Deep Learning approach applied to drone imagery for the automatic detection of forest fire

Maryeme Akhatar

Master Thesis, Msc. Geospatial Technologies

IFGI_ WWU Münster

<u>**Supervisors:**</u>

Pr. Marius Appel (WWU Münster)

Pr. Hanna Meyer (WWU Münster)

Pr. Joaquín Torres (UJI Castellon)

January 2023

ABSTRACT

Wildfires are one of the world's most costly and deadly natural disasters, damaging millions of hectares of vegetation and threatening the lives of people and animals. The risks to civilian agents and task forces are particularly high, which emphasizes the value of leveraging technology to minimize their impacts on nature and people. The use of drone imagery coupled with deep learning for automated fire detection can provide new solutions to this problem, limiting the damage that result.

In this context, our work aims to implement a solution for the automatic detection of forest fires in real time by exploiting convolutional neural networks (CNN) on drone images based on classification and segmentation models.

The methodological approach followed in this study can be broken down into three main steps: First, the comparison of two models, namely Xception Network and EfficientNetB2, for the classification of images captured during a forest burn into 'Fire' or 'No_Fire' classes. Then we will proceed to the segmentation of the images belonging to the 'Fire' class by comparing the U-Net architecture with Attention U-Net and Trans U-Net in order to choose the best performing model.

The EfficientNetB2 architecture for classification gave satisfactory results with an accuracy of 71.72%. Concerning segmentation, we adopted the U-Net model which offers a segmentation accuracy that reaches 98%. As for the deployment, a fire detection application was designed using Android Studio software by assimilating the drone's camera.

Key words: Automatic detection, Forest fires, Deep Learning, Convolutional neural networks, Classification, Semantic segmentation, Xception, EfficientNetB2, U-Net, Attention U-Net, Trans U-Net, Deployment.

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS

AG: Attention Gate.

AI: Artificial Intelligence.

ANN: Artificial Neuronal Network.

CNN: Convolutional Neural Network.

CUP: Cascaded Up-Sampler.

CV: Cross Validation.

DL: Deep Learning.

FC: Fully Connected.

FCN: Fully Convolutional Network.

GPU: Graphics Processing Unit.

IoU: Intersection Over Union.

ML: Machine Learning.

MLP: Multilayer Perceptron.

MSA: Multi-Head self-attention.

NLP: Natural Language Processing.

ReLU: Rectified Linear Unit.

ROI: Region Of Interest.

UAV: Unmanned Aerial Vehicle (Drone).

ViT : Vision Transformer.

GENERAL INTRODUCTION

1. Context

It goes without saying that during the last ten years wildfires has been a recurrent event in all the parts of the world namely north America (California, Canada) as well as around the Mediterranean basin (Iberian Peninsula, Algeria, Morocco, Italy, Greece, Turkey and France) [Ana Cristina Gonçalves, 2017]. In the other hand, the resulting accidents from wildfires constitute one of the most dangerous risks, taking into consideration the breathtaking statistics of the damages of wildfires [NICC Wildland Fire Summary and Statistics Annual Reports, 2022]. In fact each year a hundred of Million hectares of vegetation is burnt and a hundred of people die. Moreover, wildfires generate a considerable quantity of Carbone dioxide, release greenhouses gas and degrade the ecosystem. The cause of their occurrences is considered as a very complex issue. However in most cases human is the first trigger of forest fire. It is crystal clear that climate change and the extreme meteorological imbalance that we have been feeling these last years have contributed in the ecosystem degradation, but they are not a direct and principal trigger of the wildfires. Hence, once the fire is triggered, it may transform into an incontrollable propagation that causes a huge loss of both life and vegetation. Henceforth, the prevention seems the clue of the fight against wildfires.

In this frame, different methods have been applied with the purpose to control this phenomenon. As a matter of fact, wildfires have been traditionally controlled by using helicopters, aeronefs, positioning sensors…However these techniques are still few and not covering large areas. They also face some limitations such as the delay of the detection, false alarms...Nevertheless many researchers have been investigating the remote and non-material detection of fires using only satellite or aerial imagery.

Fire detection through remote sensing analysis assumes several advantages such as the high precision and the efficient detection in large zones. Thanks to image analysis algorithms, wildfires could be precisely detected and localized. Nonetheless it is a time consuming task that need calculations as well as a robust acquisition and processing systems. The automating of the process of the detection using drone imagery and machine learning allows overcoming these limitations. Drones are very useful to acquire the data from remote and difficult areas and then to process it. Furthermore Machine Learning becomes one of the most useful techniques in the early, quick and precise detection of any phenomena.

2. Motivation

Firefighters face a clear direct professional risk. Their work is dangerous and difficult because of many reasons. First of all the difficulty of predicting fire propagation pushes firefighters to deal with forest fire without preventing its location, which makes the situation less controllable. Moreover the smoke can cause respiratory and healthy troubles which may lead to grave diseases [EPA, 2021]. The use of drones seems very useful for the control of the fire and the orientation of firefighters without risking their life. In particular a drone with an RGB camera can provide information about the propagation and the direction of the fire. The acquired images would be transferred then to the concerned people for processing in order to determine the scope of the loss and act immediately. In this context deep learning plays a significant role in the automating of fire detection by offering a good accuracy and precision, cost effective solution and time saving process [Suwei Yang et al., 2021]. Being one of the most used deep learning architecture in image processing, convolutional neural network has shown a robustness and efficiency in detecting objects. In order to have an automatic detection of fire from drone images, this work aims to develop a CNN models for both the classification of acquired images by UAV and for the semantic segmentation by delimiting fire zones on the image.

3. Objectives

The main objective of this project is the automatic detection of wildfire by using UAV images acquired by an RGB camera. This would be achieved by exploiting some deep learning architecture for image processing: convolutional neural network particularly. To reach that, we specified the following sub-objectives:

- Exploration of the potential of deep learning architectures in the automatic detection of forest fires.
- The use of CNN models allowing the classification of drone images.
- Developing models of semantic segmentation of fire images.

Research questions

- *How are EfficientNet and Xception algorithms performing in detecting fire?*
- *How are the predictions, using each of U-Net, Attention U-Net and Trans U-Net, accurate in delimitating fire?*
- *Which of the methods is more reliable for real time fire detection?*

4. Thesis Structure

This document is composed of three chapters in which one is for the literature review and two last ones for the technical and practical parts of the project. So the first chapter is dedicated to generalities and knowledge acquired from the literature review. Then the following chapter would tackle the methodology and the data processing. Finally the last chapter will present the results and discuss them.

CHAPTER I : LITERATURE REVIEW

1. General Information about Deep Learning and its Applications

Since the discovery of the computer, scientists were asking if the computer could be intelligent [Lovelace, 1842]. Nowadays, artificial intelligence AI is a successful field with tremendous applications and active research in different domains. Human counts on intelligent software in facilitating routine work, understanding speeches and images, make medical diagnostics and sustain fundamental research [Bengio et al., 2015].

It is also important to distinguish between the Machine Learning and Deep Learning. Actually Machine Learning is a mean of data analysis considered as a brunch of AI, based on the idea that the system can learn from the data, generate the model and then take decision automatically with a minimal human intervention [Lemly et al., 2017]. It aims then to automatize the tasks of the construction of analytical models in order to perform cognitive tasks like object detection or translation in natural language [Samuel, 1959]. Deep Learning is another function of AI that imitates human brain in the processing of the data and creating models used in decision making. Deep Learning is a subfield of Machine Learning, which has networks that can learn without surveillance from non-structured and non-labeled data. They are also known as Neural Networks. Deep learning is particularly useful in the domains with huge amount of multidimensional data [Iqbal H. Sarker, 2021].

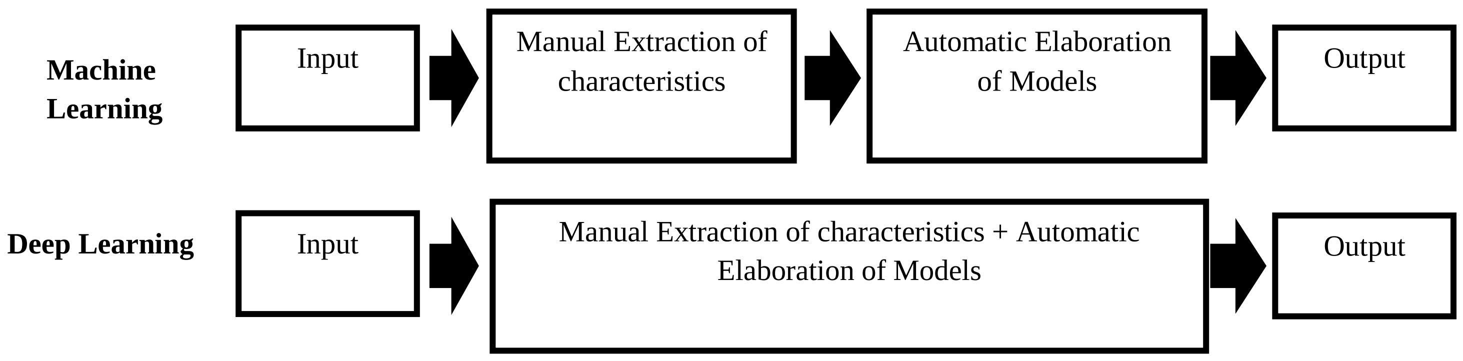

Machine Learning
Input
Manual Extraction of characteristics
Automatic Elaboration of Models
Output
Deep Learning
Input
Manual Extraction of characteristics + Automatic Elaboration of Models
Output

Figure 1: Process of elaboration of an analytical model

1.1. Neural Networks

1.1.1. Model of an Artificial Neural Network

A neuron is fundamentally made of an integrator that performs the linear combination of the inputs plus a bias ‘b’. This result would be then transformed thanks to the activation function ‘f’ to an output ‘a’.

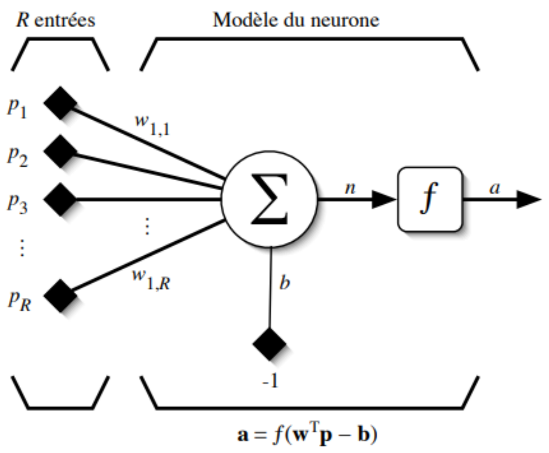


Figure 2: Model of an Artificial Neural Network (Parizau, 2004)

1.1.2. Model of a Neural Network:

A neural network is consequently a network of several neurons, usually structured in layers. Neurons of the same layer are limited to the model inputs (fully connected layer), for each connection a weight is attributed w.

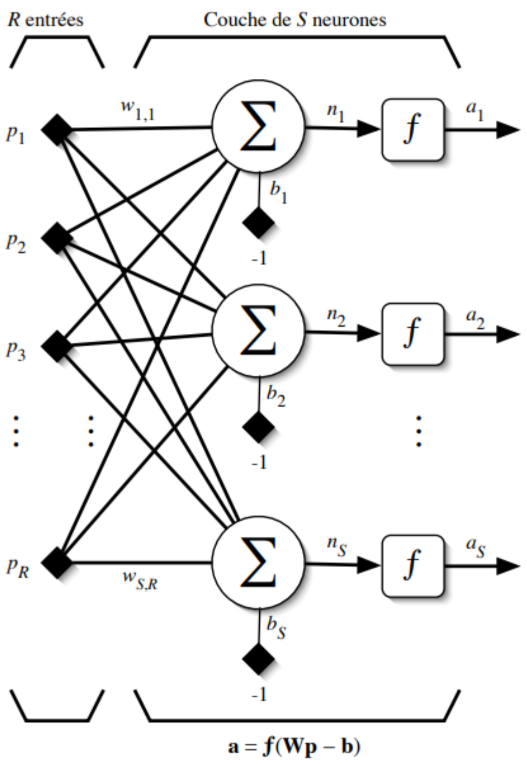


Figure 3: Model of a Neural Network

1.1.3. Activation Function :

The activation function is used in a neural network in order to calculate the weighted sum of inputs and bias, it is useful to decide if a neuron would be activated or not. [Nwankpa et al., 2018]

There are different activation functions [Nwankpa et al., 2018]

- Rectified Linear Unit (ReLu) :

ReLU is the most efficient and used activation function. It is also too fast and it returns a 0 if the impact is a negative and conserve it if it's a positive.

$$f_{relu}(x) = max(0,x) \qquad (1.4)$$

- Sigmoïd

The Sigmoid is a nonlinear activation function used primarily in forward propagation neural networks. It is a bounded differentiable real function, defined for real input values with positive derivatives everywhere and some degree of regularity.

$$f_{sigm}(x) = \frac{1}{1 + e^{-x}} \qquad (1.2)$$

- Hyperbolic Tangent

The hyperbolic tangent activation function allows eliminating input values in the range of [-1; 1].

$$f_{tanh}(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}} \qquad (1.3)$$

- Softmax

Softmax function is another type of activation function used in neural computation; it is used to calculate the distribution of the probability from a vector of real numbers. The Softmax function produces an output in the range of values between 0 and 1, with the sum of the probabilities equal to 1

$$f(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \qquad (1.4)$$

1.2. Convolution Neural Network CNN:

This type of neural network is used in the processing of the data disposing of a structure like a grid such as an image. Its name stems from a mathematical linear operation between matrices called the convolution [Goodfellow et al., 2016], [R. Yamashita et al., 2018].

1.2.1. Constitutive layers of a CNN

a) Convolution Layer :

Convolution is a specialized type of linear operation used for feature extraction, where a matrix, called a kernel, is applied to the input which is a matrix called a tensor.

In digital images, pixel values are stored in a two-dimensional (2D) grid, i.e. a matrix, and a small parameter grid called a kernel, acting as a feature extractor, is applied to every position in the image, which makes CNNs very efficient for image processing since a feature can appear anywhere in the image [Yamashita et al., 2018].

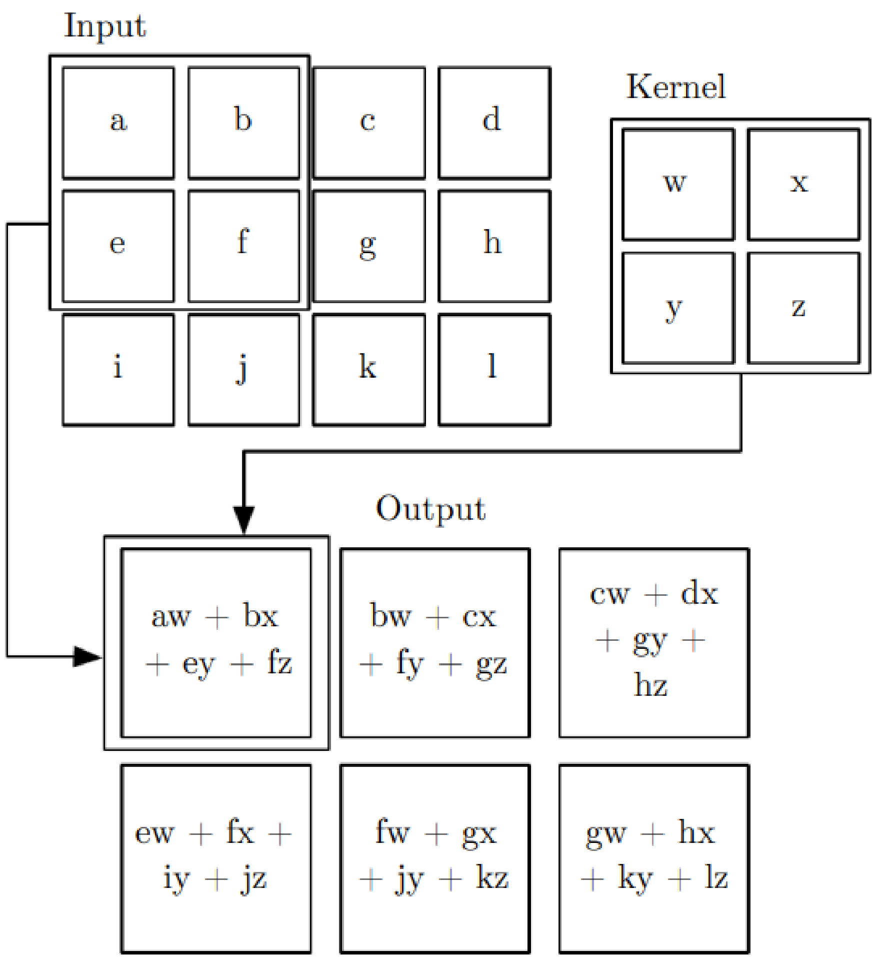


Figure 4: representation of the convolution (Y. Bengio et al., 2015)

The mathematical equation of the convolution is as follows:

$$\boldsymbol{S(i,j) = (K * I)(i,j) = \sum_{m}\sum_{n} I(i-m, j-n)K(m,n)} \qquad (1.1)$$

Two key hyper parameters that define the convolution operation are the size and the number of kernels. The former is usually 3×3, but sometimes 5×5 or 7×7. The latter is arbitrary and determines the depth of the output feature maps [Yamashita et al., 2018].

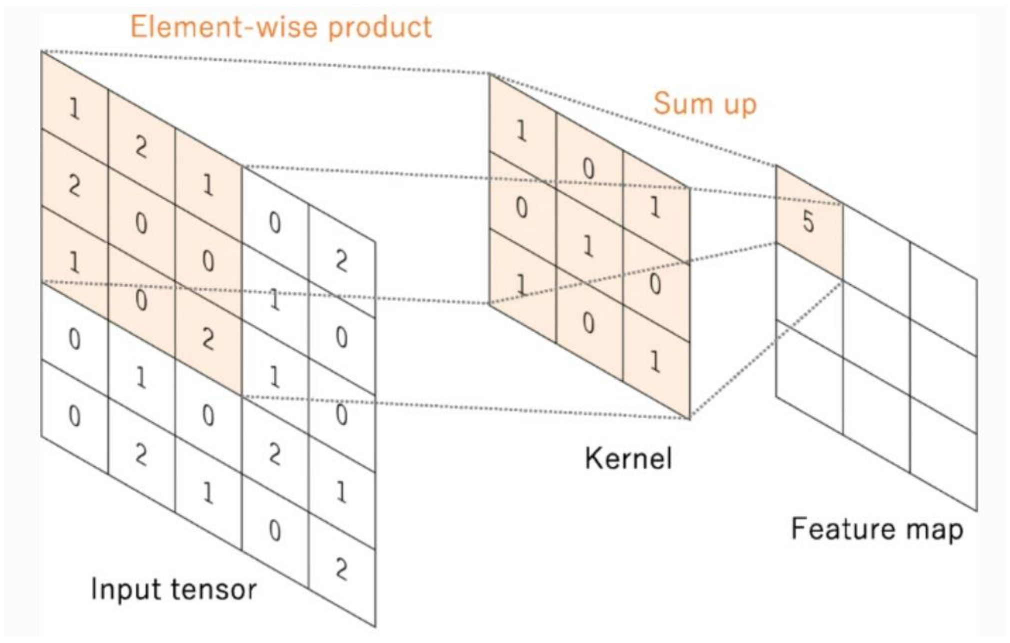


Figure 5: Example of a convolution by kernel of size 3x3

- Non-linear activation function

The outputs of a linear operation such as convolution are then passed through a nonlinear activation function. Although smooth nonlinear functions, such as the sigmoid function or the hyperbolic tangent (tanh) function, are more representative of the behavior of a biological neuron, the most commonly used nonlinear activation function today is the linear unit rectified (ReLU), which simply computes the function : f(x) = max(0, x) [Yamashita et al., 2018].

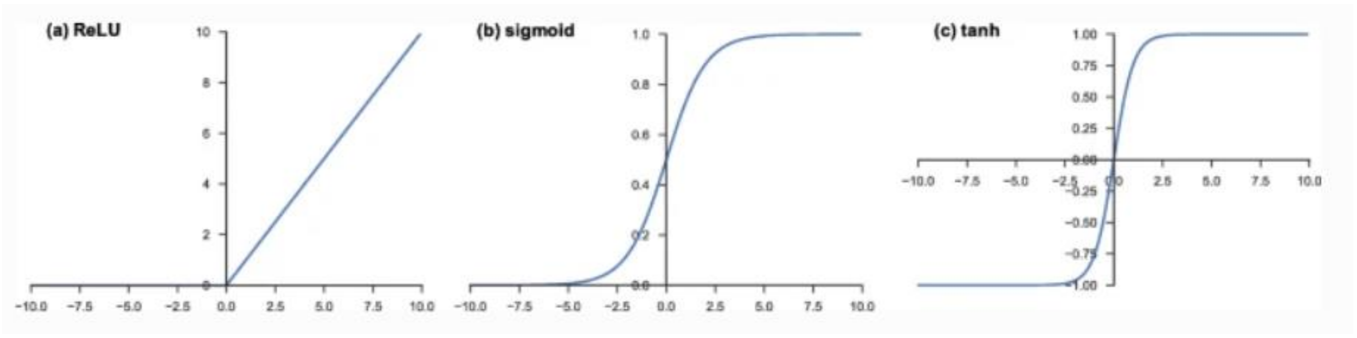


Figure 6: the most used activation functions

b) Pooling layer :

Pooling layers are responsible for reducing the spatial dimensions (width x height) of the input volume for the next convolutional layer. The operation performed by this layer is also called downsampling, because reducing the size results in a simultaneous loss of information. However, such a loss is beneficial for the network because the decrease in size leads to a reduction in the computational load for the following layers of the network, and also helps to avoid over-learning. “Average pooling” and “max pooling” are the most commonly used methods [Voulodimos, 2017]. The "max pooling" returns the maximum value in a rectangular neighborhood and the "average pooling" returns the average of the values of a rectangular neighborhood [Bengio et al., 2015].

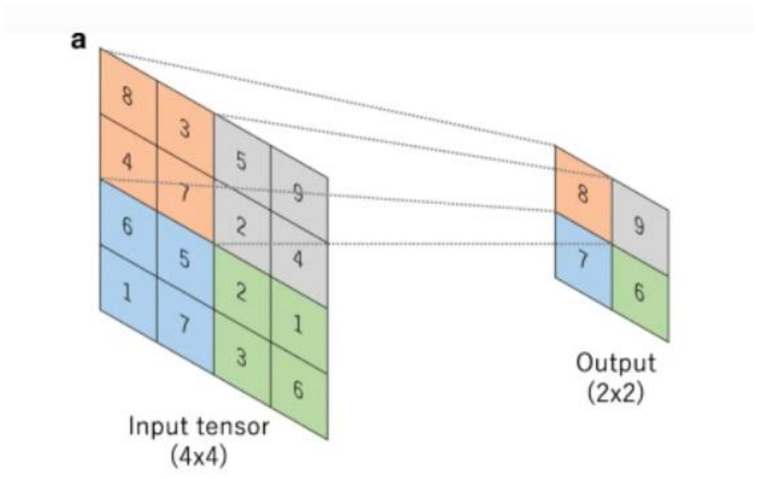


Figure 7: Example of max pooling by a filter of size 2x2 [R. Yamashita et al., 2018]

c) The Fully Connected Layer :

The output feature maps of the final convolution or pooling layer are usually flattened, that is, transformed into a one-dimensional (1D) array of numbers (or vector), and connected to one or more layers fully connected, also called dense layers, in which each input is connected to each output by a weight. Once the features extracted by the convolution layers and downsampled by the pooling layers are created, they are matched by a subset of fully connected layers to the final outputs of the network, such as the probabilities of each class in classification tasks. The final fully connected layer usually has the same number of output nodes as the number of classes. Each fully connected layer is followed by a nonlinear activation function [Yamashita et al., 2018].

So the fully connected layer contains neurons that are directly connected to neurons in the two adjacent layers. This is analogous to the way neurons are arranged in traditional forms of artificial neural networks.

- The dropout :

The fully connected layer has an immense number of connections, therefore millions of parameters to calculate. So we try, during the training phase, to reduce this number of connections by using several algorithms.

Dropout is considered as the approach that makes deep learning methods work in practice, due to both its effectiveness in preventing over-learning and its low computational requirements. The basic version of the dropout was developed by Hinton et al. (2012) and Srivastava et al. (2014). The dropout means that at each iteration, for each learning pattern, each node can be dropped according to the Bernoulli distribution with a probability of 1-**p**, where **p** is the probability that the node is retained in the network (Piotrowski and al., 2020).

