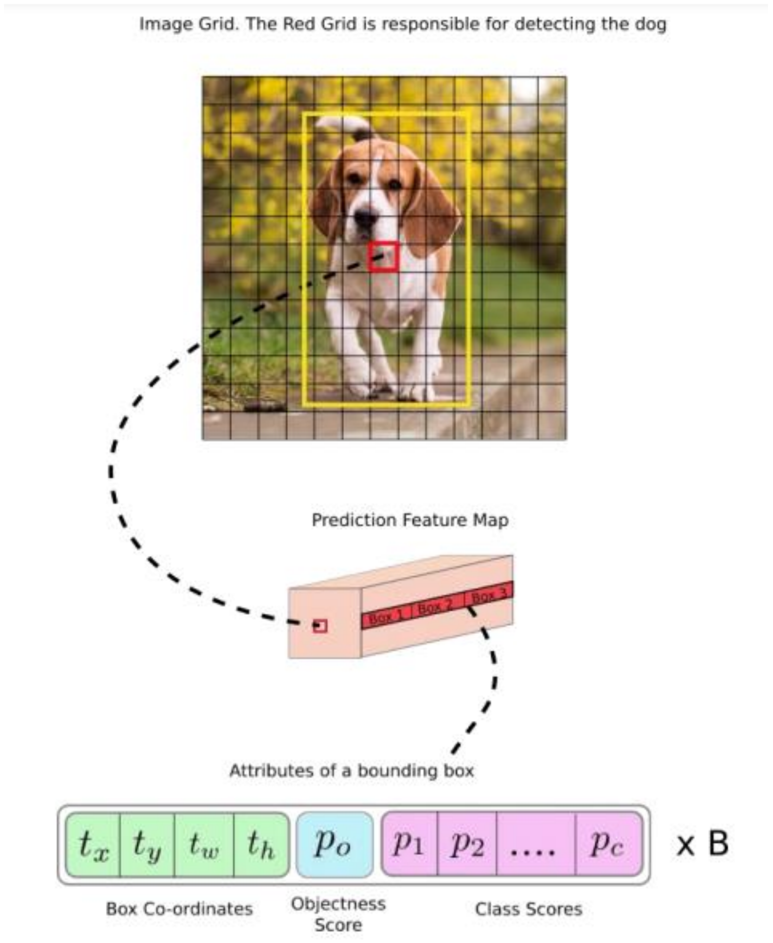


Figure 3.9. YOLOv3 detect an object by applying a 1 × 1 kernel (Kathuria, 2018).

Following the initial detection at the 82nd layer, the procedure is repeated in YOLOv3. However, before proceeding to the next two detection layers for making predictions, the feature maps at the 79th and 91st layers undergo upsampling. Subsequently, through downsampling with strides of 16 and 8 respectively, the feature maps acquire dimensions of 26 × 26 and 52 × 52, corresponding to the 94th and 106th detection layers.

Furthermore, the inclusion of detection layers at different scales in YOLOv3 serves to mitigate the challenge of detecting small objects, which was a notable limitation in YOLOv2. The feature map with a larger spatial dimension provides more intricate details, making the detection layer with a size of 52 × 52 ideal for detecting smaller objects. Conversely, the detection layer with a size of 13 × 13 is dedicated to detecting larger objects (Kathuria, 2018).

By employing concatenation with the shallower layers following upsampling in the deeper layer (specifically, concatenating with layer 61st before reaching layer 91st and with layer 36 before reaching layer 103rd, as depicted in Figure 3.9), the YOLOv3 model effectively preserves fine-grained features from earlier layers. This preservation of features greatly aids the large-scale detection layer in accurately detecting small objects (Kathuria, 2018).

In our research study, we employed this specific version of YOLO to conduct our analysis.

## 3.3 Evaluation metrics of object detection performance

The evaluation of deep learning models involves determining their effectiveness in identifying wildfires. Depending on the study's objective, different evaluation metrics can be measured to assess the deep learning models, including accuracy, precision, recall, and F1-score. Therefore, this section presents the most important evaluation metrics for assessing deep learning models.

### 3.3.1 Confusion matrix

The confusion matrix is an important measurement for classification that is used to summarize the model performance. To calculate these evaluation metrics, it is essential to first understand some crucial metrics, such as True Positive (TP), True Negative (TN), False Negative (FN), and False Positive (FP), which are obtained from the confusion matrix (Lokanath M., 2017).

- TP represents cases where the model correctly predicts the presence of a fire in the input image or video.
- TN refers to instances where the model correctly identifies the absence of a fire.
- FN occurs when the model incorrectly predicts the absence of a fire.
- FP, on the other hand, represents cases where the model incorrectly identifies the presence of a fire.

| | | **Actual Value** | |
|---|---|---|---|
| | | **Positive** | **Negative** |
| **Predicted value** | **Positive** | TP | FP |
| | **Negative** | FN | TN |

Table 3.1.The Confusion Matrix of classification model.

### 3.3.2 Intersection over Union (IoU)

The IoU metric measures the degree of overlap between the predicted boundary and the ground truth annotation's boundaries, yielding a score ranging from 0 to 1. Its purpose is to assess the quality of a prediction, with higher scores indicating better accuracy. Figure 3.10 provides a visual representation of the mathematical formula used to calculate the IoU score (Abdulkader A., 2020).

$$IoU = \frac{Area\ of\ Intersection/Overlap}{Area\ of\ Union}$$

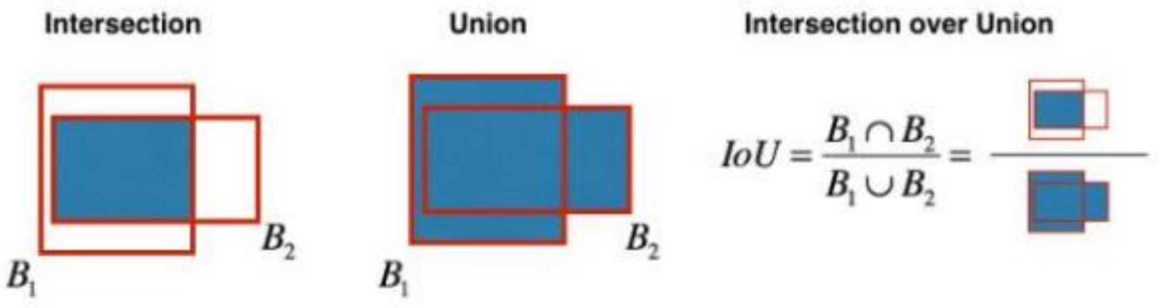


Figure 3.10.Graphical view of the IoU equation (Manishgupta , s.d.).

In order to evaluate the effectiveness of the detection, it is necessary to establish a threshold, usually set at 0.5 (ranging from 0 to 1), in such a way that:

- If the Intersection over Union (IoU) is greater than or equal to 0.5, the object detection is classified as a true positive.
- If IoU is less than 0.5, it is considered a false detection and classified as a false positive.

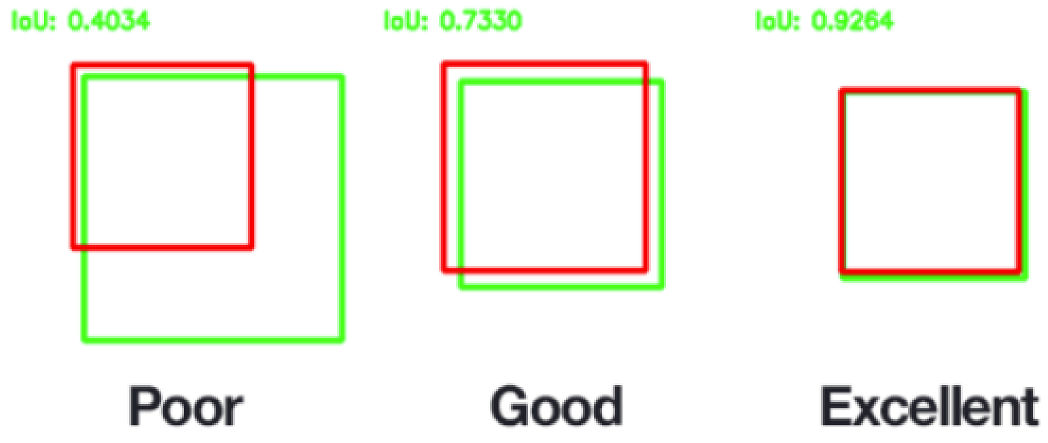


Figure 3.11. Higher IoU, better performance (StackOverflow, s.d.).

### 3.3.3 Non-maximum suppression

Non-Maximum Suppression (NMS) is a crucial technique employed in YOLO models to enhance the accuracy and efficiency of object detection. When detecting objects in an image, it is common for multiple bounding boxes to be generated for a single object. These bounding boxes might overlap or be positioned differently, yet they essentially represent the same object. NMS serves the purpose of identifying and eliminating redundant or erroneous bounding boxes, resulting in the selection of a single bounding box for each object in the image.

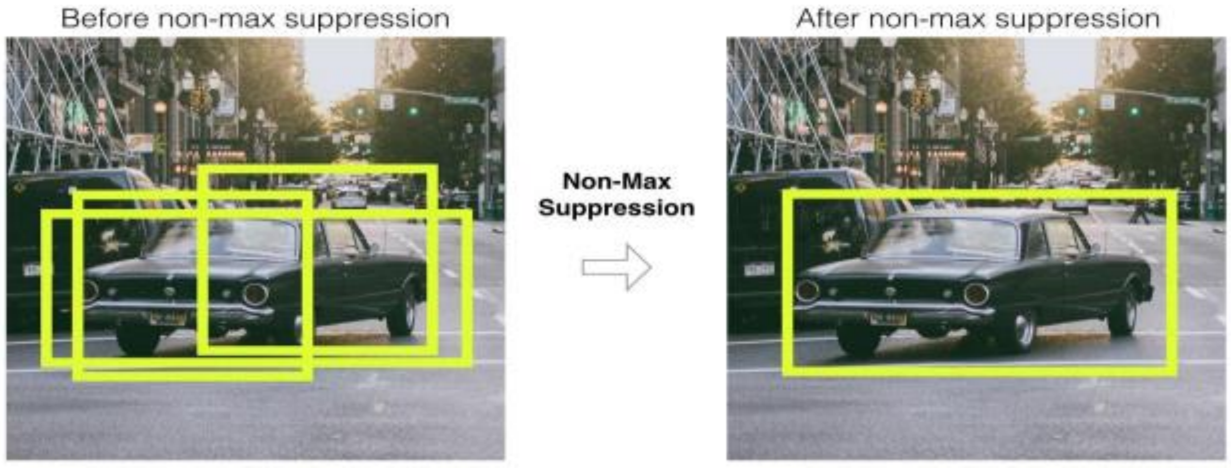


Figure 3.12.Visualization of non-maximum suppression (Li G., 2018).

