
MINI PROJET PYTHON

Présenté à

**L'Ecole Nationale d'Electronique et des
Télécommunications de Sfax**

Génie Informatique Industrielle

Par

Guerchi Jihen

**Apprentissage profond et traitement d'image
pour la détection du feu à base de CNN et
langage Python**

Dr. Koubaa Yasmine

Encadrant

Année Universitaire : 2022-2023

Remerciements

En préambule à ce projet je remercie Dieu, tout puissant, qui m'a aidé et m'a donné la patience, le courage et la volonté d'entamer et de terminer ce projet.

Je souhaite adresser mes remerciements les plus sincères aux personnes qui m'ont apporté leur aide et qui ont contribué à l'élaboration de ce Mini projet étude qu'à la réussite de cette formidable année universitaire.

Ces remerciements vont tout d'abord au corps professoral et administratif de l'école nationale de l'électronique et de communication de Sfax pour la richesse et la qualité de leurs enseignements et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.

Je tiens à remercier sincèrement Madame Koubaa Yasmine enseignante à l'ENET'COM de Sfax et notre encadrant dans ce mini projet, elle a toujours montré l'écoute et la disponibilité tout au long de la réalisation et des séances de TP , ainsi pour l'inspiration, son soutien moral, son encouragement, ses conseils bénéfiques et le temps qu'elle a bien voulu me consacrer pour que ce projet voit le jour.

Sommaire

Sommaire.....	iii
Liste des figures	v
Liste des Tableaux	vi
Liste des abréviations.....	vii
Introduction Générale	1
Chapitre 1 :.....	2
Etat de l’art.....	2
1.1. Introduction.....	3
1.2. Etude de l’existant	3
1.2.1. Techniques classiques de détection d’incendie.....	3
1.2.2. Limites et inconvénients des détecteurs classiques.....	4
1.3. Contexte du projet	5
1.3.1. Problématique	5
1.3.2. Solution proposée.....	5
1.3.3. Description et objectifs du projet	5
1.4. Incendie et feux de forêt en Tunisie	6
1.7. Conclusion	8
Chapitre2 :.....	9
Généralités sur l’intelligence artificielle	9
3.1. Introduction.....	10
3.2. Présentation de l’Intelligence Artificielle.....	10
3.3. Domaines d’applications	11
3.4. Techniques de l’intelligence artificielle	12
3.4.1. Machine Learning	12
3.4.2. Deep Learning.....	18

3.5.	Algorithme proposé.....	23
3.5.1.	Approche proposée.....	23
3.5.2.	Etude comparative.....	24
3.6.	Conclusion	25
Chapitre 3 :.....		i
Réalisation Pratique et Validation.....		i
4.1.	Introduction.....	ii
4.2.	Choix de logiciel	ii
4.2.1.	Langage de programmation	iv
4.2.4.	Outils et bibliothèques utilisés	iv
4.3.	Mise en oeuvre de l'environnement de travail	Error! Bookmark not defined.
4.3.3.	Préparation de l'environnement de travail...	Error! Bookmark not defined.
4.5.4.	Réalisation pratique	Error! Bookmark not defined.
4.6.	Détection d'incendie par Machine Learning	vi
4.6.1.	Base de données utilisée	vi
4.6.2.	Préparation du fichier xml	vi
4.6.3.	Traitement de la vidéo	vii
4.6.4.	Exécution	vii
4.7.1.	Augmentation du data.....	viii
4.7.2.	Architecture de modèle CNN.....	x
4.7.3.	Résultats obtenus et discussion	xi
4.8.2.	Programmation de module GPS.....	xii
4.8.3.	Exécution	14
4.8.4.	Réalisation réelle	14
4.11.	Conclusion	16
Conclusion générale et perspective		18
Bibliographiques		19

Liste des figures

Figure 1. 1: Détecteur de monoxyde	Figure 1. 2: Détecteur avertisseur4
Figure 1. 3: Objectifs de type SMART6	
Figure 1. 4: Incendie déclenchée à la montagne de Wargha Sakiet Sidi Youssef Kef 20217	
Figure 1. 5: Nombre des incendies en 2021 comparé à la moyenne 2008-20207	
Figure 3. 1: Intelligence Artificielle, Machine Learning et Deep Learning11	
Figure 3. 2: Exemple de voiture autonome12	
Figure 3. 3: Robot chirurgical12	
Figure 3. 4: Structure de l'apprentissage automatique (Machine learning).....13	
Figure 3. 5:Etapes de préparation de données pour les modèles à apprentissage automatique.....15	
Figure 3. 6: Illustration du KPPV16	
Figure 3. 7: Support Vector Machine (SVM).....17	
Figure 3. 8: Exemple de régression linéaire17	
Figure 3. 9: Réseau RNN19	
Figure 3. 10: L'algorithme Yolo (You Only Look Once).....19	
Figure 3. 11: Réseau neuronal convolutif20	
Figure 3. 12: Architecture simple de CNN20	
Figure 3. 13: Max Pooling sur une fenêtre 2 x222	
Figure 3. 14: Exemple d'opération entièrement connecté.....23	
Figure 3. 15: Exemple montrant l'étiquette codée de la couche de sortie CNN23	
Figure 3. 1 : Logo Anaconda..... ii	
Figure 3. 2 : LOGO Google colab iii	
Figure 3.3: Logo du classificateur cascade iii	
Figure 3. 4: Logo de la plateforme ThingsBoard iii	
Figure 3. 6: Logo d'Open CV..... iv	

Liste des Tableaux

Tableau 3. 1: Avantages et inconvénients de KNN.....	16
Tableau 3. 2: Différentes couches de cnn.....	21
Tableau 3. 3: Comparaison selon les caractéristiques entre les deux types d'apprentissages. .	24
Tableau 4. 1: environnement matériel	Error! Bookmark not defined.
Tableau 4. 2: Utilité des logiciels par rapport au projet	ii

Liste des abréviations

DAAF : Détecteur Autonome Avertisseur de Fumée

SMART : Spécifique Mesurable Atténuable Réaliste Temporel

EFFIS: European Forest Fire Information System

GPS: Global Positioning System

IA: Intelligence Artificielle

ML: Machine Learning

DL: Deep Learning

KPPV: K plus porches voisins

SVM: Support Vector Machine

RNN: Recursive Neural Network

MLP: Multilayer Perceptron MLP

CNN: Convolutional Neural Network

Yolo: You Only Look Once

SSD: Single Shot Multibox Detector

CSS: Cascading Style Sheets

HTML: Hyper Text Markup Language

XML: Language de balisage extensible

Open CV: Open-Source Computer Vision Library

DA: Data Augmentation

Introduction Générale

Les feux et les incendies de forêt influencent de plusieurs façons sur la diversité biologique. Ils sont une importante source d'émission de carbone et participent au réchauffement de la planète et aux changements dans la biodiversité.