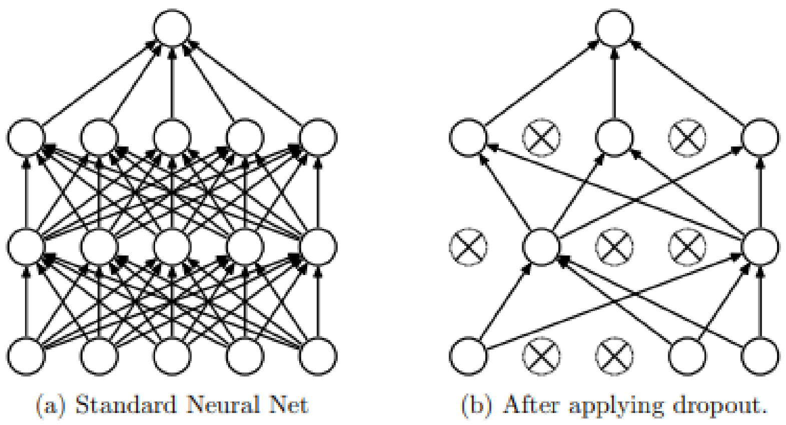


Figure 8: Abandoned neural network model

1.2.2. Examples of CNN architectures :

- AlexNet :

In 2012 Alex Krizhevesky et al. proposed a deeper and broader CNN model compared to LeNet and won the most difficult ImageNet challenge for visual object recognition, called

ImageNet Large Scale Visual Recognition Challenge (ILSVRC) in 2012. AlexNet achieved recognition accuracy state-of-the-art compared to all traditional machine learning and computer vision approaches. This is an important significant breakthrough in machine learning and computer vision for visual recognition and classification tasks and is the point in the story where interest in the deep learning has grown rapidly.

The first convolutional layer performs convolution and max pooling with local response normalization (LRN) where 96 different receptive filters, of size 11×11 are used. Max pooling operations are performed with 3×3 filters with a step size of 2. The same operations are performed in the second layer with 5×5 filters. 3×3 filters are used in the third, fourth and fifth convolutional layers with 384, 384 and 296 feature maps respectively. Two fully connected (FC) layers are used with dropout, followed by a Softmax layer at the end [Alom et al., 2018].

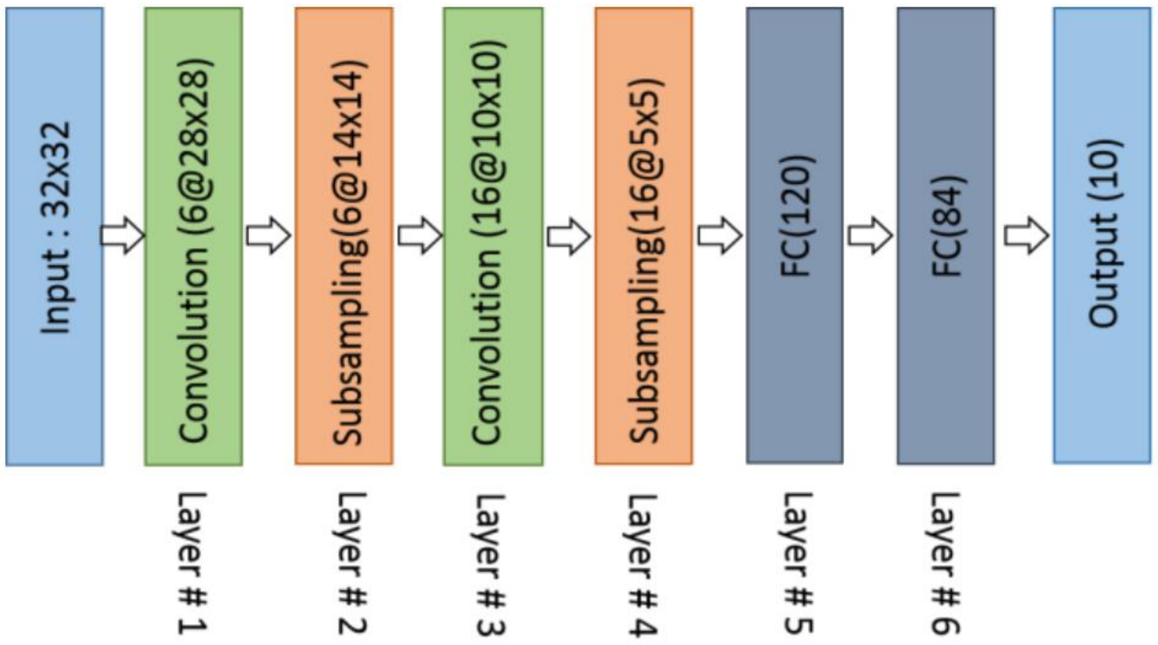


Figure9: Architecture of AlexNet (M. Alom et al., 2018)

- Network in Network :

This network model, which has slight differences from the AlexNet, introduced two innovative concepts. The first is the use of multiple convolution layers. These convolutions are performed using a 1×1 filter, which helps to add additional nonlinearity in the networks. Additionally, it increases the depth of the network, which can then be regularized using dropout. Instead of an FC layer, the global average pooling is also used, which represents the second new concept and allows a significant reduction in the number of model parameters. In addition, average pooling significantly improves network architecture. [Alzubaidi et al., 2021].

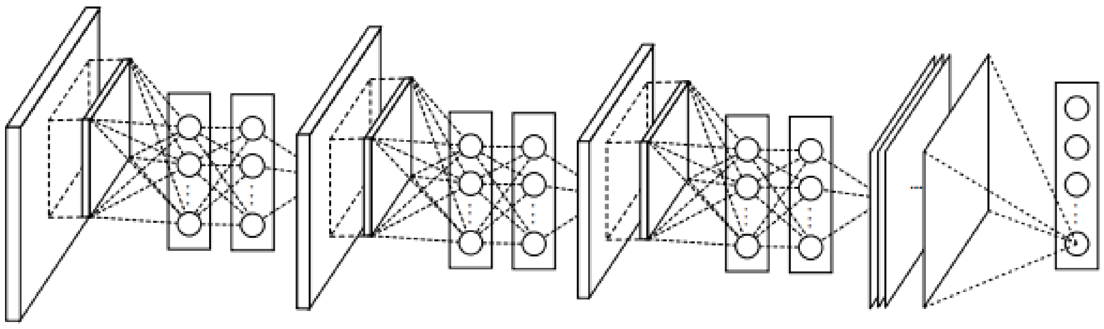


Figure10: Architecture of NIN

- ZefNet :

Before 2013, the CNN learning mechanism was basically built on the basis of trial and error, which made it difficult to understand the precise goal following the improvement. This issue limited the performance of deep CNNs on images. In response, Zeiler and Fergus introduced DeconvNet (a de-convolutional multilayer neural network) in 2013. This method later became known as ZefNet, which was developed in order to visualize the network quantitatively. The purpose of visualizing network activity was to monitor CNN performance by understanding neuron activation. By reversing the order of the convolutions and pooling layers, ZefNet works like a pass-through CNN. This type of reverse mapping throws the output of the convolution layer backwards to create visually observable image shapes that, in turn, give a neural interpretation of the internal representation of the features learned at each layer [Alzubaidi et al., 2021].

- VGG :

After CNN was found to be effective in the field of image recognition, Simonyan and Zisserman proposed a simple and efficient design principle for CNN. This innovative design was called Visual Geometry Group (VGG). A multi-layered model, it has nineteen more layers than ZefNet and AlexNet to deeply simulate the relationships of network representation capability. VGG has inserted a layer of 3×3 filters rather than ZefNet's 5×5 and 11×11 filters. This showed experimentally that the parallel assignment of these small size filters could produce the same influence as the large size filters. In other words, these small size filters made the receptive field as effective as the large size filters (7×7 and 5×5). By reducing the number of parameters, the use of small size filters has made it possible to reduce the complexity of the calculations. These results established a new research trend for working with small size filters in CNN networks. Furthermore, by inserting 1 × 1 convolutions in the middle of the layers of convolutions, the VGG regulates the complexity of the network [Alzubaidi et al., 2021]

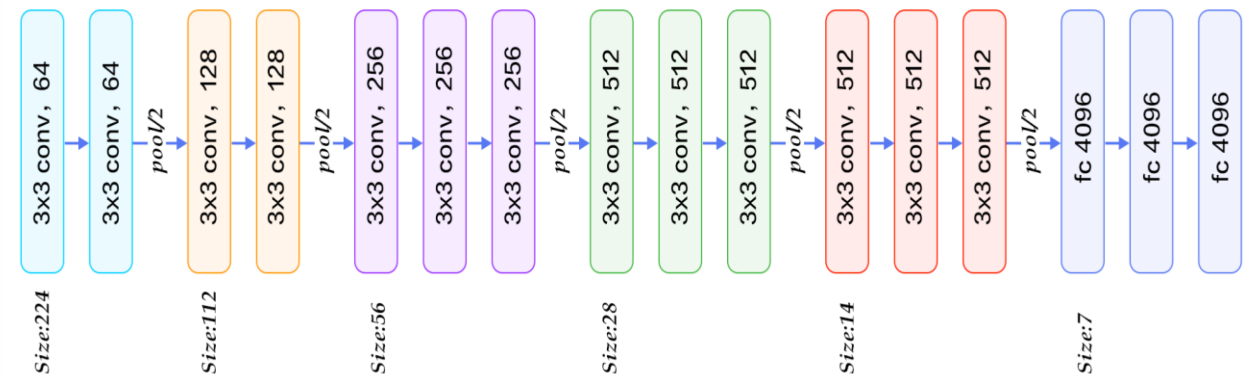


Figure11: Architecture of VGG model

- Encoder-Decoder :

a) U-Net :

U-NET is a neural network model dedicated to Computer Vision tasks and more particularly to semantic segmentation problems.

It consists of a contraction path and an expansion path. The contraction path follows the typical architecture of a convolutional network. It consists of the repeated application of two 3x3 convolutions, each followed by a rectified linear unit (ReLU) and a 2x2 max pooling operation with a step of 2 for downsampling. At each downsampling step, the number of feature channels is doubled. Each step of the expansion path consists of an oversampling of the feature map followed by a 2x2 convolution, a concatenation with the corresponding cropped feature map of the contracting path, and two 3x3 convolutions, each followed by

ReLU. Cropping is necessary due to the loss of border pixels with each convolution. At the final layer, a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes. In total, the network comprises 23 convolutional layers and has no fully connected layers [Ronneberger et al., 2015].

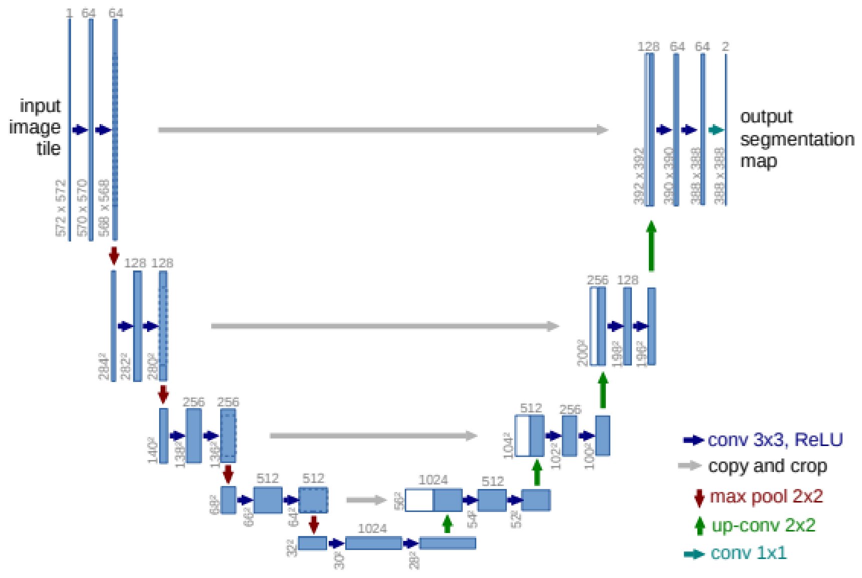


Figure 12: Architecture of U-Net (Ronneberger et al., 2015)

1.3. Applications of deep learning :

1.3.1. Image classification :

a) Definition :

Automatic image classification consists of automatically assigning a class to an image using a classification system. We thus find the classification of objects, scenes, textures, the recognition of faces, fingerprints and characters. There are two main types of learning: supervised learning and unsupervised learning. In the supervised approach, each image is associated with a label that describes its class membership. In the unsupervised approach, the available data does not have labels.

b) Performance Metrics of the classification :

- Confusion Matrix:

Confusion matrixes have been around for a long time in the evaluation of scientific models and engineering applications and are commonly used in many different fields such as computer vision (CV), natural language processing (NLP), etc. In its simplest form, a confusion matrix shows the performance of a binary classifier in a two-row, two-column table and represents the percentages of the four possible classification outcomes: True Positive (TP), False Positive (FP), True Negative (TN) and False Negative (FN).

Let's take the example of a binary classifier that predicts between a class A and a class B.

		Prediction	
		Class A	Class B
Reality	Class A	TP	FP
	Class B	FN	TN

Table 1: Example of a confusion Matrix

- Precision :

Precision represents the proportion of correctly classified elements in a class:

$$\boldsymbol{Precision} = \frac{TP}{TP + FP} \quad (1.4)$$

- Recall :

It is the proportion of well-classified elements compared to all the elements of a class:

$$\boldsymbol{Recall} = \frac{TP}{TP + FN} \quad (1.5)$$

- F-score :

Measures the compromise between the precision and the recall:

$$\boldsymbol{F - score} = \frac{2 * Precision * Recall}{Precision + Recall} \quad (1.6)$$

- Receiver operating characteristic (ROC) :

The ROC curve is a representation of the true positive rate versus the false positive rate. Its interest is to overcome the size of the test data in the event that the data is unbalanced.

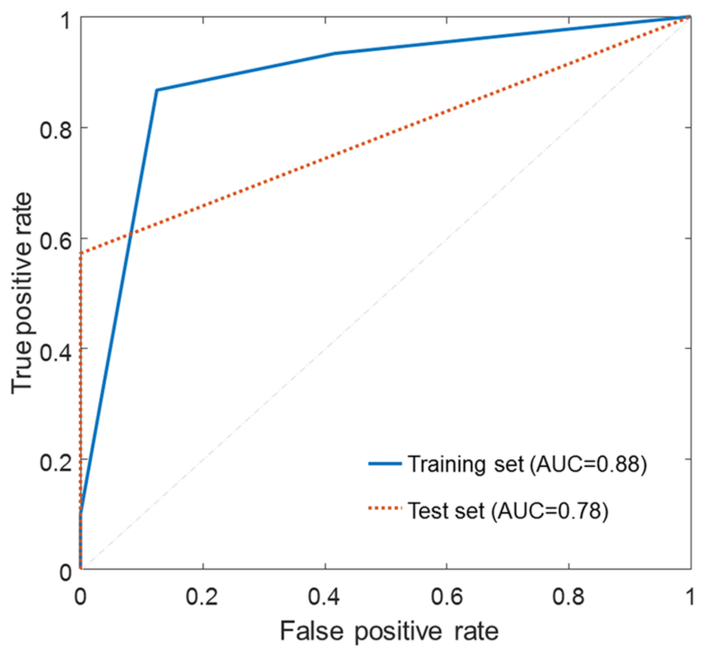


Figure 13: Example of ROC curve

1.3.2. Semantic Segmentation:

Semantic segmentation, or image segmentation, consists in grouping the parts of an image which belong to the same class of objects. It is a form of pixel-level prediction because each pixel in an image is classified according to a category.

Semantic segmentation is different from object detection because it does not predict bounding boxes around objects. The distinction is not made between different instances of the same object. For example, there could be several cars in the scene and all of them would have the same label.

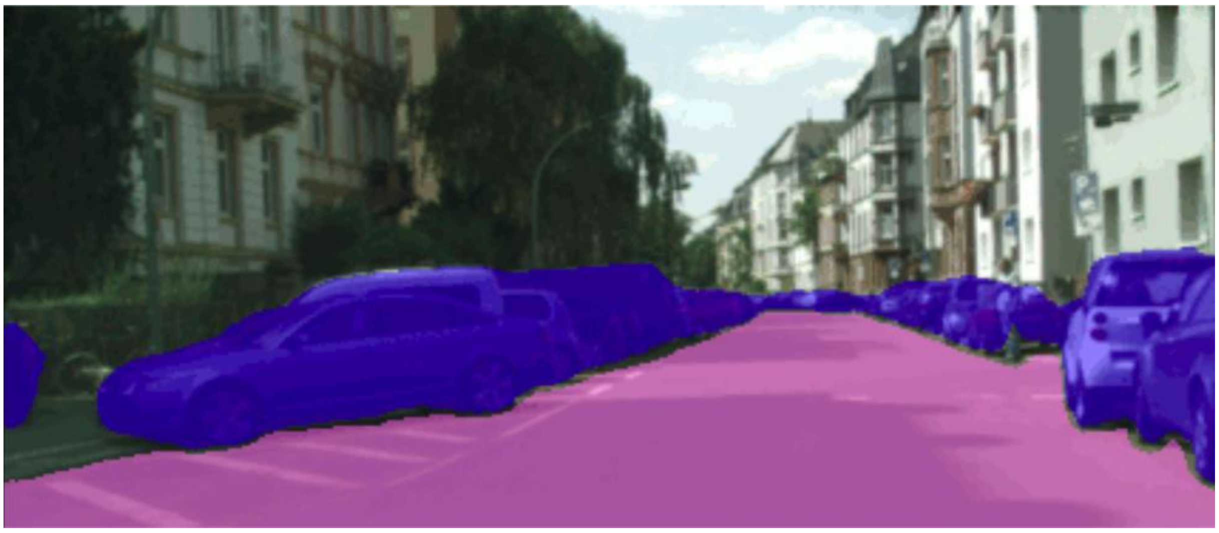


Figure14: Example of Semantic Segmentation

For the semantic segmentation task, the spatial information has to be kept, that's why no fully connected layer is used. This is why they are called fully convolutional networks. The convolution layers coupled with the downsampling layers produce a low resolution tensor containing the high level information.

By taking this tensor, we need to produce high resolution segmentation outputs. To do this, we add additional convolution layers coupled with oversampling layers that increase the size of the spatial tensor. As we increase the resolution, we decrease the number of channels as we return to low level information. This is called an encoder-decoder structure

1.4. Transformers :

1.4.1. Attention Mechanism :

Human beings can naturally and efficiently find salient regions in complex scenes. Motivated by this observation, attention mechanisms were introduced into computer vision in an attempt to mimic this aspect of the human visual system. Such an attention mechanism can be viewed as a dynamic weight adjustment process based on the characteristics of the input image. Attention mechanisms have been highly successful in many visual tasks, including image classification, object detection, and semantic segmentation.

Over the past decade, the attention mechanism has played an increasingly important role in computer vision. Figure 15 briefly summarizes the history of attention-based models in computer vision in the era of deep learning.

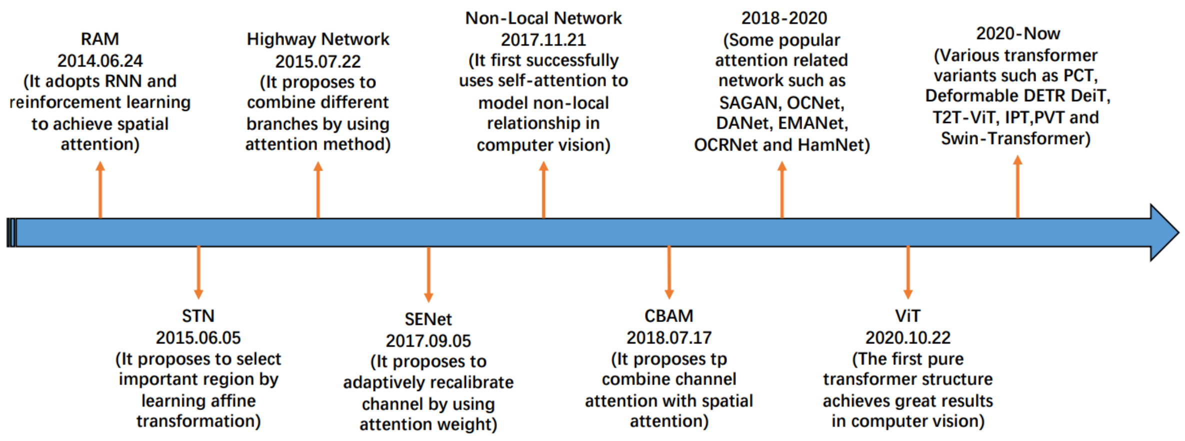

RAM
2014.06.24
(It adopts RNN and reinforcement learning to achieve spatial attention)
Highway Network
2015.07.22
(It proposes to combine different branches by using attention method)
Non-Local Network
2017.11.21
(It first successfully uses self-attention to model non-local relationship in computer vision)
2018-2020
(Some popular attention related network such as SAGAN, OCNet, DANet, EMANet, OCRNet and HamNet)
2020-Now
(Various transformer variants such as PCT, Deformable DETR DeiT, T2T-ViT, IPT,PVT and Swin-Transformer)
STN
2015.06.05
(It proposes to select important region by learning affine transformation)
SENet
2017.09.05
(It proposes to adaptively recalibrate channel by using attention weight)
CBAM
2018.07.17
(It proposes tp combine channel attention with spatial attention)
ViT
2020.10.22
(The first pure transformer structure achieves great results in computer vision)

Figure 15: Evolution of Attention mechanism in computer vision (Guo et al., 2021)

- Multi-head attention :

Usually attention mechanism realizes the following operations :

1- Each query vector $= s_{t-1}$, is matched against a database of keys to calculate a score value. This matching operation is calculated as the inner product of the specific query under consideration with each key vector k_i :

$e_{q,k_i} = q * k_i$

2- We apply a Softmax activation function to the scores to have the weights:
$\alpha_{q,k_i} = softmax(e_{q,k_i})$

3- Attention is then calculated by a weighted sum of the value vectors, v_{k_i}, where each value vector is associated with a corresponding key :
$attention(V,q,K) = \Sigma(\alpha_{q,k_i} * v_{k_i})$

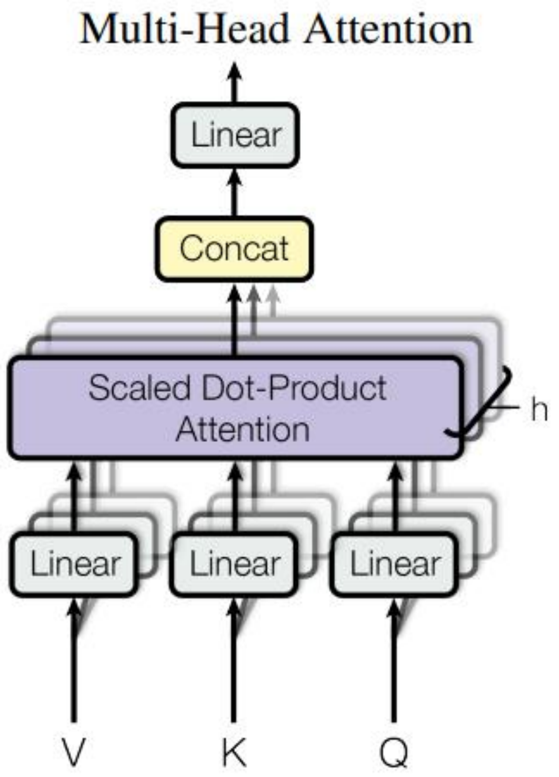


Figure 16: Architecture of multi-head attention (Vaswani et al., 2017)

1.4.2. Transformers:

The article "Attention Is All You Need" introduces a new architecture called Transformer. As the title of the article indicates, it uses the attention mechanism that we have seen previously. Transformer is an architecture for transforming one sequence into another using two parts (encoder and decoder), but it differs from the previously described encoder-decoder models because it does not involve recurrent networks.

The team that presented the paper proved that an architecture featuring only attention mechanisms without any RNNs (recurrent neural networks) can improve results in translation and other tasks.

The transformer architecture is as follows:

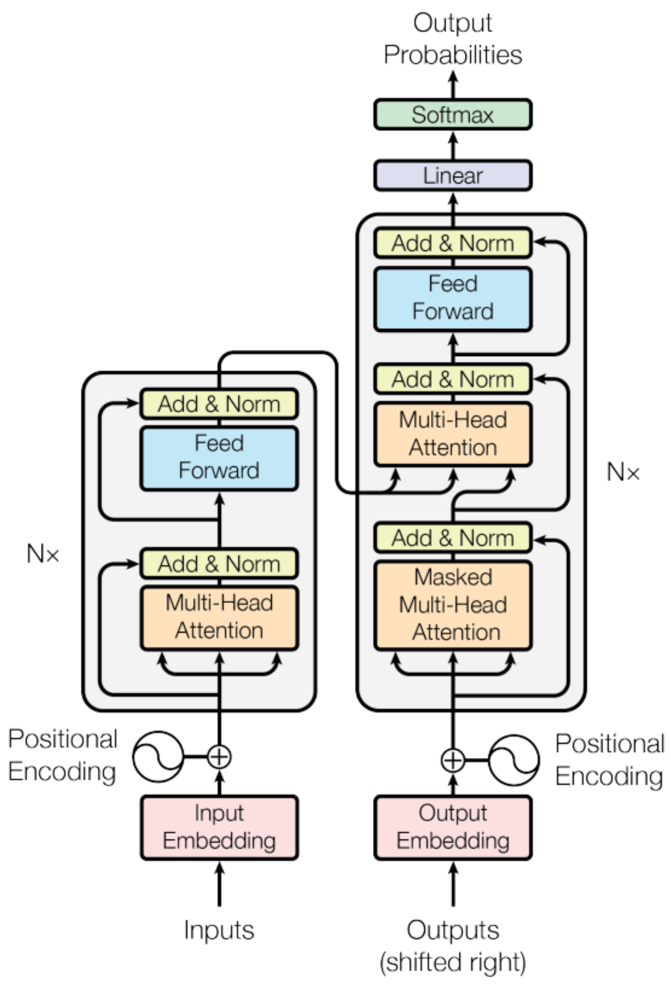


Figure 17: Architecture of a transformer (Vaswani et al., 2017)

The encoder is on the left and the decoder on the right. Both encoder and decoder are composed of modules that can be stacked multiple times on top of each other, which is described as **Nx** in the figure. We see that the modules mainly consist of multi-head attention layers and Feed Forward layers. The inputs and the outputs (sentences) are first integrated in an n-dimensional space since we cannot directly use character strings.

