Forest Fires Detection using Deep Transfer Learning

Mimoun YANDOUZI[1]
Lab. LSI, ENSAO
Mohammed First University
Oujda, Morocco

Mounir GRARI[2]
Lab. MATSI, ESTO
Mohammed First University
Oujda, Morocco

Idriss IDRISSI[3]
Lab. MATSI, ESTO
Mohammed First University
Oujda, Morocco

Mohammed BOUKABOUS[4]
Lab. MATSI, ESTO
Mohammed First University
Oujda, Morocco

Omar MOUSSAOUI[5]
Lab. MATSI, ESTO
Mohammed First University
Oujda, Morocco

Mostafa AZIZI[6]
Lab. MATSI, ESTO
Mohammed First University
Oujda, Morocco

Kamal GHOUMID[7]
Lab. LSI, ENSAO
Mohammed First University
Oujda, Morocco

Aissa KERKOUR ELMIAD[8]
Lab. LARI, FSO
Mohammed First University
Oujda, Morocco

***Abstract*—Forests are vital ecosystems composed of various plant and animal species that have evolved over years to coexist. Such ecosystems are often threatened by wildfires that can start either naturally, as a result of lightning strikes, or unintentionally caused by humans. In general, human-caused fires are more severe and expensive to fight because they are frequently located in inaccessible areas. Wildfires can spread quickly and become extremely dangerous, causing damage to homes and facilities, as well as killing people and animals. Early discovery of wildfires is vital to protect lives, property, and resources. Reinforced imaging technologies can play a key role to detect wildfires earlier. By applying deep learning (DL) over a dataset of images (collected using drones, planes, and satellites), we target to automate the forest fire detection. In this paper, we focus on building a DL model specifically to detect wildfires using transfer learning techniques from the best pretrained DL computer vision architectures available nowadays, such as VGG16, VGG19, Inceptionv3, ResNet50, ResNet50V2, InceptionResNetV2, Xception, Dense-Net, MobileNet, MobileNetV2, and NASNetMobile. Our proposed approach attained a detection rate of more than 99.9% over multiple metrics, proving that it could be used in real-world forest fire detection applications.**

Keywords*—Forest fires; wildfires; deep learning; transfer learning; computer vision; convolutional neural networks (CNN)

I. Introduction

Forests are one of the most important natural resources on our planet. They support a diverse range of plants and animals' lives, play an important role in climate regulation, and provide numerous economic and social benefits. However, forests are vulnerable to damage and destruction, and wildfires are one of their most serious threats [1]. A wildfire can start accidentally (with a spark from a campfire), or it can be deliberately set by someone who intends to cause harm. Wildfires can quickly spread, destroying everything in their path. They can also cause significant environmental damage, such as the death of trees and other plants, the removal of soil, and the release of harmful emissions into the atmosphere [2].

In recent years, wildfires have become increasingly severe. Wildfires raged through Algeria, Tunisia, and Morocco in mid-July 2021, as well as Italy and Greece in the Mediterranean. The fires, which are believed to have been started by arsonists, burned through thousands of acres of land, killing dozens of people and injuring many more [3], [4]. The fires were especially devastating to the local economies, causing widespread agricultural damage as well as the destruction of businesses and homes. Furthermore, the tourism industry suffered as many people canceled their trips to the affected areas. Despite the efforts of firefighters and volunteers, the fires kept burning for weeks, leaving a trail of destruction in their wake.

We can do a lot to reduce the risk of wildfires, such as properly managing forests and using fire-resistant materials when building houses and other structures. In addition, we need to address the root cause of these fires.

The detection of wildfires at a preliminary phase is critical for protecting people, property, and resources. Imaging technology has the potential to aid in the early detection of wildfires. High-altitude drones, aircraft, and satellites can detect wildfire heat signatures by top shooting the fire area [5].

In this paper, we aim to build the most accurate DL model for forest fires using transfer learning out of the most achieving and well-known computer vision architectures pre-trained models available today, such as VGG, Inceptionv3, ResNet50, InceptionResNetV2, Xception, Dense-Net, MobileNet, and NASNetMobile.

The rest of the paper is organized as follows. The second section provides a focus on the used techniques, while the third section deals with the related works. Then the fourth section presents our proposed method. Before concluding, the fifth section examines and discusses our study's findings.

II. Background

A. Computer Vision (CV)

CV is the process of extracting useful information from digital images. This data could be used for tasks such as object recognition, scene description, and motion tracking [6]. There are two kinds of computer vision algorithms: low-level and high-level. Low-level algorithms work on individual pixels; whereas, high-level ones work on more abstract features such as edges and corners. Low-level algorithms are typically faster and more accurate, but they are also more complex and require more processing power. High-level algorithms are less accurate, but they are also faster and easier to implement. On the other hand, deep learning has shown great promise in this area and has been used to achieve impressive results in tasks such as object recognition and scene understanding [7]. Deep learning-based computer vision can recognize objects in images, recognize faces, and read text. It can also be used for automatic image tagging and classification [8].

