Investigation of Combining Deep Learning Object Recognition with Drones for Forest Fire Detection and Monitoring

Mimoun YANDOUZI[1], Mounir GRARI[2], Mohammed BERRAHAL[3], Idriss IDRISSI[4], Omar MOUSSAOUI[5], Mostafa AZIZI[6], Kamal GHOUMID[7], Aissa KERKOUR ELMIAD[8]
Lab. LSI, ENSAO, Mohammed First University, Oujda, Morocco[1, 7]
Lab. MATSI, ESTO, Mohammed First University, Oujda, Morocco[2, 3, 4, 5, 6]
Lab. LARI, FSO, Mohammed First University, Oujda, Morocco[8]

***Abstract*—Forest fires are a global environmental problem that can cause significant damage to natural resources and human lives. The increasing frequency and severity of forest fires have resulted in substantial losses of natural resources. To mitigate this, an effective fire detection and monitoring system is crucial. This work aims to explore and review the current advancement in the field of forest fire detection and monitoring using both drones or unmanned aerial vehicles (UAVs), and deep learning techniques. The utilization of drones fully equipped with specific sensors and cameras provides a cost-effective and efficient solution for real-time monitoring and early fire detection. In this paper, we conduct a comprehensive analysis of the latest developments in deep learning object detection, such as YOLO (You Only Look Once), R-CNN (Region-based Convolutional Neural Network), and their variants, with a focus on their potential application in the field of forest fire monitoring. The performed experiments show promising results in multiple metrics, making it a valuable tool for fire detection and monitoring.**

***Keywords*—Forest fire; deep learning; drones; unmanned aerial vehicles; object detection; YOLO; Faster R-CNN**

I. Introduction

Forests are critical for our planet. They regulate our climate, purify our air and water, and are home to countless plants and animals. Sadly, forests around the world are under real threats from climate change, deforestation, and other human activities. One of the most devastating impacts on forests are wildfires. More recently, forest fires have become an annual phenomenon across the world. Statistics show that millions of acres of forests are yearly burnt. This has caused tremendous loss of forest resources, major economic damages to forest organizations, and lives of humans and animals. The rise in wildfires is largely attributed to climate change. Warmer temperatures and drier conditions make it easier for fires to start and spread. When a wildfire starts, it can spread quickly through the entire forest. Wildfires can have a significant impact on the environment and the economy. They can permanently damage forests, which can take years to be restored. In addition, wildfires can result in substantial financial damage as companies and industries relying on forests are compelled to shut down or move [1].

A wide range of techniques are used to detect and monitor forest fires, mainly we retain two main techniques based on sensors and imagery. The first one is based on deployed sensors that can detect environmental measurements such as temperature, humidity, gas levels, etc. These sensors, which may be strategically positioned throughout the forest, will notify authorities if they detect any fire. The second technique is imagery-based; it uses images coming from fixed cameras, satellites, or drones. It provides authorities with a bird-eye view of the fire and its precise location [2, 3].

Recently, with the great advances in deep learning (DL) and its applications, new opportunities are available to the problem-solving of computer vision in the field of forest fires monitoring. We remind that the traditional approach uses only visual analysis of images taken by satellites or aerial cameras. The existing system only detects wildfires within the camera view; it cannot identify the exact location of the fire [4]. This gap led us to propose, in this paper, an automated system that helps identify potential forest fires using object detection algorithms, such as YOLO (You Only Look Once) and R-CNN (Region-based Convolutional Neural Net-work).

The rest of this paper is organized as follows: The Section II presents the background of our research; the Section III gives a detailed overview of related works; the Section IV describes our proposed method; and before concluding this work, the Section V presents the obtained results and the discussion of the efficiency of our proposed system.

II. Background

A. Computer Vision (CV)

The biological vision is an inspiring model for computer vision. The mammalian visual system can decipher a complex scene in an instant, sophisticated enough to distinguish, ripe fruit from a poisonous berry, or fire from the sun. Similarly, computer vision is an emerging field that is rapidly evolving, making significant progress in recent years.

In general, computer vision is concerned with the automatic extraction, analysis, and understanding of information from images. This can be a difficult task, as images are often

cluttered and contain complex noises. However, recent advances in ma-chine learning have allowed for significant progress in this area. For example, deep learning (a subset of machine learning) is a powerful technique that has been used to achieve amazing results in computer vision [5].

There are many different applications of computer vision, including object recognition, face detection, and scene understanding [6].

B. Deep Learning (DL)

Deep learning, inspired by the brain, allows a computer to deeply learn from data by creating relevant large neuronal network models [7].

Deep learning is a relatively new field of machine learning, and it is already having a major impact. It is used in a variety of applications, including image recognition, speech recognition, and natural language processing [8].

Deep learning is a powerful tool that can be used to solve many complex problems. However, requires an important processing power and a large amount of data in order to train fast its models and learn effectively [9].