The transformer does not use recurrence or convolution, so we can perform the calculations of several modules simultaneously. However, the position of the words in the sentences will be lost in this case. To remedy this problem, position embedding is introduced, which allows the program to learn the position of each word.

1.4.3. Vision Transformer

While the Transformer architecture has become the highest standard for tasks involving natural language processing (NLP), its computer vision (CV) use cases remain few. In computer vision, attention is either used in conjunction with convolutional networks (CNNs) or used to override certain aspects of convolutional networks while keeping their entire composition intact. However, after the publication of the article "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale", it turned out that this dependency on CNNs is not mandatory, and a pure transformer applied directly to sequences of image patches can perform exceptionally well on image classification tasks.

The Vision Transformer (ViT) is a visual model based on the architecture of a transformer originally designed for text-based tasks. The ViT model represents an input image as a series of image patches, like the series of word embeddings used when using text transformers, and directly predicts class labels for it. The ViT exhibits extraordinary performance when trained with enough data, surpassing the performance of a similar state-of-the-art CNN with four times less computing resources.

The operating steps of a ViT are as follows:

- Patch Partitioning

In computer vision, each patch is assigned to its own query vector, key and value. The image is partitioned into a sequence of patches (usually 16x16) which will then be flattened to have a vector. Then, we apply a linear projection to these vectors:

$$\boldsymbol{z} = \boldsymbol{W x_i} + \boldsymbol{b}$$

- Position Encoder :

In order to avoid the loss of the information of the positions of the patches, we apply the position embedding. Unlike RNNs, transformers allow calculations to be performed on different patches simultaneously, so we must include the embedding position to allow the program to learn the position of each patch :

$$\boldsymbol{PE_{pos,2i}} = \mathbf{sin}\left(\frac{\boldsymbol{pos}}{\mathbf{1000}^{\frac{2i}{d_{model}}}}\right)$$

$$PE_{pos,2i+1} = \cos\left(\frac{pos}{1000^{\frac{2i}{d_{model}}}}\right)$$

- Class embedding :

The class embedding is the introduction of a token, calculated according to the different patches containing the essential information of the image at the beginning of the sequence of patches. A Learnable Class Integration (CLS) is added to the sequence of integrated patches. We use this class embedding to predict the output.

- Training of the model by a supervised classification.
- Fine tuning of data for image classification

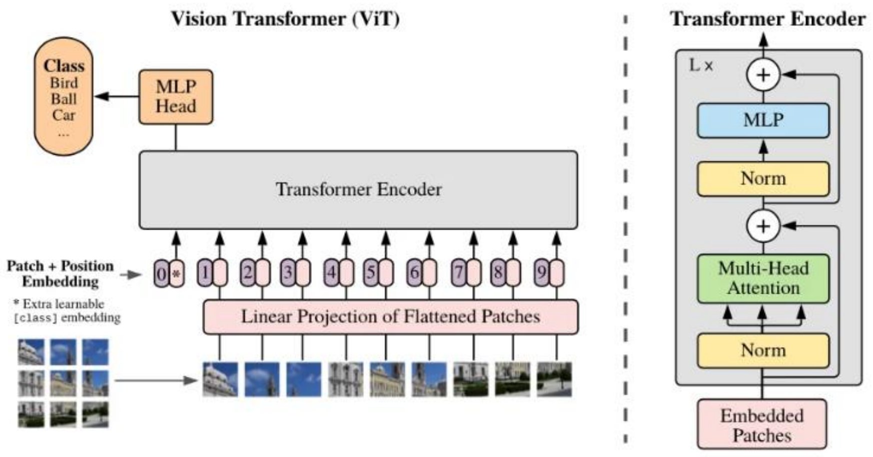


Figure 18: Architecture of ViT (Dosovitskiy et al., 2021)

1.4.4. Examples of architectures based on the transformer

a. Attention U-Net :

Attention U-Net is a deep learning model based on the U-Net structure [Ronneberger et al. 2015]. The U-Net Attention mechanism is created by adding an "attention gate AG" to jump connections in the U-Net model.

Instead of concatenating each oversampling layer in the expansion phase with the appropriate contraction phase layer, the oversampling layer is concatenated with the output of the attention mechanism, a function of the oversampling layer, the pre-sampling and the aforementioned contraction phase layer [John et al., 2022].

The advantage of U-Net Attention is its much reduced complexity compared to U-Net. Previous applications of Attention U-Net have shown it to perform better than U-Net in image segmentation. U-Net Attention represents the state of the art in terms of efficiency and performance for semantic segmentation.

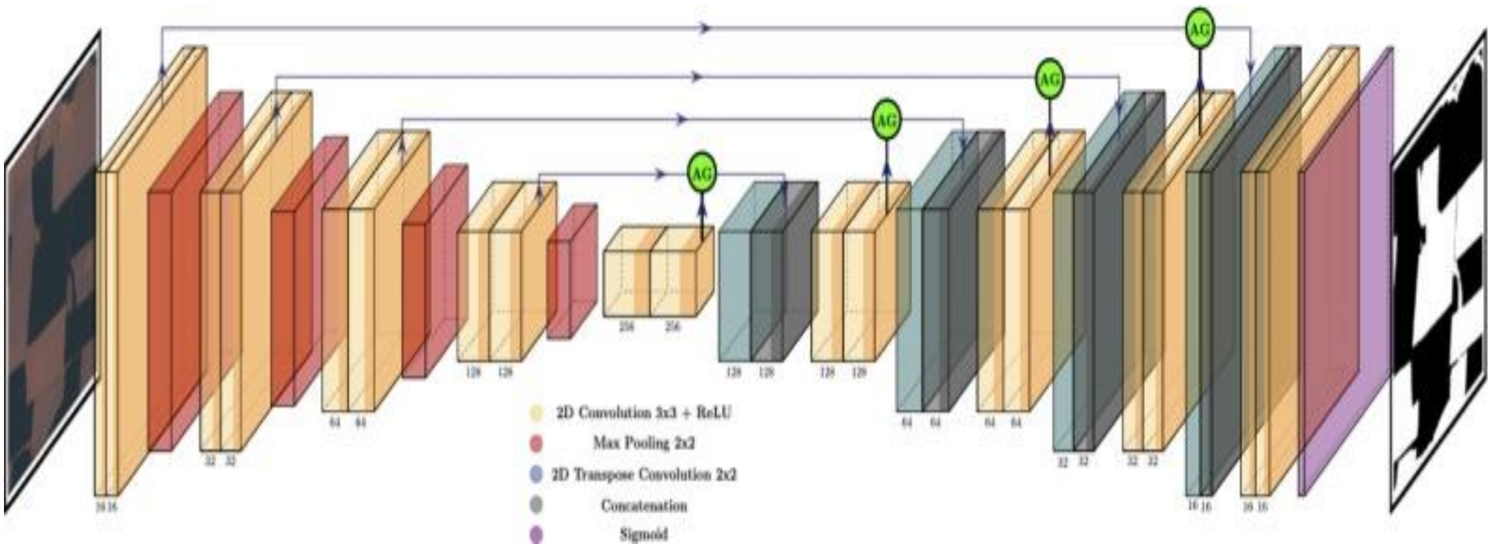


Figure 19: Architecture of Attention U-Net

b. Trans U-Net :

In order to compensate for the loss of resolution of the characteristics brought by the transformers, Chen et al. (2021) proposed the TransU-Net model which uses a hybrid CNN-Transformer architecture to exploit both the detailed high-resolution spatial information of CNN features and the global context encoded by the transformers. This design preserves the benefits of transformers and also benefits from image segmentation.

The CNN unit first extracts the global features from the image to produce a feature map. Next, the feature map is partitioned into patches that will serve as the input sequence into the Transformer unit.

The decoder, called “Cascade UpSampler”, consists of several upsampling layers in order to generate a mask having the same size as the initial image.

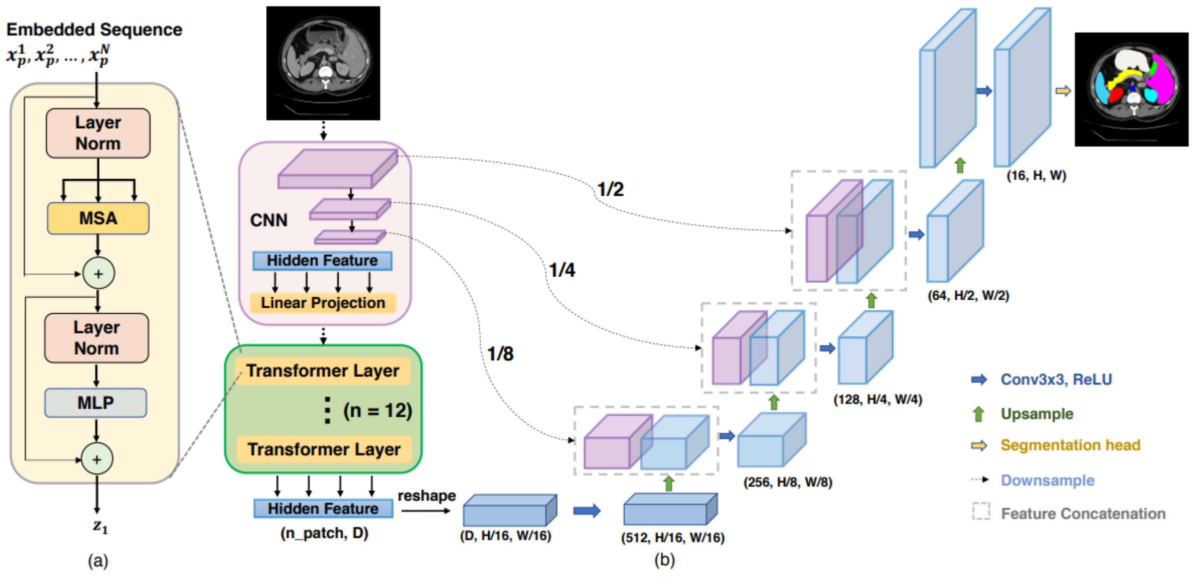


Figure 20: Architecture of Trans U-net (J.Chen et al. 2021)

2. Conclusion :

The introduction of deep neural networks in image analysis and processing has made it possible to automate several heavy tasks such as image classification and segmentation. These networks differed from each other by the number, nature and chronology of the layers. In 2020, the notions of attention mechanism and vision transformers were introduced. Combining these mechanisms with simple convolutional neural networks produces impressive results in natural language processing. A study on the contribution of these mechanisms in the classification and segmentation of images is necessary.

CHAPITRE 2 : GENERAL INFORMATION ON FOREST FIRES

1. Introduction :

In recent years, wildfires have severely damaged forests, wildlife habitats, farms, residential areas and ecosystems. Generally speaking, fires occur in the summer, because it is the times of year most vulnerable to forest fires because the combined effects of drought and low soil water content that favor the outbreak of fires.

To fight against forest fires, the implementation of an approach to anticipate the spread of fires will be essential to reduce losses.

We will see in this chapter how fires spread and their different aspects for a good understanding of the phenomenon. We will then focus on the different methods of detecting forest fires and we will focus on the method of this project: forest fire detection using deep learning.

2. General Information on Forest Fires:

2.1. Propagation of Forest Fires

First, a forest fire is a fire that spreads over a minimum area of one hectare, in one piece. The notion of "forest fire" does not only refer to forests in the strict sense, but also to formations under forests: maquis (low, closed and dense plant formation, growing on siliceous soils), scrubland (low but rather open and growing on calcareous soils) and moors (plant formations on acid soils, composed of broom and small shrubs).

For ignition and combustion, three factors must be respected, each in appropriate proportions: the fuel, which can be any object that can burn, an external heat source (flame or spark) and oxygen, which is needed to feed the fire.

The spread of fire is the result of three main processes that take place simultaneously:

- Combustion of solid particles, which releases the heat

- the transfer of the heat emitted and its absorption by the unburned fuel
- Inflammation

The transport of the heat emitted by combustion is ensured by three processes:

- ➢ The Conduction :

Conduction is the transfer of heat within the object. Heat is transferred from the high temperature area to the low temperature area.

- ➢ Thermal Radiation :

Thermal radiation is the propagation of energy in the form of electromagnetic waves. In other words, this phenomenon corresponds to heat transfer without mass transfer.

This thermal radiation can cause fires to spread through heat moving through the air. This heat can dramatically increase the temperature of the fuel at a distance, causing it to flare up.

- ➢ The convection :

Convection is a transfer of heat through a fluid (liquid or gas). It plays an important role in a fire and can be of two types: natural or forced.

Natural gravitational convection is associated with the slope of the terrain, which varies the temperature profile and with the transfer of energy favored by the presence of hot zones on fire. For forced convection, it is the movement of air masses due to the wind which is an aggravating factor in the spread of forest fires.

2.2. Types of Forest Fires :

From the ground to the sky, the vegetation can be divided into four distinct strata:

• The muscinal stratum: made up of mosses and lichens on the ground and which does not exceed a few centimeters in height, very flammable, and difficult to detect.
• The herbaceous stratum: consisting mainly of annual plants, highly flammable, the wind can spread fire over large areas.
• The shrubby stratum: made up of woody plants of low height (less than 12 meters), of average flammability, it quickly transmits fire to the upper strata.

• The tree layer: trees taller than 12 meters, rarely the source of a fire, it nevertheless allows the spread of flames when it is reached.

Depending on these types of vegetation and their characteristics, as well as the climatic conditions, fire can take different forms, we distinguish:

- Ground Fire :

This type of fire burns the organic matter contained in litter, humus or peat bogs. Powered by incandescence with combustion, their propagation speeds are low.

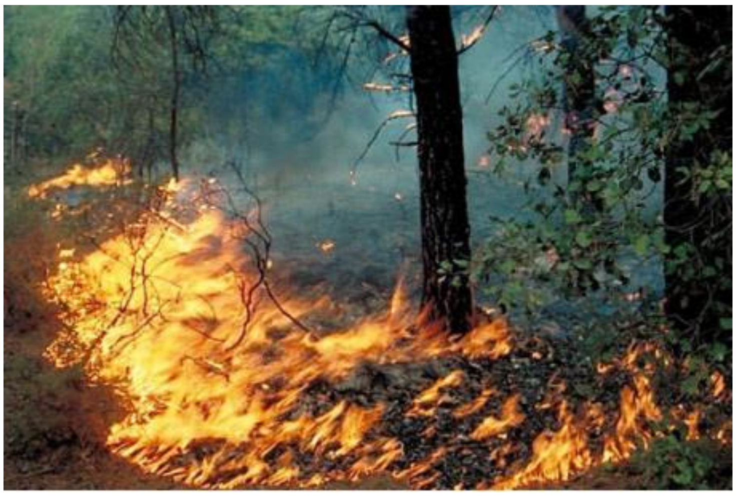


Figure 21: Ground Fire Illustration

- Surface Fire :

These are fires that spread through the undergrowth of forests. They burn the upper part of the muscinal stratum, the herbaceous and shrubby stratum. They are the most common fires and spread quickly.

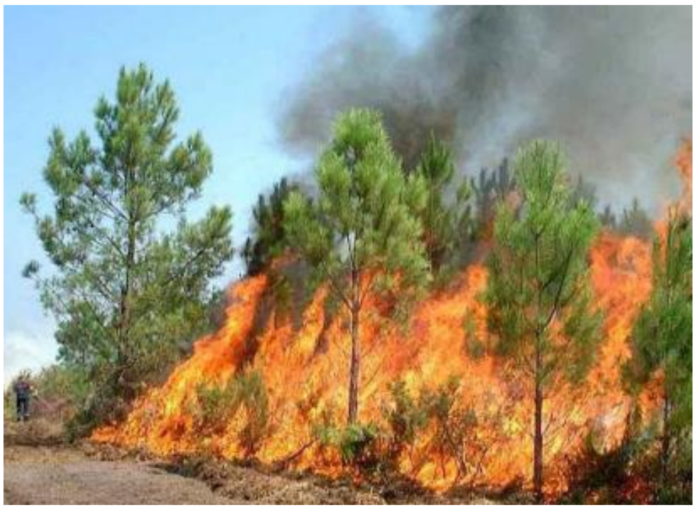


Figure 22: Surface Fire Illustration

- Treetop Fires :

They spread through the tree layer and form a crown of fire. They generally release large amounts of energy and their speed of propagation is very high. They are all the more intense and difficult to control when the wind is strong and the fuel is dry.

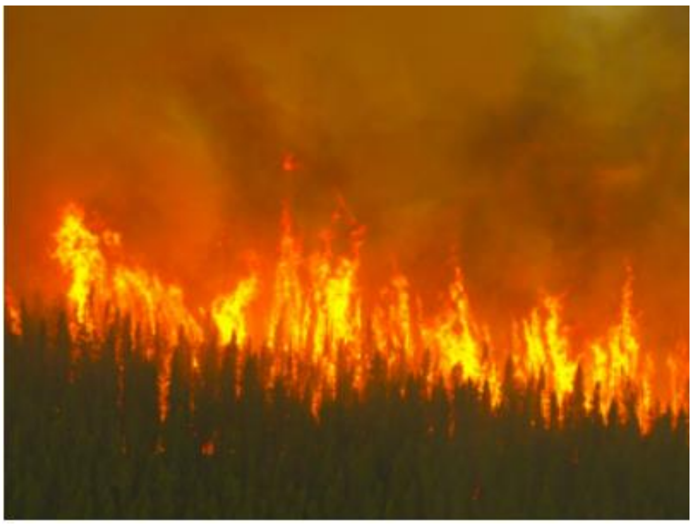


Figure 23: Treetop Fire Illustration

2.3. Causes and consequences of forest fires:

A few decades ago, forest fires were generally a natural activity caused by rare natural phenomena such as a volcanic eruption or an earthquake produced in very specific geographical areas. But today these natural causes are much less frequent and now give way to human activities, whether accidental or not (cigarette butts, badly controlled barbecue fire, work, etc.).

As for their effects, forest fires can generate devastating environmental, social, economic and other consequences. Indeed, forest fires threaten humans and these infrastructures but also the environment. They represent an ecological danger by killing many animals, they destroy the habitats of fauna and flora which affects their population and their life cycle, they increase the levels of carbon dioxide in the atmosphere, contributing to the greenhouse effect and climate change.

3. Methods used in the literature for fire detection:

Fire detection is becoming more and more attractive due to its significant application in monitoring systems of different industries such as military, social and economic security. Several previous studies have dealt with the issue of fire detection in real time by processing images from digital cameras. Their approaches are different due to several aspects. Indeed, there are traditional methods which are generally based on the stationing of personnel in watchtowers, or the use of visual and infrared images from helicopters or fixed-wing aircraft to be able to monitor fires. Other methods consist of the use of satellite imagery which provides a portrait of the situation of fires over a large territory, on which we then base ourselves to develop models.

There are also methods based on modeling the spread of fires by processing aerial images, taking into account the parameters that influence this spread such as: the topography of the land, the wind, the density of vegetation. Finally, there are new methods based on deep learning and neural networks for fire detection using unmanned aerial vehicles. The number of articles on fire detection in the literature is growing rapidly. In the following, we will first present a general overview of conventional studies based on fire feature extraction and fire spread models, then we will focus on deep learning detection methods.

3.1. Previous Studies:

Q. Huang et al. (2020) have developed a method based on the processing of aerial images from several sources (satellites, aircraft, UAVs), in order to extract the distribution of vegetation in the study area and be able to combine it with a model that takes environmental factors into account for more accurate detection of fire spread. The processing process consists of extracting aerial images of the region's plant profile, and then converting it to a grid of cells called cellular automata format. To do so, they followed the following steps:

→ Aerial images were taken to perform vegetation segmentation by choosing points within the vegetation areas as reference points, thus classifying the other points with pixel values close enough to the reference point values as vegetation.

→ an I matrix was generated by combining color space rules including RGB rules and hue, saturation, intensity (HSI) rules with frequency characteristics.

→ The image is segmented into vegetation relative to the rest using a similarity measure using the Mahalanobis distance (Benaichouche et al., 2013). The result of the segmentation is a binary image that represents the distribution of vegetation in the region.

Once the vegetation distribution is obtained, it is combined with the principle of heat conduction in the Rothermel model. A model that can indicate the process and state of the spread of forest fires, and which is built based on the parameters that clearly affect the spread of fire, namely: the speed and direction of the wind, the rate of spread of the fire fuels, vegetation density and slope.

Rothermel's surface fire spread model is based on conservation of energy, assuming fire spread matches the Huygens model (L. Cunbin et al., 2014). Based on energy conservation, Huang et al. (2020) carried out heat conduction mechanisms on the modeling of the spread of forest fires which represent a set of rules according to the state of the vegetation (OF: on fire, BO: burnt, NF: no fire). For example, one of these rules states that if the current state of a cell is $n(i, j, t) = OF$, its next state is $n(i, j, t + 1) = BO$ and so on. And to improve the model, some rules take into consideration the probability of impact of topology, slope, wind, and vegetation density.

- The influence of the topography:

Several models suggest that topography is an important factor affecting fire behavior. Indeed, a flat terrain or a positive slope can contribute to accelerate the spread of fire, while an environment with a rough terrain with negative slopes can slow down the spread of fire (Q.Huang et al., 2020).

- The Influence of the wind :

The wind can act in several ways on the spread of fire. In fact, it brings oxygen thus activating the combustion, it lowers the flames on the vegetation, and it modifies the direction of the fire. To model the relationship between fire and wind, a probability of fire spread was calculated based on wind direction and the state of neighboring cells (Q.Huang et al., 2020).

- The influence of Vegetation density:

Several models ignore the influence of vegetation density since they assume a constant base probability. Yet the distribution of vegetation is not even, where denser vegetation not only increases the rate of spread, but also provides more fuel thus increasing burn time. The extraction of the real vegetation density carried out made it possible to integrate this factor into the modeling through a linear relationship that links this density and the probability of propagation (Q.Huang et al., 2020).

The results of this model were applied to the 'RIM fire', which took place in the United States, to compare them with the fire in reality between August 19 and 27, 2013.

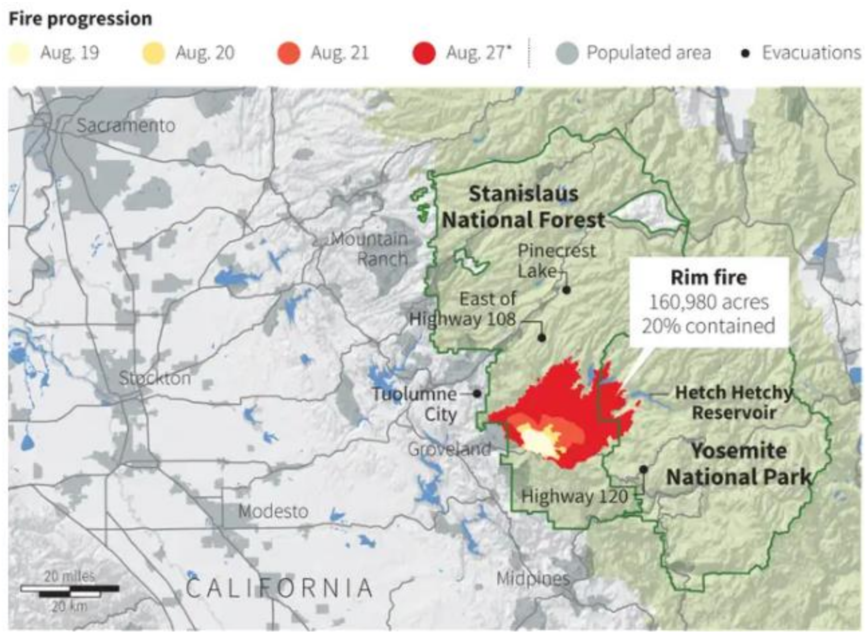


Figure 24: The spread of the 'RIM fire' between August 19 and 27, 2013 (businessinsider.com)

The simulation results in Figure 24 illustrate estimated fire perimeters at different time points that are similar to the fire progression in Figure 25. This correspondence verifies the accuracy of the model. The high alignment between modeling results and reality is consistent. Indeed, they showed the fire region for four time points (64, 106, 181, and 545) in the simulation in Figure 25, which shows consistency with the four time points in Figure 24 capturing the fire region on the 19th August, 20th August, 21st August and 27th August.