B. Convolutional Neural Networks (CNN)

CNNs are a type of deep learning (DL) network, which means they are made up of multiple layers of neurons organized in a hierarchical structure. A deep learning network's goal is to learn representations of input data by gradually extracting more and more information from it. Because of their ability to learn features of the input data, CNNs are particularly well-suited for computer vision tasks [9]. This is accomplished through a process known as feature extraction, which involves identifying the important features in the data and representing them in a way that the network can learn from. In contrast, traditional machine learning algorithms require the programmer to explicitly specify which features the algorithm should use [10].

CNNs have the ability to learn features that are specific to the task at hand, which is one of their advantages [11], [12]. It is made up of several layers, each with its own function. The first layer is the input layer, which receives input data in the form of a numerical value matrix and feeds it into a series of convolutional and pooling layers. This combination of convolutional and pooling layers is known as a kernel. The output layer is the final layer in a CNN; it produces the network's results (see Fig. 1).

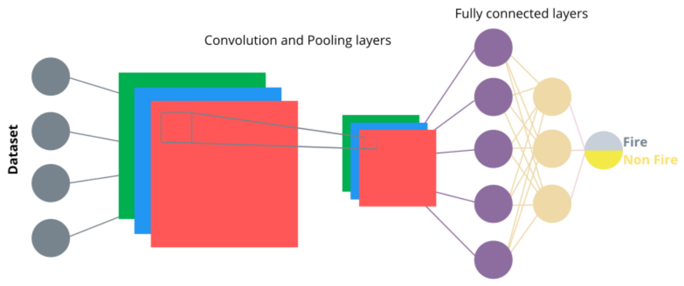

Fig. 1. CNN Layers Architecture.

We list below the most successful deep learning CNN architectures for image recognition:

- VGG is a deep learning architecture (created by the Visual Geometry Group) that was also created for the ILSVRC contest (ImageNet Large Scale Visual Recognition Challenge). The VGG16 is made up of 16 layers, including five convolutional layers, four dense layers, and a final fully connected layer, while the VGG19 model has 19 layers [13].
- Inception is a deep learning model that performed well in the ImageNet competition. It is made up of a deep convolutional network with a large number of layers (more than 20) [14].
- ResNet (Residual neural network) is a deep neural network that performed well in the ILSVRC. It is made up of a deep convolutional network with a large number of layers (more than 100), one of ResNet's major purposes is a so-called “identity shortcut connection” that skips one or more layers [15].
- InceptionResNetV2 is a variant of the original Inception-v2 model, designed to improve its performance on the ImageNet dataset. The model is composed of an Inception module followed by a ResNet module [16].
- Xception (Xtreme Inception) is a deep learning model, based on a CNN with a large number of layers (more than 150) [17].
- DenseNet is a CNN model with a large number of layers (more than 500), designed to increase the number of connections between neurons. This helps to improve the overall accuracy of the network.[18].
- MobileNet is a deep learning framework that enables developers to create sophisticated neural networks for mobile devices. It is designed to be efficient and lightweight, making it suitable for running on a wide range of mobile devices. MobileNet50 version has a depth of 50 layers and can be used for both classification and detection tasks [19].
- NASNet (Neural Architecture Search Network) is a CNN model trained on the ImageNet dataset [20]. It automates network architecture engineering by searching for the best algorithm to achieve the best performance on a certain task, while automatically configuring the number of layers, the number and type of neurons in each layer, and the architecture of the network. The NASNetMobile version is suitable for mobile devices [21].

III. Related Work

Dutta S. et al [22] proposed a hybrid architecture of separable convolution neural networks and digital image processing employing thresholding and segmentation for reliably detecting small-scale forest burning, which generally heralds the beginning of more terrible catastrophes. Performance examination of the test data on the suggested design provided outstanding results in terms of high sensitivity (98.10 %) and specificity (87.09 %).

Aslan S. et al [23] proposed a smoke detection approach based on Deep Convolutional Generative Adversarial Neural Networks (DC-GANs). In order to ensure a robust representation of sequences with and without smoke, the training framework includes regular training of a DCGAN with real pictures and noise vectors, as well as training the discriminator separately using smoke images without the generator. With a TNR of 99.45% and a TPR of 86.23%, the suggested approach is able to identify smoke pictures in real time with minimal false positives.

Wang Y. et al [24] proposed a forest fire image identification system based on traditional image processing methods and convolutional neural networks, and an adaptive pooling methodology was established to identify fire automatically. Using this technique, the features of the fire flame may be segmented and learned in advance. It has been shown in experiments that the adaptive pooling convolutional neural network approach has greater performance and a higher recognition rate, with an accuracy as high as 90.7%.

Chen Y. et al [25] proposed a UAV-based forest fire detection approach based on a convolutional neural network method in order to identify a probable fire in its early stages. Experimentation with generated flames in an indoor testbed proves that the suggested fire detection system works.

We will make a comparative study between a wide range of deep learning models, practically all of those that have demonstrated their effectiveness in computer vision, in order to propose the most accurate model possible for forest fire detection

IV. Proposed Method

Our proposed solution is to detect wildfires before they spread out of control using drones and deep learning algorithms. Drones are used to fly over forests and identify hot spots that could spark a wildfire. Once a hot spot has been identified, the deep learning algorithm can be used to determine whether it is a wildfire, and notify the authorities via a cloud server (Fig. 2).