Toutes ces conséquences atroces sont causées par les incendies répétitifs qui menacent alors la nature et contribuent au déséquilibre environnemental.

La lutte contre les incendies demeure un sujet primordial qui nécessite beaucoup de ressources matérielles et humaines car il est lié à plusieurs facteurs tel que les facteurs climatiques (vent, sécheresse...) et le facteur de temps qui joue un rôle important dans la maîtrise de feu d'où l'emploi de plusieurs détecteurs.

Les technologies traditionnelles de détection des incendies et des feux telles que les détecteurs de fumée, ne sont pas adaptées aux grands espaces. La détection classique des fumées est difficile en raison de limites de technologies cités précédemment, des fausses alarmes, des retards de détection et d'autres soucis qui se manifestent fréquemment alors il est essentiel d'avoir un outil embarqué permettant de détecter les potentiels de départ du feu en temps réel, de mettre à jour la base de données et d'avoir un taux de reconnaissance élevé.

La surveillance des forêts et des réserves naturelles qui balayent plusieurs hectares est assurée généralement par les agents forestiers qui essaient de superviser l'espace tout le long de la journée. Il paraît bénéfique d'employer des nouvelles technologies comme l'intelligence artificielle et l'internet des objets.

Dans ce cadre précis et dans le cadre d'un mini projet en employant le langage de programmation python. On est chargé de réaliser un modèle de classification d'image qui permet de distinguer la présence du feu qui sera intégré par la suite dans un système embarqué qui permet de reconnaître le feu dans un flux vidéo en temps réel, de localiser le système et de traquer sa trajectoire pendant son déplacement ainsi que sa supervision via une plateforme en ligne. En résumant, dans ce mini projet on a essayé de développer la partie importante du projet qui nécessite beaucoup de recherche et de découverte pour atteindre l'objectif souhaité.

Les différentes étapes suivies pour la réalisation de ce projet sont présentées par le biais du présent rapport qui s'articule autour de quatre chapitres à savoir :

- Le premier chapitre aura pour but de représenter une véritable étude de l'existant, le contexte général du projet, quelques généralités sur les incendies en Tunisie ainsi que les objectifs spécifiques du projet.
- Le deuxième chapitre présentera quelques définitions sur l'intelligence artificielle, Machine Learning et Deep Learning en se basant sur les techniques récemment utilisées.
- Le dernier chapitre sera consacré à la réalisation pratique de notre projet en mettant l'accent sur les différentes parties de mise en œuvre au niveau du code.

Et enfin, on finira par une conclusion qui résume notre travail en présentant les éventuelles perspectives.

Chapitre 1 :

Etat de l'art

1.1. Introduction

L'intelligence artificielle est devenue l'un des domaines les plus dynamiques et les plus importants de la technologie, avec des progrès rapides dans la recherche et le développement de nouvelles applications. Il peut servir d'outil essentiel de lutte contre les incendies pour toutes personnes chargées par les services d'incendie en particulier dans et autour des centres urbains.

Dans ce chapitre on s'intéresse à présenter le contexte général du projet, les problématiques, les limites des technologies déjà existantes dans le marché pour la détection d'incendie. Par la suite on présentera la solution proposée, ses objectifs, sa valeur ajoutée et sa contribution efficace au cours des interventions, et enfin on termine par l'énoncé de quelques techniques utilisées dans la mise en œuvre pratique.

1.2. Etude de l'existant

De nos jours, où quand l'informatique contrôle tout le monde, il existe des systèmes sécurisés de télésurveillance et de contrôle. Dans cette partie, on va étudier quelques techniques classiques de détection d'incendie ainsi que leurs limites et inconvénients.

1.2.1. Techniques classiques de détection d'incendie

Il existe de nombreux systèmes qui ont pour rôle la détection de fumée, les plus reconnus dans le marché sont le détecteur avertisseur autonome de fumée (DAAF) et le détecteur de monoxyde de carbone.

➤ Détecteur de monoxyde de carbone

Le détecteur de monoxyde de carbone permet de vérifier que l'atmosphère contrôlée ne contient pas de dose anormale de monoxyde de carbone, reflet d'une combustion incomplète. Ce gaz mortel, inodore et incolore ne peut être identifié que par un détecteur spécifique. Ce type de détecteur diminue les risques d'incendies en émettant un signal sonore à la présence d'un niveau dangereux de monoxyde de carbone.

➤ Détecteur avertisseur autonome de fumée (DAAP)

Le détecteur avertisseur autonome de fumée est un détecteur de fumée couplé à une alarme. Ce petit appareil détecte la fumée dans les premiers instants d'un incendie et

déclenche une alarme. Il donne le temps soit de maîtriser un feu naissant, soit de fuir s'il y a trop de fumée. Très facile à poser, ce type d'appareil est en général alimenté par une pile 9 V.

Les figures 1.1 et 1.2 représentent respectivement le détecteur de monoxyde de carbone 9 V et le détecteur avertisseur autonome de fumée.



Figure 1. 1: Détecteur de monoxyde de carbone 9v



Figure 1. 2: Détecteur avertisseur autonome de fumée

Les autres types de détecteur de fumée sont basés sur les mêmes principes. Ils sont tous presque basés au moins sur un capteur surveillant qui mesure un paramètre et le transforme en un signal électrique exploitable, d'une partie traitement qui analyse les informations délivrées et d'une partie alarme.