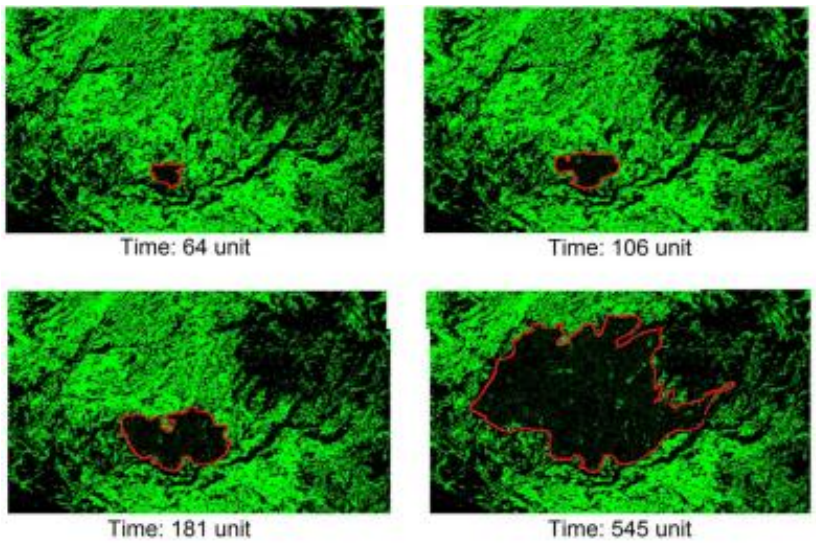


Figure 25: Prediction of fire regions (Huang et al., 2020)

Lin et al. (2011) proposed a fire detection method based on several processing steps of aerial images from RGB cameras in order to achieve fully automatic fire monitoring, while taking into account the static and dynamic characteristics of fires. Indeed, fire exhibits unique visual signatures which can be divided into static and dynamic types.

The static visual characteristics of fire include both spectral information and spatial structure. The spectral information mainly refers to the brightness and color of the flame, to which they added the spatial structure that presents the shape and size of the flame. These characteristics change constantly to avoid erroneous detection of sources having the same color as the lights. Taking into account the static and dynamic characteristics of fire, Mingxiu Lin et al. (2011) used in their approach a decisional strategy in several stages, the latter consisted in considering criteria coming from the characteristics of the flame, namely:

- The moving region of the flame;
- Color segmentation;
- The random nature of the size of the zone;
- The likelihood of the edges;
- The pointed corner;

- Circularity;

Based on an analysis of these characters, the authors developed an intelligent system of the proposed strategy. This system allows using an algorithm, first, to process the image in gray level using a time difference to obtain the moving region, then to apply a segmentation of the colors in order to extract the regions. Image interest, then calculates shape characteristics such as area size, and edge likelihood, and finally uses polygonal and irregular flame characteristics such as sharp corners and circularity to identify the fire.

They applied the algorithm to images containing fire and light to test its performance. The results showed that the decision strategy was able to distinguish between environmental light changes, background color interference, and misidentification of light.

3.2. Deep learning methods for fire detection:

With the advent of artificial intelligence technologies, and the development of the field of computer vision, more particularly those of deep learning and their applications in the fight against forest fires, several articles dealing with this subject have seen the in recent years in order to provide solutions to this problem which affects several sectors.

The use of UAVs with visual and infrared cameras has in turn aided in the accurate management of fires by providing real-time imagery at low cost, and significant ability to cover large areas. Also, their integration with DL techniques have seen remarkable progress, especially CNN architectures which have shown great performance in image processing tasks.

Yi Zhao et al. (2018) developed in their article "Saliency Detection and Deep Learning-Based Wildfire Identification in UAV Imagery" a method for localizing and segmenting fires by combining two approaches, namely: salient segmentation as a data augmentation tool, and the 'Logistic Regression' classifier which is a binary classification method. They carried out their approach in two main steps. First, a region of interest (ROI) is extracted from the images using salient segmentation, in order to calculate the color and texture characteristics of these ROIs. Then, apply two independent 'Logistic Regression' classifiers to be able to determine if the ROI belongs to fire or smoke, in order to segment it afterwards. The model used in this method is Fire_Net of deep convolutional neural network "DCNN", its architecture is shown in the figure below:

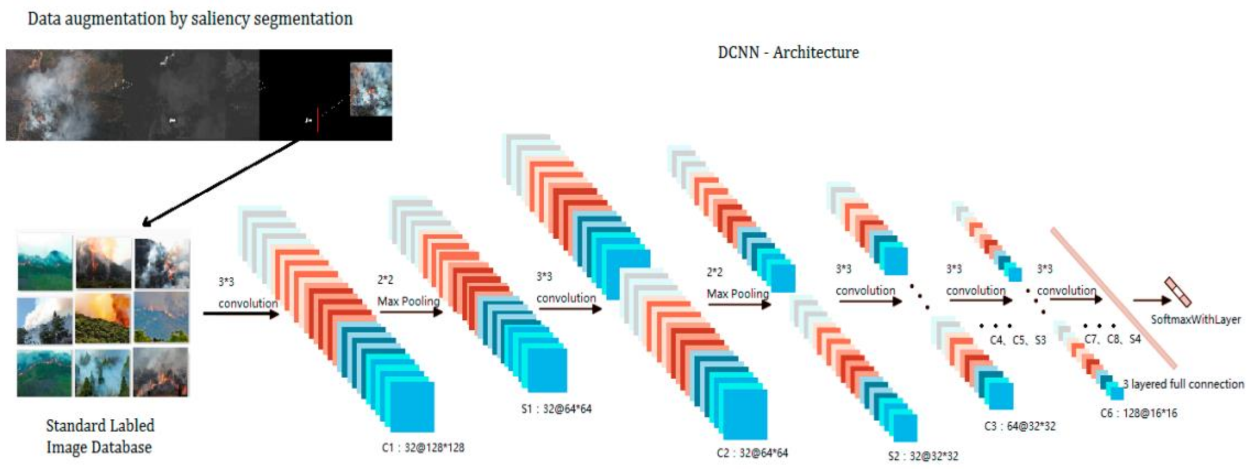


Figure 26: Architecture of Fire_Net (Yi Zhao et al., 2018)

To review their approach, Yi Zhao et al. (2018) tested their model on the database before and after augmentation by salient segmentation to assess its performance. In fact, they achieved a 1% improvement in the accuracy of their results, ranging from 97% without an increase to 98% after segmentation.

Rafik Ghali et al. (2022) also proposed the use of CNN-based methods by exploiting high-resolution RGB images captured by drones for the classification of images into two classes; “Fire” and “Non-Fire”, followed by segmentation to detect fires. For fire detection and classification, they used a hybrid model by combining two models: EfficientNet-B5 and DenseNet-201 according to the following architecture:

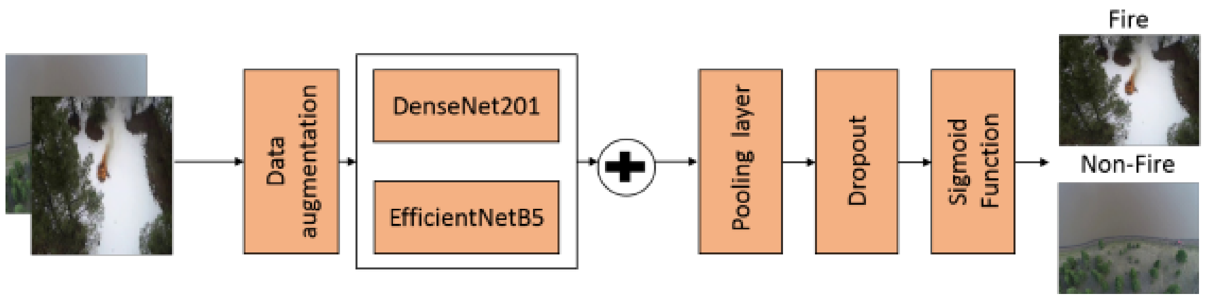


Figure 27: Classification Architecture used by (Rafik Ghali et al., 2022)

They then carried out segmentation by exploiting a CNN model, EfficientSeg, and two processors, TransUNet and TransFire:

- EfficientSeg : which is a semantic segmentation method based on the U-Net architecture

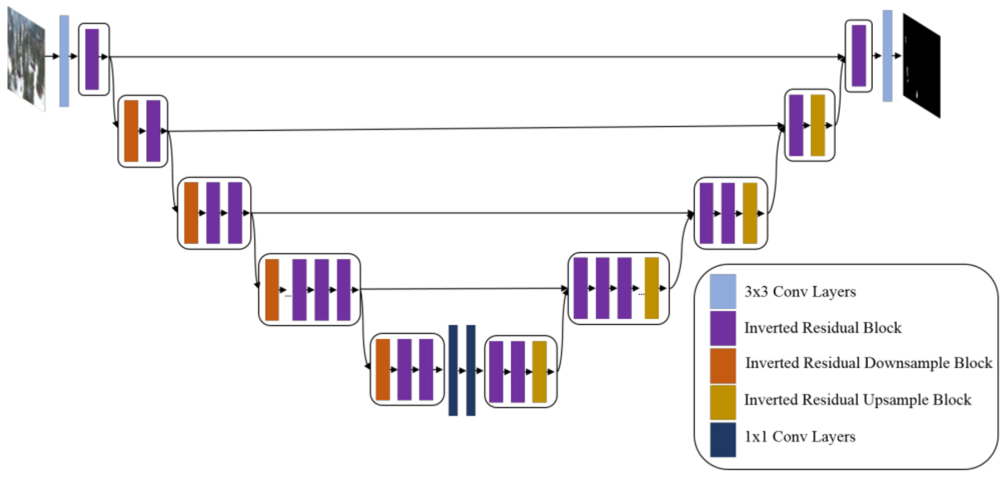

3x3 Conv Layers
Inverted Residual Block
Inverted Residual Downsample Block
Inverted Residual Upsample Block
1x1 Conv Layers

Figure 28: Architecture of EfficientSeg (Rafik Ghali et al., 2022)

- TransU_Net: which is a transformer based on the U-Net architecture and which uses the dependency between input and output data.

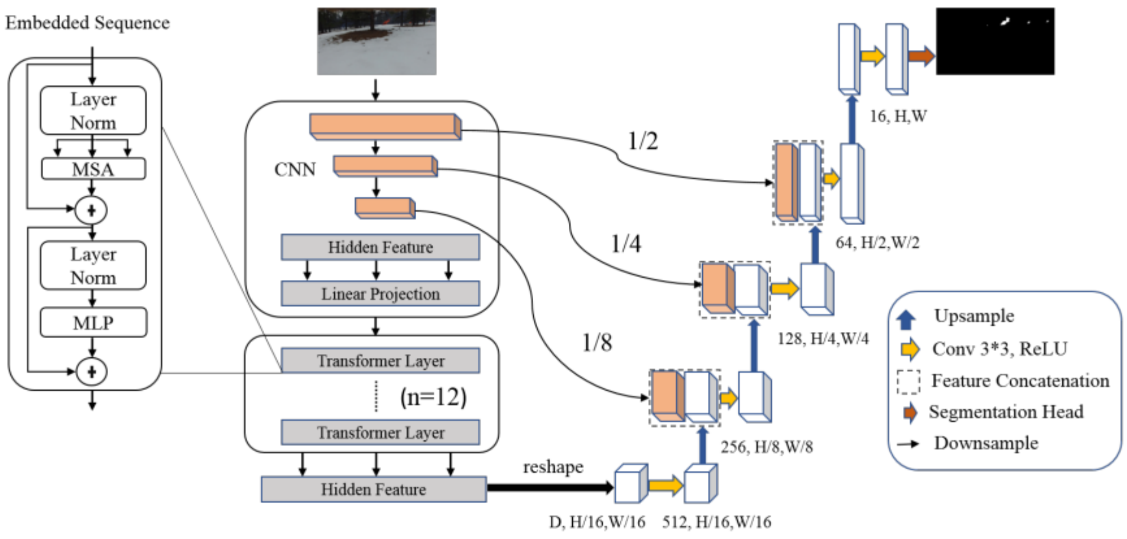


Figure 29: Architecture of TransUNet (Rafik Ghali et al., 2022)

- TransFire: is a model based on the MedT architecture (Medical Transformer) which is a method of segmenting medical images and which does not require a large database.

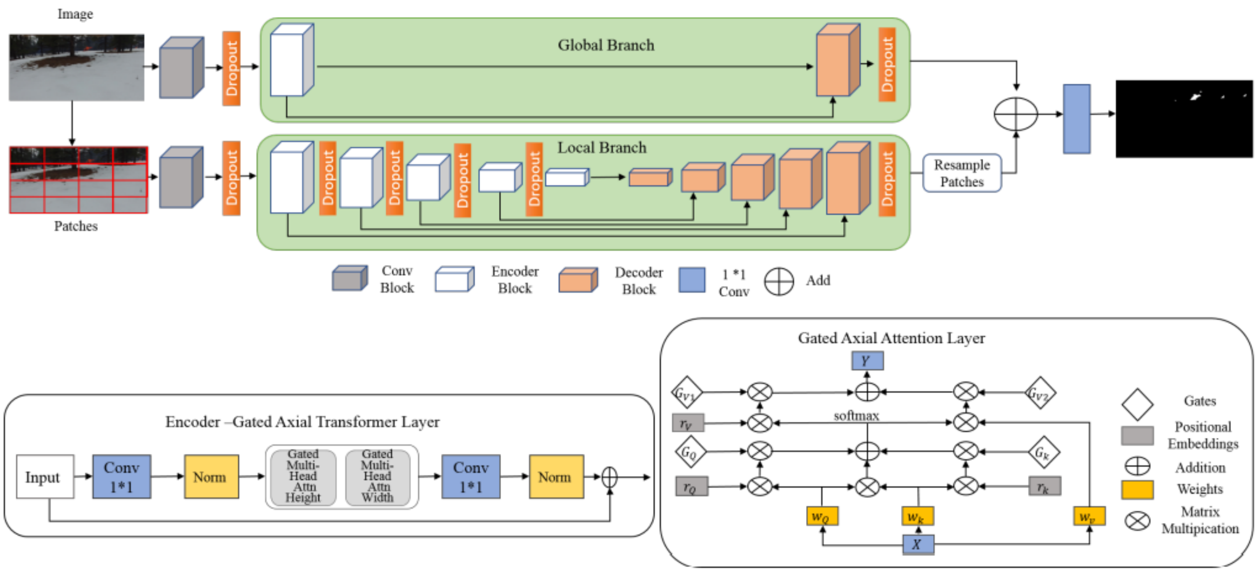


Figure 30: Architecture of TransFire (Rafik Ghali et al., 2022)

The analysis of the classification by the hybrid model adopted showed better results compared to the use of each model separately, resulting in an accuracy of 85.12% in a calculation time of 0.018s thus allowing a better classification of fires. Concerning the segmentation, by comparison of the three methods, it is the TransUNet model which gave the best results with an average precision of 99.90% in a calculation time of 0.29s, which allows a more effective separation between the fire and the other details in the picture.

4. Conclusion :

In recent years, several previous studies have dealt with the problem of detecting and delimiting forest fires in time. Their approaches differ widely due to several factors, such as: the image acquisition device and the types of sensors with which they are associated as well as the methods of processing the acquired images. Several detection methods are used, such as conventional methods based on fire characteristics, fire spread models and deep learning detection methods. Considering all the works presented in this chapter, the integration of drone imagery in deep learning techniques has brought new solutions and answers by providing good results for the task of fire detection.

Due to this multitude of deep learning approaches and architectures that can be applied, we have focused our present work on the classification and segmentation of fires via convolutional neural networks which have shown their performance in these tasks.

CHAPITRE 3 : METHODOLOGY AND MATERIALS

1. Introduction :

This chapter is a detailed presentation of the methodology followed for the detection of forest fires by classification and their delimitation by segmentation of drone images using deep learning. We will first present the work environment and the data used. Next, we will discuss two main sections. The first is a presentation of the CNN networks that will allow the classification of images into 'Fire' and 'No Fire'. The second section consists in exploring the different image segmentation architectures in order to choose the best one between them.

2. Methodology :

The methodology that we propose for the automatic detection of forest fires by drone consists of a succession of implementation steps. First, a classification of the acquired images is essential to be able to know if there is fire in the site or not. Then, if the classification returns the presence of fire, then comes the stage of semantic segmentation. Finally, the resulting models will be embedded in a mobile application. The flowchart (Figure 31) illustrates the main steps followed in our methodology.

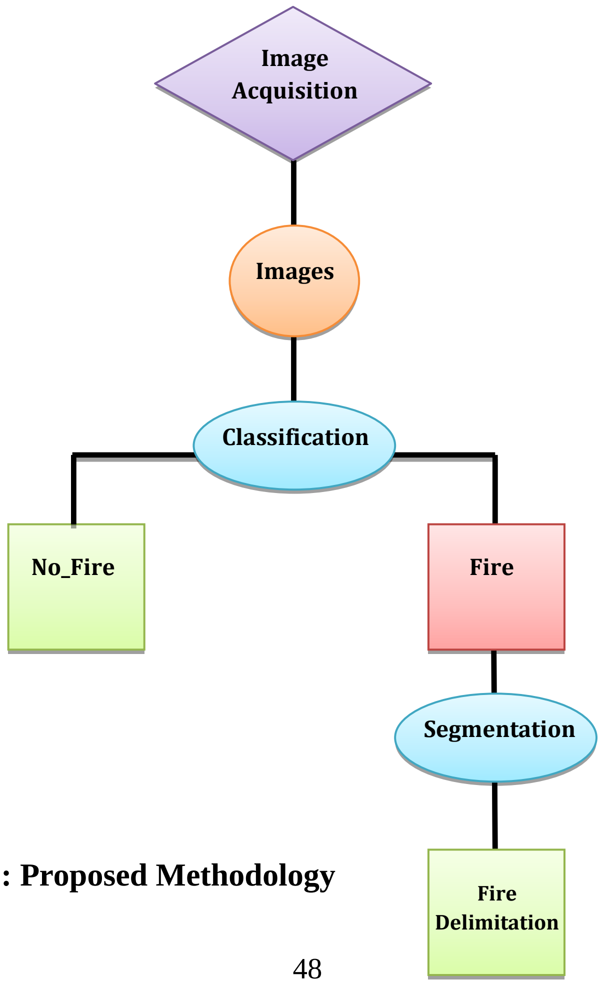


Figure31: Proposed Methodology

3. Working Environment :

The realization of this work requires the use of programming using deep learning models. As a programming language, we opted for python and the code was executed at the Anaconda Prompt terminal and the 'jupyter notebook'.

- Python : it is an interpreted cross-paradigm and cross-platform programming language. It promotes structured, functional and object-oriented imperative programming. Python is one of the most used programming languages in Deep Learning thanks to its high-level tools and its easy-to-use syntax.

- Jupyter Notebook : it is a web-based interactive programming environment for creating Jupyter Notebook documents. The term "notebook" can refer to many different entities such as Jupyter web application, Jupyter Python web server, or Jupyter document format. A Jupyter Notebook document is a JSON document. It follows a pattern containing an ordered list of input/output cells. These can contain code, text (using Markdown), mathematical formulas, graphics, and interactive media. This document usually ends with the ".ipynb" extension

4. Data Acquisition :

The input data are images from the FLAME (Fire Luminosity Airborne-based Machine learning Evaluation) dataset. The images in this dataset were captured by drones during a prescribed burn of a pile of trees and straws in the state of Arizona in the United States. These images were divided into two directories, one for image classification and another for semantic segmentation.

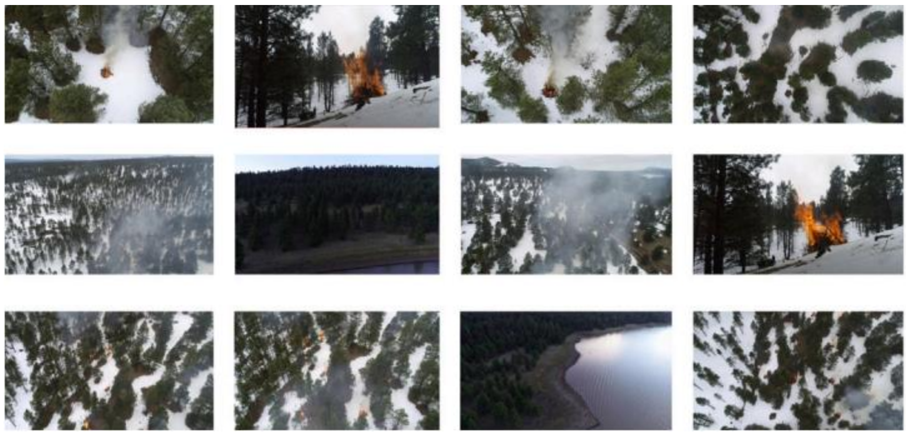


Figure 32: Images' Extract of FLAME Dataset

Regarding the acquisition equipment, Table 2 presents the drones and cameras used in this acquisition mission.

Table 2: Drones used for image acquisition

Phantom 3 Professional, DJI.	
Matrice 200, DJI.	
Zenmuse X4S, DJI.	

5. Classification Methodology :

5.1. Data Presentation :

- Data Structure :

Image classification models require three types of data:

- ✓ Training data: These are the labeled images that will be processed first to allow the model to "learn", i.e. to optimize its parameters.
- ✓ Validation data: These are images different from the training one; they make it possible to ensure the optimization of the learning process by modifying the hyper-parameters of the model.
- ✓ Test data: These are labeled images used to evaluate the performance of the model.

The images for classification have been structured as follows:

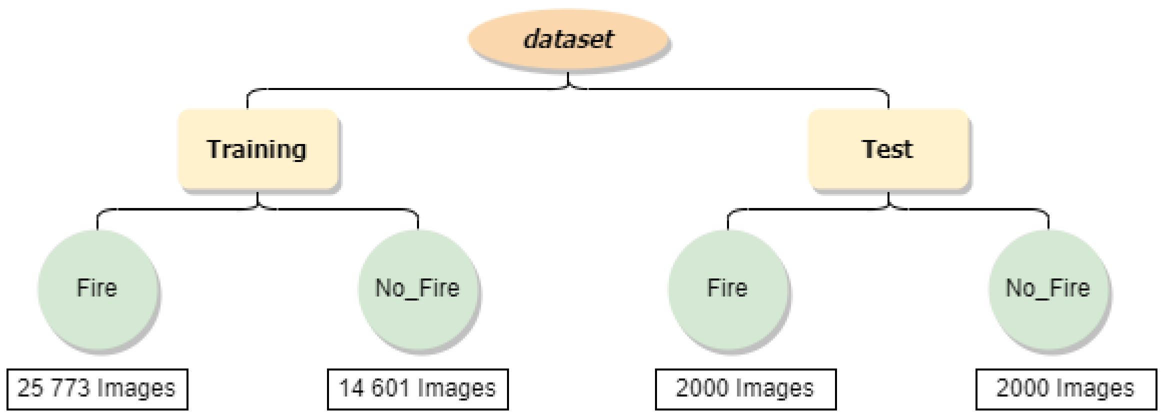


Figure 33: Structuration of Classification Data

The classification we want to perform is a supervised classification. So the images must be labeled. Instead of creating a second folder containing the label of each image, we put the images of each class in a subfolder; the label in this case is the name of the subfolder: Fire/No_Fire.

5.2. CNN Models used :

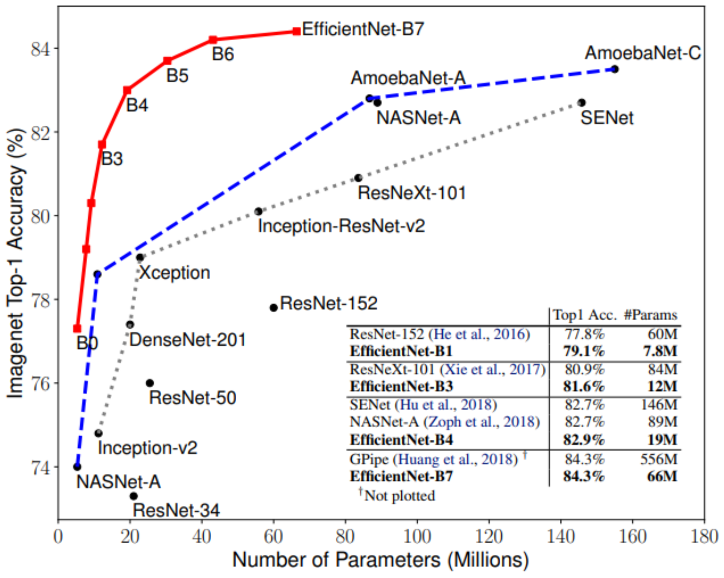
5.2.1. Xception :

It is a convolutional neural network architecture entirely based on separable convolution layers. It has 36 convolution layers that form the basis for extracting network features. These layers are structured into 14 modules, all of which have residual linear connections around them except for the first and last module. Each convolution layer is followed by a logistic regression layer. In short, the Xception architecture is a linear stack of separable convolution layers with residual connections. This makes the architecture very easy to define and modify.