C. Object Detection

Object detection is the task of detecting instances of objects in an image, regardless of their position or orientation. This can be a very difficult task, as there can be a great deal of variation in the appearance of objects. For example, two different people can have very different opinions on what constitutes a "tree". Despite this challenge, object detection is a very important task in many applications, such as security, car self-driving, and robotics. In addition to detecting objects in an image, CV can also be used in following their movement. This is very helpful in keeping track of people or vehicles in a security system, or in avoiding obstacles for autonomous vehicles. There are many different types of techniques for object detection, but one of the most popular ones are those DL-based. DL is able to learn from both unstructured and unlabeled data. This makes it ideal for object detection, as it can learn to identify objects from a variety of different sources [10]. Furthermore, one of the advantages of deep learning for object detection is that it can learn to identify objects that are not easily detectable by humans. For example, deep learning can be used to detect objects in images that are blurry or have low contrast. Additionally, it can be used to detect objects that are occluded or partially hidden. Another advantage of deep learning for object detection is that it can learn to identify objects from a variety of different views. This is helpful because it means that the object detector will be more robust and will be able to identify objects even when they are not perfectly visible.

There are a few different object detection models that are popular among developers and researchers [11], including YOLO, and Faster R-CNN models. Each of these models has its own strengths and weaknesses. Object detection algorithms can be broadly classified into two categories: one-stage and two-stage. One-stage algorithms detect objects in a single step, whereas two-stage algorithms divide the process into two phases. The first phase uses a classifier to determine potential object positions, and the second phase employs a region proposal method to pinpoint the objects' most probable locations, as depicted in Fig. 1. One-stage algorithms are faster but less accurate, while two-stage algorithms are slower, but more accurate. One-stage algorithms are favored in real-time scenarios where processing speed is more prioritized than accuracy, whereas two-stage ones are utilized when accuracy is of utmost importance.

The components of object detection architecture typically comprise five parts: the input, backbone, neck, dense layer, and sparse layer. The input is the image fed into the network, which is usually pre-processed for standardization such as resizing or normalization. The backbone, which is typically a pre-trained convolutional neural network, extract features from the input image. The neck combines the extracted features from the backbone. The dense layer is a fully connected layer that generates the final object detection results using the combined features from the neck. The Sparse layer performs similarly to the dense layer, with one key difference: its connections are sparse, meaning that not every neuron in the current layer is linked to every neuron in the prior layer. This architectural design is often used to optimize neural network performance by reducing the number of parameters and improving computational efficiency. The Sparse layer is commonly employed in feature extraction and object detection tasks, and it also outputs the final object detection results.

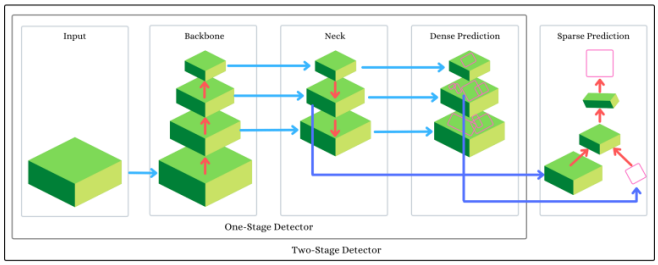

Fig. 1. Object detection architectures: one-stage and two-stage approaches.

1) R-CNN (Region-based CNNs) and its variants: R-CNN (Region-based Convolutional Neural Network) and its variants are a family of deep neural network architectures for image classification and object detection [12, 13]. R-CNN was originally proposed for object detection in natural images. The main idea of R-CNN is to use a CNN to process a region of an image, extract features from it, and then classify it. The R-CNN architecture has been successful in many object detection tasks, including detecting objects in both natural images and video.

Variants of R-CNN include Fast R-CNN [14] and Faster R-CNN [15]. Fast R-CNN is an improvement over R-CNN that uses a Region Proposal Network (RPN) to propose regions, rather than using a sliding window. Faster R-CNN is an even further improvement that shares convolutional layers between the RPN and the classifier, resulting in even faster performance [16].

2) YOLO (You Only Look Once): The YOLO object detection system is a widely used method for detecting objects in images and videos, originally created by Joseph Redmon and Ali Farhadi [17, 18]. YOLO is a real-time object detection system that is fast and accurate. YOLO has been used in a

variety of applications and is considered to be the best of object detection systems of the literature.

YOLO is a deep learning model that is able to effectively identify objects in images and video frames. This is done by first partitioning an image into a set of grid cells, and then using a specially designed neural network to predict the bounding box coordinates and class probabilities for each cell.

YOLO is constantly being improved and its new versions are being released. The first version, YOLOv1, was released in 2016. YOLOv2 was released in 2016. YOLOv3 was released in 2018. YOLOv4 was released in 2020. YOLOv5 was released in 2021. And the latest YOLOv6 [19], YOLOv7 [20], and YOLOv8 [21] (v6 and v7 released in 2022, and v8 released in 2023), included further improvements in terms of accuracy, speed, and they introduced a new segmentation pipeline.

III. Related Works

Zheng et al. [22] worked on the classification of dynamic scenes to facilitate the process of detection and tracking of objects and thus improve the performance of visual surveillance. The proposed model, Bi-heterogeneous convolutional neural network (Bi-CNN), extracts spatial and temporal information from the video sequences to categorize them. The model was trained and tested on a dataset composed of drone videos. They achieved a mean accuracy of 93%.