1.2.2. Limites et inconvénients des détecteurs classiques

Tous systèmes de détection classiques doivent avoir quelques contraintes et désavantages, parmi lesquels on cite les suivants :

- ✓ Une zone de détection assez limitée et réduite et ne doit pas dépasser une surface donnée d'où l'association de plusieurs capteurs dans les grands bâtiments.
- ✓ Généralement utilisés dans les espaces fermés tels que les logements, les résidences et les extensions de maison.
- ✓ Non performants dans les zones vastes et étendues.
- ✓ Durée de vie limitée qui influe de façon directe sur son fonctionnement normal.
- ✓ Risque d'alarmes non justifiées
- ✓ L'alarme du détecteur est peut-être dominée par le bruit des chaînes stéréo, téléviseurs, climatiseurs et d'autres appareils électroniques
- ✓ Disfonctionnement lors de coupure de courant

1.3. Contexte du projet

1.3.1. Problématique

Les forêts, les montagnes et les sites loin de la population témoignent de nombreuses lacunes au niveau de la surveillance et de détection d'incendie, parmi ces lacunes, on cite :

- ✓ Perte de temps de détection et d'intervention en urgence surtout lors des incendies.
- ✓ Manque de moyen sophistiqué et innovant avec un système de supervision en temps réel faible dans les montagnes et les forêts.
- ✓ Absence d'outils de détection de feux dans les forêts.
- ✓ Systèmes d'information géographique faibles avec un manque d'informations nécessaires pour quantifier le risque.
- ✓ Les vérifications reposent encore sur les gardiens des forêts ce qui introduit les fausses alertes des citoyens et les complots avec ceux qui provoquent des incendies notamment les pratiquants de contrebande.
- ✓ Perte de vie humaines, de ressources naturelles et des ressources animales.

1.3.2. Solution proposée

Notre projet a été proposé dans le but de répondre à un ensemble de besoins spécifiques et précis et qui permet de contribuer efficacement à lutter contre les incendies de forêt par utilisation d'un système embarqué qui sera localisé au niveau des forêts.

1.3.3. Description et objectifs du projet

Après l'étape de fixation des problématiques et l'étude de l'existant à laquelle on a décrit les limitations liées aux détecteurs classiques d'incendie qui sont performants dans les espaces à une surface assez réduite et non pas adaptable aux espaces vastes, on passe à présenter notre solution présentée par notre drone détecteur d'incendie qui analyse une vidéo en temps réel via le concept de traitement d'image et alerte l'utilisateur en cas de présence de feu. On essaye de rendre nos objectifs de type SMART (S pour spécifique, M pour mesurable, A pour atteignable, R pour réaliste, T pour temporel).



Figure 1. 3: Objectifs de type SMART

On cite ci-dessous les principaux objectifs de notre projet :

- ✓ Détecter les potentiels de départ de feu.
- ✓ Alerter l'utilisateur en cas de présence de feu.
- ✓ Qualifier et quantifier le risque.
- ✓ Surveiller l'air d'essai en temps réel pour la collection des informations riches et capables d'aider efficacement les décideurs pour la prise de décisions.
- ✓ Localiser géographiquement l'incendie.
- ✓ Renforcer et améliorer le système d'information géographique.
- ✓ Contribuer à intervenir en urgence et en temps réel lors des catastrophes...
- ✓ Contribuer à l'amélioration de systèmes de lutte contre les malfaiteurs de contre bande.

1.4. Incendie et feux de forêt en Tunisie

L'état des incendies des forêts a atteint un niveau très préoccupant dans différents pays du monde, notamment aux Etats Unis Américains et en région méditerranéenne : Turquie, Grèce, Espagne, Algérie et Tunisie.

En prenant le cas de la Tunisie, 45 incendies (d'une superficie brûlée supérieure à 30 ha) correspondant à une superficie cumulée de 26854 ha ont été cartographiés par la plateforme EFFIS (Européen Forest Fire Information System) qui fournit de l'information fiable sur les incendies des forêts en Europe et dans les pays du moyen orient et de l'Afrique du Nord à la date du 23 aout 2021. Selon EFFIS, cette superficie représente en moyenne 80% environ de la surface totale brûlée si l'on tient compte des feux de forêts inférieur à 30ha. Cette superficie représente 4,64 fois la valeur moyenne des années 2008-2020 (5789 ha) à la même date comme représente les deux figures ci-dessous. La surface brûlée en Tunisie dépasse 2,2% de la superficie totale des forêts [1].

Les figures 1.4 et 1.5 représentent respectivement un exemple d'incendie déclenché à la montagne de Wargha Sakiet Sidi Youssef en 2021 et l'évolution du nombre des incendies en 2021 comparé à la moyenne 2008-2020.



Figure 1. 4: Incendie déclenchée à la montagne de Wargha Sakiet Sidi Youssef Kef 2021

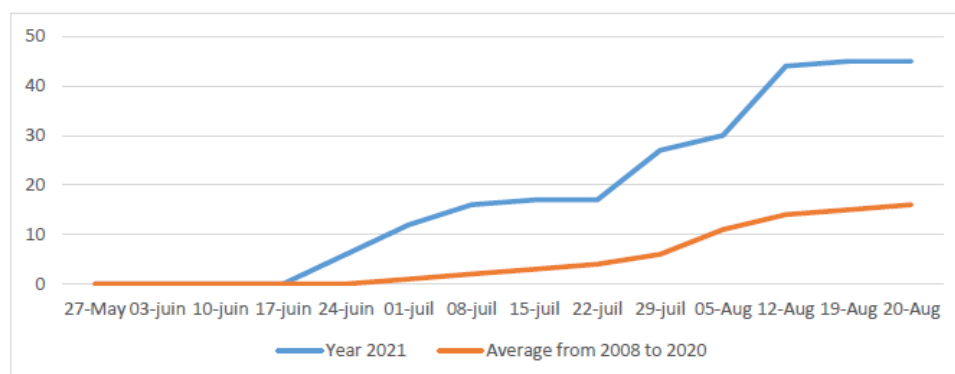


Figure 1. 5: Nombre des incendies en 2021 comparé à la moyenne 2008-2020

La progression intensive du nombre d'incendie par an et l'augmentation remarquable de la superficie brûlée dans les pays méditerranéens et plus spécifiquement la Tunisie déjà étudiée dans la partie précédente est due au manque de moyen d'alerte rapide des incendies alors l'emploi des drones et des technologies modernes sera une nécessité pour faciliter la détection, le temps de réponse, l'intervention rapide pour l'extinction et la gestion des urgences.