5.2.2. EfficientNetB2 :

EfficientNetB2 is a model introduced by M.Tan et al. in 2020. Its architecture is based on the 'Compound Scaling' method. Unlike conventional practice which arbitrarily scales the architecture factors, this model uniformly scales network width, network depth, and resolution with a set of fixed scaling coefficients. The input images in our project are large in size, so the compound scaling method makes sense as the array needs to have a higher number of layers to increase the receptive field and a higher number of channels to capture patterns thinner on larger images.

Figure 34 summarizes the performance of ImageNet models, where EfficientNet architectures significantly outperform other image classification models.



	Top1 Acc.	#Params
ResNet-152 (He et al., 2016)	77.8%	60M
EfficientNet-B1	**79.1%**	**7.8M**
ResNeXt-101 (Xie et al., 2017)	80.9%	84M
EfficientNet-B3	**81.6%**	**12M**
SENet (Hu et al., 2018)	82.7%	146M
NASNet-A (Zoph et al., 2018)	82.7%	89M
EfficientNet-B4	**82.9%**	**19M**
GPipe (Huang et al., 2018) †	84.3%	556M
EfficientNet-B7	**84.3%**	**66M**

†Not plotted

Figure 34: Performance of ImageNet models (Tan et al., 2020)

6. Segementation Methodology :

This step consists in segmenting the images belonging to the class ‘Fire’, in order to generate their binary masks presenting the location and the shape of the fire. This segmentation will first be carried out based on the U-Net architecture, then using the attention mechanisms by Attention U-Net, then the integration of transformers in Trans U-Net.

6.1. Data Preparation :

In order to train our models, we have a database containing 2003 pairs of images and masks of dimension 3480x2160. The couples are organized in the form of the folders below, thus making it possible to separate the training couples from those of validation and test:

Segmentation Folder :

- **Training :**
  ├── Images/*.jpg
  ├── Masks/*.png
- **Validation :**
  ├── Images/*.jpg
  ├── Masks/*.png
- **Test :**
  ├── Images/*.jpg
  ├── Masks/*.png

6.2. Adopted Methods :

In order to train our models, we have a database containing 2003 pairs of images and masks of dimension 3480x2160. The couples are organized in the form of the folders below, thus making it possible to separate the training couples from those of validation and test.

6.2.1. U-Net :

The choice to use the U-Net architecture as a training model is motivated by several reasons. U-Net is a network that has a simple architecture and easy to implement, but also very robust to make good predictions even with a little training data. U-Net's outstanding characteristics are its “encoder-decoder” structure, and the “skip connections”. The encoder block is used to extract features from the image, while the decoder block is used to recover the image to its original size from the extracted features and to produce the final segmentation result. Jump connections combine low-level functionality in the encoder block with high-level functionality in the decoder block (Song-Toan Tran et al. 2021). Regarding the architecture of our model, we chose the U-Net model defined by Samsoshoara et al. in 2020.

6.2.2. Attention U-Net :

Attention U-Net is a model based on the U-Net architecture in which attention blocks (AG) are added as explained in the first chapter. The integration of the attention mechanism highlights the important characteristics transmitted by connection jumps. The extracted information is used to disambiguate noisy and irrelevant responses in connection hops. This process is performed just before the concatenation operation to merge only the relevant activations (Oktay et al. 2018).

6.2.3. Trans U-Net :

Trans U-Net is a Vision Transformer (ViT) based on the U-Net architecture. It uses global dependencies between input and output data by exploiting Self-Attention methods. It is an encoder-decoder; the encoder uses a hybrid CNN transformer architecture composed of VGG16 and ViT to extract the features. It contains MLP and MSA blocks. The decoder uses CUP (Cascaded up-sampler) blocks to decode the extracted features and display the binary segmentation mask. Each CUP includes a 3×3 convolutional layer, a ReLU activation function, and two upsampling operators.

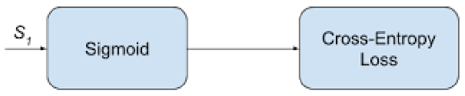
7. Parameters and training functions used

The training of the chosen models is the most important step in our approach, it is during which we define the various parameters which can influence the performance of the models such as: the Framework used, the loss function, the function of optimization, the activation function…

Machines

Framework

Loss Function



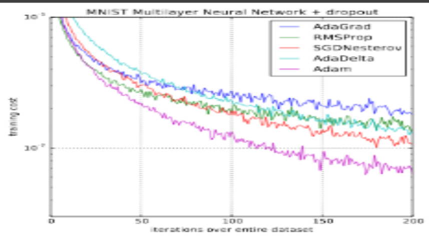
Binary Cross-Entropy

$$CE = -t_1 log(f(s_1)) - (1 - t_1) log(1 - f(s_1))$$

$$f(s_i) = \frac{1}{1 + e^{-s_i}}$$

Optimization Function

Adam Optimizer



Activation Function

Sigmoïd

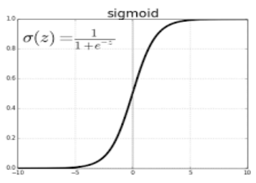


Figure 35: Parameters and Training Functions

7.1. Training Functions

7.1.1. Loss Function

The loss function used is Binary Cross-Entropy (Binary_Crossentropy) since the output nature is binary, fire and background. The choice of this function is also motivated by the fact that it strongly penalizes forecasts that are optimistic but erroneous. In addition, it is compatible with the sigmoid activation function used in this case.

7.1.2. Optimisation Function

We have chosen the Adam optimizer as the optimization algorithm. It is an efficient stochastic optimization method that requires only first-order gradients with little memory. It is robust in the field of machine learning and more efficient in comparison with other optimizers [Diederik and Jimmy, 2014].

7.1.3. Activation Function

As an activation function, we have adopted for the intermediate layers the ReLU function. This is the simplest and most used activation function. It allows only positive values to pass into the following layers of the neural network. Since the Sigmoid function is efficient for binary classification, we used it in the final layer.

7.2. Training Parameters

In order to achieve a high-performance model, and which gives the best results, it is necessary to choose its training parameters carefully. This involves testing different values of the model's hyperparameters and choosing those that give the best accuracy. This approach is called hyperparameter optimization; it can be used for filter count, batch size, epoch count…

7.2.1. Number of filters

The choice of the number of filters is important factor acting on the optimization of our models. To determine it in our case, we chose a fixed value based on similar studies.

7.2.2. Batch size

The batch size is a hyperparameter that describes the number of samples to process before updating the internal model parameters. Several factors go into the choice of batch size, namely: the size of the data, the machine used and its memory capacity.

7.2.3. Learning rate

The learning rate is a hyperparameter that controls the degree to which the model changes in response to the estimated error each time the model weights are updated. To choose it, we opted for the manual choice of a fixed value based on similar studies.

7.2.4. Number of epochs

The number of epochs is the number of complete passes through the training data set. Its choice is related to the size of the database, the depth of the model, and the GPU capacity of the machine used.

8. Validation of proposed models

The evaluation of the models used for the classification and segmentation of forest fires can be based on:

- ✓ Visual Validation:

This is the first validation to be carried out after the end of training. It consists, in the case of classification, in visualizing whether the class assigned to the input image by the model is the real one, and in the case of segmentation, comparing the masks resulting from the models with the real masks.

- ✓ Validation by metrics :

The calculation of the metrics gives us an idea of the precision and the quality of the results obtained by the different models, as it allows controlling the choice of the parameters of the latter. These metrics are mainly calculated based on the TP, FP, FN and TN values explained in the first chapter, and are as follows:

- Cross-validation
- Confusion Matrix
- The recall index
- The precision index
- F1-scoreValidation par jeu de données :

The validation set is used to examine the results of the model by comparing these predictions with the real labels, while evaluating its performance by calculating the above metrics.

9. Conclusion

This chapter presents the methodology adopted for the classification and segmentation of forest fires by comparing several models in order to choose the most efficient. We can draw from this chapter the main methodological steps such as: data preparation, choice of classification and segmentation models, training of these models and their validation in order to choose the most efficient model.

CHAPITRE 4: PRESENTATION AND ANALYSIS OF RESULTS

1. Introduction :

The objective of this chapter is to present and analyze the results obtained in each step in the process of automatic detection of forest fires. It is divided into two main parts. The first is devoted to the results obtained by image classification by comparing the EfficientNetB2 and Xception models, as well as their evaluations based on different metrics. In the second part, we will present the results of the Deep Learning models used for semantic segmentation, namely: U-Net, AttentionU-Net which is a CNN model based on the attention mechanism, and TransU-Net, a hybrid CNN model - Transformer, to choose the most optimal model for our project.

2. Classification Results :

2.1. Training

In this step, we define the different parameters used to train our models such as: the loss function, the optimization function, batch size, etc.

Since the size of the dataset images is 254 × 254, so a pre-processing of the training images is needed. This involves scaling the input images to size 256x256 for the Xception Network training and 260×260 for the EfficientNet model training. This choice of size was based on the architectures of the models, ensuring that this conversion will not degrade the quality of the images and that they are large enough so that the spectral information of the lights remains preserved.

As a loss function, we used the function 'Binary_Crossentropy' since it is a binary classification since in our case there are only two classes 'Fire' and 'No Fire'. Looking at similar studies, we chose a value of 10-3 for the learning rate, and we adopted the Adam optimizer. We've added Keras' ModelCheckpoint function as a callbacks function, which allows saving the model after each epoch runs.

The Xception model was defined layer by layer according to its architecture while the EfficientNetB2 model was imported from Keras Applications which is a set of CNN models available with pre-trained weights on a very large database such as ‘ImageNet’. Table 3 shows the characteristics of the imported EfficientNetB2 model:

Table 3: EfficientNetB2 characteristics (Keras.io)

Model	**Size (Mo)**	**Top-1 Accuracy**	**Top-5 Accuracy**	**Parameters**	**Depth**	**Time (ms) per inference step (CPU)**	**Time (ms) per inference step (GPU)**
EfficientNetB2	36	80.1	94.9	9.2M	186	80.8	6.5

→ Top-1 Accuracy and Top-5 Accuracy refer to the performance of the model on the ImageNet validation dataset.

→ Depth refers to the topological depth of the network. This includes activation layers, batch normalization layers, etc.

→ Time per inference step is the average of 30 batches and 10 repetitions.

The choice of hyperparameters was based on the concept of 'hyperparameter tuning' using the 'Grid Search' method. It is an optimization method (hyperparameter optimization) that will allow us to test a series of hyperparameters and compare their performance to deduce the best setting. This method consists of defining a grid whose row corresponds to the values of one hyperparameter and the column to the values of another, the model makes a prediction for each possible combination of values in the grid to choose the one that gives the best results. In our case, we applied this method on the number of epochs and the size of the batches:

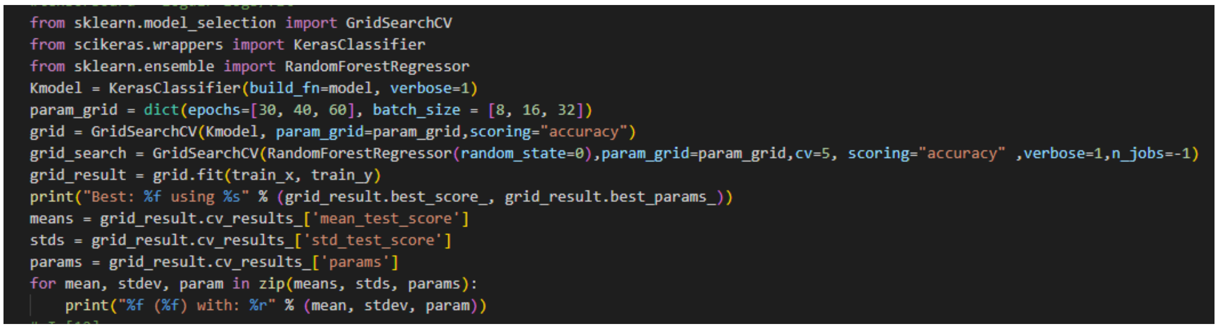
- Number of epochs : [30, 40, 50]
- Batch Sizes : [8, 16, 32]

So the grid created by the ‘grid search’ method is as follows:

Table 4: Combinations of hyperparameters by Grid Search

	8	16	32
30	Combination 1	Combination 2	Combination 3
40	Combination 3	Combination 5	Combination 6
50	Combination 7	Combination 8	Combination 9

The code in Figure 36 shows the implementation of this method in our model:



```
from sklearn.model_selection import GridSearchCV
from scikeras.wrappers import KerasClassifier
from sklearn.ensemble import RandomForestRegressor
Kmodel = KerasClassifier(build_fn=model, verbose=1)
param_grid = dict(epochs=[30, 40, 60], batch_size = [8, 16, 32])
grid = GridSearchCV(Kmodel, param_grid=param_grid,scoring="accuracy")
grid_search = GridSearchCV(RandomForestRegressor(random_state=0),param_grid=param_grid,cv=5, scoring="accuracy" ,verbose=1,n_jobs=-1)
grid_result = grid.fit(train_x, train_y)
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
means = grid_result.cv_results_['mean_test_score']
stds = grid_result.cv_results_['std_test_score']
params = grid_result.cv_results_['params']
for mean, stdev, param in zip(means, stds, params):
    print("%f (%f) with: %r" % (mean, stdev, param))
```

Figure 36: The use of 'Grid Search' method

Once the training process is complete, the ‘grid search’ method returns that the best combination is 40 epochs and a batch size of 32 for the Xception model and 30 epochs with a batch size of 32 for EfficientNetB2.

On peut résumer l’ensemble des paramètres d’entrainement utilisés dans le tableau 5 :

Table 5: Training Parameters

Model	**Loss Function**	**Optimisation Function**	**Learning Rate**	**Batch Size**	**Epochs**
Xception	Binary_Crossentropy	Adam Optimizer	10^{-3}	32	40
EfficientNetB2	Binary_Crossentropy	Adam Optimizer	10^{-3}	32	30

In order to avoid evaluation metrics affected by overfitting, we used the ‘K-folds’ method to perform cross-validation. This method consists of dividing the data set randomly into k groups, the model makes k predictions, each time keeping a different group for validation. This allows to have a less biased model that is not too stuck to the training data. In our case, we chose **K=8** given the large number of images in our dataset.

2.2. Validation of Results

We present in this phase the methods used for the control of the results obtained as well as their analyses. The validation of the classification results obtained is ensured by the test data set reserved for evaluating the performance of the models. This dataset contains 4000 images of which 2000 belong to the 'Fire' class, and 2000 images to the 'No Fire' class. The evaluation was carried out based on the calculations of the validation metrics, namely: the confusion matrix, the recall index, the precision index, and the F1-score.

Figures 37 and 38 show the confusion matrices of the Xception and EfficientNetB2 models respectively.

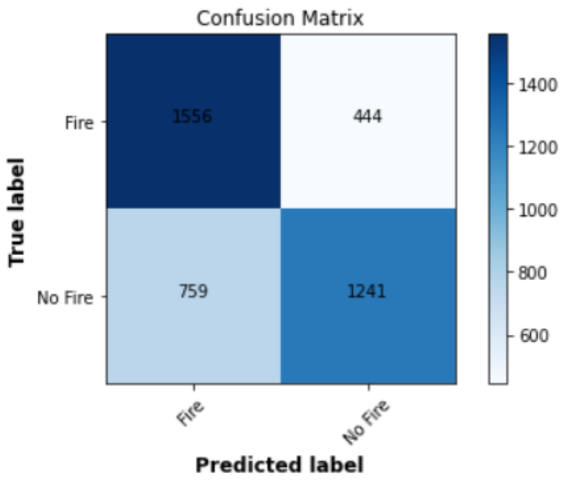


Figure 37: Confusion Matrix of Xception Model

Since the predicted classes are in columns and the actual classes in rows, we note that:

- 1556 images of the class 'Fire' were well classified while 444 are badly classified.
- 1241 images of the 'No Fire' class were correctly predicted by the model, while 759 images of this class were classified as 'Fire'.

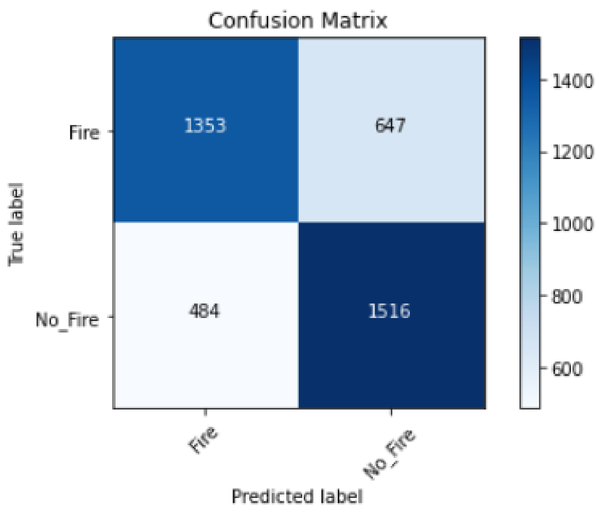


Figure 38: Confusion Matrix of EfficientNetB2 Model

- 1353 images of the class 'Fire' were well classified while 484 are badly classified.
- 1516 images of the class 'No Fire' were correctly predicted by the model as 'No Fire', although 647 images of the same class were classified in the wrong class.

We find that the number of true positives and true negatives is high compared to the number of false positives and false negatives in both models. From these results, we find that the separation between the classes performed by these models gives satisfactory results. The large difference between the number of images of the two classes (25,773 fire class images and 14,601 no_fire class images) during the training phase disturbs the model in the predictions, which explains the presence of a large number of false positives and false negatives.

Now comparing the validation metrics of the two models:

Table 6: Calculated Evaluation Metrics

Model	Indice of Recall %	Precision of test %	F1-score %
Xception	67.19	69.90	68.51
EfficientNetB2	73.65	71.72	72.41

According to Table 6, both models show satisfactory results. Indeed, the EfficientNetB2 model is better in terms of spotting fires given its high recall index, and it is a little more precise, i.e. it makes less error in predicting fires by compared to Xception.

2.3. Interpretability

Despite the good performance of the models in terms of image classification, we still do not know the reasons why each model makes a certain prediction. Judging the performance of the model based only on its accuracy is no longer sufficient as it can rely on the wrong parts of the image to predict. Interpretability makes it possible to explain the thought process of the model to predict an image. Understanding the behavior of our machine learning model is very important to be able to exploit it in an optimal way.

To understand this behavior, we used the **LIME** python library. **LIME** is based on the process of disturbing images by following the following steps:

- Generation of several samples similar to our input image by activating and deactivating some super-pixels of the image
- Prediction of the class of each of the generated images using our trained model
- Calculation of the weight of each superpixel
- Fitting a linear regression model using these weights
- Show superpixels that influence prediction.

Let's test this process on our models. We use the following image for this:

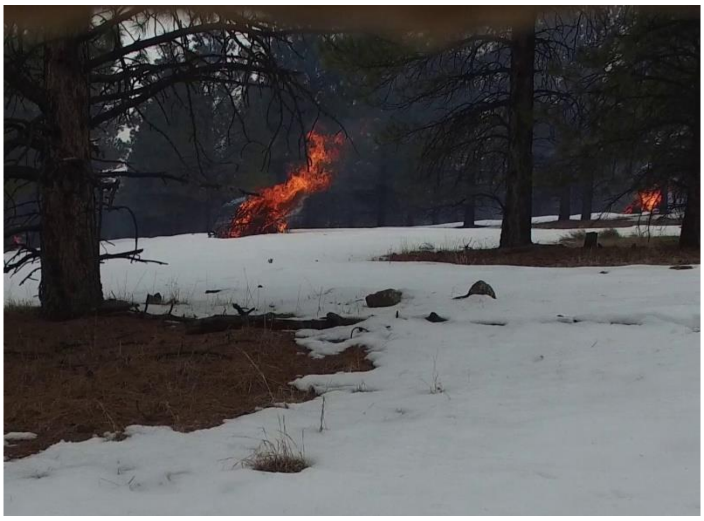


Figure 39: Image subject of interpretability

After the resizing and the prediction of the image by the models (Fire), the image is divided into superpixels:

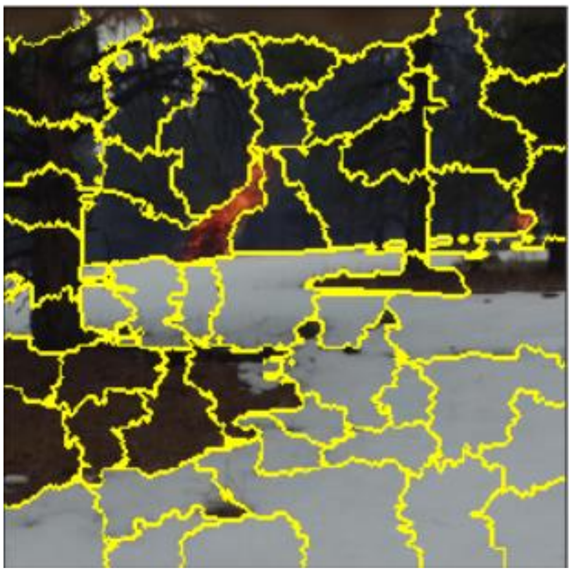


Figure 40: Image split into superpixels

Then, we move on to the disturbance of the image. LIME generates several images by combining different superpixels each time, for example the image of fig 40:

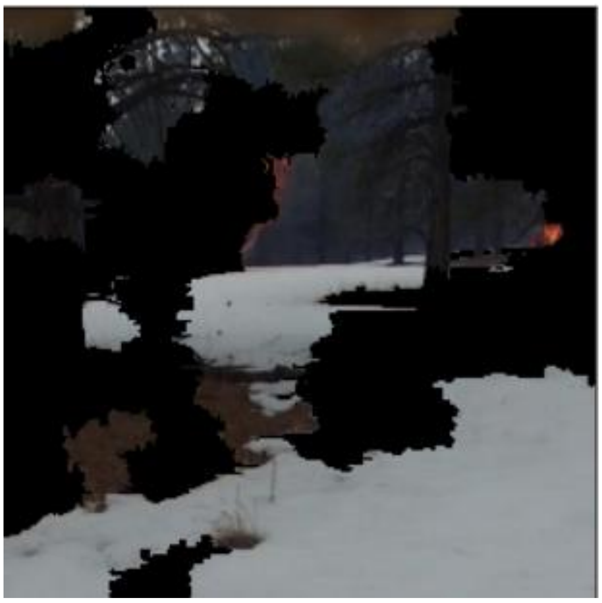


Figure 41: Example of the perturbation results

The model will predict each combination of superpixels and assign a weight. Finally, only the superpixels having a high weight are displayed.

Figure 42 presents the interpretability result of the Xception model by displaying the first five superpixels in terms of weight. The image does not show the part with lights. Therefore, the model considers that the characteristics of the fire are those presented in Figure 41. We find that, despite the satisfactory results of the quantitative validation, the learning process of the model is not successful.

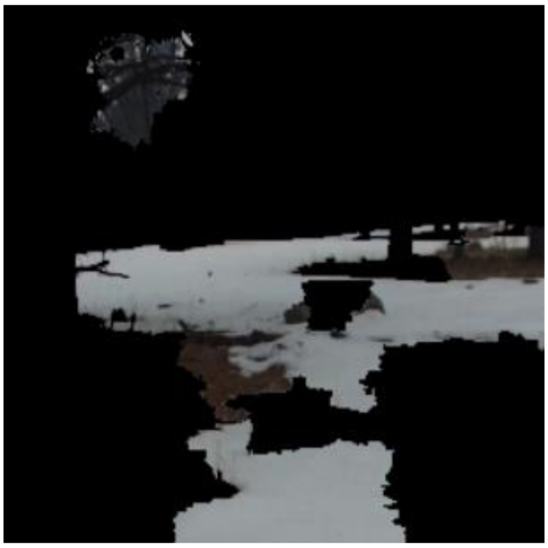


Figure 42: Result of interpretability by Xception

The result of the interpretability of the EffficientNetB2 model is shown in Figure 43:

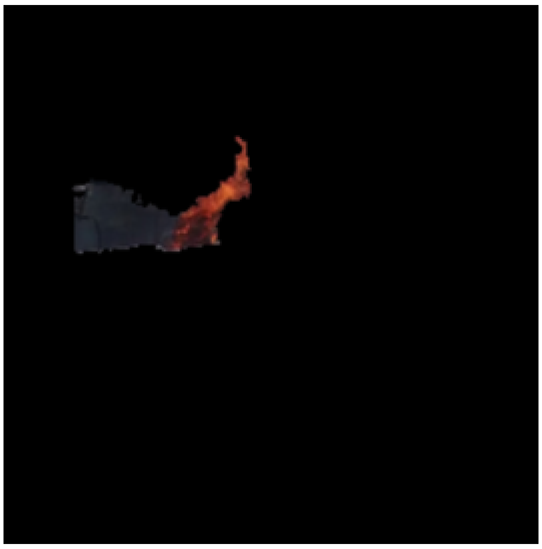


Figure 43: Result of interpretability by EfficientNetB2

By displaying only the superpixel with the greatest weight, we notice that it is indeed the part where there are lights that allowed the model to make its prediction. Therefore, the training of the EfficientNetB2 model is successful and the model accurately predicts the presence of fires.