### 3.3.4 Accuracy

The accuracy metric is the most widely used and straightforward way to assess the performance of a trained deep learning model. It measures the number of correct predictions made by the model out of the total number of predictions (Bouguettaya A., 2022), as specified by Eq1:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (3\text{-}1)$$

### 3.3.5 Precision

Precision is a critical evaluation metric that represents the proportion of true positives (correct predictions) to the total number of predicted positives (total predictions). This metric indicates the accuracy of predicted bounding boxes relative to the ground truth boxes (Bazaga A., 2019). The formula for calculating precision is as follows:

$$Precision = \frac{TP}{TP + FP} \qquad (3\text{-}2)$$

### 3.3.6 Specificity

Specificity is an additional evaluation metric that is helpful for evaluating model performance. It indicates how many non-fire instances were incorrectly predicted as fire by the model (Khan S., 2018). The following equation is used to determine the specificity rate:

$$S = \frac{TN}{TN + FP} \qquad (3\text{-}3)$$

### 3.3.7 Recall

Recall is another essential evaluation metric used for object detection, also known as sensitivity. It measures the proportion of true positives to the total number of positives in the ground truth (Mohana M., 2019), it can be calculated by:

$$Recall = \frac{TP}{TP + FN} \qquad (3\text{-}4)$$

### 3.3.8 F1-score

The F1-score is an evaluation metric that takes into account both Precision and Recall rates. This metric represents a weighted harmonic mean of the Precision and Recall rates (Bouguettaya A., 2022) and is calculated using the following equation:

$$F1_{score} = \frac{2 * precision * Recall}{precision + Recall} \qquad (3\text{-}5)$$

### 3.3.9 Average precision (AP) and mean Average precision

The AP and mAP are commonly used evaluation metrics for assessing the performance of deep learning-based object detection algorithms. The AP metric is computed as the area under the precision-recall curve, which varies between 0 and 1 for all Recall values. A higher score indicates a better model, and a lower score indicates the opposite. The mAP is the average of the AP scores across all classes, which requires calculating the AP for each class and averaging them. In the literature, the terms AP and mAP are often used interchangeably (Bouguettaya A., 2022). The equations for calculating these metrics are as follows:

$$AP = \int_0^1 P(R)\, d(R) \qquad (3\text{-}6)$$

$$mAP = \frac{1}{N}\sum_{i=1}^{N} APi \qquad (3\text{-}7)$$

Where P, R, and N denote Precision rate, Recall rate, and the number of classes, respectively.

The evaluation metrics employed to assess our model's performance encompass Precision, Recall, F1-Score, and Accuracy. These metrics were carefully selected based on their relevance and significance in providing a comprehensive and robust analysis of our model's predictive capabilities.

## 3.4 Development environment

### 3.4.1 Programming language Python

Python was the chosen software environment for implementing our project. It is a versatile, interpreted, object-oriented, high-level programming language with dynamic semantics, which supports rapid application development and scripting. Python's ability to act as a "glue language" connecting existing components together is a significant advantage. Its simple syntax emphasizes readability, making program maintenance easier, while its module and package system encourage modularity and code reuse. Python's popularity among programmers worldwide can be attributed to its high productivity and ease of use, enabling fast and straightforward edit-test-debug cycles. Additionally, Python's widespread use as a tool for web development, data analysis, artificial intelligence, and automation is attributed, in part, to its capabilities in numerical computing, programming, and graphical visualization, which also make it a favored choice for a broad range of scientific and engineering applications. Moreover, Python and its libraries are easily accessible and freely available for download in both source and binary forms on the majority of platforms, allowing developers to rapidly create applications using the language's extensive pre-written code and modules (Python, n.d.).

### 3.4.2 Main libraries

Neural Networks have been heavily used in the last few years to solve different and difficult problems in various fields. Since then, many frameworks that facilitate the implementation of these networks have appeared. These frameworks, such as TensorFlow, PyTorch, and Keras, provide a high-level interface for building, training, and deploying neural network models, simplifying the process and allowing researchers and developers to focus on the high-level design of their models. The libraries adopted in this project are:

#### 3.4.2.1 TensorFlow

TensorFlow, an open-source software library created by the Google Brain team, serves as a powerful tool for high-performance numerical computation and machine learning. It is specifically designed to encompass every aspect of the machine learning process, ranging from model training to deployment on various devices. It utilizes data flow graphs, where nodes represent mathematical computations (operations) and edges represent tensors (multi-dimensional arrays) that flow between the nodes. TensorFlow, being language and platform independent, supports various programming languages including Python, C++, and R. It also has a tool called TensorBoard that allows for in-depth visualization of models during the training process, providing a web interface for the representation of computational graphs and the understanding of how the parameters and performance change; TensorBoard is an effective tool for the representation of network modeling and performance (Abadi M., 2016). TensorFlow provides a comprehensive interface for expressing machine learning algorithms, along with a Distribution Strategy API that enables the distribution of training across diverse hardware devices without necessitating any modifications to the model. This functionality proves valuable when dealing with larger-scale machine learning tasks (TensorFlow. , n.d.).

The TensorFlow architectures let users use multiple CPUs, GPUs or TPUs to operate the computations or calculations in single application programming interface (API).

#### 3.4.2.2 ImageAI

ImageAI, a Python library, provides assistance to developers, researchers, and students in constructing applications and systems that possess self-contained Deep Learning and Computer Vision functionalities. By offering a collection of state-of-the-art Machine Learning algorithms, ImageAI simplifies the creation of such applications using only concise lines of code. The library provides diverse range of classes and functions that enable powerful computer vision tasks, including image recognition, object detection in images and videos, and video detection analysis. These capabilities can be seamlessly integrated into Python programs, whether for web applications or operating systems such as Windows, Linux, or macOS. ImageAI currently offers support for image prediction and training, utilizing four distinct Machine Learning algorithms that have been trained on the ImageNet-1000 dataset. Additionally, ImageAI enables object detection, video detection, and object tracking through the utilization of RetinaNet, YOLOv3, and TinyYOLOv3, which have been trained on the COCO dataset. Furthermore, ImageAI provides the capability for users to train custom models for the purpose of detecting and recognizing new objects (Renas et al., 2023).

Eventually, ImageAI aims to expand its support encompass a broader range of specialized aspects domains within the field of Computer Vision.

ImageAI is actually built on top of several other popular Python libraries, including:

- **Keras**

Keras is a popular open-source neural network library written in Python that is built to run on top of various platforms, including TensorFlow. Its design has been focused on being user-friendly, modular, and efficient (Keras, 2020). Keras achieves this by using a high-level API that acts as a wrapper over the low-level APIs that handle computations. By taking advantage of these low-level APIs, Keras enables faster model definition and training with minimal coding effort required from the user. Keras is also flexible in its execution, allowing it to run on both CPUs and GPUs. The focus of development is on a simple interface and on enabling rapid test execution (Keras, n.d.).

- **OpenCV**

OpenCV is an open-source computer vision library written in C and C++. It is compatible with Linux, Windows, and Mac OS X operating systems and actively being developed to interface with other programming languages, including Python, Ruby, and MATLAB. The library is designed with an emphasis on computational efficiency and real-time applications, optimized in C, and capable of utilizing multicore processors. Its primary objective is to provide an accessible infrastructure for computer vision that enables the quick creation of complex vision applications and affords a comprehensive solution for computer vision requirements including a general-purpose Machine Learning Library (MLL), which focuses on statistical pattern recognition and clustering, making it a valuable tool for computer vision and machine learning applications (Bradski, 2008).

- **Matplotlib**

Matplotlib, a Python programming language package, is a highly commendable library for creating 2D and 3D graphics. It generates figures of scientific publication quality, offering a wide range of output formats for both print and interactive environments across various platforms. The design philosophy behind matplotlib emphasizes simplicity, allowing users to effortlessly create plots with just a few commands or even a single one. It strives to make simple tasks straightforward and tackle complex ones. Notably, matplotlib is built on the solid foundation of the Numpy and Scipy framework, enabling it to seamlessly adapt to different operating systems and graphics backends, making it a versatile and adaptable data visualization tool (Ari N., 2014).

- **NumPy**

NumPy, or Numerical Python, has gained significant popularity among machine learning practitioners, researchers, and algorithm developers. It is built on the foundations of Python, one of the most widely used programming languages, and serves as an essential resource for scientific computing. The core strength of NumPy lies in its provision of multidimensional arrays, which play a vital role in various computational tasks. By leveraging these arrays, users can efficiently perform complex operations and functions, thereby enhancing their ability to manipulate and analyze data. As a result, NumPy has become an indispensable tool for professionals working in these domains (Nishino, 2017).

## 3.5 Data preparation and labeling

Object detection datasets and models possess tremendous potential as they allow computers to recognize the content of each pixel, thereby improving the accuracy of detecting specific object pictures and enhancing the technologies used in various domains. However, to fully unlock the futuristic potential of object detection and deep learning, it is imperative to train our datasets efficiently; in this regard, selecting the optimal method is an essential aspect. For instance, using YOLOv3 with a self-trained dataset requires labeled images with bounding boxes. Therefore, the LabelImg tool was utilized to label the dataset for this project.

### 3.5.1 Data gathering

The collection of the dataset mainly relayed on the famous FLAME (Fire Luminosity Airborne-based Machine learning Evaluation) dataset, The FLAME dataset consists of fire images captured by drones during a controlled burning of piled detritus in a pine forest in a Ponderosa pine forest on Observatory Mesa, Arizona, USA. The prescribed fire took place on January 16th, 2020 with a temperature of 43◦F (∼ 6°C) and partly cloudy conditions and no wind (Shamsoshoara A., 2021).

The FLAME dataset contains 39,375 images which include 25,018 of type ‘‘fire’’ and 14,357 of type ‘‘non-fire’’. After eliminating images that did not contain fire and those where the fire was not visible within the “fire” set, we selected 2,000 images from the subset of remaining images; thereafter, we manually labeled them.