1.5. Conclusion

Dans ce chapitre on a présenté quelques généralités sur les feux de forêts en Tunisie, leur influence négative sur les ressources de l'espace forestier tunisien, On a bien traité la problématique après une bonne étude des solutions existantes sur le marché, afin de fixer les objectifs sur lesquels notre solution est basée.

Comme l'étude de ce projet est établie dans le cadre d'un mini projet on a besoin en premier lieu de connaître les technologies utilisés qui fera l'objet du chapitre suivant



Chapitre 2:

Généralités sur l'Intelligence Artificielle

3.1. Introduction

L'IA (Intelligence Artificielle) est basée sur une démarche d'apprentissage afin de reproduire une partie de l'intelligence humaine à travers une application, un système ou un processus. La reconnaissance faciale, les perceptions visuelles et autres sont des exemples de systèmes d'intelligence artificielle existants.

Dans ce chapitre, on présente un aperçu sur l'intelligence artificielle, son historique, ses diverses applications et ses techniques les plus célèbres dans la classification intelligente, on propose par la suite des approches pour la détection rapide des incendies basées sur des techniques de machine Learning et Deep Learning.

3.2. Présentation de l'Intelligence Artificielle

L'intelligence artificielle est un ensemble de techniques permettant à des machines d'accomplir des tâches et de résoudre des problèmes normalement réservés aux êtres humains. Elle désigne la possibilité pour une machine de reproduire des comportements humains, tels que le raisonnement, la planification et la créativité.

L'IA est née dans les années 50, quand une poignée de pionniers du domaine naissant de l'informatique, ont commencé à se demander si les ordinateurs pouvaient être amenés à penser comme l'être humain et à agrandir l'effort d'automatiser les tâches intellectuelles normalement effectuées par les humains.

L'IA est un domaine général qui englobe l'apprentissage automatique et l'apprentissage en profondeur, mais qui comprend également beaucoup plus d'approches qui n'impliquent aucun apprentissage. Les programmes d'échecs initiaux, par exemple, ne concernaient que des règles codées en dur élaborées par des programmeurs et ne se qualifiaient pas comme apprentissage automatique [2].

Pour y parvenir, trois composants sont nécessaires :

- Des systèmes informatiques
- Des données avec des systèmes de gestion
- Des algorithmes d'IA avancés (code)

La figure 2.1 explique la relation qui existe entre l'intelligence artificielle, Machine Learning et Deep Learning.

En premier lieu l'IA a émergé dans ces domaines. Plus tard, l'apprentissage automatique (Machine Learning) a prospéré. Actuellement vers l'apprentissage approfondi (Deep Learning).

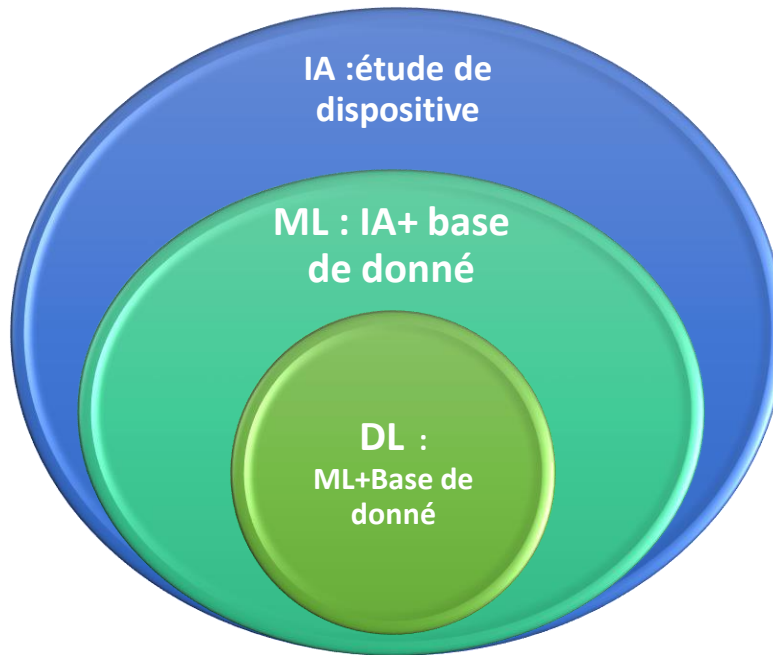


Figure 2. 1: Intelligence Artificielle, Machine Learning et Deep Learning

3.3. Domaines d'applications

L'intelligence artificielle se retrouve implémentée dans un nombre grandissant de domaines d'applications. Parmi lesquels on peut citer les suivants :

- ✚ Les réseaux sociaux
- ✚ Les applications de navigation
- ✚ Transport
- ✚ Santé

Les figures 2.2 et 3.2 représentent respectivement un exemple de voiture autonome et d'un robot chirurgical exploitant l'IA respectivement dans le transport et dans la santé.



FIGURE 3. 2: EXEMPLE DE VOITURE AUTONOME



FIGURE 2. 3: ROBOT CHIRURGICAL

3.4. Techniques de l'intelligence artificielle

Comprendre les dernières avancées en matière d'intelligence artificielle peut sembler fastidieux et bénéfique, mais cela revient en réalité à deux concepts : l'apprentissage automatique ou ML (Machine Learning) et l'apprentissage en profondeur ou DL (Deep Learning).