This scheme has several advantages over traditional methods. First, drones can fly over large areas much faster than ground crews. Second, the deep learning algorithm can identify wildfires much more accurately than human observers can. Third, the use of drones and deep learning algorithms can help to protect firefighters and their assistants and keep them safe from danger by alerting them earlier.

Our main contribution in this paper is to build the most accurate DL model specifically for detecting wildfires in forests from the best resulting DL computer vision architectures available at the time, by leveraging previously known knowledge from pre-trained models. These models are already trained to know certain categories, and we narrow their knowledge to focus only on two categories (Fire or Non-Fire).

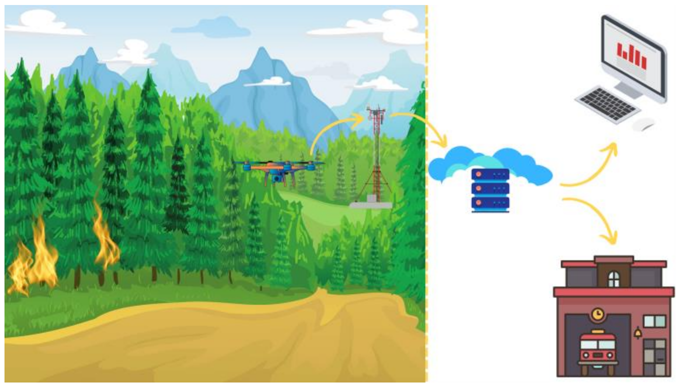

Fig. 2. Components and Functions of a UAV-Based Forest Fire Detection Platform.

A. Dataset

A common challenge in deep learning is obtaining datasets that are sufficiently large and diverse in nature for the task at hand [26], [27]. The dataset used for training our models is comprised of a large number of images captured of wildfires in different locations around the world, as well as images of forest landscapes with no fire. It was constructed by mixing and merging multiple smaller datasets from search engines and Kaggle [28], resulting in 4661 images in our new dataset; 2525 images with the label "no fire" and 2136 images with the label "fire", after cleaning some corrupted images.

In addition, we performed data augmentation on the dataset, allowing us to significantly increase the size of our training dataset and, as a result, the quality of the trained models [29]. With data augmentation, we added new data to the dataset that is similar to the original data, but with some slight modifications, which can improve the performance of neural networks learning from data and improve their accuracy [30]. We used different data augmentation techniques [31], such as:

1) *Random rotation:* this can help to reduce overfitting by creating new images that are rotated versions of existing images. This also gives the model a chance to learn how to recognize objects from different angles.

2) *Horizontal and vertical mirroring:* they can also help to reduce overfitting by providing the model with new images that are mirror images of existing images. This can also help the model learn to recognize objects that may be upside down or rotated in different orientations.

3) *Gaussian blur:* it can help to improve the robustness of the model by making the images less detailed and more forgiving of small changes. This can help the model to generalize better to new data.

4) *Pixel level augmentation:* it can help to improve the model's ability to learn from small changes in the input data. This can be useful for learning from data that may be noisy or have low resolution.

B. *Building the Models*

A model can be trained in a variety of ways. In this section, we will stare at transferring pre-trained models to a new task. Transfer learning models are typically constructed by first training them on a large dataset, such as the ImageNet dataset. This model is then used as the "base model" for another model trained on a smaller dataset (see Fig. 3). In our case, the smaller dataset is often a more specialized dataset (images of fires and forests). The smaller model is then tuned to better fit the dataset on which it is being used. This process is frequently repeated, with the final model trained on a dataset even smaller than the original. This process of model training is commonly referred to as fine-tuning [32]. We used this same strategy for each of the state-of-the-art models, importing the pre-trained DL model class [33], while ensuring that we can add our own custom input and output layers according to our data. While leveraging the previously learned weights during the initial training on the old data, a massive amount of time and space is saved while minimizing the model's complexity. (see Fig. 3).

Afterward, we inserted a fully connected and output layer (new classifier) after the pre-trained model was imported so that new real learning could take place; the fully connected layer is a flatten layer and a dense layer with 512 neurons [34]. The sigmoid activation function is used in the output layer with only one output neuron matching the binary label in our data (Fire or Not). Finally, we train our models on 80% of the augmented new dataset (Training set) and validate the obtained results with the remaining 20% (Validation set).

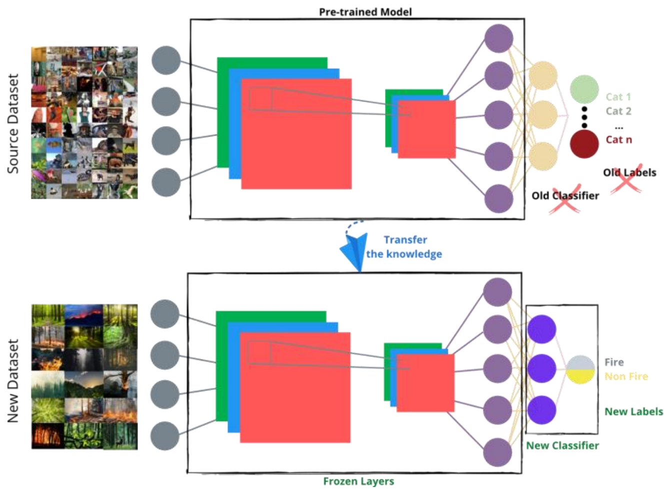

Fig. 3. Transfer Learning Technique.