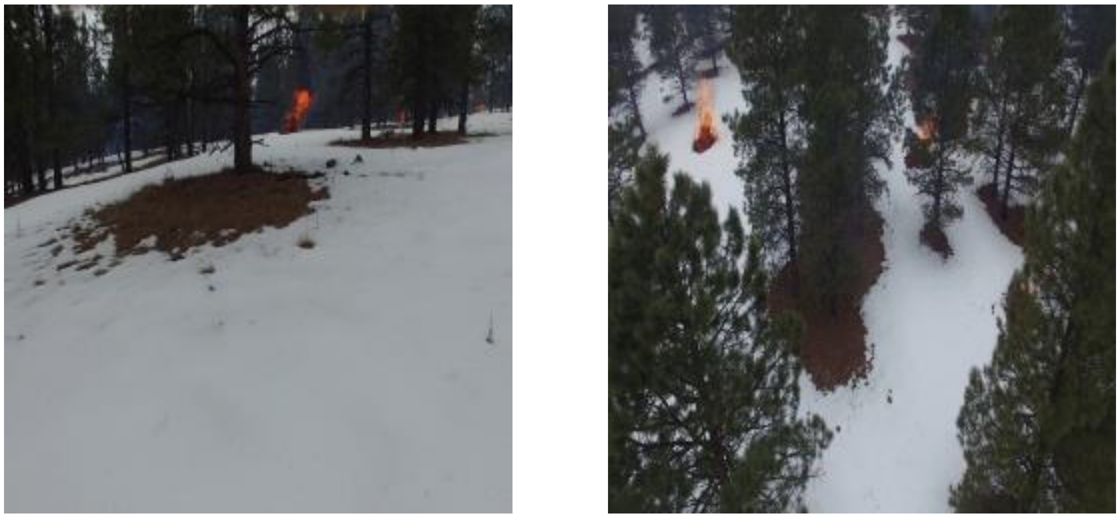


Figure 3.13.Images from the FLAME dataset.

### 3.5.2 Data splitting

The dataset, containing all the features, was prepared to be fed into the machine learning algorithms. Before using the algorithm, it was recommended to perform data splitting.

This technique was utilized to divide the existing data into two subsets, commonly known as the training and validation sets. These subsets serve the purpose of constructing feature sets that enable the model to acquire knowledge from the data.

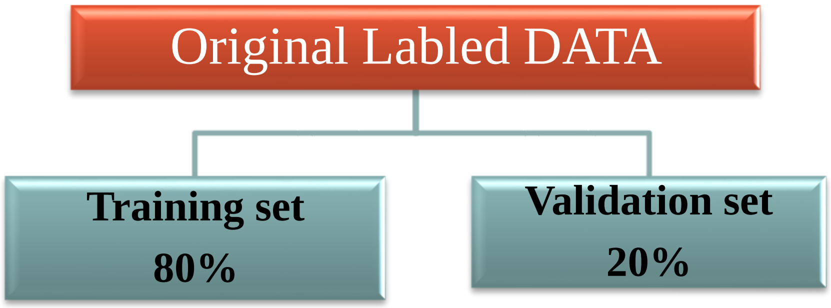


Figure 3.14. Data splitting.

- **Training set**

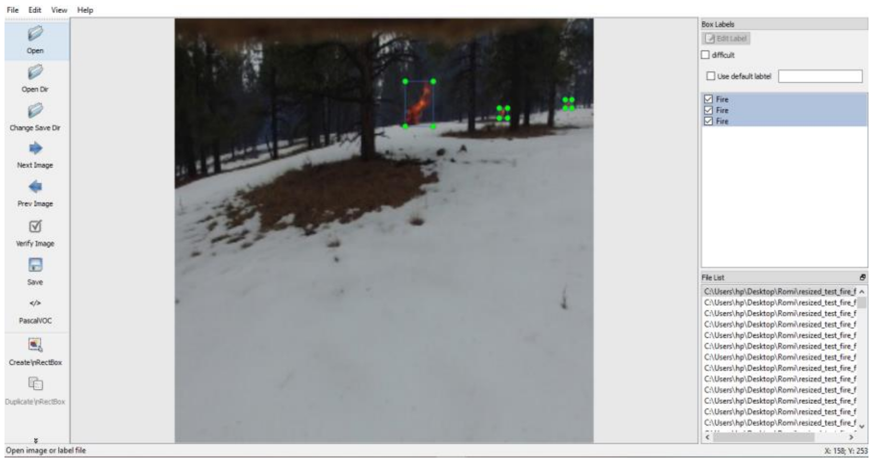
The training set constitutes a portion of the created dataset, specifically chosen for the purpose of enabling the model to learn essential features, fitting parameters, and adjusting weights to comprehend the empirical associations within these features. Typically, 80% of the original dataset is partitioned and designated as the training set.

- **Validation set**

The validation set served the purpose of validating the model's hypotheses. It was comprised of a subset of the created dataset that remained untouched during any of the model's training stages. Typically, this data is withheld from the network until the completion of training. The final model was applied to the validation data in order to obtain an accurate assessment of its performance when deployed on real-world data. For the analysis of the network's performance, a mini-validation set comprising 381 images was utilized.

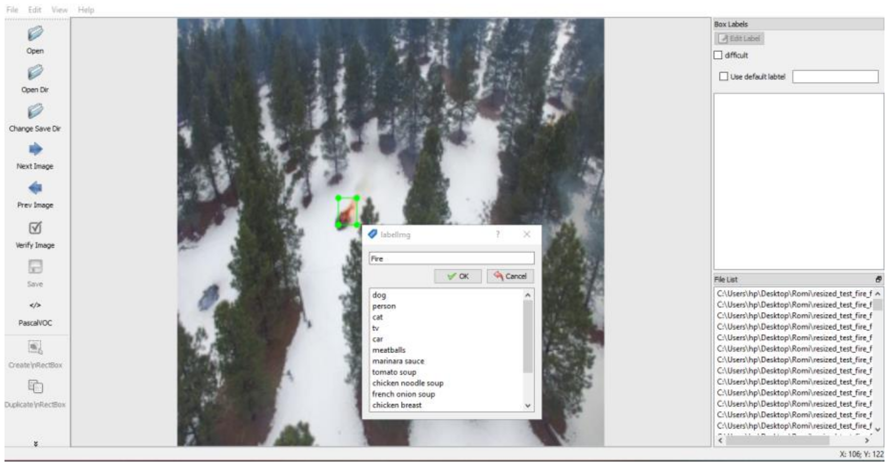
### 3.5.3 LabelImg

Data constitutes a fundamental requirement in the realm of machine learning and is indispensable in the process of training models designed for computer vision tasks, necessitating the inclusion of accurate and precise labels to enable effective learning. The software under consideration serves the specific purpose of facilitating this crucial data labeling task. The LabelImg application is a graphical tool used for annotating images. It is programmed in the Python language and has a graphical user interface built with Qt. The application uses bounding boxes to label objects in an image, then generates a file that contains the classes and coordinates of the bounding boxes for the image. The annotations are saved as XML files in the PASCAL VOC format, which is the same format used by ImageNet. Besides, it also supports YOLO and Create ML formats (Renas et al., 2023).



C:\Users\hp\Desktop\Romi\resized_test_fire_f

Figure 3.15. Labeling data with LabelImg.



Fire

dog
person
cat
tv
car
meatballs
marinara sauce
tomato soup
chicken noodle soup
french onion soup
chicken breast

C:\Users\hp\Desktop\Romi\resized_test_fire_f

Figure 3.16. Data labeling using LabelImg.

### 3.5.4 Transfer learning

Transfer learning is a machine learning technique that leverages pre-trained neural networks to enhance the training process and achieve better results with limited data. By utilizing a pre-trained model trained on a large dataset, the learned parameters and weightings serve as a foundation for training the model with new data. Typically, the last layer of the network is replaced with a separate classification layer, while the convolutional layers' weightings are partially frozen, preserving general features such as edges and corners. This process, known as fine-tuning, optimizes the network by retraining the classification layers while reusing the learned features from earlier layers.

Transfer learning is particularly effective in computer vision and natural language processing, enabling the efficient training of deep neural networks with smaller datasets.

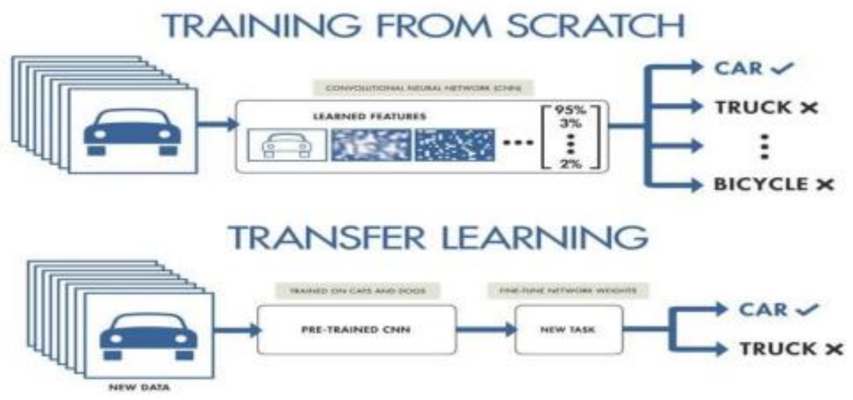


Figure 3.17. An example of transfer learning.