Jiao et al. [23] proposed a model based on YOLOv3 that can be deployed in architecture using UAVs. The developed platform is presented with all the technical choices of the UAV and the analysis station. The DL model is therefore deployed on a ground station. They were able to achieve a speed of photo transmission of 3.2 images per second with a fire recognition rate of 83%. The same authors, in a second work [24], upgraded the equipment used to reach a transmission speed of 30 frames per second and an accuracy of 91%.

Lohit et al. [25] used object detection to solve a post-fire problem related to reforestation. The authors use a drone equipped with a Raspberry Pi board on which deep learning models are deployed. A comparative study is performed between the models DenseNet121, Resnet152, and MobileNetv2. The dataset used is composed of UAV images, UAV Dataset (from Kaggle), and Open-source photos. The best results were obtained when using the model DenseNet121 with an accuracy of 93.1%.

Wang et al [26] initially chose the YOLOv4 object detection architecture as the neural network's backbone. Due to the large number of parameters, heavy computational load, and significant memory requirements, this model is not suitable for implementation on embedded development kits with limited computational power. As a result, they replaced the YOLOv4 model's backbone with a MobileNetV3 model to create an initial lightweight YOLO + MobileNet model and reduce the number of parameters in the model as well as the computational load. The model was then further compressed by removing redundant parts of the proposed network structure. Finally, using knowledge distillation, they improved the detection accuracy of the compressed model and obtained the final model.

Yanık et al. [27] presented a new drone-based architecture for smoke and fire recognition tasks in low-cost forests equipped with image processing and object detection capabilities. To do so, they used a drone equipped with a Raspberry on which a lightweight deep learning model based on MobileNet is deployed. The study focuses on the issue of battery consumption in order to increase the number of flight hours of the UAV. The proposed model "ssdlite mobilenet" is tested on four variants of parameters related to the number of images in the training and testing phases on the COCO dataset.

However, most of the time, object detection systems tend to be inaccurate and inefficient for detecting potential forest fires when image quality is bad or when the fire area is relatively small. As such, an accurate and efficient object detection system is required to detect potential forest fires from images or videos immediately. In this paper, we propose an automated system that uses an object detection technique to detect potential forest fires and quickly alert the appropriate authorities. In this way, they can rapidly respond to such crises and reduce the impact of the fire on forests.

IV. Proposed Method

According to the above research works, computer vision-based methods provide improvements over traditional methods. This is where our research comes in, we propose an automated system to detect fire in forests by using drones. The drone is equipped with a high-resolution camera, which films the area to be inspected for fire hazards. Video from the Drone's camera is transferred to an object detection system. The object detection system uses different techniques to automatically detect potential forest fires from a video. The coordinates of the detected fires are then transferred to the Geographic Information System (GIS). The GIS then creates a map of the detected fires and sends the map to the fire department. The map shows the location of the fire and the drone's current location. The fire department then sends a fire patrol to this location. The drone continues to film the area and continues to send videos to our system. The object detection system keeps following the fire and sending the updated fire location to the GIS. The GIS then updates the map and sends it to the fire department. The Fire patrol is then able to follow the fire and put it out (see Fig. 2).

The methodology proposed consists of five key steps, outlined in Fig. 3.

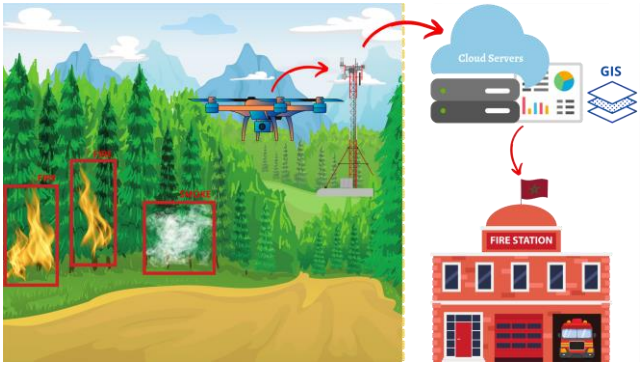

Fig. 2. Proposed architecture.

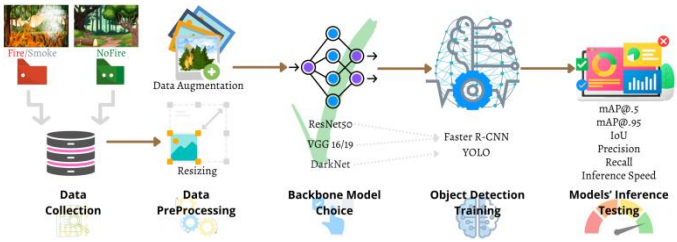

Fig. 3. Proposed method.