V. Results and Discussions

A. *Hardware and Software Characteristics*

In order to get our results, we used TensorFlow on an HPC system with the following hardware specifications:

- 2x Intel Gold 6148 (2.4 GHz/20 cores) CPUs
- 2x NVIDIA Tesla V100 graphics cards, each having 32GB of RAM

TensorFlow v2.7.0 was used in our experiments, it is an open-source data analysis and machine learning software library. It was first developed by engineers and researchers at the Google Brain team in 2015. TensorFlow provides a wide range of capabilities for data analysis and machine learning, including numerical computing, linear algebra, graph processing, and deep learning [35].

B. *Evaluation Metrics*

It is necessary to have a proper evaluation metric in place in order to find the best model during the training phase [36]. When evaluating deep learning models, certain metrics must be used, such as Accuracy, Precision, Recall, and Loss. In order to calculate these metrics, four different parameters are used [37]:

- True Positive (TP): is the total of successfully categorized positive class records.
- True Negative (TN): is the total of successfully categorized negative class records.
- False Positive (FP): is the total of incorrectly categorized positive class records.
- False Negative (FN): is the total of incorrectly categorized negative class records.

1) Accuracy: Accuracy is the percentage of correctly classified items. It is the most basic and common evaluation metric for classification tasks. The accuracy is simply the ratio of correctly predicted labels out of all predicted labels [34]:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (1)$$

2) Loss: It is a measure of how far off the algorithm is from the desired output. The lower the loss, the better the algorithm is performing [38]. The cross-entropy loss is a commonly used loss metric for classification problems. It is calculated by this formula:

$$Loss = -\sum_i y_i * \log(p_i) \quad (2)$$

Where y_i is the output of the true label and p_i is its predicted probability. The cross-entropy loss is used to assess the performance of a classifier by penalizing incorrect predictions.

The higher the cross-entropy loss, the more incorrect predictions the classifier is making.

3) Precision: Precision is a metric estimating how well a model predicts true positives. True positives are those instances that are correctly identified as positive by the model [39].

A model with high precision will correctly identify the most positive examples, while a model with low precision will misclassify many positive examples as negative. The Precision metric is given by:

$$Precision = \frac{TP}{TP+FP} \quad (3)$$

4) Recall: The recall is the ratio of correctly predicted positive instances to all positive instances (see formula (4)). It is also known as the true positive rate or sensitivity. A model with high recall is capable of detecting the most positive instances [39]. A model with low recall is not informative as it classifies most positive instances as negative.

$$Recall = \frac{TP}{TP+FN} \quad (4)$$

In general, Recall should be used alongside other metrics such as Precision and Accuracy to get a complete picture of a model's performance.

5) The number of parameters: The number of parameters in deep learning refers to the number of variables that are used to define the structure of the neural network. These variables can be the weights and biases of the network, the size of the network, or the type of activation function used. It reflects the learning capacity of the model; A deep learning model with a large number of parameters has the capacity to learn more complex patterns than a model with fewer parameters [40]. The number of parameters also determines the amount of memory required to store the model; A model with a large number of parameters requires more memory than a model with fewer parameters [41]. The number of parameters in the models will vary from the original pre-trained models owing to the change in the fully connected layers (the convolution base was unmodified as it was frozen).

C. Evaluating the Results

In Table I and Figures 4-7, we present the obtained results on multiple metrics; the accuracy, loss, precision, and recall, along with the number of parameters for each model (this number differs from the original pre-trained models, due to our new classifier). To maximize effectiveness, we trained all of the models over one hundred epochs. These obtained results show that the ResNet50, VGG16, and VGG19 algorithms have higher accuracies, lower losses, and higher recalls and precisions, achieving a near-perfect score. Meanwhile, MobileNet and DenseNet came in second place with more than 97% in three metrics (accuracy, precision, and recall), but with a loss of around 5 to 6%; On the other hand, MobileNetV2 achieves close results, more than 96% in the three metrics and a loss of more than 9%. Then, Xception, which received more than 94% in the three metrics, and a high Loss averaging 14 to 15%. ResNet50V2 obtained mediocre results; even though its first version (ResNet50) got good results, around 84-85% in the three metrics, but with high losses up to 34% (higher errors are related to high loss, which means that the model does not do a good job). NASNetMobile and InceptionV3 performed similarly to ResNet50V2 in all metrics. On the other side, the mixed-model InceptionResNetV2 performed the worst in the accuracy, precision, and loss metrics, but reached the best score in the recall metric (100%). This shows that the model has a low false-negative rate (down to zero), but with a high false-positive rate due to the low precision results. At the end of this discussion, the best models retained are ResNet50, VGG16, and VGG19. Then, we compared their number of parameters. They have the respective numbers: ResNet50 (24.6~ million), VGG16 (~14.9 million), and VGG19 (~20.2 million). if we are looking for a model with the best learning capacity, ResNet50 is the accurate candidate; On the other hand, if we are targeting a lightweight model to deploy on a limited resource and battery-connected devices such as a drone or an IoT thing [42], VGG16 is the suitable one among the three. It has60% fewer parameters in comparison with the ResNet50. With fewer performances, DenseNet is the best lightweight model (after VGG16) with only 7.5 million parameters. Also, if we prioritize model size, MobileNet will be the best choice with only 3,7m parameters and an accuracy close to 98%, MobileNetV2 is the lightest model in this case study with only 2,9m with a modest accuracy of more than 96% just a little behind it first version MobileNet.