2.4. Evaluation of EfficientNetB2 Model

According to the validation methods mentioned above, the EfficientNetB2 model is more adequate than the Xception model for the classification of forest fires. However, before the final validation of this model, it is necessary to ensure that it has not undergone over-learning. In our case, overfitting of the model is very likely given the similarities between the training, validation and test dataset.

Cross-validation when training the model allows it to be presented with different possibilities in order to prepare it to predict images other than those of the dataset. The analysis of the trend of the validation curve compared to that of training is a relevant indicator to study the occurrence of overlearning. We generated the progression curves of the precision and the loss function during the training and the validation of the model:

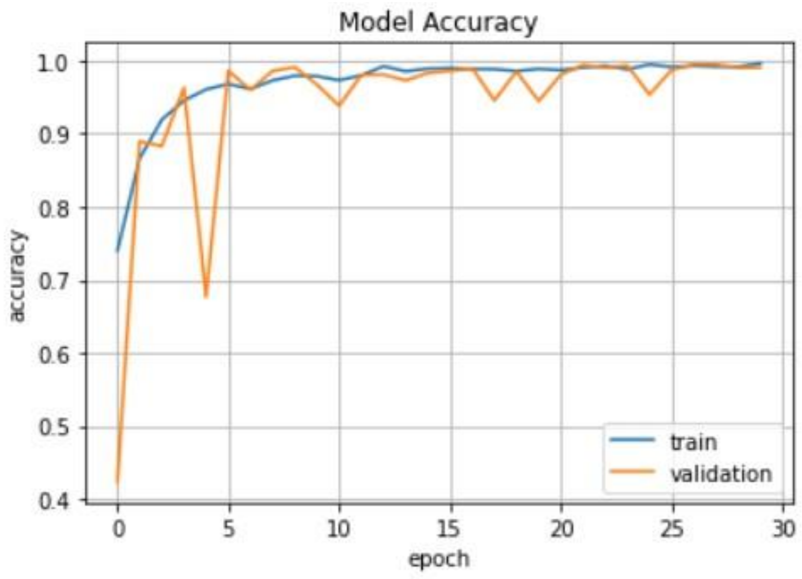


Figure 44: Accuracy progression during training

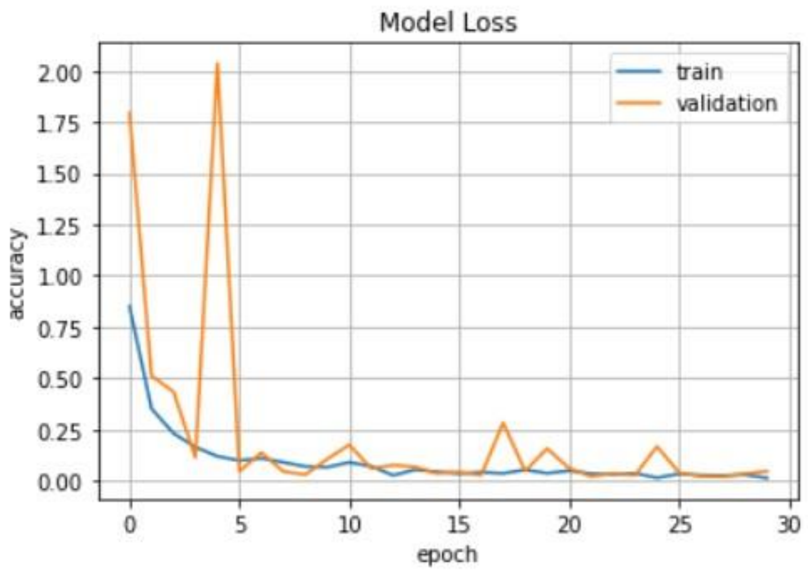


Figure 45: Loss Function progression during training

During the first epochs, validation precision drops are expected since the model is at the first learning step and the parameters are not yet adapted correctly. Despite the fluctuations of the validation curves, we notice that they follow the trend of the training curves. This is an indicator that the model was trained in a correct way thanks to the large size of the dataset as well as the 'hyperparameter tuning' which allowed us to optimize this process.

3. Segmentation Results :

In order to obtain a powerful and precise model given the critical domain of its use, we opted for a comparison between three semantic segmentation models known by their performance, namely: U-Net, Attention U-net, and Trans U-Net.

3.1. Data Preparation :

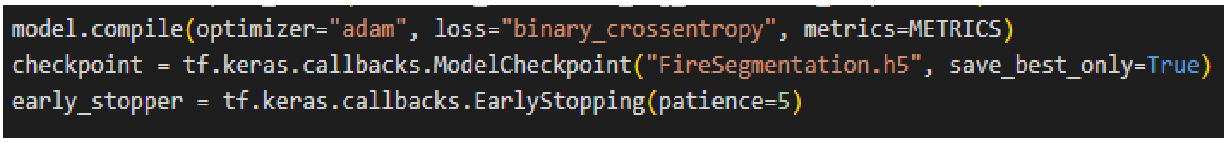
The FLAME Dataset segmentation dataset includes 2003 images and their masks. The number of frames in the dataset largely affects the performance of the attention mechanism and transformers. Therefore, to properly evaluate the results, training these models by different dataset sizes is necessary. Therefore, we will use three datasets for training the models:

Table 7: Datasets used for training the segmentation models

Dataset	Number of images	Number of Masks	Reference
1	2003	2003	FLAME Dataset
2	251	251	Randomly generated from the FLAME Dataset
3	12 018	12 018	Augmentation of images of the FLAME Dataset

3.2. Training of the U-Net model

Before moving on to training our model, we first start by preparing our data. These are 2003 images in ".jpg" format with their masks in ".png" format divided into two folders; 'Pictures' and 'Masks'. We used cross-validation to divide the images into training and validation datasets. Next, we defined the training parameters. Indeed, we adopted 'Binary_Crosstropy' as loss function since it is a binary semantic segmentation: 'Fire' and 'No Fire'. For the optimizer, we chose Adam optimizer with a learning rate of 10-3. We used the ModelCheckpoint function to be able to save the weights of the model after execution of each epoch in ".h5" format. As well as the addition of the Earlystopping callback function with a patience of 5, which allows the training to be automatically stopped after the execution of five epochs if the validation error does not reduce, or if the performance of the model does not increase. Figure 46 shows the integration of these parameters in the model:



```
model.compile(optimizer="adam", loss="binary_crossentropy", metrics=METRICS)
checkpoint = tf.keras.callbacks.ModelCheckpoint("FireSegmentation.h5", save_best_only=True)
early_stopper = tf.keras.callbacks.EarlyStopping(patience=5)
```

Figure 46: Parameters of U-Net model

The architecture of the U-Net model presented in the methodology part is as follows:

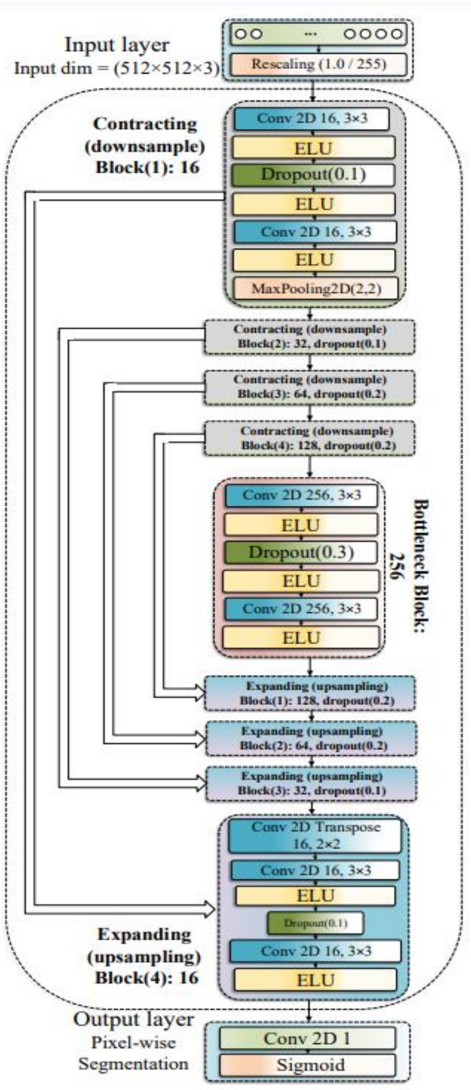
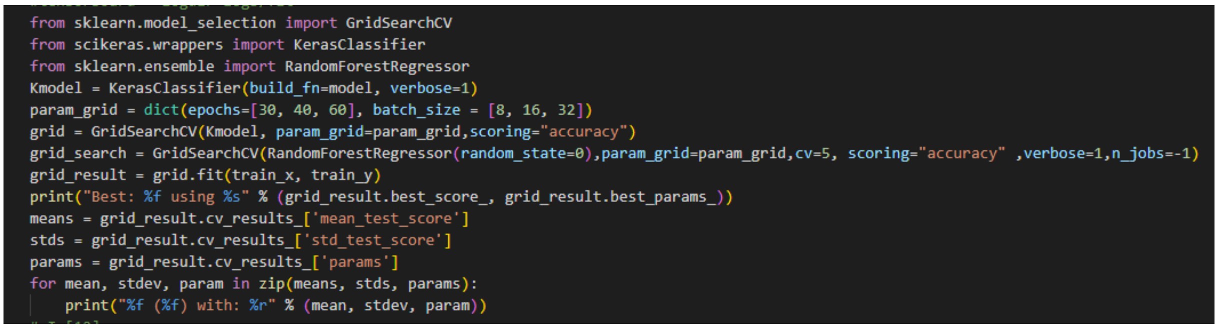

Input layer
Input dim = (512×512×3)
Rescaling (1.0 / 255)
Contracting (downsample) Block(1): 16
Conv 2D 16, 3×3
ELU
Dropout(0.1)
ELU
Conv 2D 16, 3×3
ELU
MaxPooling2D(2,2)
Contracting (downsample) Block(2): 32, dropout(0.1)
Contracting (downsample) Block(3): 64, dropout(0.2)
Contracting (downsample) Block(4): 128, dropout(0.2)
Bottleneck Block: 256
Conv 2D 256, 3×3
ELU
Dropout(0.3)
ELU
Conv 2D 256, 3×3
ELU
Expanding (upsampling) Block(1): 128, dropout(0.2)
Expanding (upsampling) Block(2): 64, dropout(0.2)
Expanding (upsampling) Block(3): 32, dropout(0.1)
Expanding (upsampling) Block(4): 16
Conv 2D Transpose 16, 2×2
Conv 2D 16, 3×3
ELU
Dropout(0.1)
Conv 2D 16, 3×3
ELU
Output layer
Pixel-wise Segmentation
Conv 2D 1
Sigmoid

Figure 47: Architecture of the used U-Net model (A. Samsoshora et al., 2020)

The optimization of the model's hyperparameters by the 'hyperparameter tuning' method, explained in the previous chapter, has been deployed in the training of the segmentation model:



```
from sklearn.model_selection import GridSearchCV
from scikeras.wrappers import KerasClassifier
from sklearn.ensemble import RandomForestRegressor
Kmodel = KerasClassifier(build_fn=model, verbose=1)
param_grid = dict(epochs=[30, 40, 60], batch_size = [8, 16, 32])
grid = GridSearchCV(Kmodel, param_grid=param_grid,scoring="accuracy")
grid_search = GridSearchCV(RandomForestRegressor(random_state=0),param_grid=param_grid,cv=5, scoring="accuracy" ,verbose=1,n_jobs=-1)
grid_result = grid.fit(train_x, train_y)
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
means = grid_result.cv_results_['mean_test_score']
stds = grid_result.cv_results_['std_test_score']
params = grid_result.cv_results_['params']
for mean, stdev, param in zip(means, stds, params):
    print("%f (%f) with: %r" % (mean, stdev, param))
```

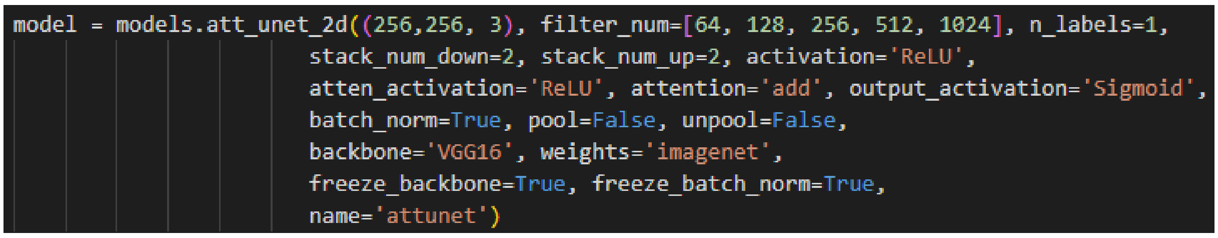
Figure 48: Use of 'Grid Search' method

- Number of epochs: [30, 40, 50]
- Batch Sizes : [8, 16, 32]

Once the model training is complete, the 'grid search' method returns that the best combination is that of 40 epochs and a batch size of 32.

3.3. Training of Attention U-Net Model

For training this model, we structured the training data in the same way as before. The model used is part of the **Keras_unet_collection** collection. This is an online CNN template library containing functions that build the basic architecture of U-Net variants for template customization.

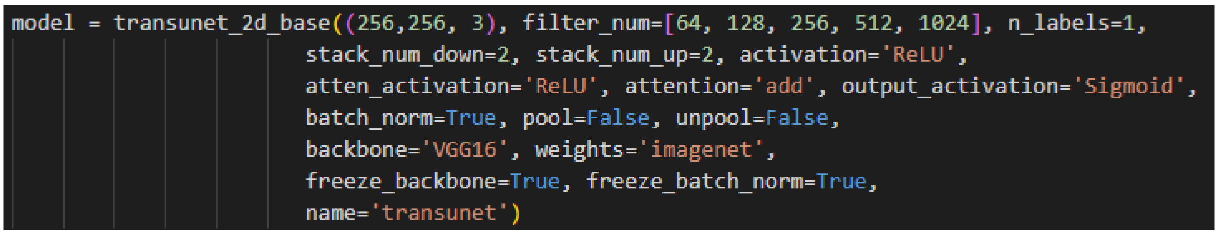


```
model = models.att_unet_2d((256,256, 3), filter_num=[64, 128, 256, 512, 1024], n_labels=1,
                           stack_num_down=2, stack_num_up=2, activation='ReLU',
                           atten_activation='ReLU', attention='add', output_activation='Sigmoid',
                           batch_norm=True, pool=False, unpool=False,
                           backbone='VGG16', weights='imagenet',
                           freeze_backbone=True, freeze_batch_norm=True,
                           name='attunet')
```

Figure 49: Importing the Attention_UNet model from keras_unet_collection

3.4. Training of Trans U-Net Model

Similar to the Attention U-Net model, the training of Trans U-Net was carried out using the aforementioned collection.



```
model = transunet_2d_base((256,256, 3), filter_num=[64, 128, 256, 512, 1024], n_labels=1,
                          stack_num_down=2, stack_num_up=2, activation='ReLU',
                          atten_activation='ReLU', attention='add', output_activation='Sigmoid',
                          batch_norm=True, pool=False, unpool=False,
                          backbone='VGG16', weights='imagenet',
                          freeze_backbone=True, freeze_batch_norm=True,
                          name='transunet')
```

Figure 50: Importing the trans_unet model from keras_unet_collection

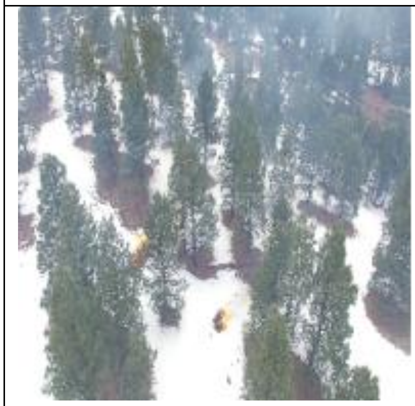
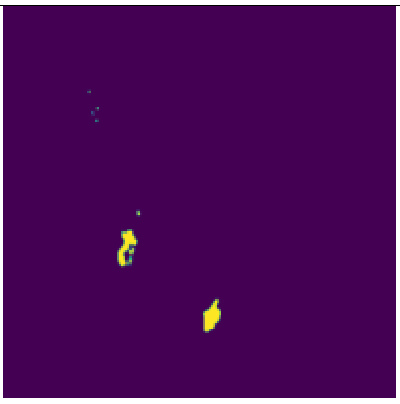
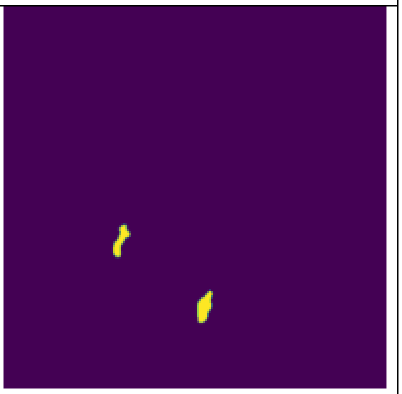
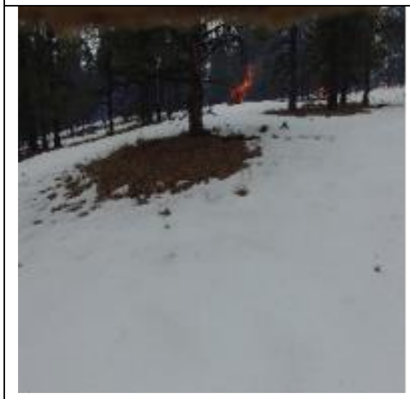
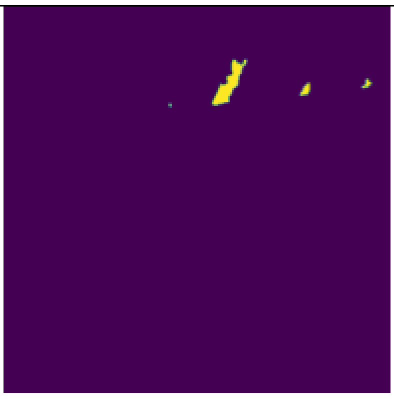
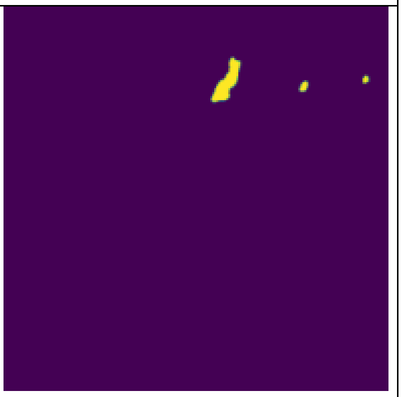
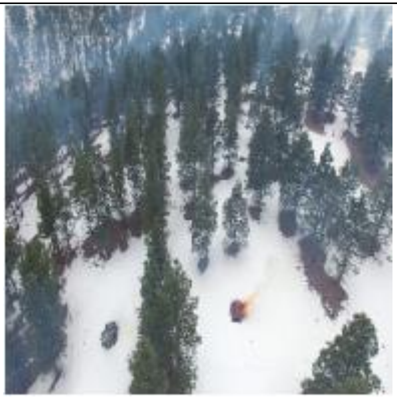
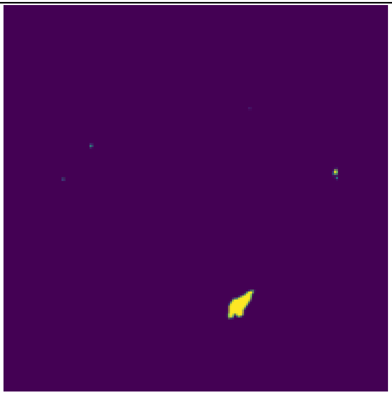
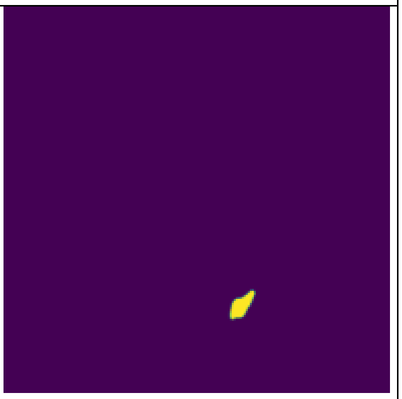
3.5. Evaluation of the Results :

3.5.1. Dataset 1

3.5.1.1. Visual Validation :

This is the first validation to be performed after training the models. It is a question of visually examining the masks of the images resulting from the predictions by comparing them with the real masks.

Table 8: Segmentation Results using U-Net (Dataset 1)

Original Images	Mask	Predicted Mask by U-Net
		
		
		

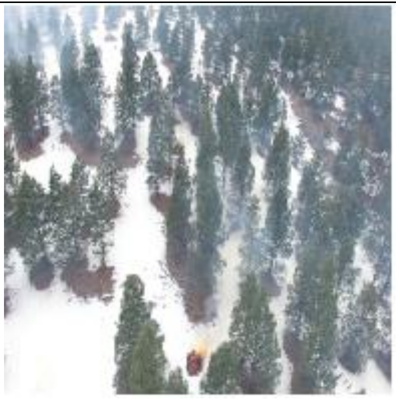
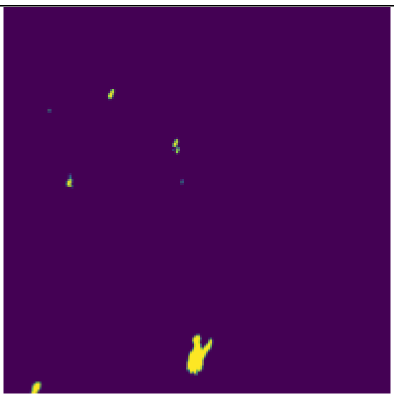
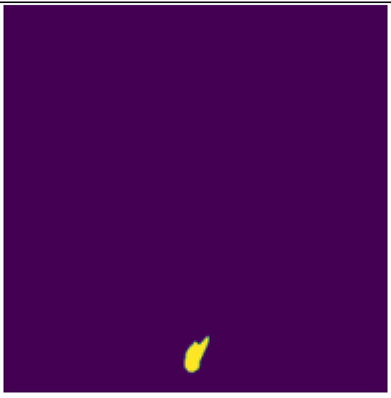
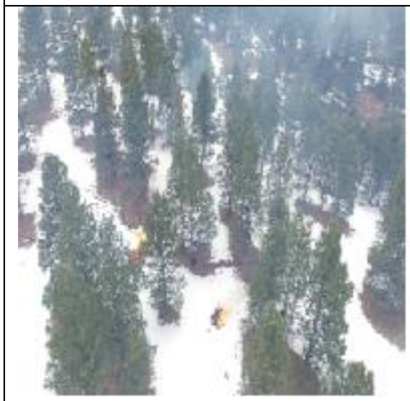
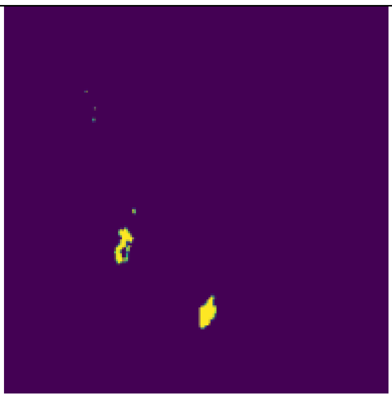
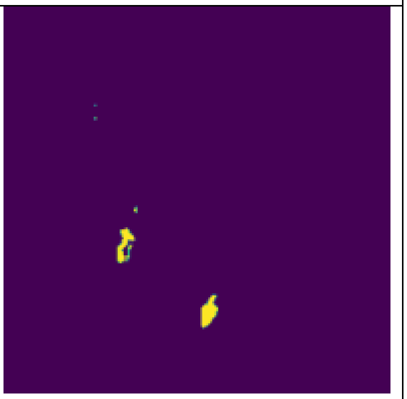
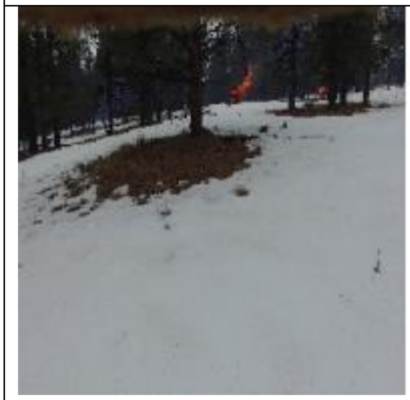
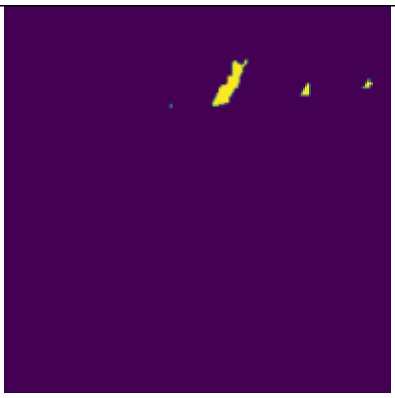
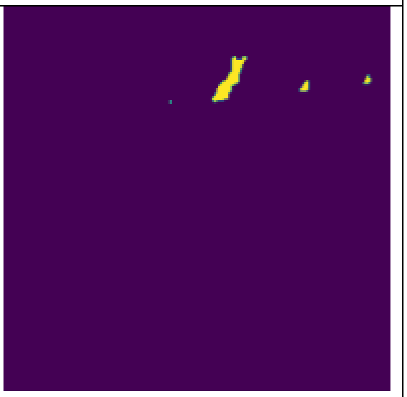
		