A. Dataset Collection

The first step in our proposed method is to collect a dataset of images containing forest fires. Our dataset (after data augmentation) reaches a total of 4236 images with the labels Fire and Smoke. These photos were taken with both ground-level cameras and aerial drones. The ground cameras were used to capture detailed images of forest fires in real-time, providing accurate representations of the fires (some images were shot by us on the campus, see Fig. 4). Aerial drone images, on the other hand, were used to provide a broader perspective by capturing larger areas of an entire forest, allowing us to monitor the extent and size of the fire's affected area. Additionally, photos were gathered from publicly accessible datasets such as online image libraries and websites. These images, which depict real-world scenarios of forest fires, were used to further expand the dataset and provide a more diverse dataset for an effective learning.

To facilitate the labeling process, we leveraged the open-source software Label-Studio, which required significant time and effort from our project team. To streamline the task, we divided the dataset among all members, enabling us to complete the process efficiently.

It is worth noting that we have taken the necessary precautions to ensure that the images are diverse in terms of the various types of fires, fire intensities, and environments in which they occurred. This was done to ensure that the model could detect fires accurately in a variety of conditions, thus improving its performance and generalization ability.

Fig. 4. Images of the man-made supervised fires.

B. Data Preprocessing

The next step in our proposed method is to preprocess and augment the collected dataset. Preprocessing includes cropping and resizing the images to the required size (640x640), as well as converting the images to a standard format such as JPG or PNG (JPG in our case). Data preprocessing plays a crucial role in object detection and can determine the success or failure of an object detection system. By resizing the images to a standard size, we can ensure that all the data is of the same size, making it easier to work with and compare.

Augmentation involves applying various distortions and transformations to the images, such as rotations, horizontal flips, vertical flips, and random crops, to increase the variety of the dataset and make it more robust. Data preprocessing is performed to increase the available training data and improve the ability of the object detection system to recognize objects from various perspectives. Resizing and augmenting the data help to increase the chances of success in object detection.

C. Backbone Model Choice

When selecting a backbone model for object detection, we have several options to consider, including VGG-16, VGG-19, and ResNet50, which have been found to be effective in our previous research [4]. It is important to carefully evaluate the strengths and weaknesses of each model in our object detection system to determine which one performs best on our specific dataset. To do this, we can benchmark each model and compare its performance for the Faster R-CNN model. DarkNet is specifically used as the backbone model for YOLOv6, v7, and v8.

ResNet (Residual Network) is a deep learning model introduced by Microsoft Research in 2015 [28]. It is known for its ability to train very deep neural networks with hundreds or even thousands of layers, using a technique called skip connections or shortcut connections. These connections allow the model to learn residual functions, or the difference between the input and the desired output, rather than trying to learn the entire mapping from scratch. This helps to alleviate the vanishing gradient problem and enables ResNet to achieve very good performance on a variety of tasks. In this work, we will utilize three backbone combinations with the faster R-CNN architecture, including:

- C4 feature extractor: a type of feature extractor used to extract relevant information from the feature maps produced by a convolutional neural network for object detection.
- Feature Pyramid Network (FPN): a type of neural network architecture used for object detection that combines high-resolution and semantically strong features to produce a multi-scale feature representation.
- 5 levels of down-sampling (DC5): a design choice in a feature extractor where the image is down-sampled five times to produce a lower resolution version while preserving important features, making object detection easier and faster to process.

VGG was developed by the Visual Geometry Group at the University of Oxford [29], it is a convolutional neural network architecture known for its simplicity and good performance on image classification tasks. It consists of a series of convolutional and max pooling layers, followed by a few fully-connected layers. VGG-16 and VGG-19 are two variations of the VGG model that differ in the number of layers and the number of parameters.

DarkNet is a neural network framework developed by Joseph Redmon [17]. It is the basis for the YOLO object detection algorithm, which is known for its speed and real-time performance. DarkNet consists of a series of convolutional and max pooling layers, followed by multiple fully-connected layers. It is designed to be simple and easy to extend, making it a popular choice for researchers and practitioners working on object detection and other computer vision tasks [30].

D. Object Detection Training

The fourth step in our proposed method involves fine-tuning and training our object detection models on the preprocessed and augmented dataset. The dataset is split into train (70%), validation (20%), and test (10%) sets. We use a variety of models including YOLOv6, v7, and v8, and faster R-CNN (RS50 and VGG16/19). The backbone model serves as the foundation for our models. We have selected faster R-CNN as it is a more efficient and precise object detection algorithm compared to R-CNN and Fast R-CNN, and it has the capability to process complex images and learn high-level features with fast inference speed [16]. YOLO, particularly its newer versions, is also well-known for its speed and accurate results.

We train our models using labeled data, which typically consists of bounding boxes around objects in the image and information about the class of objects contained within the box. The input data for faster R-CNN is in the form of TensorFlow record (TFRecord) files, while YOLO uses TXT annotations and YAML config files. The goal of this step is to create a highly accurate and reliable model for detecting forest fires.

E. Models Evaluation

Finally, we evaluate the models' performance on the collected dataset by using the test set (10%) to assess the mean average precision and inference speed of each model. The testing phase is crucial as it enables us to measure the models' performance on previously unseen data and helps us to determine the overall efficiency of the models in detecting different types of forest fires.