For the other models, ResNet50V2 has about the same number of parameters as ResNet50, while Xception and InceptionV3 have respectively ~24.6m and ~21.9m, but produced modest results. Despite its high number of parameters (55.1m), InceptionResNetV2 is the poorest model in our case study, indicating that deeper networks or more neurons do not always produce the best results.

TABLE I. ACHIEVED RESULTS FOR THE IMPLEMENTED MODELS

Deep Learning Algorithm	Number of parameters	Accuracy	Loss	Precision	Recall
● **VGG16**	14.977.857	99.81%	0.49%	99.77%	99.89%
● **VGG19**	20.287.553	99.78%	0.48%	99.83%	99.78%
● **InceptionV3**	22.852.385	83.23%	37.48%	83.47%	87.21%
● **ResNet50**	24.637.313	99.94%	0.19%	99.94%	99.94%
● **ResNet50V2**	24.614.401	84.89%	34.89%	85.20%	87.68%
● **Inception ResNetV2**	55.124.193	55.19%	68.78%	55.19%	100%
● **Xception**	21.911.081	94.09%	15.54%	94.58%	94.84%
● **DenseNet**	7.562.817	97.50%	6.88%	97.84%	97.62%
● **MobileNet**	3.754.177	97.87%	5.24%	98.02%	98.13%
● **MobileNetV2**	2.914.369	96.28%	9.53%	96.74%	96.47%
● **NASNetMobile**	4.811.413	85.09%	33.14%	84.13%	89.98%

NASNet Mobile is the lightweight model in our case study (~4.8m) it is an edge devices model however its performance is insufficient for our purposes.

Fig. 8 shows predicted image samples that demonstrate that our system can almost perfectly distinguish between fire and normal forest state regardless of all the features and variety of objects (people. snow. different types of trees. etc.). Fig. 9 shows the incorrectly predicted images using the VGG16 and ResNet50 models (these images are collected from the Web and not seen by the model neither in training nor in validation) which can easily explain why the model wrongfully predicted the bad labels. The most likely explanation for the negative results is that they can really deceive the human eye; in the first image the sun and its radiation in the clouds and the lake can be easily misinterpreted as fire because we see the same features and patterns of flames. In the second and third images our system incorrectly misidentifies fog as fire.

No system is perfect, but these findings show that deep learning can be extremely accurate in detecting wildfires, with a success rate of more than 99.9%. Thanks to its ability to identify the unique signatures emitted by wildfires, this can provide an early warning of a wildfire, allowing fire crews to be dispatched to the scene before it spreads too far. This solution could be a valuable tool for fire departments and other emergency responders in identifying and responding to wildfires.

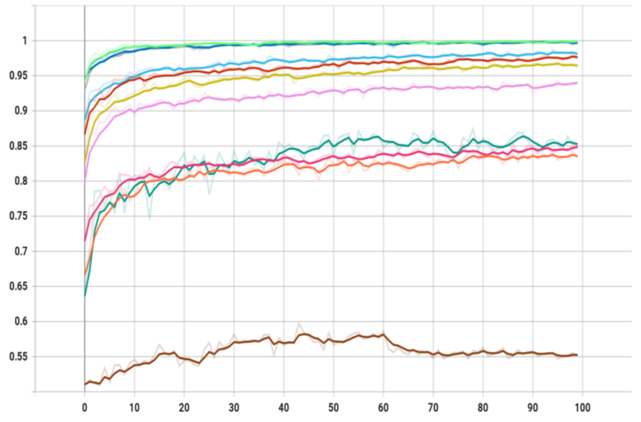
Fig. 4. Achieved Accuracy (over 100 Epochs) for the Implemented Models.

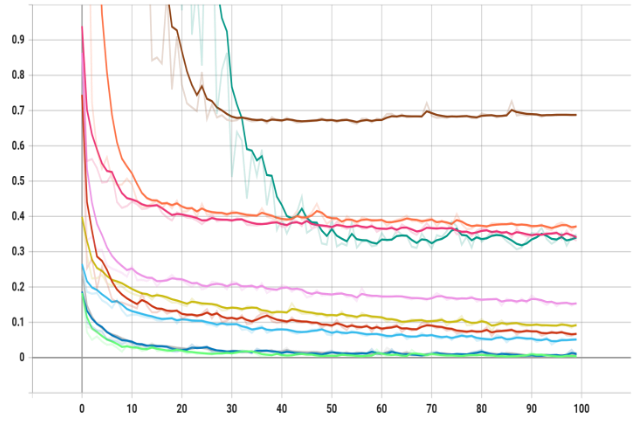
Fig. 5. Achieved Loss (over 100 Epochs) for the Implemented Models.