## 3.6 Model training

### 3.6.1 Work station

An efficient implementation of an Object Detection Models will be valid according to precise hardware conditions. For our case, we used:

| | | |
|---|---|---|
| Graphic Card Nvidia (GeForceRTX3090) | GPU frequency | 1400 MHz |
| | Memory capacity | 24 Go |
| | Memory type | GDDR6X |
| | Memory frequency | 2438 MHz |
| | Heat dissipator | Triple |
| Intel Core i9-12900KF | Cache | 30 MB |
| | Max Turbo frequency | 5.20 GHz |
| | Total Cores | 16 |
| | Total threads | 24 |

| | |
|---|---|
| Motherboard | Gigabyte Z690M AORUS Elite DDR4 |
| Power supply | Gigabyte Aorus 850 Watt |
| CPU Cooling | Gigabyte All-in-one liquid cooling system |
| GPU cooling | Tube/Pure 12 ARGB Fan 120*2 |
| Hard Disk | • SSD (Boot) M2. PCIe: 1 TO<br>• HDD (Storage) 5400 RPM 6Gb/S / 2 TO |
| RAM memory | 64 GO (4*16 Go) PG Spectrix D456 RGB DDR4 /PC4-28800 |

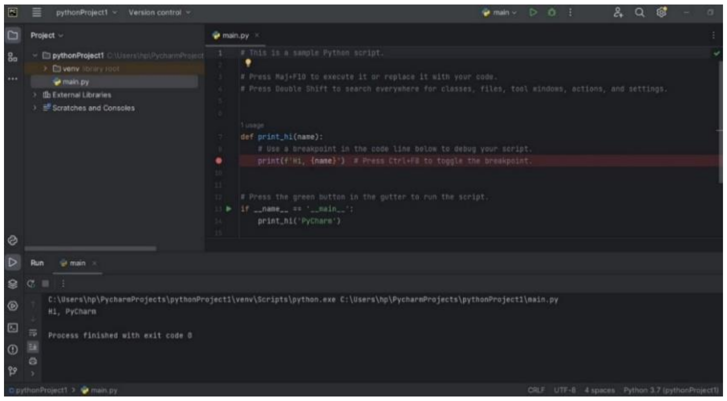
Table 3.2. Training Machine characteristics.

### 3.6.2 Anaconda

Anaconda, a free software distribution of Python designed specifically for machine learning and data science, is available for Windows, macOS, and Linux. It offers different editions to fulfil individual, team, and enterprise needs, and it also provides support for the R programming language (Root, 2020). Anaconda Navigator serves as a user-friendly graphical user interface (GUI) where different integrated development environments (IDEs) can be accessed and launched effortlessly, eliminating the need for command-line inputs. With the Navigator, applications can be managed, conda packages can be controlled, and environments and channels can be efficiently handled, all without relying on the command line interface (CLI). Anaconda includes the conda program, which acts as both a package manager and an environment manager, ensuring that each package version has the necessary dependencies for proper functionality. Additionally, Anaconda simplifies the installation and management of additional packages through its own package manager called conda. The distribution comes pre-installed with more than 1,500 packages commonly used in data analysis and scientific computing, including popular libraries like NumPy, Pandas, SciPy, and Matplotlib. One of the notable advantages of Anaconda is that it provides a consistent environment for projects, ensuring that code runs the same way across different machines regardless of the operating system or other installed software. This saves significant effort and time, especially when collaborating with others or deploying code to production environments. In summary, Anaconda offers a comprehensive solution for Python/R data science and machine learning on a single machine.

### 3.6.3 PyCharm

PyCharm is an IDE (Integrated Development Environment) that offers a consistent user experience across different operating systems like Windows, macOS, and Linux, providing a comprehensive set of essential tools to facilitate productive Python, web, and data science development. It comes in two editions: Professional and Community. The PyCharm Community Edition is tailored for "pure Python" development, is open-source, and completely free, albeit with limited features. On the other hand, the Professional edition is a commercial version that provides an extensive range of tools and features for a more comprehensive development experience.
PyCharm is compatible with various versions of Python, including Python 2 (specifically version 2.7) and Python 3 (from version 3.6 up to version 3.12) (PyCharm, n.d.).



```
# This is a sample Python script.

# Press Maj+F10 to execute it or replace it with your code.
# Press Double Shift to search everywhere for classes, files, tool windows, actions, and settings.


1 usage
def print_hi(name):
    # Use a breakpoint in the code line below to debug your script.
    print(f'Hi, {name}')  # Press Ctrl+F8 to toggle the breakpoint.


# Press the green button in the gutter to run the script.
if __name__ == '__main__':
    print_hi('PyCharm')
```

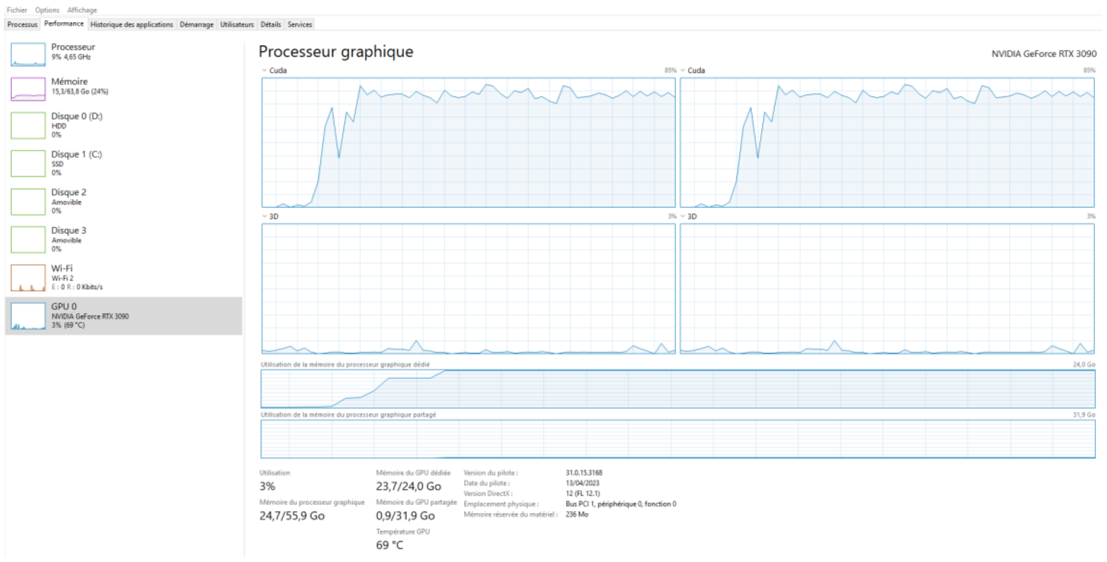
```
C:\Users\hp\PycharmProjects\pythonProject1\venv\Scripts\python.exe C:\Users\hp\PycharmProjects\pythonProject1\main.py
Hi, PyCharm

Process finished with exit code 0
```

Figure 3.18. PyCharm environnement.

### 3.6.4 CUDA

NVIDIA has achieved significant advancements in the development of their GPUs, featuring a variety of architectures. In 2006, NVIDIA introduced their own highly parallel architecture known as compute unified device architecture (CUDA), revolutionizing the programming model for GPUs. CUDA serves as a parallel programming model, utilizing the parallel compute engine of NVIDIA GPUs to efficiently tackle large-scale computational problems. It extends the capabilities of the C programming language and is open source.

A CUDA program is divided into two distinct phases that can be executed on either the host (CPU) or the device (GPU), offering the flexibility to switch between CPU and GPU processing based on specific task requirements (Kalaiselvi, 2017).


Fichier Options Affichage
Processus Performance Historique des applications Démarrage Utilisateurs Détails Services
Processeur graphique
NVIDIA GeForce RTX 3090
Cuda
85%
3D
3%
Utilisation de la mémoire du processeur graphique dédié
24,0 Go
Utilisation de la mémoire du processeur graphique partagé
31,9 Go

Processeur
9% 4,65 GHz

Mémoire
15,3/63,8 Go (24%)

Disque 0 (D:)
HDD
0%

Disque 1 (C:)
SSD
0%

Disque 2
Amovible
0%

Disque 3
Amovible
0%

Wi-Fi
Wi-Fi 2
E : 0 R : 0 Kbits/s

GPU 0
NVIDIA GeForce RTX 3090
3% (69 °C)

Utilisation
3%

Mémoire du processeur graphique
24,7/55,9 Go

Mémoire du GPU dédiée
23,7/24,0 Go

Mémoire du GPU partagée
0,9/31,9 Go

Température GPU
69 °C

Version du pilote : 31.0.15.3168
Date du pilote : 13/04/2023
Version DirectX : 12 (FL 12.1)
Emplacement physique : Bus PCI 1, périphérique 0, fonction 0
Mémoire réservée du matériel : 236 Mo

Figure 3.19. GPU and CUDA consumptions.

## 3.7 Results

This section presents the experimental results obtained from a series of conducted tests that aimed to evaluate the performance and effectiveness of the proposed models. The tests involved varying batch sizes and epochs until the GPU memory of our workstation was fully utilized. Our strategy was to fix a specific batch size and vary the number of epochs. The batch sizes used were 8, 16, and 29, while the epochs were set as 30, 60, and 100.

As a first step, we selected a batch size of 8 for the three aforementioned epochs.

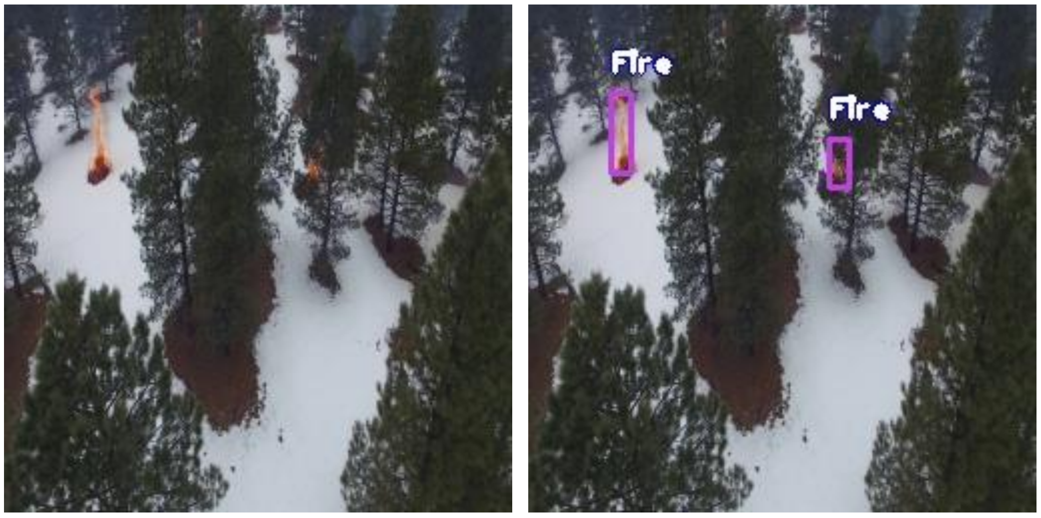


Figure 3.20. The obtained results corresponding to batch size 8 epoch 30.