Table 9: Segmentation Results using Att U-Net (Dataset 1)

Original Images	Mask	Predicted Masks Att U-Net
		
		

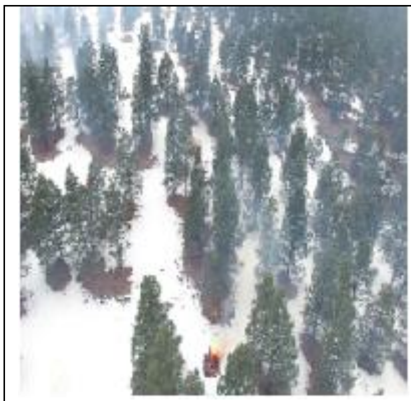
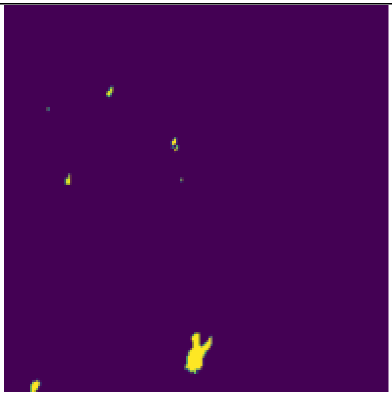
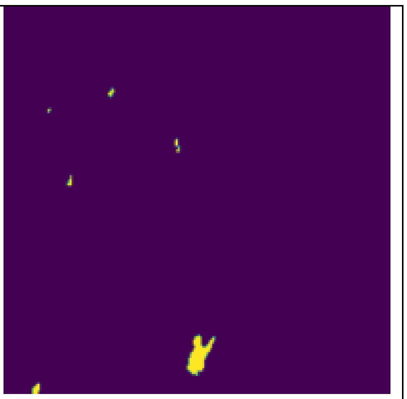
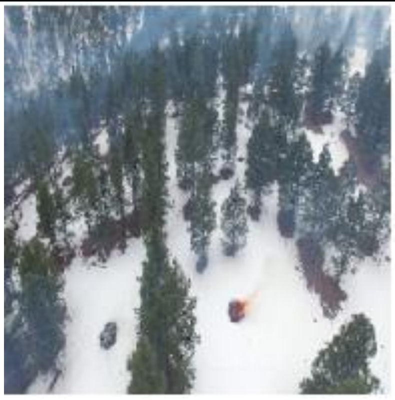
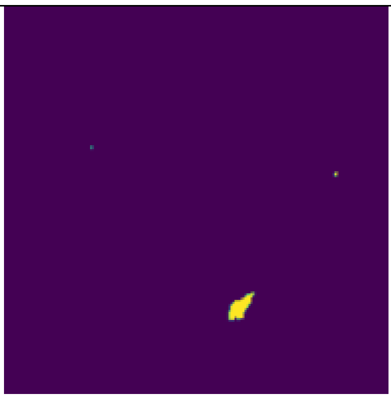
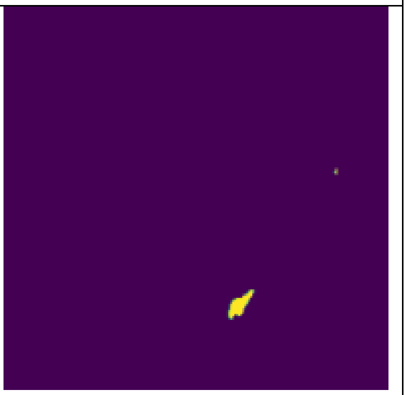
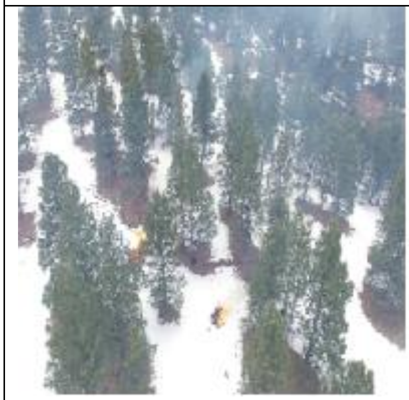
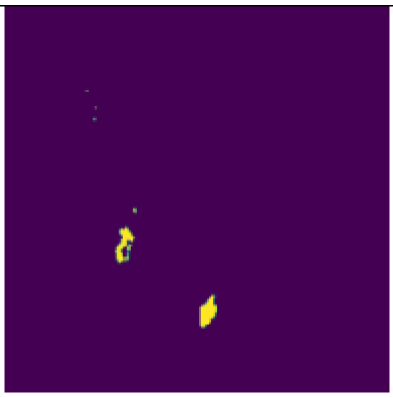
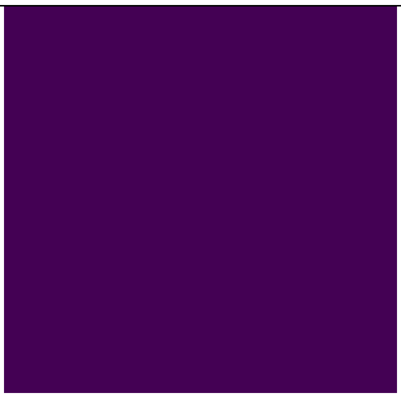
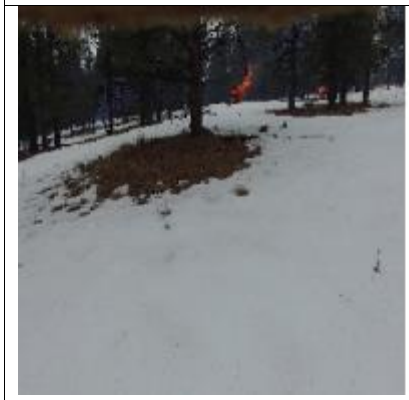
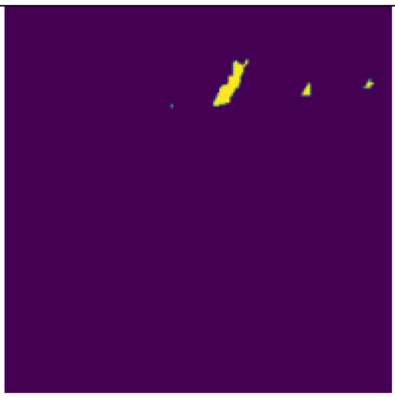
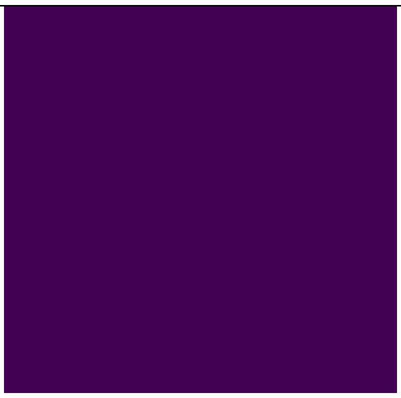
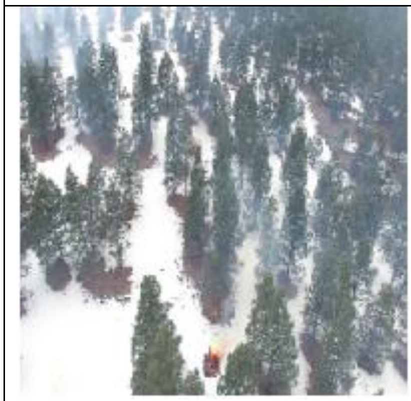
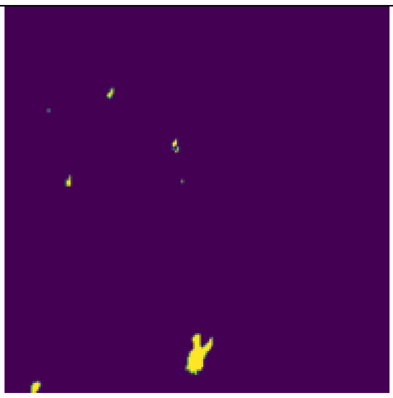
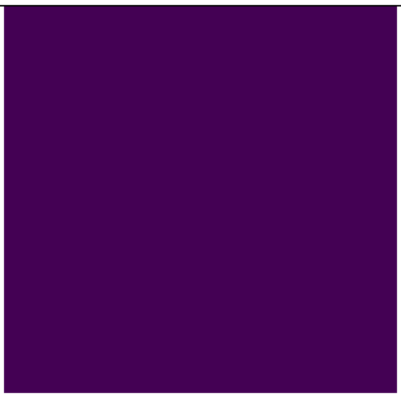
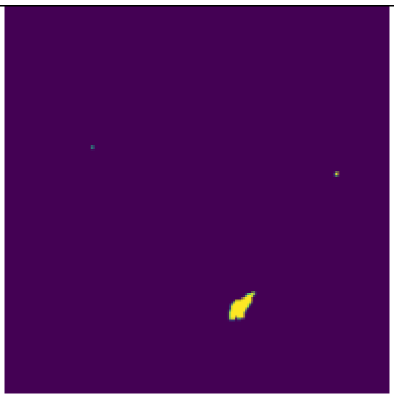
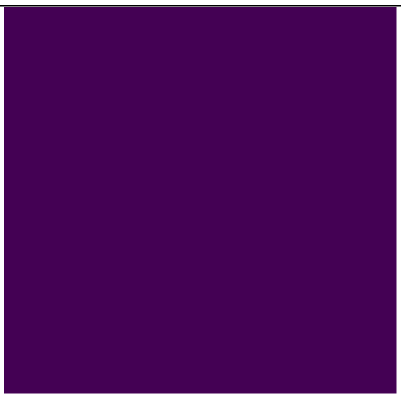
		
		

Table 10 : Segmentation Results using Trans U-Net (Dataset 1)

Original Images	Mask	Predicted Mask by Trans U-Net
		
		
		

We notice that the masks generated by U-Net and Attention U-Net are almost identical to the real masks. We can deduce that the two models are validated qualitatively. As for the Trans U-Net model, it did not generate masks. In addition, the resulting '.h5' model after training is very large (1.6 GB). These results are due to the fact that processors require huge datasets having up to 10 million images to give good results. So the Trans U-Net is not validated.

In the following, we will only focus on the two models U-Net and Attention U-Net.

3.5.1.2. Quantitative Validation:

This step consists of an evaluation of the segmentation results by the two models used based on the test dataset. The metrics adopted for this purpose and their results are presented in the following table 11:

Table 11: Evaluation metrics of the segmentation using U-Net et Attention U-Net

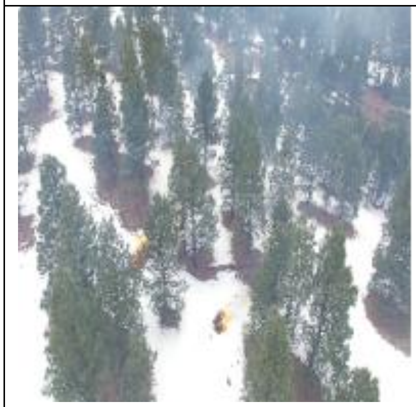
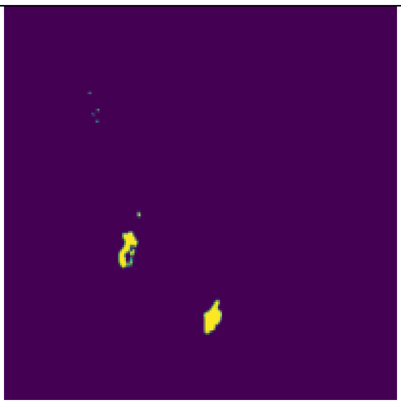
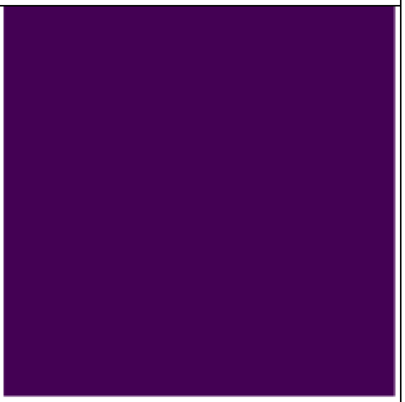
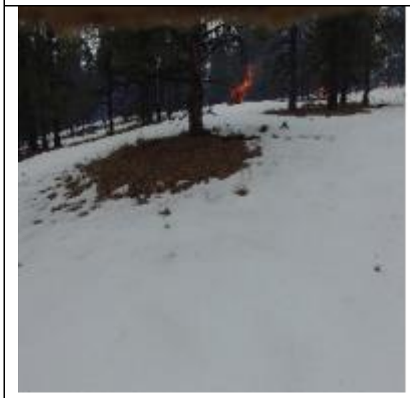
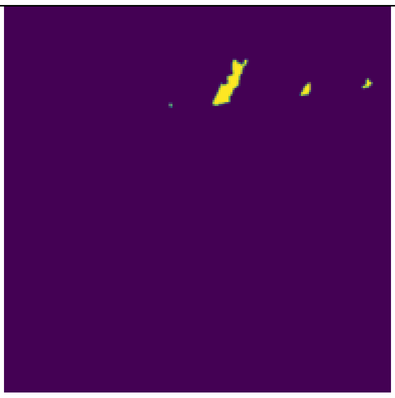
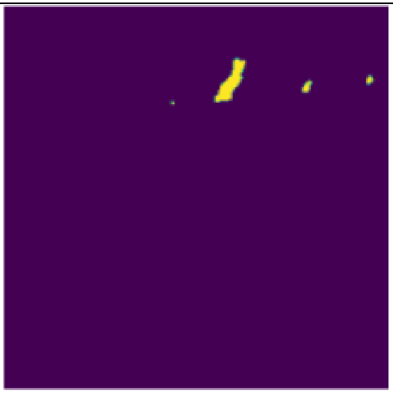
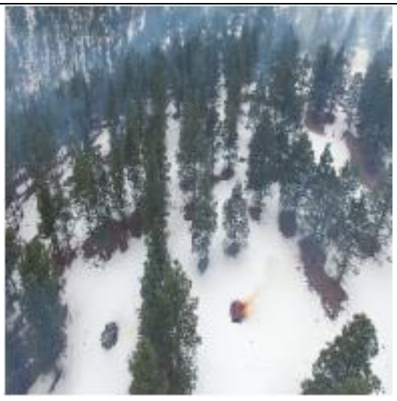
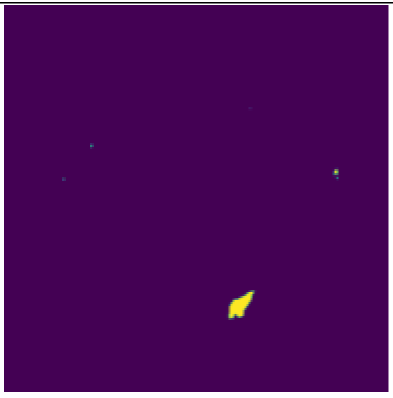
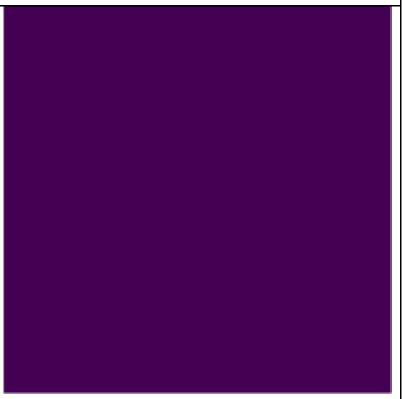
Model	Precision %	Recall %	IoU %
U-Net	88.6	90.7	81.4
Attention U-Net	98.4	92.8	89.7

Table 11 shows that the U-Net Attention model performs much better than the U-Net model. The richness of the dataset allowed the model to master the classification of pixels, hence the high value of the IoU (intersection over union). These metric values explain the great similarity between the masks generated by this model and the ground truth masks.

3.5.2. Dataset 2 :

3.5.2.1. Visual Validation :

Table 12: Segmentation Results using U-Net (Dataset 2)

Original Images	Mask	Predicted Mask U-Net
		
		
		

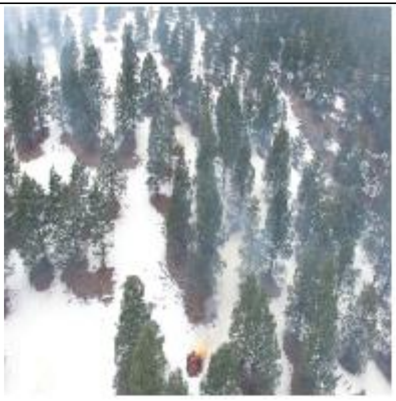
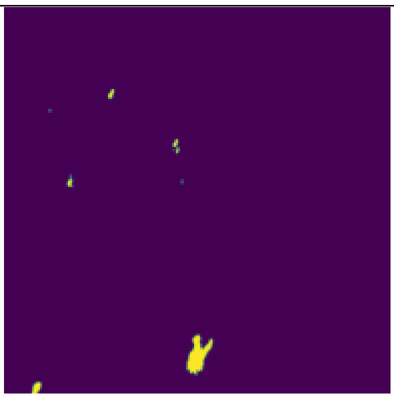
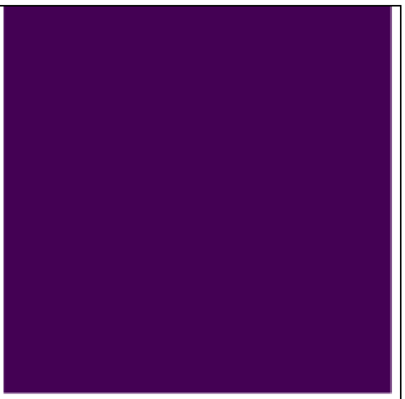
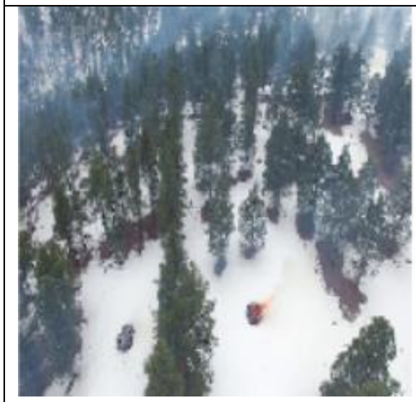
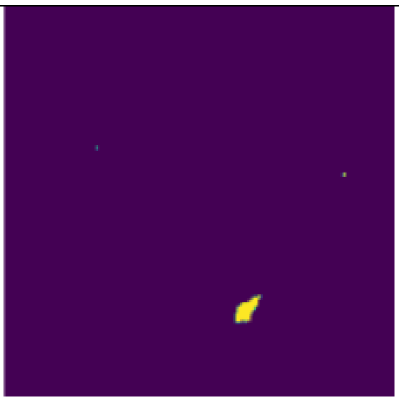
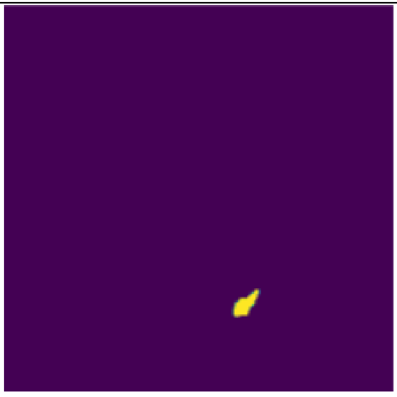
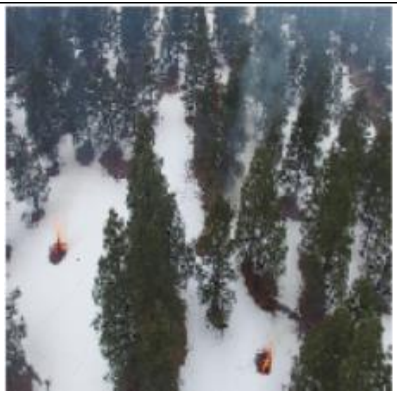
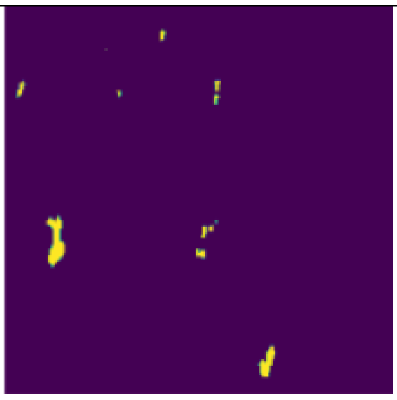
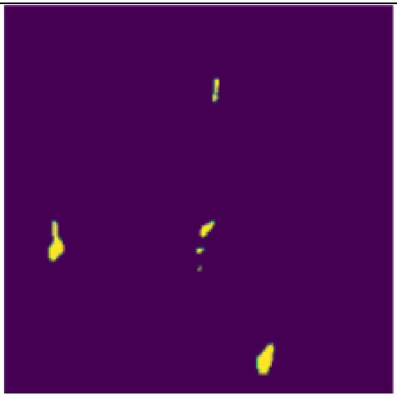
		

Table 13: Segmentation Results using Att U-Net (Dataset 2)

Original Images	Mask	Predicted Mask Att U-Net
		
		

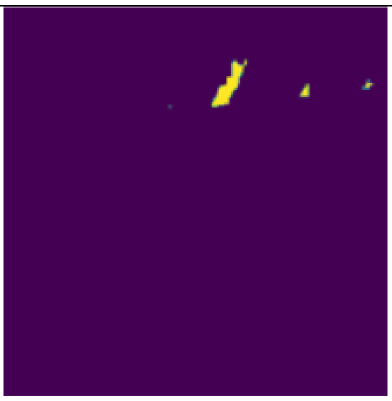
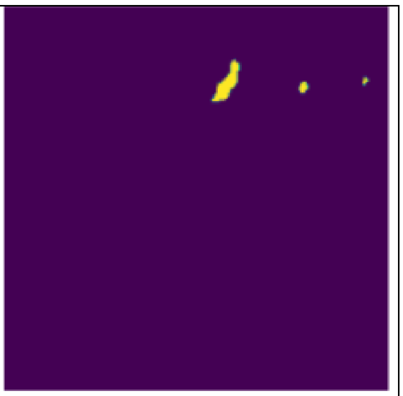
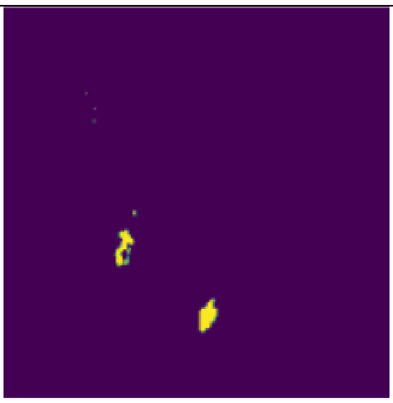
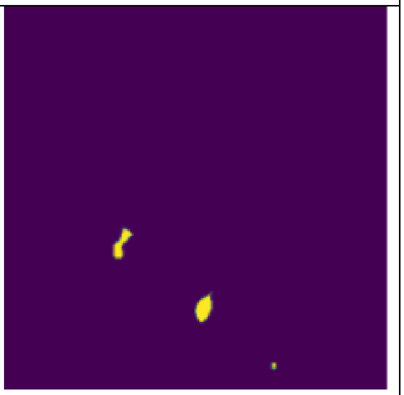
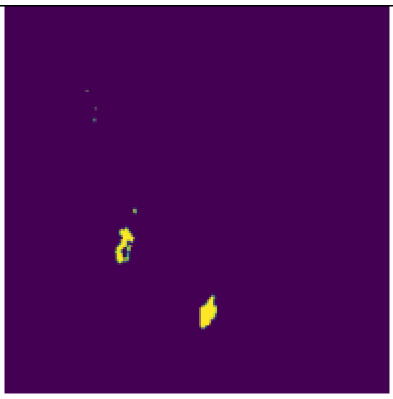
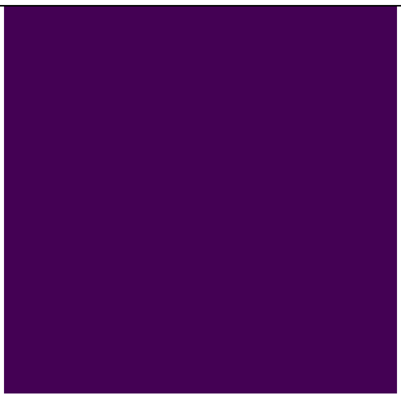
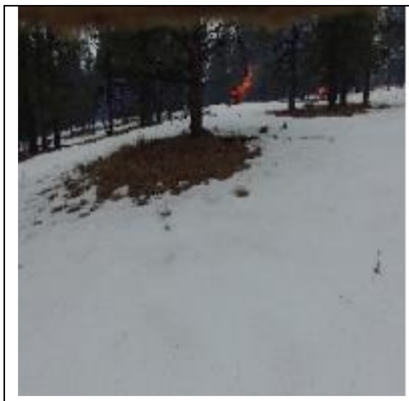
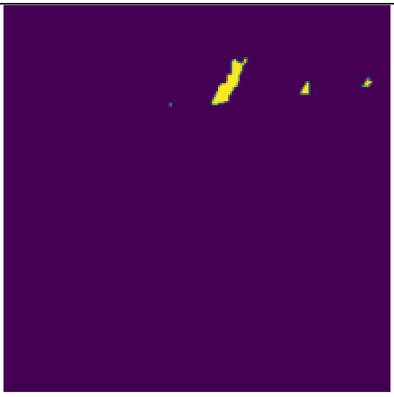
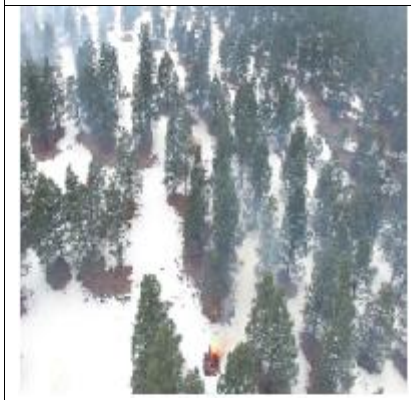
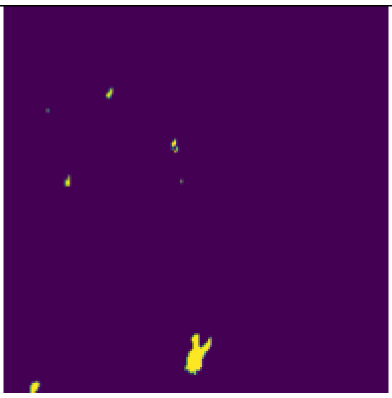
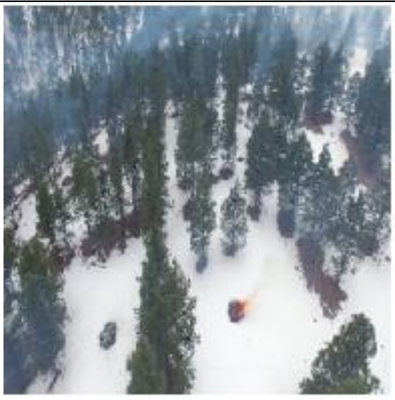
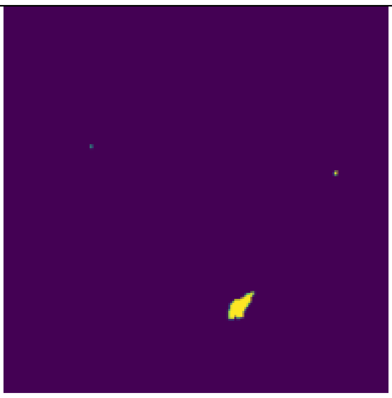
		
		

Table 14: Segmentation Results using Trans U-Net (Dataset 2)

Original Images	Mask	Predicted Mask Trans U-Net
		

From the first table, the U-Net model predicted only one image. This image was part of the second training dataset while the others were not. So, if the accuracy of the model is high, we can directly see that it is over-learning, otherwise, the number of images was insufficient to train the model.