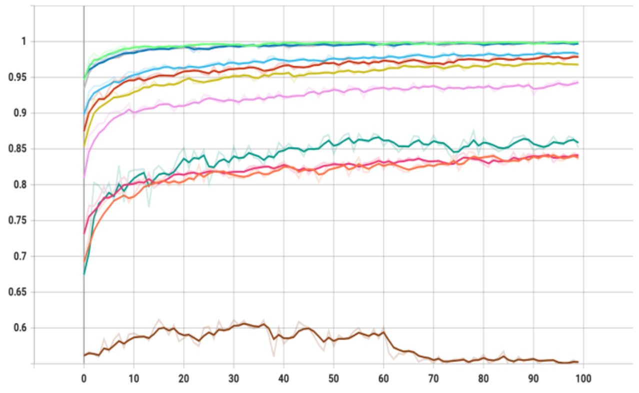
Fig. 6. Achieved Precision (over 100 Epochs) for the Implemented Models.

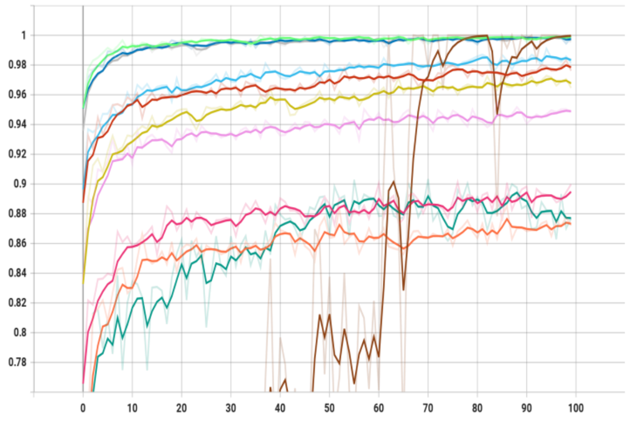
Fig. 7. Achieved Recall (over 100 Epochs) for the Implemented Models.

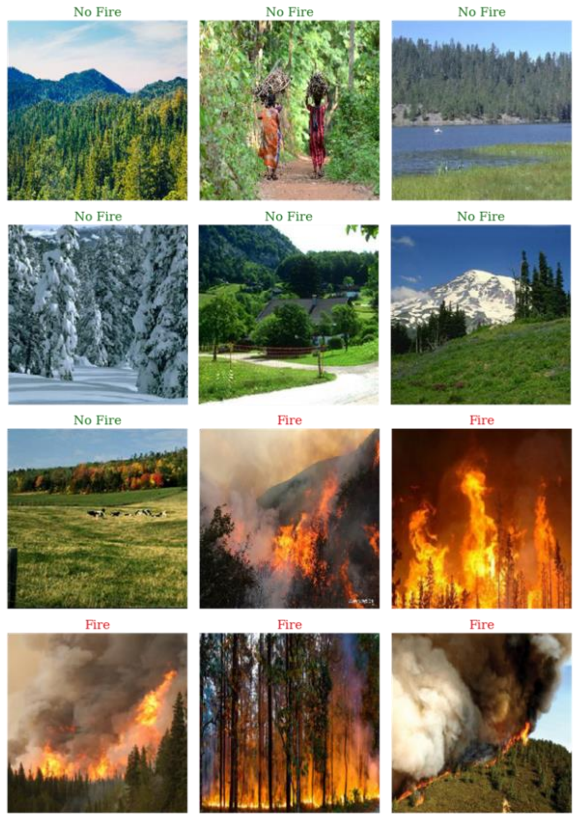

Fig. 8. Examples of Predicted Wildfires.

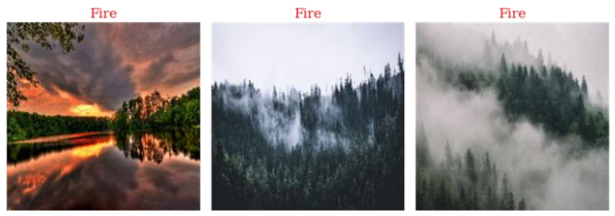

Fig. 9. Examples of Wrongly Predicted Wildfires.

In the future, we will deal with the incorrect negative cases, in which the system can be confused between flames and the sun, fog and clouds, and smoke. Furthermore, we will try to implement these models as the feature extractor backbone in other DL algorithms such as the R-CNN (Region-Based Convolutional Neural Networks) family [43], SSD (Single Shot Detector) [44], or applying YOLO (You Only Look Once) [45], [46] in order to detect not only fires but also its precise coordinates.

VI. Conclusion

Deep learning has revolutionized computer vision by enabling computers to learn from data to recognize patterns and classify objects with high accuracy. This has led to the development of powerful computer vision algorithms and applications that can detect and identify objects in photos and videos with a high degree of accuracy. Our proposed approach in this paper involves building a deep learning model specifically for detecting wildfires in forests using the transfer learning technique. Our discussion based on the obtained results has given us VGG16 and ResNet50 as relevant models for our issue; they are able to achieve higher scores in accuracy, recall and precision of more than 99.9% and a loss down to 0.19% for ResNet50 and down to 0.48% for VGG16. Fire departments and other emergency responders may benefit from these techniques to better identify and control wildfires before they spread too far. Through future works we will try to improve and develop these models by using object detection approaches such as the R-CNN family, SSD and YOLO to identify fires based on their precise location coordinates.