3.4.1. Machine Learning

L'apprentissage automatique est un domaine de recherche en informatique qui traite des méthodes d'identification et de mise en œuvre de systèmes et algorithmes par lesquels un ordinateur peut apprendre, ce domaine a souvent été associé à l'intelligence artificielle et plus spécifiquement l'intelligence computationnelle.

L'intelligence computationnelle est une méthode d'analyse de données qui pointe vers la création automatique de modèles analytiques. Autrement dit, permettant à un ordinateur d'élaborer des concepts, d'évaluer, prendre des décisions et prévoir les options futures. [3]

Le processus d'apprentissage nécessite un ensemble de données comme suit :

- ❖ **Ensemble de données pour l'entraînement** : c'est la base de connaissance utilisée pour entraîner, notre algorithme d'apprentissage, pendant cette phase, les

paramètres du modèle peuvent être réglés et ajustés en fonction des performances obtenues.

❖ **Ensemble de données pour le test** : cela est utilisé juste pour évaluer les performances du modèle sur les données non-vues.

La théorie de l'apprentissage utilise des outils mathématiques dérivés de la théorie des probabilités et de la théorie de l'information. Cela vous permet d'évaluer l'optimalité de certaines méthodes par rapport aux autres. On peut citer deux types d'algorithme d'apprentissage automatique comme représente la figure ci-dessous :

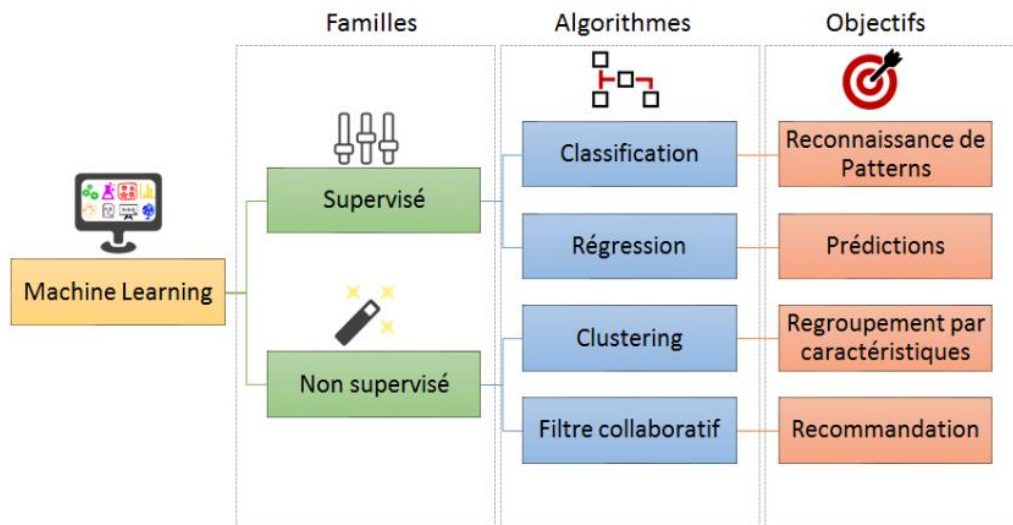


FIGURE 2. 4: STRUCTURE DE L'APPRENTISSAGE AUTOMATIQUE (MACHINE LEARNING)

🔗 Apprentissage supervisé

L'apprentissage supervisé est la tâche d'apprentissage automatique la plus simple et la plus connue. Il est basé sur un certain nombre d'exemples pré classifiés, dans lesquels est connu à priori la catégorie à laquelle appartient chacune des entrées utilisées comme exemples. Dans ce cas, la question cruciale est le problème de généralisation, après l'analyse d'un échantillon d'exemples, le système devrait produire un modèle qui devrait fonctionner pour toutes les entrées possibles.

L'ensemble de données pour l'entraînement, est constitué de données étiquetées, c'est-à-dire d'objets et de leurs classes associées. Cet ensemble d'exemples étiquetés constitue donc l'ensemble d'apprentissage [4].

Apprentissage non supervisé

La deuxième classe d'algorithmes d'apprentissage automatique est appelée apprentissage non supervisé, dans ce cas, on n'étiquète pas les données au préalable, on laisse plutôt l'algorithme arriver à sa conclusion. Ce type d'apprentissage est important car il est beaucoup plus commun dans le cerveau humain que l'apprentissage supervisé.

Les algorithmes d'apprentissage non supervisé sont particulièrement utilisés dans les problèmes de clustering, dans lesquels, étant donné une collection d'objets, on veut être en mesure de comprendre et de montrer leurs relations. Une approche standard consiste à définir une mesure de similarité entre deux objets, puis à rechercher tout groupe d'objets plus similaires les uns aux autres, par rapport aux objets des autres clusters. Par exemple, dans le cas précédent des e-mails spam/ non spam, l'algorithme peut être capable de trouver des éléments communs à tous les spam (par exemple, la présence de mots mal orthographiés). Bien que cela puisse fournir une classification meilleure qu'aléatoire, il n'est pas clair que les spam/non spam puissent être facilement séparés [5].

Détails sur l'extraction et nettoyage de données

❖ **Extraction de données**

Le prétraitement des données dans les systèmes d'apprentissage automatique implique à la fois l'extraction des données et des caractéristiques. Le prétraitement des images consiste à convertir des données brutes en données préparées. Par ailleurs, l'étape de l'extraction des caractéristiques traite ensuite ces données préparées de manière à créer des attributs utiles pour l'apprentissage automatique [6].

❖ **Données brutes et données transformées**

Les données à traiter par les systèmes d'apprentissage peuvent être brutes (dans un lac de données) ou transformées (dans un entrepôt de données). Les données brutes sont des données sous leur forme source non traitées pour l'apprentissage automatique. Les données transformées d'un entrepôt de données peuvent avoir été transformées en se basant sur leur forme d'origine pour être utilisées à des fins d'analyse. En plus, les données envoyées à partir de systèmes de diffusion en continu qui seront utilisées par des modèles d'apprentissage automatique pour la prédiction sont considérées comme des données brutes [6].