In terms of performance metrics, the achieved Precision, Recall, F1-Score, and Accuracy values were 98%, 96.66667%, 97.328767% and 94.666667% respectively.

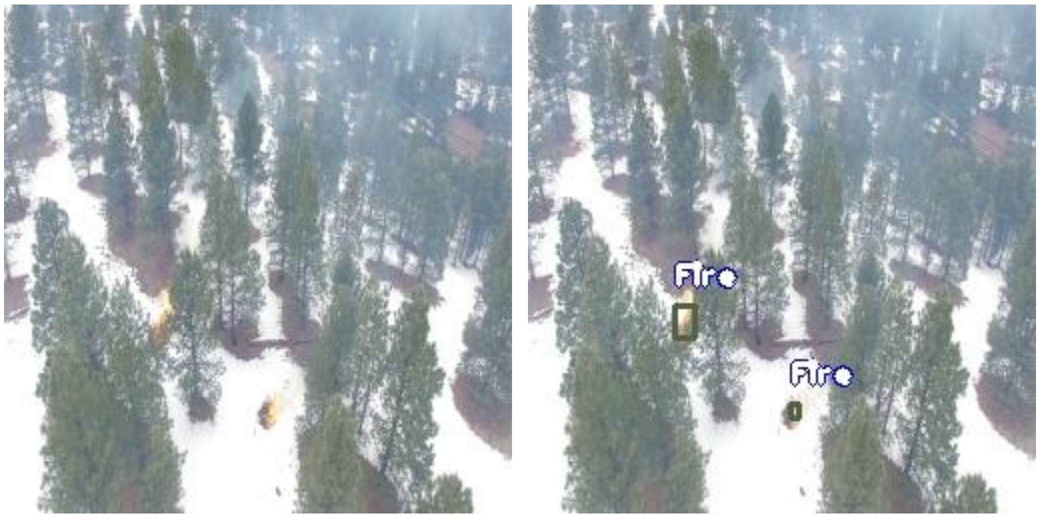


Figure 3.21. The obtained results corresponding to batch size 8 epoch 60.

In terms of performance metrics, the achieved Precision, Recall, F1-Score, and Accuracy values were 98%, 95.33%, 96.648576%, and 93.33% respectively.

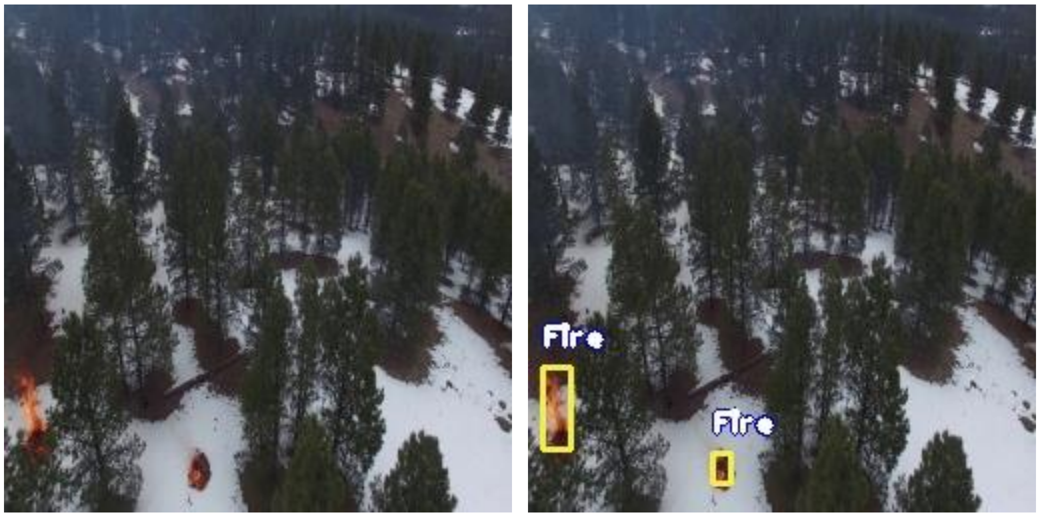


Figure 3.22. The obtained results corresponding to batch size 8 epoch 100.

In terms of performance metrics, the achieved Precision, Recall, F1-Score, and Accuracy values were 98%, 95.33%, 96.648276%, and 93.33% respectively.

After analyzing the performance results of the model with a batch size of 8 and different epochs, it was observed that the model performed better with 30 epochs.

Subsequently, the algorithm was utilized with a batch size of 16 throughout the three epochs.

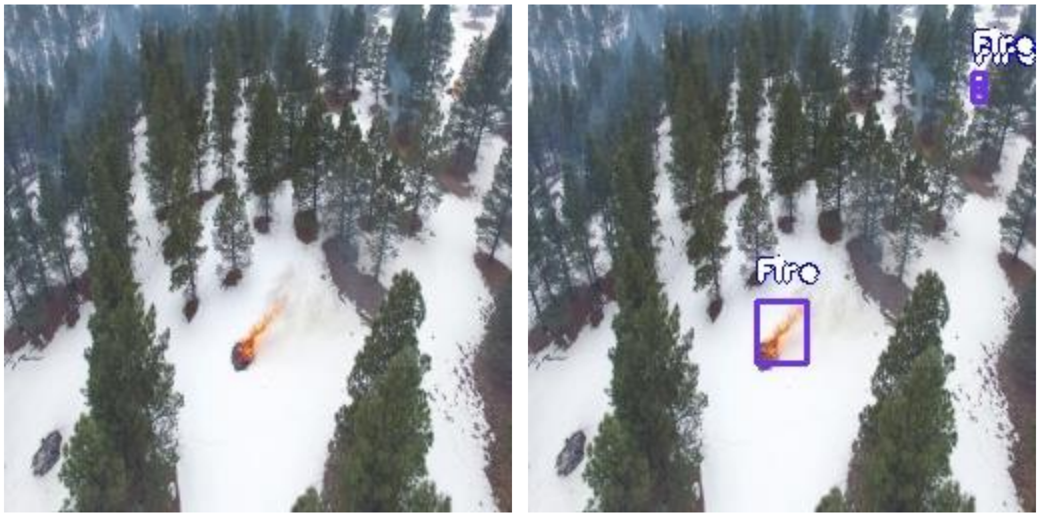


Figure 3.23. The obtained results corresponding to batch size 16 epoch 30.

In terms of performance metrics, the achieved Precision, Recall, F1-Score, and Accuracy values were 98%, 98%, 98%, and 94.666667% respectively.

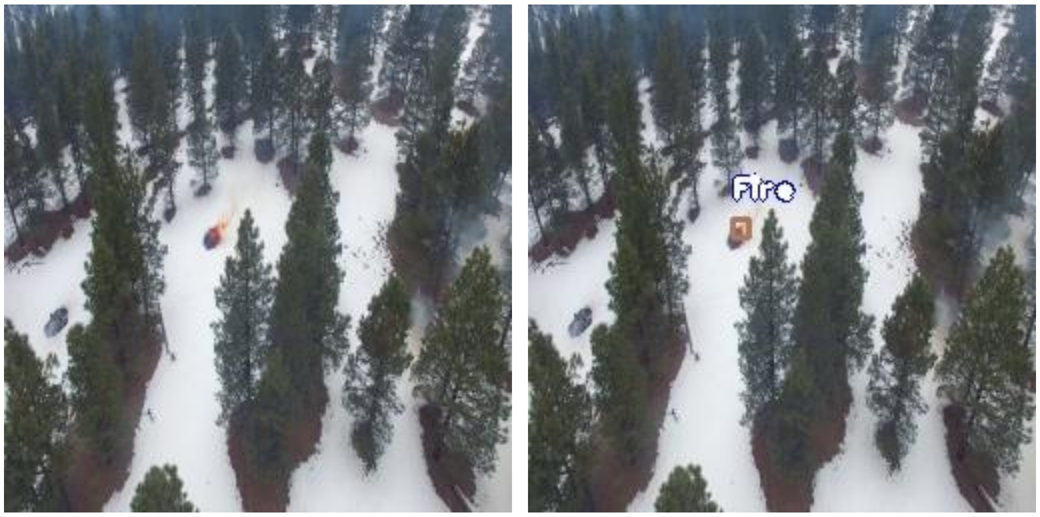


Figure 3.24. The obtained results corresponding to batch size 16 epoch 60.

In terms of performance metrics, the achieved Precision, Recall, F1-Score, and Accuracy values were 98%, 95.33%, 96.648276%, and 93.33% respectively.

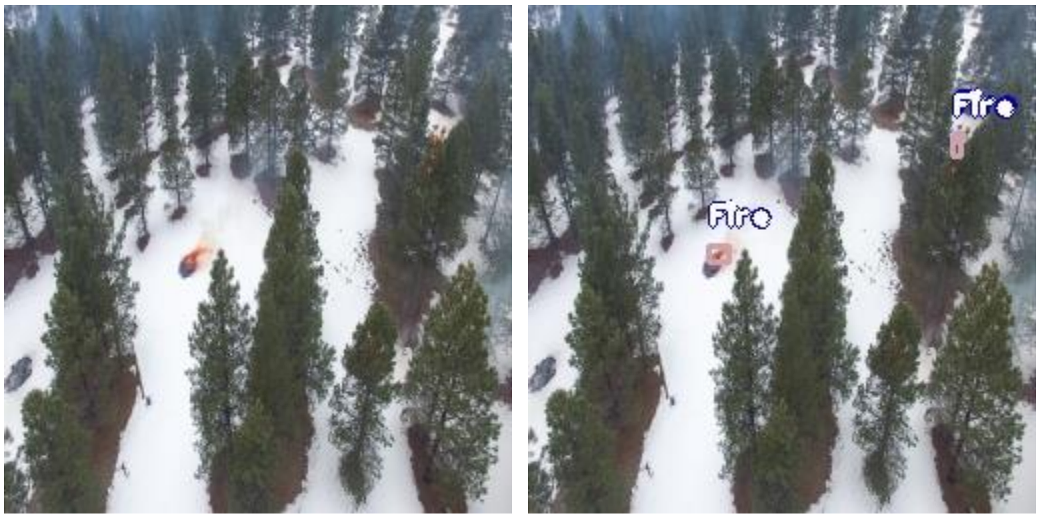


Figure 3.25. The obtained results corresponding to batch size 16 epoch 100.

In terms of performance metrics, the achieved Precision, Recall, F1-Score, and Accuracy values were 98%, 95.33%, 96.648576%, and 93.33% respectively.