Acknowledgment

This work is supported by the Mohammed First University under the PARA1 Program (Low-cost and real-time Forest Fire Detection System based on Wireless Sensor Networks - SDF-RCSF). The used 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, 2022.

[2] 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.

[3] Africanews, “Wildfires bring devastation to Algeria, Tunisia | Africanews.” https://www.africanews.com/2021/08/12/wildfires-bring-devastation-to-algeria-tunisia/ (accessed Jan. 26, 2022).

[4] Africanews, “Forest fires rage in northern Morocco | Africanews.” https://www.africanews.com/2021/08/16/forest-fires-rage-in-northern-morocco/ (accessed Jan. 26, 2022).

[5] 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, 2022.

[6] J. Peters, “Foundations of Computer Vision - Computational Geometry, Visual Image Structures and Object Shape Detection,” 2017, pp. 1–443. Accessed: Jan. 26, 2022. [Online]. Available: https://books.google.com/books/about/Foundations_of_Computer_Vision.html?hl=fr&id=CtxmDgAAQBAJ

[7] A. Kherraki, M. Maqbool, and R. El Ouazzani, “Traffic Scene Semantic Segmentation by Using Several Deep Convolutional Neural Networks,” 2021 3rd IEEE Middle East North Africa Commun. Conf., pp. 1–6, Dec. 2021, doi: 10.1109/MENACOMM50742.2021.9678270.

[8] 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.

[9] 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, Feb. 2022, doi: 10.11591/IJEECS.V25.I2.PP.

[10] I. Idrissi, M. Azizi, and O. Moussaoui, “IoT security with Deep Learning-based Intrusion Detection Systems: A systematic literature review,” in 4th International Conference on Intelligent Computing in Data Sciences, ICDS 2020, Nov. 2020, pp. 1–10. doi: 10.1109/ICDS50568.2020.9268713.

[11] 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.

[12] I. Idrissi, M. Boukabous, M. Azizi, O. Moussaoui, and H. El Fadili, “Toward a deep learning-based intrusion detection system for IoT against botnet attacks,” IAES Int. J. Artif. Intell., vol. 10, no. 1, pp. 110–120, Mar. 2021, doi: 10.11591/ijai.v10.i1.pp110-120.

[13] K. Simonyan and A. Zisserman, “Very Deep Convolutional Networks for Large-Scale Image Recognition,” 3rd Int. Conf. Learn. Represent. ICLR 2015 - Conf. Track Proc., Sep. 2014, Accessed: Jan. 26, 2022. [Online]. Available: https://arxiv.org/abs/1409.1556v6

[14] C. Szegedy et al., “Going Deeper with Convolutions,” Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., vol. 07-12-June-2015, pp. 1–9, Sep. 2014, doi: 10.1109/CVPR.2015.7298594.

[15] 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-December, pp. 770–778, Dec. 2015, doi: 10.1109/CVPR.2016.90.

[16] C. Szegedy, S. Ioffe, V. Vanhoucke, and A. A. Alemi, “Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning,” 31st AAAI Conf. Artif. Intell. AAAI 2017, pp. 4278–4284, Feb. 2016, Accessed: Feb. 12, 2022. [Online]. Available: https://arxiv.org/abs/1602.07261v2

[17] F. Chollet, “Xception: Deep Learning with Depthwise Separable Convolutions,” Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-January, pp. 1800–1807, Oct. 2016, doi: 10.1109/CVPR.2017.195.

[18] G. Huang, Z. Liu, L. Van Der Maaten, and K. Q. Weinberger, “Densely Connected Convolutional Networks,” Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017, vol. 2017-January, pp. 2261–2269, Aug. 2016, doi: 10.1109/CVPR.2017.243.

[19] A. G. Howard et al., “MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications,” arXiv:1704.04861, Apr. 2017, Accessed: Jul. 14, 2022. [Online]. Available: http://arxiv.org/abs/1704.04861

[20] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, “Learning Transferable Architectures for Scalable Image Recognition,” Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit., pp. 8697–8710, Jul. 2017, doi: 10.1109/CVPR.2018.00907.

[21] B. Zoph and Q. V. Le, “Neural Architecture Search with Reinforcement Learning,” 5th Int. Conf. Learn. Represent. ICLR 2017 - Conf. Track Proc., Nov. 2016, Accessed: Feb. 12, 2022. [Online]. Available: https://arxiv.org/abs/1611.01578v2

[22] S. Dutta and S. Ghosh, “Forest Fire Detection Using Combined Architecture of Separable Convolution and Image Processing,” 2021 1st Int. Conf. Artif. Intell. Data Anal. CAIDA 2021, pp. 36–41, Apr. 2021, doi: 10.1109/CAIDA51941.2021.9425170.

[23] S. Aslan, U. Gudukbay, B. U. Toreyin, and A. Enis Cetin, “Early Wildfire Smoke Detection Based on Motion-based Geometric Image Transformation and Deep Convolutional Generative Adversarial Networks,” ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc., vol. 2019-May, pp. 8315–8319, May 2019, doi: 10.1109/ICASSP.2019.8683629.