V. Results and Discussions

In this section, we describe the results and discussion of our proposed method for object detection using a drone-mounted camera. To assess the accuracy of the proposed system, we have evaluated the results of object detection against a dataset of real forest fires. Moreover, the system was tested on fake and real forest fires and smokes to study the robustness of the proposed system.

A. Hardware Characteristics

The experimental setup used in this work consisted of a drone (DJI Mavic Air) equipped with a high-resolution camera, a computer, and the proposed object detection system. To run the proposed object detection model, we used a high-performance computing machine with the following hardware specifications:

- Two Intel Gold 6148 (2.4GHz/20-core) processors.
- Two NVIDIA Tesla V100 graphics cards, each having 32GB of RAM.

B. Evaluation Metrics

There are a variety of different metrics that can be used to evaluate the performance of an object detection algorithm, including:

1) Average Precision (AP): It is a fairly straightforward metric that simply measures the average precision of the detector across all classes. This is a good metric to get a general idea of how well the detector is performing. However, it doesn't give any insight into how well the detector is performing in specific classes. AP is calculated by first computing the precision-recall curve for a given set of detections, then taking the average of the precision values at regularly spaced recall levels. Given a set of detections [31], the formula for average precision (AP) is:

$$AP = \sum_{k=0}^{k=n-1}[Recalls(k) - Recalls(k+1) * Precisions(k)] \quad (1)$$

Where Recalls(n)=0, Precisions(n)=1, and n=Number of thresholds.

2) Mean Average Precision (mAP): It is a more sophisticated metric that takes into account the precision of the detector in each class. This is a good metric to get a more detailed picture of how well the detector is performing. However, it can be more difficult to interpret than AP. The mAP metric is usually reported at several confidence thresholds (e.g., 0.5, 0.95). The formula for mean average precision (mAP) is:

$$mAP = \frac{1}{N}\sum_{i=1}^{N} APi \quad (2)$$

Where APi is the Average Precision of class i and N is the number of classes.

3) Intersection over Union (IoU): It is a metric that measures the amount of overlap between the detected object and the ground truth object. This is a good metric to make sure that the detected object is a good match for the ground truth object. However, it can be more difficult to be interpreted than AP or mAP. Given two rectangles, with coordinates (x1, y1, x2, y2) and (x3, y3, x4, y4), the formula for Intersection over Union (IoU) is:

$$IoU = \frac{(Area\ of\ Intersection)}{(Area\ of\ Union)} \quad (3)$$

$$Area\ of\ Intersection = (min(x2, x4) - max(x1, x3)) * (min(y2, y4) - max(y1, y3)) \quad (4)$$

$$Area\ of\ Union = (x2 - x1) * (y2 - y1) + (x4 - x3) * (y4 - y3) - Area\ of\ Intersection \quad (5)$$

In general, AP is the primary metric used to measure the performance of an object detection model. However, mAP is also commonly used as it provides a more thorough overview of the model's performance. IoU is used as a complementary metric to provide insights into how well the model is doing in terms of localization.

C. Evaluating the Results

In this study, several object detection models were evaluated for their performance in detecting forest fires and smoke. The models included Faster R-CNN (with different backbone networks) and YOLO models (v6, v7, and v8 with different architectures and computational requirements) trained on a dataset of forest fire and smoke images; we trained the Faster RCNN models over 500000 iterations and 1000 epochs for the YOLO models. The models were trained and evaluated using several metrics, including mAP at 0.5 and 0.95 IoU thresholds, recall, and precision. The inference time (s/image) was also measured on two Nvidia Graphics cards V100.

The YOLO models v6, v7 and v8 although they appeared successively in time, they are not necessarily progressively improved versions, and the meaning of (n) is nano, (s) is small, (l) is large model, and (x) is extra-large model (in the case of YOLOv7; this model does not provide a large version). The nano (n), small (s), and large (l) variations of the YOLO models have different numbers of layers and parameters, which can affect their accuracy and inference time.

Table I summarizes the achieved results for the implemented models on the testing set. The Faster R-CNN models with the ResNet-50 (RS50) and Feature Pyramid Network (FPN) backbones showed the best performance, with the Faster R-CNN (RS50) FPN achieving a mAP@0.5 of 90.57% and a mAP@0.95 of 80.34%. This model also had the lowest inference time among the Faster R-CNN models, with an average of 0.0281 seconds per image. On the other hand, the YOLO models showed slightly lower performance compared to the Faster R-CNN models, with YOLOv8n achieving a mAP@0.5 of 89.45% and a mAP@0.95 of 79.28%. However, the YOLO models had a much lower inference time, with YOLOv8n having an average of 0.0011 seconds per image.

Fig. 5 to 10 shows the performance of Faster R-CNN (RS50) FPN and YOLOv8n over iterations on the validation set. These figures show that both models achieved a relatively stable performance over iterations, with Faster R-CNN (RS50) FPN achieving a higher mAP@0.5 and mAP@0.95, and YOLOv8n having a lower loss.