The Attention U-Net model generated visually acceptable results despite the size of the dataset. However, the possibility of overfitting remains valid.

As for the first dataset, the number of images is not sufficient to properly train the Trans U-Net model, which explains the results above.

3.5.2.2. Quantitative Validation :

Table 15: Evaluation Metrics of the segmentation using U-Net et Attention U-Net

Model	Precision %	Recall %	IoU %
U-Net	99.8	81.5	49.8
Attention U-Net	99.8	87.4	49.7

The values of the metrics of the two models are very similar. First, the high validation accuracy of the U-Net model confirms our hypothesis that it follows overfitting. Regarding the U-Net Attention model, the prediction results per test set are contradictory with the IoU value of 49.7%. This value means that only 49.7% of the pixels are common between the real masks and those generated by the model. While in the previous part, we saw that the two masks are almost similar.

3.5.3. Dataset 3 :

This dataset is reserved for testing the Trans U-Net model to highlight the fact that it requires a very large number of images.

3.5.3.1. Data Preparation :

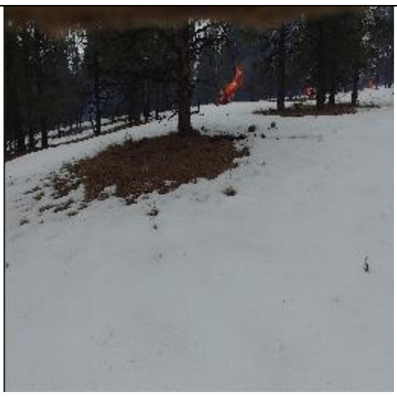
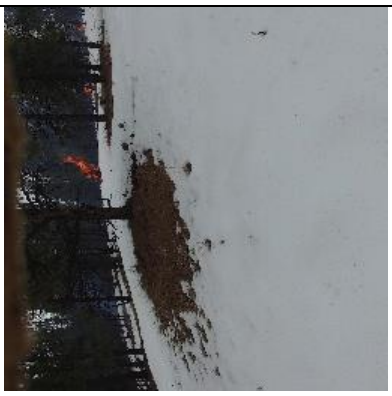
The images in this dataset are the result of an augmentation of the initial data from the ‘FLAME Dataste’. The objective of the augmentation is to enrich the dataset by modifying the initial images. Regarding the case of semantic segmentation, each image must have its corresponding mask, so the same modifications must be made to the mask corresponding to each image.

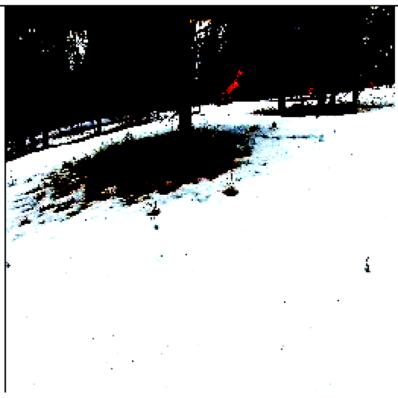
We achieved this augmentation using the python '**Albumentations**' library. ‘**Albumentations**’ efficiently implements a wide variety of image transformation operations. The choice of transformations must be based on the nature of the object studied. For example in our case, a zoom operation can generate images that do not contain fire. We chose the following five functions to augment our dataset:

- RandomRotate90: a random rotation of 90
- GridDistortion: slight image disturbance
- Horizontalflip: horizontal flip (mirror effect)
- Verticalflip: vertical flip
- RandomContrast: random modification of the contrast

These operations will make it possible to diversify our dataset while keeping the spectral information of the fire.

Table 16: Augmentation Examples of an image from FLAME Dataset

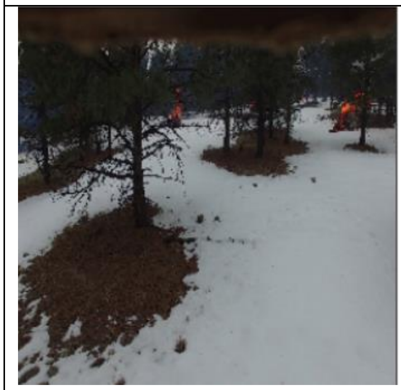
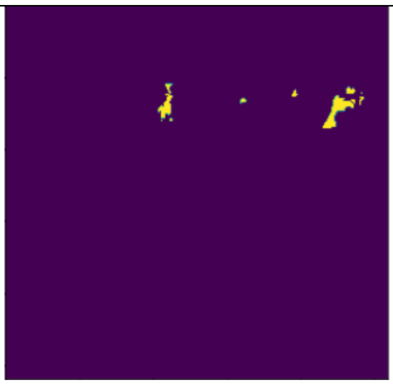
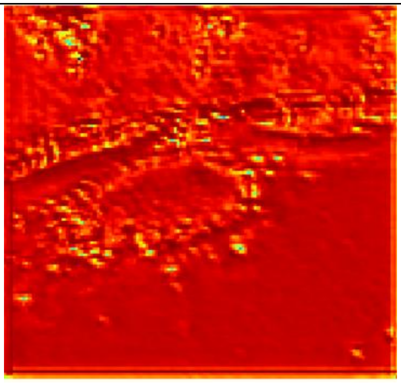
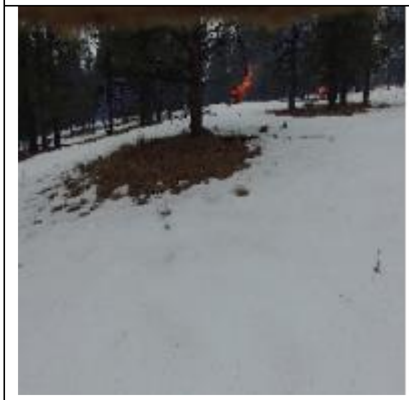
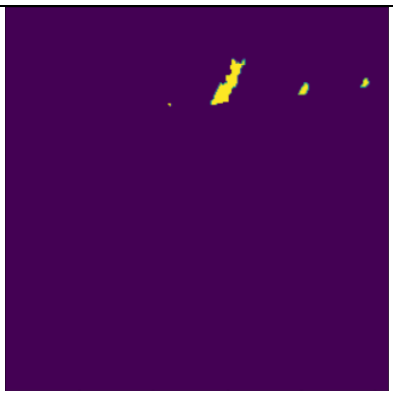
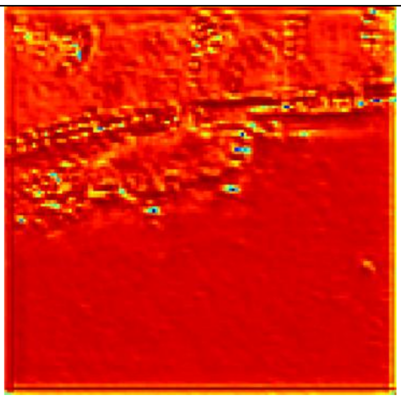
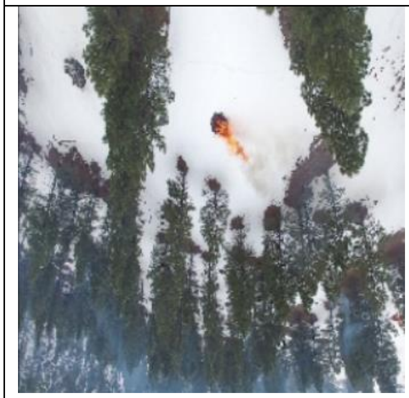
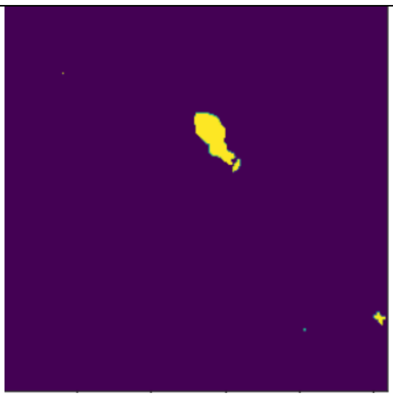
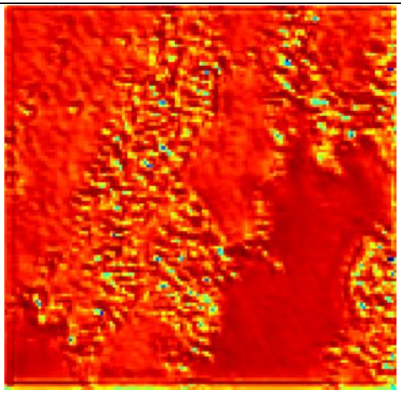
Original Image	
RandomRotate90	

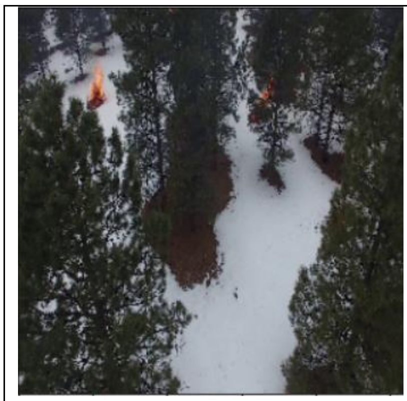
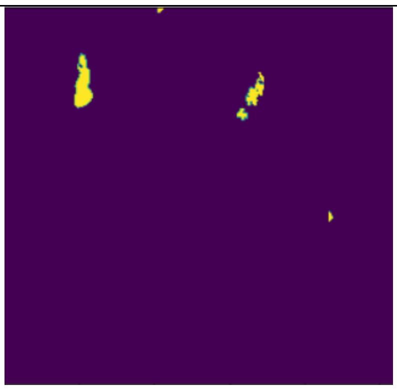
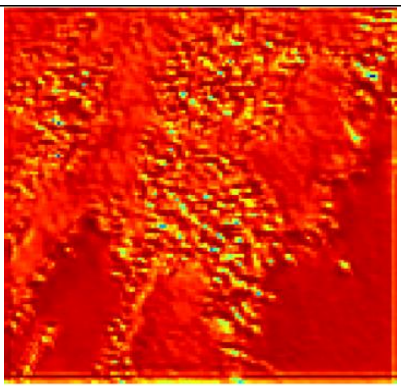
GridDistortion	
Horizontalflip	
Verticalflip	
RandomContrast	

Thanks to this increase, the new dataset contains 12,018 images and 12,018 masks.

3.5.3.2. Visual Validation

Table 17: Results of the prediction using Trans U-Net model (Dataset 3)

Original Images	Mask	Predicted Mask Trans U-Net
		
		
		

First of all, we can see that the high number of images allowed the model to train enough to generate a mask, unlike the first two tests. However, the resulting segmentation masks are very disturbed and far from reality. This proves that vision transformers require a very large number of images to generate acceptable results.

The presentation of the results of the quantitative validation is not necessary in this case since the model is clearly poorly trained.

3.5.4. General Comparaison

Table 18: General Comparison of Segmentation methods

Datasets	Validation	U-Net	Att U-Net	Trans U-Net
1	Visual	✓	✓	✗
	Quantitative	✓	✓	✗
2	Visual	✗	✓	✗
	Quantitative	✓	✓	✗
3	Visual	-	-	✗
	Quantitative	-	-	✗

4. Conclusion :

Our methodology, explained in the previous chapter, has yielded considerable results for the classification and semantic segmentation of forest fire images. The comparison of the two CNN classification architectures allowed us to highlight the contribution of the ‘Compound Scaling’, used in the EfficientNetB2 model, in the classification of images. Admittedly, the evaluation metrics of the two models EfficientNetB2 and Xception are close, but the interpretability showed us that only the EfficientNet model actually detects the presence of lights.

Regarding image segmentation, we compared three powerful semantic segmentation networks: U-Net, Attention U-Net and Trans U-Net. In order to evaluate the impact of numbers of images on the performance of these models, we used three datasets containing the same type of images with different numbers. The role of the attention mechanism in semantic segmentation was highlighted by the results. Indeed, the U-Net Attention model outperformed other models in visual validation and quantitative validation. Increasing the dataset proved to be insufficient to properly train the Trans U-Net model which requires a huge number of images.

GENERAL DISCUSSION

In this end-of-studies work, we propose an approach to optimize the detection and delimitation of forest fires by drone.

First, we explored the use of EfficientNet, a model other than the classic ImageNet classification models. This model is characterized by 'compound scaling', a method which makes it possible to uniformly scale the architecture parameters of the model. From the analysis of the evaluation metrics, we found that both models are validated quantitatively. However, the interpretability shows us that the Xception model is based on the bad characteristics to predict the class of the image, unlike EfficientNet which correctly predicts the presence of fire.

The majority of previous studies on image classification rely solely on evaluation metrics to evaluate their models. The interpretability results of our models show that no matter how strong the evaluation metrics are, they can mislead us.

In order to optimize the forest fire delineation process, we compared three segmentation models: U-Net, Attention U-Net and Trans U-Net. Three data sets of different sizes were created. The objective of this comparison was to highlight the contribution of the attention mechanism and vision transformers in semantic segmentation.

We concluded from this comparison that the size and representativeness of the dataset play a major role in the performance of the model. In the case of a dataset with a low number of images, the occurrence of overtraining is very likely. Therefore, the visual validation of the model by images similar to those used during training is not sufficient to decide on its performance. We also saw the requirements of vision processors in terms of dataset size.

The contribution of attention mechanisms on semantic segmentation was significant. The model showed a remarkable performance improvement over the U-Net model.

GENERAL CONCLUSION

Automation and speed in the fire detection process are two essential criteria for the management and preservation of natural resources and human life. In order to achieve this end, the use of convolutional neural networks based on drone imagery is the best approach.

The realization of our graduation project was carried out following mainly these two main steps: first, the training of two classification models with a database containing thousands of images to choose the model allowing the exact classification of an input image into two classes: 'Fire' or 'No_Fire'. Then, a comparison between three powerful models in the semantic segmentation of images and choose the most accurate model to generate masks for images containing fires. Moreover the implementation of a mobile application where the two models of classification and segmentation are deployed could not be achieved in this project, given its complexity.

With a view to refining and improving this work, we would like to make the following recommendations:

- The use of a more representative dataset whose images actually capture a natural forest fire.
- The use of a large data set to be able to benefit from the advantages of convolutional neural networks combined with transformers such as the Trans U-Net model.
- The variation of the hyperparameters of the models in order to improve them.
- Design of a model that not only allows the detection but also the monitoring of the spread of fire as well as the prediction of areas that can be burned.
- Creation of a specific Android Studio project for the deployment of semantic segmentation models

REFERENCES

- **Adam P. Piotrowski, Jaroslaw J. Napiorkowski, Agnieszka E. Piotrowska**, Impact of deep learning-based dropout on shallow neural networks applied to stream temperature modelling,Earth-Science Reviews,Volume 201,2020,103076,ISSN 0012-8252, https://doi.org/10.1016/j.earscirev.2019.103076.
- **Alom, Md Zahangir, et al**. "The history began from alexnet: A comprehensive survey on deep learning approaches." arXiv preprint arXiv:1803.01164 (2018).
- **Alzubaidi, Laith, et al**. "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions." Journal of big Data 8.1 (2021): 1-74.
- **Ana Cristina Gonçalves**, Mediterranean Identities — Environment, Society, Culture (pp.305-335) Chapter: The Fire in the Mediterranean Region: A Case Study of Forest Fires in Portugal, 2017.
- **Athanasios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, Eftychios Protopapadakis**, "Deep Learning for Computer Vision: A Brief Review", Computational Intelligence and Neuroscience, vol. 2018, Article ID 7068349, 13 pages, 2018. https://doi.org/10.1155/2018/7068349
- **A. N. Benaichouche, H. Oulhadj, and P. Siarry**, “Improved spatial fuzzy c-means clustering for image segmentation using pso initialization, mahalanobis distance and post-segmentation correction,” Digital Signal Processing, vol. 23, no. 5, pp. 1390–1400, 2013.
- **Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla**. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." IEEE transactions on pattern analysis and machine intelligence 39.12 (2017): 2481-2495.
- **Bijen Khagi, Goo-Rak Kwon**, "Pixel-Label-Based Segmentation of Cross-Sectional Brain MRI Using Simplified SegNet Architecture-Based CNN", Journal of Healthcare Engineering, vol. 2018, Article ID 3640705, 8 pages, 2018. https://doi.org/10.1155/2018/3640705
- **Christopher M. Bishop**, « Pattern Recognition And Machine Learning ».
- **David John, Ce Zhang**, An attention-based U-Net for detecting deforestation within satellite sensor imagery, International Journal of Applied Earth Observation and Geoinformation, Volume 107, 2022, 102685, ISSN 1569-8432, https://doi.org/10.1016/j.jag.2022.102685.

- **Dosovitskiy, Alexey, et al**. "An image is worth 16x16 words: Transformers for image recognition at scale." arXiv preprint arXiv:2010.11929 (2020).
- **EPA United States Environment Protection Agency**, 2021.
- **Florent** (2018) Deep Learning, les fonctions d'activation disponible sur https://www.supinfo.com/articles/single/7923-deep-learning-fonctions-activation
- **Ghali, Rafik, Moulay A. Akhloufi, and Wided Souidene Mseddi**. "Deep learning and transformer approaches for UAV-based wildfire detection and segmentation." *Sensors* 22.5 (2022): 1977.
- **Guo, Meng-Hao, et al**. "Attention mechanisms in computer vision: A survey." Computational Visual Media (2022): 1-38.
- **Goodfellow, I., Bengio, Y., Courville, A**. (2016) Deep Learning. The MIT Press. 800p.
- **Huang, Qiyuan, et al**. "Wildfire spread modeling with aerial image processing." *2020 IEEE 21st International Symposium on" A World of Wireless, Mobile and Multimedia Networks"(WoWMoM)*. IEEE, 2020.
- **Iqbal H. Sarker**, Deep Learning: A Comprehensive Overview on Techniques, Taxonomy, Applications and Research Directions, 2021.
- **Janiesch, C., Zschech, P. & Heinrich, K**. Machine learning and deep learning. Electron Markets 31, 685–695 (2021). https://doi.org/10.1007/s12525-021-00475-2
- **J. Lemley, S. Bazrafkan and P. Corcoran**, "Smart Augmentation Learning an Optimal Data Augmentation Strategy," in *IEEE Access*, vol. 5, pp. 5858-5869, 2017, doi: 10.1109/ACCESS.2017.2696121.
- **Jordan, M. I., & Mitchell, T. M**. (2015). Machine learning: Trends, perspectives, and prospects. *Science, 349*(6245), 255–260. https://doi.org/10.1126/science.aaa8415
- **Kingma, Diederik P., and Jimmy Ba**. "Adam: A method for stochastic optimization." *arXiv preprint arXiv:1412.6980* (2014).
- **L. Cunbin, Z. Jing, T. Baoguo, and Z. Ye,** "Analysis of forest fire spread trend surrounding transmission line based on rothermel model and huygens principle," International Journal of Multimedia and Ubiquitous Engineering, vol. 9, no. 9, pp. 51–60, 2014.
- **Lin, Ming Xiu, et al**. "An intelligent fire-detection method based on image processing." *Advanced Engineering Forum*. Vol. 2. Trans Tech Publications Ltd, 2012.

- **LeCun, Y., Bengio, Y. & Hinton, G**. Deep learning. Nature 521, 436–444 (2015). https://doi.org/10.1038/nature14539
- **NICC Wildland Fire Summary and Statistics annual reports**, 2022.
- **Nwankpa, Chigozie, et al**. "Activation functions: Comparison of trends in practice and research for deep learning." arXiv preprint arXiv:1811.03378 (2018).
- **Oktay, Ozan, et al**. "Attention u-net: Learning where to look for the pancreas." arXiv preprint arXiv:1804.03999 (2018).

- **Oktay, Ozan, et al**. "Attention u-net: Learning where to look for the pancreas." *arXiv preprint arXiv:1804.03999* (2018).

- **Pelletier, Claude**. Méthodologie de détection des feux de forêt à partir d'images satellitaires NOAA. Université du Québec à Chicoutimi, 2001.
- **Suwei Yang**, Massimo Lupascu and Kuldeep S. Meel, Predicting Forest Fire Using Remote Sensing Data And Machine Learning, School of Computing, National University of Singapore 2 Department of Geography, National University of Singapore, 2021
- **Shamsoshoara, Alireza, et al**. "Aerial Imagery Pile burn detection using Deep Learning: the FLAME dataset." *Computer Networks* 193 (2021): 108001.
- **Samuel, Arthur L.**“Some Studies in Machine Learning Using the Game of Checkers.” *IBM J. Res. Dev.* 3 (1959): 210-229.
- **S. Albawi, T. A. Mohammed and S. Al-Zawi**, "Understanding of a convolutional neural network," 2017 International Conference on Engineering and Technology (ICET), 2017, pp. 1-6, doi: 10.1109/ICEngTechnol.2017.8308186.
- **Shifa Zhong, Kai Zhang, Majid Bagheri et al**, Environmental Science & Technology 2021 55 (19), 12741-12754 DOI: 10.1021/acs.est.1c01339
- **Tran, S.-T.; Cheng, C.-H.; Nguyen, T.-T.; Le, M.-H.; Liu, D.-G**. TMD-Unet: Triple-Unet with Multi-Scale Input Features and Dense Skip Connection for Medical Image Segmentation. Healthcare 2021, 9, 54. https://doi.org/10.3390/ healthcare9010054
- **Tan, Mingxing, and Quoc Le**. "Efficientnet: Rethinking model scaling for convolutional neural networks." *International conference on machine learning*. PMLR, 2019.

- **Tan, Mingxing, and Quoc Le**. "Efficientnet: Rethinking model scaling for convolutional neural networks." International conference on machine learning. PMLR, 2019.

- **Vaswani, Ashish, et al**. "Attention is all you need." Advances in neural information processing systems 30 (2017).
- **Yamashita, R., Nishio, M., Do, R.K.G. et al**. Convolutional neural networks: an overview and application in radiology. Insights Imaging 9, 611–629 (2018). https://doi.org/10.1007/s13244-018-0639-9
- **Zhao, Yi, et al**. "Saliency detection and deep learning-based wildfire identification in UAV imagery." *Sensors* 18.3 (2018): 712.