Based on the analysis of the performance results, it was observed that the model with a batch size of 16 achieved better performance with 30 epochs.

Finally, the batch size of 29 was utilized for each of the three distinct epochs.

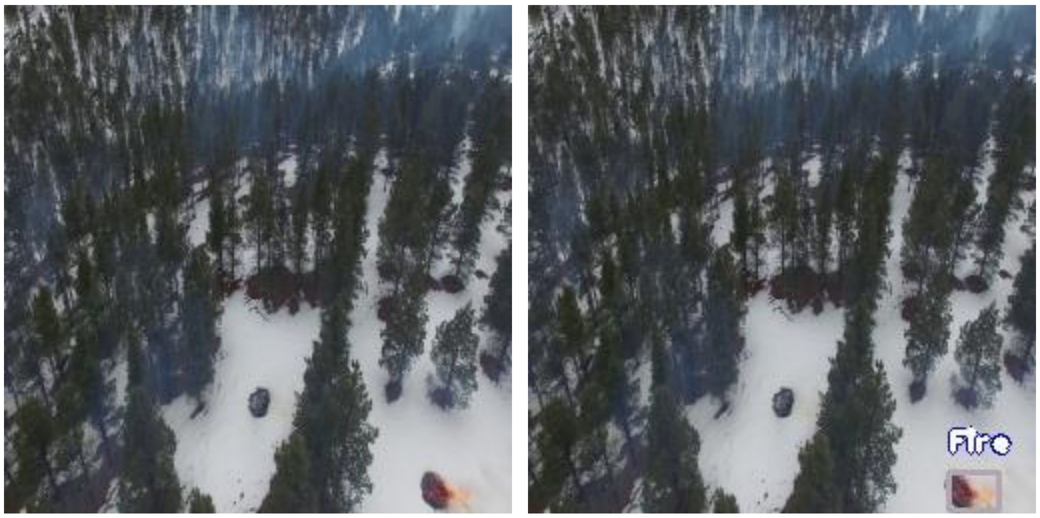


Figure 3.26. The obtained results corresponding to batch size 29 epoch 30.

In terms of performance metrics, the achieved Precision, Recall, F1-Score, and Accuracy values were 100%, 96.66667%, 98.305085%, and 96.66667% respectively.

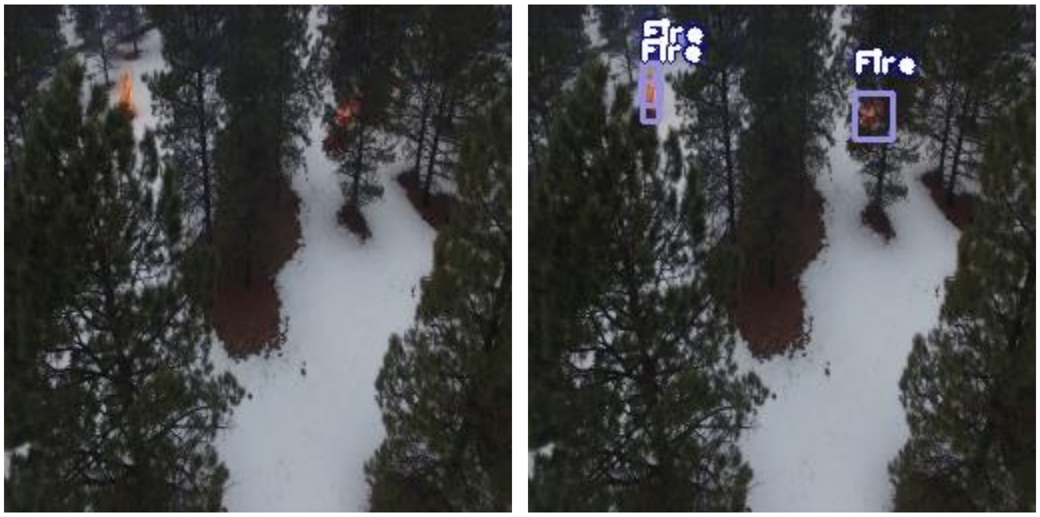


Figure 3.27. The obtained results corresponding to batch size 29 epoch 60.

In terms of performance metrics, the achieved Precision, Recall, F1-Score, and Accuracy values were 100%, 95.33%, 97.610922%, and 95.33% respectively.

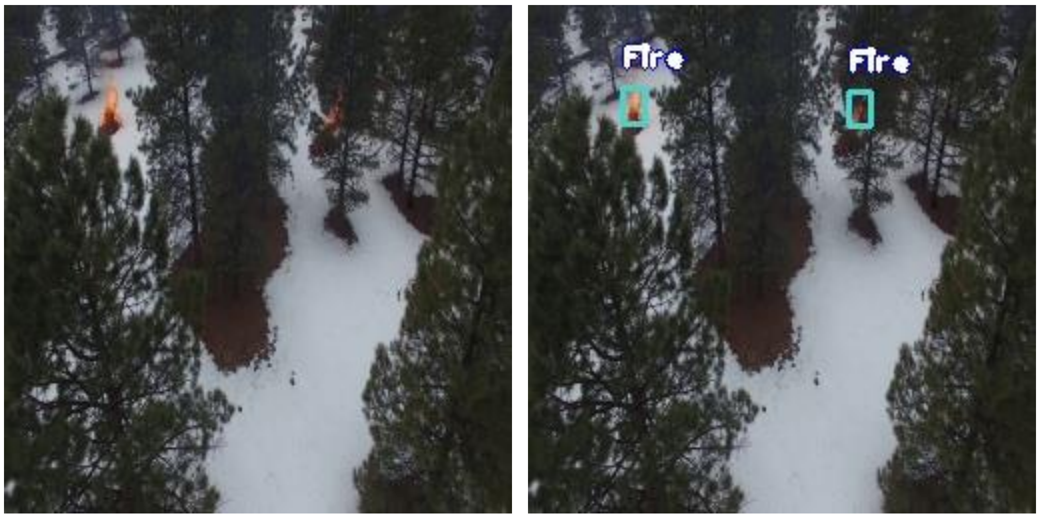


Figure 3.28. The obtained results corresponding to batch size 29 epoch 100.

In terms of performance metrics, the achieved Precision, Recall, F1-Score, and Accuracy values were 100%, 96.66667%, 98.305085%, and 96.666667% respectively.

Through a comprehensive analysis of the performance results obtained from the model with a batch size of 29 and varying epochs, it was determined that the model exhibited superior performance, specifically with 30 epochs.

| Batch Size | Epochs | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|---|
| Batch size 8 | 30 | 94.66667% | 98% | 96.66667% | 97.328767% |
| | 60 | 93.33% | 98% | 95.33% | 96.648276% |
| | 100 | 93.33% | 98% | 95.33% | 96.648276% |
| Batch size 16 | 30 | 94.666667% | 98% | 98% | 98% |
| | 60 | 93.33% | 98% | 95.33% | 96.648276% |
| | 100 | 93.33% | 98% | 95.33% | 96.648276% |
| Batch size 29 | 30 | 96.66667% | 100% | 96.66667% | 98.305085% |
| | 60 | 95.33% | 100% | 95.33% | 97.610922% |
| | 100 | 96.666667% | 100% | 96.66667% | 98.305085% |

Table 3.3. Comparison between the created models.

For evaluating the performance of each created model, we selected Accuracy, Precision, Recall, and F1-Score as evaluation metrics. Subsequently, a comparative table 3.3 was generated to facilitate the comparison between our models. By conducting a comparative analysis, it becomes evident that the model utilizing a batch size of 29 demonstrates superior performance, surpassing all other models in terms of the aforementioned metrics. The results obtained from the selected model in terms of Precision, Recall, F1-Score, and Accuracy are respectively as follows: 100%, 96.66667%, 98.305085%, and 96.666667%, which are highly satisfactory outcomes.

The study we carried out has provided us with the valuable insight that increasing the batch size results in a more performant model. Additionally, it has shed light on the benefits of higher batch sizes, such as improved convergence speed and enhanced stability, during the training process.

During the study of the batch size effect on our model, it was discovered that the epoch parameter also has an influence on its performance. It is known that increasing the number of epochs leads to improved outcomes; however, in our specific study, it was determined that the optimal performance was achieved with 30 epochs. This finding indicate that our model reaches its peak performance at this number of epochs, rendering further epochs unnecessary.

In conclusion, based on our analysis, it can be deduced that the model with a batch size 29 and epoch 30 exhibiting the highest performance.

## 3.8 Conclusion

In this chapter, we have provided an overview of the YOLO model architecture and its various versions, up to version 3. We specifically utilized version 3 in our work, employing transfer learning with the FLAME aerial image dataset. Furthermore, we explored various performance evaluation measures for assessing the effectiveness of deep learning models, including recall, F1-Score, precision, and accuracy. Moreover, we introduced PyCharm as a widely utilized training environment in the field of deep learning, explained our choice of libraries, outlined the selection of training parameters, and provided an explanation of the data labeling process.

The core analysis of this chapter centers on comparing the model's performance under different batch sizes and epochs. Various combinations are tested and evaluated. After analyzing the final results, we determined that the model achieved the highest level of performance when trained with a batch size of 29 and 30 epochs.

Finally, we discussed the implications and significance of these findings, emphasizing the potential for improving deep learning models by carefully selecting these parameters.

# GENERAL CONCLUSION

Forest fires are recognized as a major risk in Algeria under the framework of the law 04-20, which pertains to the prevention of significant risks and disaster management, due to their potential to cause extensive economic and ecological losses as well as endangering human lives. Therefore, there is growing attention paid to early detection and monitoring of forest fires to preserve natural resources and safeguard people and property by adopting faster and more efficient methods.

As shown in the research study, a model has been designed and developed for the early detection and monitoring of forest fires using UAVs and Artificial Intelligence. The initial section of this thesis focuses on providing an overview of the Algerian forest, including statistical data of forest fires. Additionally, an exploration of existing methods for forest fire detection systems. The purpose is to familiarize readers with the fundamental concepts, enhance their comprehension of these ideas, and gain a clear understanding of the study's objectives. In the second chapter, convolutional neural networks and various optimization algorithms were introduced as they offer a promising solution to enhance the performance of our detection system. By incorporating these techniques, we aim to improve the effectiveness and efficiency of our system's capabilities. In the last chapter we present a real-time model evaluation and testing.