However, the YOLO models performed well in terms of inference time, with YOLOv6n, YOLOv8n, YOLOv8s, and YOLOv8l having an inference time of fewer than 0.0011 seconds per image. This makes YOLO a good choice for real-time applications such as drone data, where fast processing speed is essential. The Fast-RCNN (RS50) C4, Fast-RCNN (RS50) DC5, and Fast-RCNN (VGG19) models also showed good results, however, the processing speed was higher compared to YOLO models.

In sum, the choice between YOLO and Faster RCNN models for the task of forest fire detection would depend on the desired trade-off between accuracy and processing speed. For applications that prioritize high accuracy, the Faster R-CNN models, particularly the Faster R-CNN (RS50) FPN model would be the best choice. On the other hand, for real-time applications that require fast processing speeds, the YOLO models, particularly YOLOv8n, would be the best option (Fig. 11).

In conclusion, the choice between YOLO and Faster RCNN models for the task of forest fire detection would depend on the desired trade-off between accuracy and processing speed. For applications that prioritize high accuracy, the Faster RCNN models, particularly the Faster RCNN (RS50) FPN model, would be the best choice. On the other hand, for real-time applications that require fast processing speeds, the YOLO models, particularly YOLOv8n, would be the best option (Fig. 10).

TABLE I. ACHIEVED RESULTS FOR THE IMPLEMENTED MODELS (ON THE TESTING SET)

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

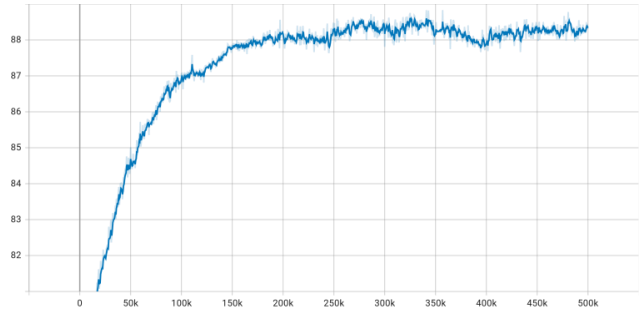

Fig. 5. Achieved mAP@0.5 over iterations for faster R-CNN (RS50) FPN (on the validation set).

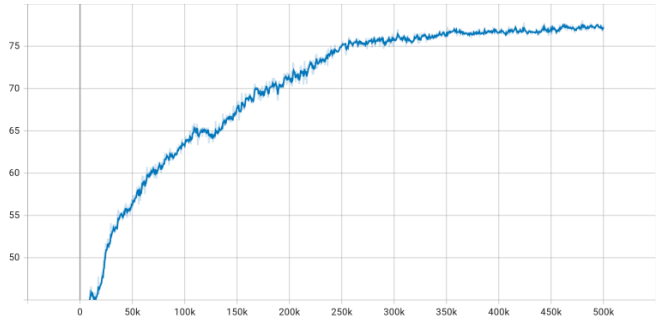

Fig. 6. Achieved mAP@0.95 over iterations for faster R-CNN (RS50) FPN (on the validation set).

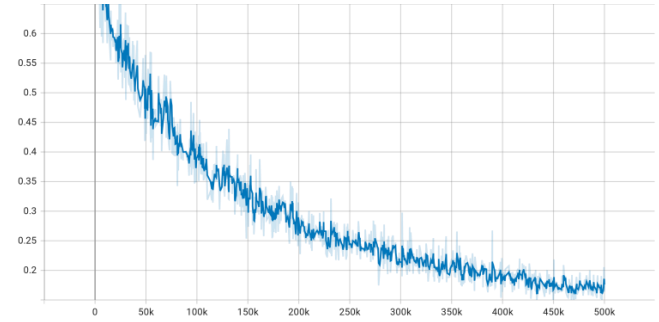

Fig. 7. Achieved loss over iterations for faster R-CNN (RS50) FPN (on the validation set).

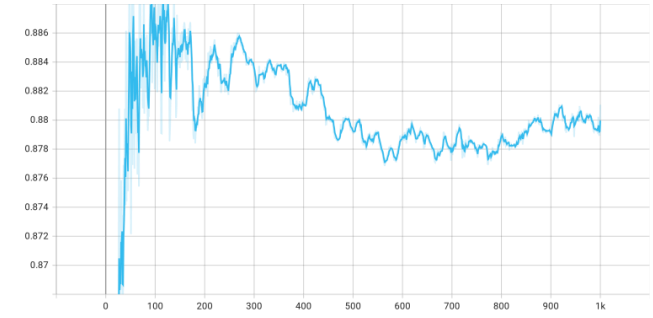

Fig. 8. Achieved mAP@0.5 over iterations for YOLOv8n (on the validation set).

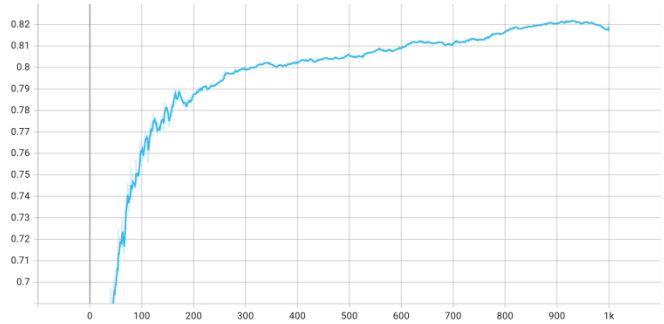

Fig. 9. Achieved mAP@0.95 over iterations for YOLOv8n (on the validation set).