❖ Prétraitement et nettoyage des données

Le prétraitement des données comprend diverses opérations. Chaque opération vise à aider le système d'apprentissage automatique à élaborer de meilleurs résultats de classification et prédiction. Le nettoyage de données consiste généralement à la suppression ou correction des données contenant des valeurs corrompues ou non valides au sein des données brutes, ou la suppression des données pour lesquels il manque un grand nombre d'informations. L'amélioration de la qualité des caractéristiques peut être réalisée à travers une égalisation d'histogramme, une normalisation, interpolation des valeurs manquantes, le découpage des anomalies et l'ajustement des valeurs ayant des distributions asymétriques et la simple suppression de caractéristiques si nombreuses valeurs sont manquantes [6].

La figure 2.5 illustre les étapes de préparation de données pour les modèles à apprentissage automatique [6].

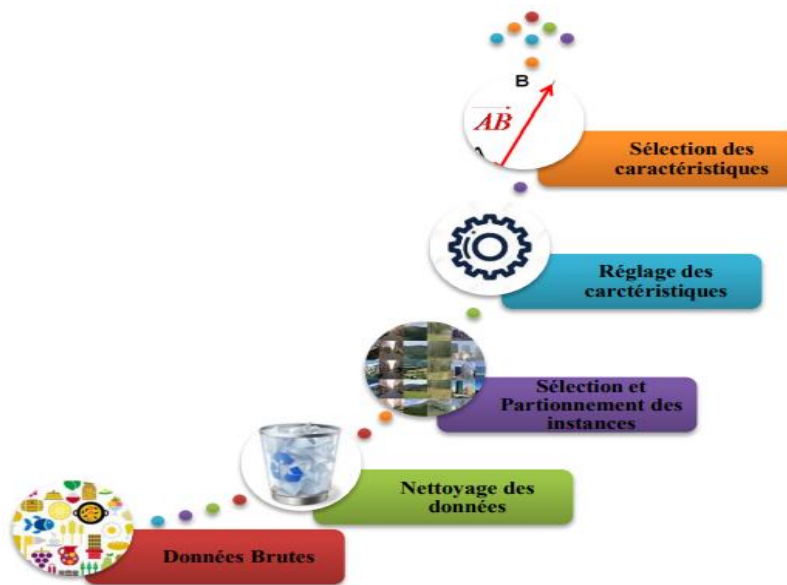


FIGURE 2. 5:ÉTAPES DE PREPARATION DE DONNEES POUR LES MODELES A APPRENTISSAGE AUTOMATIQUE

❖ Etude de quelques approches basées sur ML

Il existe des nombreuses approches basées sur machine Learning parmi lesquelles on cite :

- **Les k plus proches voisins (KPPV)**

L'algorithme des K-Nearest Neighbors (KNN) (K plus proches voisins) (figure 2.6) est un algorithme de classification supervisé. Chaque observation de l'ensemble

d'apprentissage est représentée par un point dans un espace à n dimensions ou n est le nombre de variables prédictives. Pour prédire la classe d'une observation, on cherche les k points les plus proches de cet exemple. La classe de la variable cible, est celle qui est la plus représentée parmi les k plus proches voisins. Il existe des variantes de l'algorithme ou on pondère les k observations en fonction de leur distance euclidienne à l'exemple dont on veut classer, les observations les plus éloignées de notre exemple seront considérées comme moins importantes.

Exemple : Pour $k = 3$ la classe majoritaire du point central est la classe b, mais si on change la valeur du voisinage $K = 6$ la classe majoritaire devient la classe A.

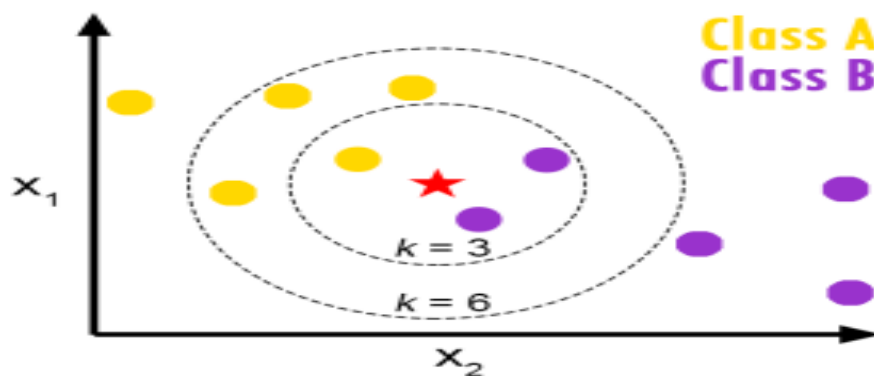


FIGURE 2. 6: ILLUSTRATION DU KPPV

Tableau 2. 1: Avantages et inconvénients de KPPV

Avantage	Inconvénients
✓ Simple à concevoir	✓ Sensible aux bruits ✓ Pour un nombre de variable prédictif très grands, le calcul de la distance devient très coûteux et lent.

- **Support-vector machine (SVM)**

Les machines à vecteur support (Support Vector Machines) (figure 2.7) sont une méthode de machine Learning très utilisée dans les classifications. Leur but est d'apprendre à placer une délimitation entre deux classes. La frontière choisie doit être aussi lointaine que possible des premiers éléments de chaque côté, ceci afin de ne pas biaiser la classification et d'optimiser la capacité de généralisation. De façon générale, un SVM aura pour tâche de

trouver un hyperplan qui sépare les catégories. Dès lors, un tel classificateur déterminera la classe du nouvel objet selon sa position dans l'hyperplan par rapport à la frontière. En effet, les hyperplans sont placés de sorte que les catégories sont séparées par un « fossé ». Les nouveaux objets sont placés sur le plan et la classification s'effectuera en fonction du côté du fossé vers lequel tombe ou penche l'objet. Ce système peut générer ce type de frontières :

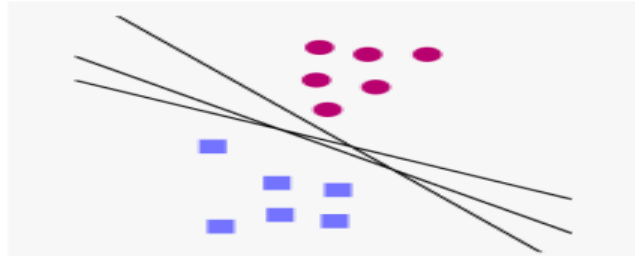


FIGURE 2. 7: SUPPORT VECTOR MACHINE (SVM)

- **La régression linéaire**

L'algorithme de régression linéaire est un algorithme d'apprentissage supervisé. Le modèle permet à partir de la variable cible ou noté aussi la variable à expliquer Y de faire une prédiction grâce à des variables explicatives X pu dite prédictives.