In our study, we tackled the issue of Fire classification and localization by leveraging state-of-the-art deep learning techniques. Specifically, we utilized the YOLOv3 algorithm, known for its exceptional object detection capabilities, to recognize fire classes through image analysis.

To construct this fire detector based on computer vision, two crucial components were required: the "FLAME" dataset and transfer learning. The subset we created from the mentioned dataset was manually labeled using the LabelImg application.

Therefore, we create an environment in Anaconda that enables us to install and collect the libraries needed in the creation of our model, this software gives us access to PyCharm where we developed our program using a pre-trained model, all these were done on a high-performance workstation which allow the training to be executed with various epochs and batch sizes.

We concluded from this experience that increasing the batch size results in a more performant model, whereas epoch 30 gave us the most satisfying results, eliminating the need for more epochs in the training process.

Finally, the model with a batch size of 29 and epoch 30 gained the superior performance, achieving exceptional metrics: Precision (100%), Recall (96.66667%), F1-Score (98.305085%), and Accuracy (96.66667%). These outcomes signify highly satisfactory model evaluation.

YOLOv3 demonstrates the advantageous ability to perform real-time execution while preserving a high level of predictive performance and accurately identifying objects with exceptional precision.

We can enhance our model by working on some aspects such as:

- Our intention is to implement the proposed fire detection system on UAVs to actively detect forest fires in real-world scenarios. This embedded solution comprise a camera and a compatible AI development kit, such as the NVIDIA Jetson card.
- Increasing the number of images used for training the model by incorporating multiple aerial databases.
- The development of a system that automatically sends alerts to relevant services in case of a fire departure.
- Evaluate the performance of our model in comparison to YOLOv5 and YOLOv7 within our research domain, and select the variant that demonstrates superior performance.

# APPENDIX

**The Supervision Team and the Work Team**

- The Supervision Team

| The Supervision Team | |
|---|---|
| **(01)**: Primary supervisor<br>**Ahmed Kechida** | Field of specialization<br>**Electronics** |
| **(02)**: Primary supervisor<br>**Amine Taberkit** | Field of specialization<br>**Electronics** |
| Associate supervisor<br>**Hocine Bentrad** | Field of specialization<br>**Propulsion** |

- The Work Team

| The Work Team: | Field of specialization | Faculty |
|---|---|---|
| Student: Romaissa Bouhamam | Avionics | Institute of Aeronautics and Space Studies |
| Student: Nesrine Touahria | Avionics | Institute of Aeronautics and Space Studies |

## 1. Proposed solution

The increasing frequency of wildfires in recent years has prompted a consideration to create a system in the industrial agriculture field that restrict the damages we have seen in our country. In the beginning, our initial thought was to utilize systems that could be applied in Algeria such as cameras and sensors. However, after conducting thorough bibliographic research, we identified several advantages in utilizing a cutting-edge system that combines drone and artificial intelligence technologies in comparison to other techniques for the early detection of fires.

Our project entails the implementation of an embedded system on an AI-oriented development kit equipped with an onboard camera which will be done by our work team.

This project is accomplished in the Research Center in Industrial Technologies (CRTI).

## 2. Value propositions

In 2021, Algeria lived horrendous wildfire in Tizi-Ouzou that captured the attention of several researchers and made them work on this thematic nevertheless until today there is no deployment of such technique in the Algerian forests, which is the reason that made us work on it to make a higher performance system and trying to make it a successful project. Considering that the studies conducted in this dilemma were exclusively focused on ground processing which, in contrast our solution is based on instantly on-board processing.

The core objective of the project centers on fire detection as the primary focus. However, it also allows for the implementation of other functionalities that rely on detection capabilities, thereby providing flexibility for diverse application needs with the possibility of adjustment in regard to the client needs enabling a vast category of clients to accomplish their missions.

Our product not only ensures safe usage without any possible risks but also extends accessibility to individuals who were previously unable to utilize it. Moreover, our product empowers staff members to effortlessly take advantage of its functionalities without the need for extensive technical expertise, thereby enhancing user-friendliness.

## 3. Work team

Our team consists of two students from the field of Avionics within the Aeronautics and Space Studies Institute.

| | **Nesrine Touahria** | **Romaissa Bouhamam** |
|---|---|---|
| **Skills** | • Communication<br>• Technical<br>• Teamwork<br>• Research<br>• Computer Programming<br>• Project Management<br>• Strategic Planning | • Technical<br>• Teamwork<br>• Research<br>• Computer Programming<br>• Customer Service<br>• Leadership<br>• Data Analysis |
| **Practical internship** | • Air Algerie Company<br>• ERMA<br>• ENNA | • Air Algerie Company<br>• ERMA |
| **Distribution of tasks and responsibilities** | • Everything related to patent writing.<br>• Legislative aspect.<br>• Market analysis, marketing. | • Responsibility for all technical aspects of project development. |
| **Team interaction and communication methods** | Regular team meetings, E-mail communication, instant messaging and calls, and video conferencing calls. | |

## 4. Project goals

Taking advantage of the recent technologies that are available and affordable nowadays to propel our country several steps ahead and decrease economic and ecological losses while preserving citizens' lives, we can not only make the process safer by using drones to reach areas that are inaccessible or considered too dangerous for firefighting crews, but also significantly expedite operations due to the maneuverability of UAVs. Moreover, the embedded system on a drone lightens the workload by remotely supplying real-time fire information to human firefighters.

The establishment of our start-up will make a significant contribution towards reducing our country's reliance on purchasing complete packages from other developed countries. By fulfilling our needs internally, we will generate job opportunities and effectively decrease unemployment rates. Additionally, this initiative has the potential to be promoted at an international level, thereby creating new sources of revenue for our country.

## 5. A timeline for the realization of the project

| | | | Months | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 1 | | Preliminary studies: selection of the production unit's location, preparation of the necessary documents. | ✓ | ✓ | | | | | |
| 2 | | equipment orders | | | ✓ | ✓ | | | |
| 3 | | Construction of a production headquarters (factory) | | ✓ | ✓ | | | | |
| ... | | equipment installation | | | | ✓ | ✓ | | |
| N | | Registering a patent in order to obtain an application code and industrial protection | | | | | | ✓ | ✓ |
| ... | | Prototype realization | | | | | | ✓ | ✓ |

- **Breaking down the ultimate objective of a patent into individual tasks**

The tasks are divided as follows:

- Model programming.
- Implementation.

Additionally, patent redaction was a shared task between the work team.

## 6. Market sector

- **Potential market:** The target sector is forest surveillance "Forest General Direction and its subdivision all over the country, civil protection" as they are in need of supplies to protect our land from any major disaster.

- **Target market:** There are several other companies to whom we can present my project since it can diminish and facilitate some staff tasks. Among these companies we can cite: Sonatrach, Sonalgaz, etc and expanding our selling to the international level.

## 7. Measure of competitiveness

My project has never been achieved in real life or been commercialized before so we don't have any competitors at the moment. However, there may be other competitors in the future, we can distinguish ourselves from our competitors by the possibility of trying our solutions on drones which have a long autonomy, and by the fact that we are each time improving our solution and adding other options.

## 8. Costs and expenses

- A high performance work station: We need a high performance work station to train our model based on Artificial Intelligence, and to prepare all our marketing and design supports : banner, videos, images, we predict that the cost of an acceptable high performance work station will be around : 1.000.000 DA.
- We need a compatible AI development kit, and also a high resolution camera, that is compatible with the kit, we can predict that the cost of both kit and camera will be around 110.000 DA.
- The startup will need a rent, electricity, and other costs as well. We did not estimate these costs since we are in the beginning in projects incubator, and we project to join a business accelerator.

| | | | |
|---|---|---|---|
| **Business Model Canvas – *BMC*** | *Project holders:*<br>1- Nesrine Touahria<br>2- Romaissa Bouhamam | *Supervisors:*<br>S- Ahmed Kechida<br>CO-S- Hocine Bentrad | *Project code:*<br>05-15-3133 |

*Start-up Project :* UAV Aerial Image-Based Forest Fire Detection Using artificial intelligence

| **Key Partners**: | **Key Activities:** | **Value Propositions:** | **Customer Relationships:** | **Customer Segments:** |
|---|---|---|---|---|
| • DGF (Forests General Directorate).<br>• CRTI (Research Center in Industrial Technologies). | • Forest fire detection system using artificial intelligence.<br>• Implementation of the algorithm on the development kit.<br>• Embed our card onto the VTOL unmanned aerial vehicle, manufactured by the CRTI research center. | • Real-time forest fires detection.<br>• On-board processing.<br>• Deployment of cutting-edge systems "drone".<br>• Real-time alerts.<br>• Adding functionalities based on the client's needs.<br>• Customization according to the client. | • Contracts.<br>• Service offerings.<br>• License disposal.<br>• Orientation and consultation regarding system usage. | • Forests General Directorate.<br>• The Algerian wilayas' forest administrations.<br>• Civil Protection General Directorate.<br>• Security-specialized companies.<br>• Sonalgaz.<br>• Sonatrach. |
| | **Key Resources:** | | **Channels:** | |
| | • Essential hardware**.**<br>• Databases.<br>• Competent individuals in computer science.<br>• Funds.<br>• Software licensing. | | • Social media.<br>• Smartphone application.<br>• Website.<br>• Participation in entrepreneurship fairs and events. | |

| **Cost Structure**: | **Revenue Streams:** |
|---|---|
| • A development kit specially designed for the utilization of embedded systems.<br>• An integrated camera that is fully compatible with the utilized development kit.<br>• A high-performance work station.<br>• Rent, electricity, and other costs. | • Selling electronic cards containing the algorithm for early detection of forest fires and industrial fire detection.<br>• Offering AI training courses. |

# REFERENCES

Abadi M., B. P. (2016). Tensorflow: a system for large-scale machine learning. . *In OSDI, Vol 16*, pages 265–283.