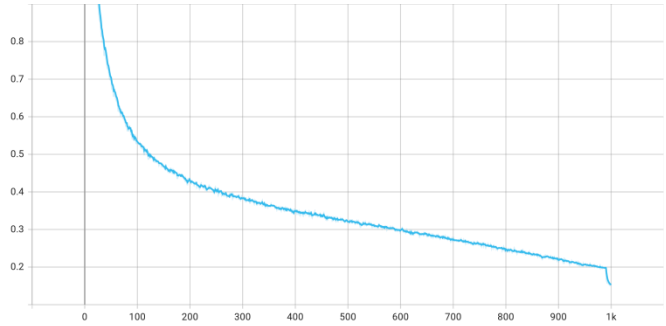

Fig. 10. Achieved loss over iterations for YOLOv8n (on the validation set).

Fig. 11. Forest fire and smoke detection by drone - examples using YOLOv8n.

VI. Conclusion

Forests are of utmost importance to maintain the balance of the ecosystem and provide various ecological, social and economic benefits. However, the increasing frequency and severity of forest fires pose a significant threat to the sustainability of forests and their functions, making early detection and prompt actions critical for limiting the damages. The use of drones fitted with sensors and cameras presents a cost-effective and efficient solution for detecting fires in real-time. The proposed method consists of four major steps, including video recording, object detection, GIS mapping, and fire department notification, to provide an efficient and cost-effective solution for real-time monitoring and early fire detection. This study conducts an extensive evaluation of the recent advancements in deep learning object detection techniques, including YOLO, Faster R-CNN, and their variants, with a specific emphasis on their suitability for forest fire monitoring. Based on the experimental findings, these techniques exhibit positive outcomes in several metrics, thereby presenting a promising tool for detecting and monitoring fires. To select the appropriate model for detecting forest fires and smoke based on drone images, it is important to find a balance between accuracy and processing speed. For higher accuracy, the Faster RCNN model is recommended, whereas for real-time applications that prioritize speed, the YOLO model, particularly the YOLOv8n version, is the better choice with a mAP@0.5 of 89.45%, a mAP@0.95 of 79.28% and an inference time of almost 0.0011 seconds per image.

As part of our upcoming tasks, we are currently exploring the utilization of thermal images captured by UAVs. Additionally, we are examining the individual contributions of each RGB layer during model training to effectively decrease the overall number of parameters.

ACKNOWLEDGMENT

This work is supported by the Mohammed First University under the PARA1 Program (Low-cost, real-time Forest Fire Detection System based on Wireless Sensor Networks - SDF-RCSF). And the computational resources of HPC-MARWAN are provided by the National Center for Scientific and Technical Research (CNRST), Rabat, Morocco.

REFERENCES

[1] M. Grari, I. Idrissi, M. Boukabous, O. Moussaoui, M. Azizi, and M. Moussaoui, "Early wildfire detection using machine learning model deployed in the fog/edge layers of IoT," Indones. J. Electr. Eng. Comput. Sci., vol. 27, no. 2, pp. 1062–1073, Aug. 2022, doi: 10.11591/IJEECS.V27.I2.PP1062-1073.

[2] M. Grari et al., "Using IoT and ML for Forest Fire Detection, Monitoring, and Prediction: a Literature Review," J. Theor. Appl. Inf. Technol., vol. 100, pp. 5445–5461, 2022.

[3] M. Yandouzi et al., "Review on forest fires detection and prediction using deep learning and drones," J. Theor. Appl. Inf. Technol., vol. 100, no. 12, pp. 4565–4576, 2022.

[4] M. Yandouzi et al., "Forest Fires Detection using Deep Transfer Learning," Int. J. Adv. Comput. Sci. Appl., vol. 13, no. 8, pp. 268–275, Oct. 2022, doi: 10.14569/IJACSA.2022.0130832.

[5] M. Berrahal and M. Azizi, "Augmented Binary Multi-Labeled CNN for Practical Facial Attribute Classification," Indones. J. Electr. Eng. Comput. Sci., vol. 23, no. 2, pp. 973–979, Aug. 2021.

[6] A. Kherraki and R. El Ouazzani, "Deep convolutional neural networks architecture for an efficient emergency vehicle classification in real-time traffic monitoring," IAES Int. J. Artif. Intell., vol. 11, no. 1, pp. 110–120, Mar. 2022.

[7] I. Idrissi, M. Azizi, and O. Moussaoui, "A Stratified IoT Deep Learning based Intrusion Detection System," in 2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET), Mar. 2022, pp. 1–8, doi: 10.1109/IRASET52964.2022.9738045.

[8] M. Boukabous and M. Azizi, "Review of Learning-Based Techniques of Sentiment Analysis for Security Purposes," in Innovations in Smart Cities Applications Volume 4, Springer, Cham, 2021, pp. 96–109.