[24] Y. Wang, L. Dang, and J. Ren, “Forest fire image recognition based on convolutional neural network:,” https://doi.org/10.1177/1748302619887689, vol. 13, Nov. 2019, doi: 10.1177/1748302619887689.

[25] Y. Chen, Y. Zhang, J. Xin, Y. Yi, D. Liu, and H. Liu, "A UAV-based Forest Fire Detection Algorithm Using Convolutional Neural Network," Chinese Control Conf. CCC, vol. 2018-July, pp. 10305–10310, Oct. 2018, doi: 10.23919/CHICC.2018.8484035.

[26] I. Idrissi, M. Azizi, and O. Moussaoui, "An unsupervised generative adversarial network based-host intrusion detection system for internet of things devices," Indones. J. Electr. Eng. Comput. Sci., vol. 25, no. 2, pp. 1140–1150, Feb. 2022, doi: 10.11591/IJEECS.V25.I2.PP1140-1150.

[27] 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. doi: doi.org/10.1007/978-3-030-66840-2_8.

[28] "Forest Fire Images | Kaggle." https://www.kaggle.com/mohnishsaiprasad/forest-fire-images (accessed Jan. 27, 2022).

[29] M. Berrahal and M. Azizi, "Optimal text-to-image synthesis model for generating portrait images using generative adversarial network techniques," Indones. J. Electr. Eng. Comput. Sci., vol. 25, no. 2, Feb. 2022, doi: 10.11591/IJEECS.V25.I2.PP.

[30] 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.

[31] A. Mikołajczyk and M. Grochowski, "Data augmentation for improving deep learning in image classification problem," 2018 Int. Interdiscip. PhD Work. IIPhDW 2018, pp. 117–122, Jun. 2018, doi: 10.1109/IIPHDW.2018.8388338.

[32] M. Boukabous and M. Azizi, "A comparative study of deep learning based language representation learning models," Indones. J. Electr. Eng. Comput. Sci., vol. 22, no. 2, pp. 1032–1040, 2021, doi: 10.11591/ijeecs.v22.i2.pp1032-1040.

[33] "Keras Applications." https://keras.io/api/applications/ (accessed Jan. 30, 2022).

[34] I. Idrissi, M. Azizi, and O. Moussaoui, "Accelerating the update of a DL-based IDS for IoT using deep transfer learning," Indones. J. Electr. Eng. Comput. Sci., vol. 23, no. 2, pp. 1059–1067, Aug. 2021, doi: 10.11591/ijeecs.v23.i2.pp1059-1067.

[35] "API Documentation| TensorFlow Core v2.7.0." https://www.tensorflow.org/api_docs (accessed Jan. 27, 2022).

[36] M. Hossin and Sulaiman, "A REVIEW ON EVALUATION METRICS FOR DATA CLASSIFICATION EVALUATIONS," IJDKP) Int. J. Data Min. Knowl. Manag. Process, vol. 5, no. 2, 2020, doi: 10.5121/ijdkp.2015.5201.

[37] "Metrics to Evaluate your Machine Learning Algorithm | by Aditya Mishra | Towards Data Science." https://towardsdatascience.com/metrics-to-evaluate-your-machine-learning-algorithm-f10ba6e38234 (accessed Sep. 13, 2020).

[38] Z. Zhang and M. R. Sabuncu, "Generalized cross entropy loss for training deep neural networks with noisy labels," in Advances in Neural Information Processing Systems, 2018, vol. 2018-Decem, pp. 8778–8788.

[39] F. Cakir, K. He, X. Xia, B. Kulis, and S. Sclaroff, "Deep Metric Learning to Rank." pp. 1861–1870, 2019.

[40] M. Geiger et al., "Scaling description of generalization with number of parameters in deep learning," J. Stat. Mech. Theory Exp., vol. 2020, no. 2, p. 023401, Feb. 2020, doi: 10.1088/1742-5468/AB633C.

[41] "The Effective Number of Parameters: An Analysis of Generalization and Regularization in Nonlinear Learning Systems." https://proceedings.neurips.cc/paper/1991/hash/d64a340bcb633f536d56e51874281454-Abstract.html (accessed Mar. 17, 2022).

[42] I. Idrissi, M. Mostafa Azizi, and O. Moussaoui, "A Lightweight Optimized Deep Learning-based Host-Intrusion Detection System Deployed on the Edge for IoT," Int. J. Comput. Digit. Syst., vol. 11, no. 1, pp. 209–216, 2022, doi: 10.12785/ijcds/110117.

[43] 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.

[44] W. Liu et al., "SSD: Single Shot MultiBox Detector," Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 9905 LNCS, pp. 21–37, Dec. 2015, doi: 10.1007/978-3-319-46448-0_2.

[45] "YOLO: Real-Time Object Detection." https://pjreddie.com/darknet/yolo/ (accessed Jul. 26, 2020).

[46] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," arXiv:1804.02767v1, 2018, Accessed: Jul. 26, 2020. [Online]. Available: https://pjreddie.com/yolo/.