Abdulkader A., &. V. (2020). Real-time vehicle and pedestrian detection, a data-driven recommendation focusing on safety as a perception to autonomous vehicles. Retrieved from http://www.diva-portal.org/smash/record.jsf?dswid=-864&pid=diva2%3A1479957

Ag, A. (2018). *Batch Normalization in Deep Learning,"*. Retrieved from Medium: https://medium.com/ai%C2%B3-theory-practice-business/batch-normalization-in-deeplearning-ca215a7a7a5d

Alkhatib, A. A. (2013). A Review on Forest Fire Detection Techniques. doi:10.1155/2014/597368

Alzubaidi, L. Z. (2021). Review of deep learning: concepts, CNN architectures, challenges, applications, future directions. J Big Data. *Springer Link*, 26-31. Retrieved from https://doi.org/10.1186/s40537-021-00444-8

Arfa A.M.T. (2008). les incendies de forets en Algérie: Strategie de prévention et plan de gestion. *Memoire de magister*. Université Mentouri SNV, Constantine.

Ari N., a. U. (2014). Matplotlib in python. *11th International Conference on Electronics, Computer and Computation (ICECCO)*, (pp. pp. 1-6,). Abuja, Nigeria, . doi:10.1109/ICECCO.2014.6997585.

Aubert G. (1991). Effets de l'incenide sur les sols forestiers. Symposium (la foret carbonisée, son present, son futurs). *Revue n°1 (les cahiers du conservatoire du littoral) n°2 (foret méditerranénne: vivreavec le feu).*

Bajocco S. and Ricotta C. (2008). Evidance of selective burning in Sardinia: which land cover classes do wildfires prefer? Landscape Ecology. 241-248.

Barmpoutis P, P. P. (2020). A Review on Early Forest Fire Detection Systems Using Optical Remote Sensing. Sensors. 20(22):6442. Retrieved from https://doi.org/10.3390/s20226442

Bazaga A., R. M.-M. (2019). A Convolutional Neural Network for the automatic diagnosis of collagen VI-related muscular dystrophies. *Appl. Soft Comput.,, vol. 85, p. 105772*. doi:10.1016/j.asoc.2019.105772.

Bouguettaya A., Z. H. (2022). A review on early wildfire detection from unmanned aerial vehicles using deep learning-based computer vision algorithms. *ELSEVIER*. Retrieved from https://doi.org/10.1016/j.sigpro.2021.108309.

Bradski, G. &. (2008). *Learning OpenCV: Computer vision with the OpenCV library.* «O'Reilly Media, Inc.". Retrieved from https://books.google.dz/books?hl=fr&lr=&id=seAgiOfu2EIC&oi=fnd&pg=PR3&dq=opencv+library&ots=hVL46fbJSf&sig=jRIbBf6OCOFup8_cD8Fk94FBhT4&redir_esc=y#v=onepage&q=opencv%20library&f=false

Branka A. (2001). Etude Comparée des incendies de forets et de leur prévention dans les départements du Var et des Landes. *Mémoire de fin d'études,*, 89 p. Université de Cergy-Pontoise, UFR des Scienes Humaines, Département de Géographie,.

Chung Y.-L., &. L.-K. (2020). Application of a Model that Combines the YOLOv3 Object Detection Algorithm and Canny Edge Detection Algorithm to Detect Highway Accidents. doi:10.3390/sym12111875

Datahacker.rs. (2018).

DGF, D. G. (2022). *Les Satistiques des superficies parcourue par les incendie.* Alger.

FOSA. (2001). *Forestry Outlook study for Africa.* Algerie.

He, K. Z. (2015). Deep Residual Learning for Image Recognition. Retrieved from https://arxiv.org/pdf/1512.03385.pdf

Ioffe, S. &. (2015). Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariat Shift. Retrieved from https://arxiv.org/pdf/1502.03167.pdf

Jakhar D., K. I. (2020). Artificial intelligence, machine learning and deep learning: definitions and differences, Clinical and Experimental Dermatology,. 131-132. Retrieved from https://doi.org/10.1111/ced.14029

Kalaiselvi, T. S. (2017). Survey of using GPU CUDA programming model in medical image analysis. *Informatics in Medicine Unlocked, 9, 133-144.* Retrieved from Survey of using GPU CUDA programming model in medical image analysis - ScienceDirect

Kathuria, A. (2018, Apr 21). Retrieved from https://towardsdatascience.com/yolo-v3-object-detection-53fb7d3bfe6b

Kazi Aoual N., R. S. (2010). *La génération des forets par l'utilisation des eaux usées.* Hammamet.

*Keras*. (n.d.). Retrieved from https://keras.io/

*Keras*. (2020). Retrieved from "Getting started with the Keras Sequential," [Online].: https://keras.io/gettingstarted/sequential-model-guide/.

Khan S., R. H. (2018). A Guide to Convolutional Neural Networks for Computer Vision,. *Synth. Lect. Comput. Vis.,, vol. 8, no. 1*. doi:10.2200/S00822ED1V01Y201712COV015.

Li G., S. Z. (2018). A New Method of Image Detection for Small Datasets under the Framework of YOLO Network. *IEEE 3rd Advanced Information Technology,Electronic and Automation Control Conference (IAEAC),*. doi:10.1109/IAEAC.2018.8577214

Lokanath M., K. K. (2017). Accurate object classification and detection by faster-RCNN,. *IOP Conf. Ser. Mater. Sci. Eng, 263, p. 052028.* doi:10.1088/1757-899X/263/5/052028.

M. Bo, L. M. (2020). Urban air pollution, climate change, and wildfires: the case study of an extended forest fire episode in northern Italy favored by drought and warm weather conditions, . *Energy Rep*, pp. 78.

*Manishgupta* . (n.d.). Retrieved from "YOLO – You Only Look Once”. : towardsdatasci-ence.com

Margerit J. (1998). Modélisation et simulations numériques de la propagation des feux de forets. 260-261. Institut National Polytechnique de Lorraine, Nancy, France.

Menegaz, M. (2018). Understanding YOLO. *hackernoon*.

Mohana M., a. R. (2019). Object Detection and Tracking using Deep Learning and Artificial Intelligence for Video Surveillance Applications. *Int. J. Adv. Comput. Sci. Appl.,, vol. 10*. doi:10.14569/IJACSA.2019.0101269.

Moreira F., R. F. (2001). Temporal (1958-1995) pattern of change in a cultural landscape of northwestern Portugal: Implications for fire occurrence Landscape Ecology. 557-567.

Namoune A. (2021). Conception et réalisation d'un drone hybride VTOL NADJAH 200. *Mémoire de fin d'etude Master*. Institut d'Aeronautique et des Etudes Spatial, Blida.

Nishino, R. O. (2017). Cupy: A numpy-compatible library for nvidia gpu calculations. *31st confernce on neural information processing systems,* (p. 151(7)). Retrieved from paper_16.pdf (learningsys.org)

Nunes M.C.S., V. M. (2005). Nunes M.C.S., Vansconcelos M.J., Pereira J.M.C., Dasgupta N., Alldredge R.JLand cover type and fire in Portugal: do fires burn land cover selectively? Lanscape Ecology. 661-673.

Prove P.L. (2017, Jul 22). *Medium.* Retrieved from https://towardsdatascience.com/types-of-convolutions-in-deep-learning-717013397f4d

*PyCharm*. (n.d.). Retrieved from PyCharm web site: https://tinyurl.com/6wpwuvy6

*Python*. (n.d.). Retrieved from https://www.python.org/doc/essays/blurb/

Redmon J., &. F. (2018). YOLOv3: An Incremental Improvement. Retrieved from https://arxiv.org/pdf/1804.02767.pdf

Redmon, J. &. (2016). YOLO9000: Better, Faster, Stronger. https://arxiv.org/pdf/1612.08242.pdf. Retrieved from https://arxiv.org/pdf/1612.08242.pdf

Redmon, J. D. (2016). You Only Look Once: Unified, Real Time Object Detection. Retrieved from https://arxiv.org/pdf/1506.02640.pdf

Renas et al. (2023). Object Detection using the ImageAI Library in Python. Retrieved from View of Object Detection using the ImageAI Library in Python (pgjsrt.com)

Root, D. (2020, Oct 8). *Medium.* Retrieved from https://towardsdatascience.com/an-overview-of-the-anaconda-distribution-9479ff1859e6

Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint*. Retrieved from 1609.04747. 1609.04747.pdf (arxiv.org)

S. Vardoulakis, G. M.-6. (2020). Lessons learned from the Australian bushfires: climate change, air pollution, and public health. *JAMA Intern. Med*, pp. 635-636. doi:10.1001/jamainternmed.2020.0703.

Saad H.H., a. A. (2021). COMPARISON OF OPTIMIZATION TECHNIQUES BASED ON GRADIENT DESCENT ALGORITHM. *A REVIEW. PalArch's Journal of Archaeology of Egypt / Egyptology*. Retrieved from https://archives.palarch.nl/

Science, O. (2018). *Overview of the YOLO Object Detection Algorithm.* Retrieved from Medium: https://odsc.medium.com/overview-of-the-yolo-object-detection-algorithm-7b52a745d3e0

Shamsoshoara A., A. F. (2021). Aerial imagery pile burn detection using deep learning: The FLAME dataset. *Computer Networks*. Retrieved from https://doi.org/10.1016/j.comnet.2021.108001.

Sharma S., a. A. (2017). Activation functions in neural networks. *Towards Data sci*. doi:6(12), 310-316.

Shinde P. P., a. S. (2018). A Review of Machine Learning and Deep Learning Applications,. *Fourth International Conference on Computing Communication Control and Automation (ICCUBEA).* India.

Short, K. C. (2017). Spatial wildfire occurrence data for the United States, 1992-2015 [FPA_FOD_20170508]. . *4th Edition. Fort Collins, CO: Forest Service Research Data Archive*. Retrieved from https://doi.org/10.2737/RDS-2013-0009.4.

*StackOverflow*. (n.d.). Retrieved from “Intersection over Union (IoU) ground truth in YOLO": https://stackoverflow.com/questions/61758075/intersection-over-union-iou-ground-truth-in-yolo

*TensorFlow*. . (n.d.). Retrieved from https://www.tensorflow.org/

Yamashita, R. N. (2018). Convolutional neural networks: an overview and application in radiology. Retrieved from https://doi.org/10.1007/s13244-018-0639-9

Yi, Y. (2020). *Intelligent System and Computing.*

Zhang W. J., Y. G. (2018). On Definition of Deep Learning,. *World Automation Congress (WAC), Stevenson, WA,*, (pp. 1-5). USA. doi:10.23919/WAC.2018.8430387.

Zhao L., &. L. (2020). Object Detection Algorithm Based on Improved YOLOv3. *Electronics, 9(3), 537*. doi:10.3390/electronics9030537