Un modèle de régression linéaire est un modèle de machine learning dont la variable cible Y est quantitative tandis que la variable X peut être quantitative ou qualitative

L'objectif est de trouver une fonction dite de prédiction ou fonction cout qui décrit la relation entre X et Y c'est-à-dire qu'à partir de valeurs connues de X, on arrive à donner une prédiction des valeurs de Y. La forme générale de la fonction recherchée est :

$$Y = f(X) \text{ avec } f(X) \text{ est une fonction linéaire}$$

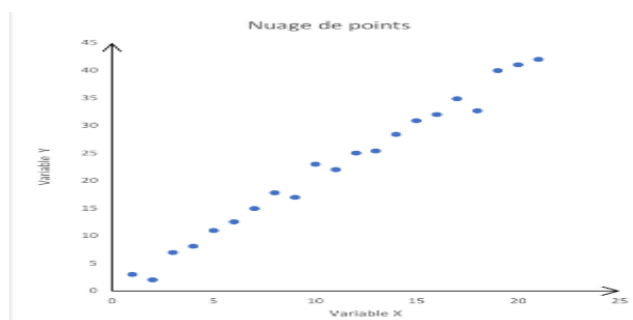


FIGURE 2. 8: EXEMPLE DE REGRESSION LINEAIRE

3.4.2. Deep Learning

L'apprentissage en profondeur (Deep Learning) est un domaine de recherche sur l'apprentissage automatique basé sur un type particulier de mécanisme d'apprentissage.

Il est caractérisé par l'effort de créer un modèle d'apprentissage à plusieurs niveaux, dans lequel les niveaux les plus profonds prennent en compte les résultats des niveaux précédents, les transformant et en faisant toujours plus d'abstraction. Cet aperçu des niveaux d'apprentissage est inspiré par la façon dont le cerveau humain traite l'information et apprend en réagissant aux stimuli externes. Chaque niveau d'apprentissage correspond, par hypothèse, à l'une des différentes zones qui composent le cortex cérébral.

➤ Exemple d'applications de Deep Learning

Les applications exploitantes le Deep Learning sont utilisées dans divers secteurs, de la conduite automatisée aux dispositifs médicaux. Grâce au Deep Learning on peut :

- ❖ Faire une colorisation des images en noir et blanc.
- ❖ Ajouter des sons à des films silencieux.
- ❖ Faire de la traduction automatique.
- ❖ Faire la classification des objets en photographies.
- ❖ Générer l'écriture automatique.
- ❖ Générer la légende d'image.
- ❖ Faire des jeux automatiques.

➤ Etude de quelques approches

❖ Recursive Neural Network (RNN)

Les réseaux de neurones artificiels récurrents sont très largement utilisés en reconnaissance de la parole ainsi qu'en traitement des langues de façon générale.

Ces modèles séquentiels sont le plus souvent construits de manière à pouvoir reconnaître les caractéristiques depuis des données et d'utiliser des patterns pour prédire le scénario suivant le plus probable.

Un RNN (Figure 2.9) est apte à faire passer l'information dans les deux sens (vers l'avant et vers l'output). Un problème majeur de ce type de réseau, partagé par les perceptrons multicouches (multi layer perceptron MLP), est ce que l'on nomme le « vanishing »

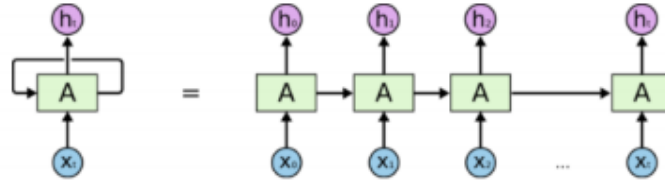


FIGURE 2. 9: RESEAU RNN

❖ Détecteur des objets en temps réel : YOLO

Les détecteurs d'objets à coup unique « Single Shot Detectors » détectent les frontières des régions et les classifient en même temps. Comme il s'agit d'un processus en une seule étape, il est beaucoup plus rapide que les détecteurs régionaux. Il convient de noter que ce type de détecteurs est peu performant lorsqu'il s'agit d'objets de petite taille. Les algorithmes populaires qui utilisent cette méthodologie sont YOLO (You Only Look Once) et SSD (Single Shot multibox Detector). Sa grande force est sa rapidité : il peut être appliqué en temps réel (jusqu'à 45 images / seconde).

Yolo est plus rapide que des R-CNN aussi, car il découpe l'image en petits blocs et génère des tenseurs pour chaque bloc. Le principe du YOLO est de faire parcourir l'image une seule fois par un réseau de neurones profonds (d'où le nom de You Only Look Once), par opposition aux méthodes dites de régions (notamment utilisées par les modèles basés sur R-CNN généralement).

Le schéma de principe de l'algorithme Yolo est affiché dans la Figure 3.10 [13].