[9] Y. Hammoudi, I. Idrissi, M. Boukabous, Y. Zerguit, and H. Bouali, "Review on maintenance of photovoltaic systems based on deep learning and internet of things," Indones. J. Electr. Eng. Comput. Sci., vol. 26, no. 2, May 2022.

[10] M. Boukabous and M. Azizi, "Crime prediction using a hybrid sentiment analysis approach based on the bidirectional encoder representations from transformers," Indones. J. Electr. Eng. Comput. Sci., vol. 25, no. 2, pp. 1131–1139, Feb. 2022, doi: 10.11591/IJEECS.V25.I2.PP1131-1139.

[11] M. Berrahal and M. Azizi, "Review of DL-Based Generation Techniques of Augmented Images using Portraits Specification," in 4th International Conference on Intelligent Computing in Data Sciences, ICDS 2020, Nov. 2020, pp. 1–8, doi: 10.1109/ICDS50568.2020.9268710.

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Region-Based Convolutional Networks for Accurate Object Detection and Segmentation," IEEE Trans. Pattern Anal. Mach. Intell., vol. 38, no. 1, pp. 142–158, Jan. 2016, doi: 10.1109/TPAMI.2015.2437384.

[13] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 580–587, Nov. 2013, doi: 10.48550/arxiv.1311.2524.

[14] R. Girshick, "Fast R-CNN," in Proceedings of the IEEE International Conference on Computer Vision, 2015, vol. 2015 Inter, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.

[15] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," IEEE Trans. Pattern Anal. Mach. Intell., vol. 39, no. 6, pp. 1137–1149, Jun. 2015, doi: 10.48550/arxiv.1506.01497.

[16] M. Boukabous and M. Azizi, "Image and video-based crime prediction using object detection and deep learning," Bull. Electr. Eng. Informatics, vol. 12, no. 3, pp. 1630–1638, Jun. 2023, doi: 10.11591/EEI.V12I3.5157.

[17] J. Redmon and A. Farhadi, " YOLO: Real-Time Object Detection," 2018. https://pjreddie.com/darknet/yolo/ (accessed Jan. 30, 2023).

[18] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," Apr. 2018, Accessed: Mar. 22, 2023. [Online]. Available: https://arxiv.org/abs/1804.02767v1.

[19] C. Li et al., "YOLOv6: A Single-Stage Object Detection Framework for Industrial Applications," Sep. 2022, doi: 10.48550/ARXIV.2209.02976.

[20] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors," arXiv Prepr. arXiv2207.02696, 2022.

[21] G. Jocher, A. Chaurasia, and J. Qiu, "YOLOv8," 2023. https://github.com/ultralytics/ultralytics (accessed Jan. 30, 2023).

[22] J. Zheng, C. Xianbin, Z. Baochang, Y. Huang, and Y. Hu, "Bi-heterogeneous Convolutional Neural Network for UAV-based dynamic scene classification," ICNS 2017 - ICNS CNS/ATM Challenges UAS Integr., Aug. 2017, doi: 10.1109/ICNSURV.2017.8011932.

[23] Z. Jiao et al., "A Deep learning based forest fire detection approach using uav and yolov3," 1st Int. Conf. Ind. Artif. Intell. IAI 2019, Jul. 2019, doi: 10.1109/ICIAI.2019.8850815.

[24] Z. Jiao et al., "A YOLOv3-based Learning Strategy for Real-time UAV-based Forest Fire Detection," Proc. 32nd Chinese Control Decis. Conf. CCDC 2020, pp. 4963–4967, Aug. 2020, doi: 10.1109/CCDC49329.2020.9163816.

[25] G. V. S. Lohit and D. Bisht, "Reforestation Using Drones and Deep Learning Techniques," 2021 7th Int. Conf. Adv. Comput. Commun. Syst. ICACCS 2021, pp. 847–852, Mar. 2021, doi: 10.1109/ICACCS51430.2021.9442053.

[26] S. Wang, J. Zhao, N. Ta, X. Zhao, M. Xiao, and H. Wei, "A real-time deep learning forest fire monitoring algorithm based on an improved Pruned + KD model," J. Real-Time Image Process. 2021 186, vol. 18, no. 6, pp. 2319–2329, May 2021, doi: 10.1007/S11554-021-01124-9.

[27] A. Yanık, M. Yanık, M. S. Güzel, and G. E. Bostancı, "Machine Learning–Based Early Fire Detection System Using a Low-Cost Drone," Adv. Sens. Image Process. IoT, pp. 1–18, Feb. 2022, doi: 10.1201/9781003221333-1.

[28] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 2016-Decem, pp. 770–778, Dec. 2015, doi: 10.1109/CVPR.2016.90.

[29] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., 2015.

[30] Redmon J Darknet: Open Source Neural Networks in C. https://pjreddie.com/darknet/. (accessed Jan. 23, 2023).

[31] Average Precision - Hasty.ai. https://hasty.ai/docs/mp-wiki/metrics/average-precision. (accessed Jan. 30, 2023).