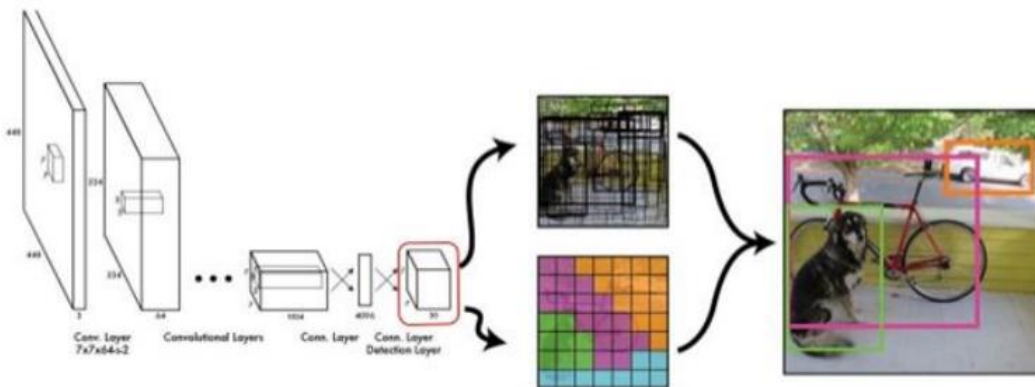


FIGURE 2. 10: L'ALGORITHME YOLO (YOU ONLY LOOK ONCE)

❖ Réseaux de neurones convolutionnels

Les réseaux CNN (figure 2.11) se concentrent principalement sur le fait que l'entrée sera composée d'images. Cela permet de centrer l'architecture à mettre en place pour répondre au mieux à la nécessité de traiter un type de données spécifiques. Les réseaux neuronaux convolutifs diffèrent des autres formes de réseaux neuronaux artificiels en ce sens. Qu'au lieu de se concentrer sur l'intégralité du domaine problématique, les connaissances sur le type spécifique d'entrées sont exploitées, cela permet à son tour de mettre en place une architecture réseau beaucoup plus performante [7].

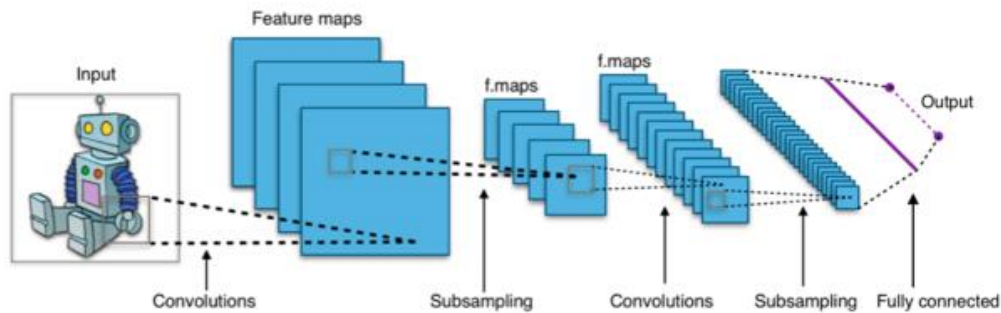


FIGURE 2. 11: RESEAU NEURONAL CONVOLUTIFS

Les CNN sont composés de trois types de couches : des couches convolutives, des couches de regroupement et des couches entièrement connectées. Lorsque ces couches sont empilées, une architecture CNN sera formée.

Une architecture CNN simplifiée pour la classification MNIST (Base de données pour la classification des chiffres) est illustrée à la figure 2.12 [8].

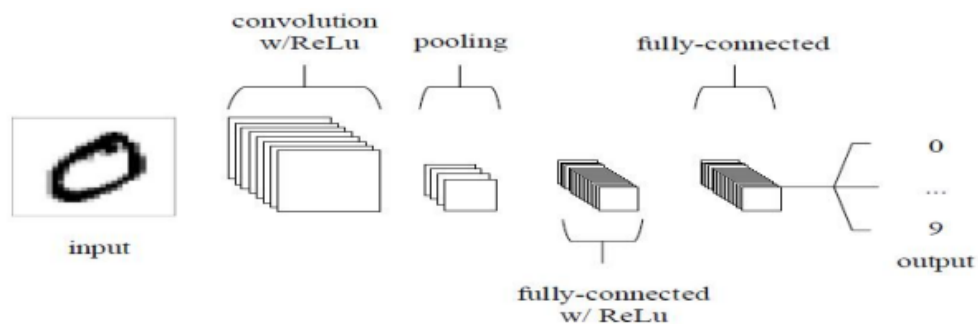


FIGURE 2. 12: ARCHITECTURE SIMPLE DE CNN

Le tableau 2.2 explique le rôle principal de chaque couche d'un réseau de neurone convolutionnels simple.

TABEAU 2. 2: DIFFERENTES COUCHES DE CNN

Couche de convolution (CONV)	Traite les données d'un champ récepteur
Couche de Pooling (POOL)	Compresser l'information en réduisant la taille de l'image intermédiaire, par un sous- échantionnage.
Couche de correction (Relu)	Appelée souvent par abus 'Relu' en référence à la fonction d'activation (Unité de rectification linéaire).
Couche entièrement connectée(FC)	La phase d'apprentissage finale, qui mappe les caractéristiques visuelles extraites aux sorties souhaitées, habituellement adaptable aux taches de classification.
Couche de sortie	Contient l'étiquette codée.

✓ Couches convolutives

Les couches convolutives constituent le noyau du réseau convolutifs. Ces couches se composent d'une grille rectangulaire de neurones qui ont un petit champ réceptif étendu à travers toute la profondeur du volume d'entrée. Ainsi, la couche convolutives est juste une convolution d'image de la couche précédente, où les poids spécifient le filtre de convolution.

La couche convolutives déterminera la sortie des neurones qui sont connectés aux régions locales de l'entrée par le calcul du produit scalaire entre leurs poids et la région connectée au volume d'entrée. Relu vise à appliquer une fonction d'activation « élémentaire » telle qu'une fonction sigmoïde à la sortie de l'activation produite par la couche précédente.

✓ Couche de Pooling

Après chaque couche convolutive, il peut y avoir une couche de Pooling. Cette couche sous échantillonne le long de la dimensionnalité spatiale de l'entrée donnée, ce qui réduira davantage le nombre de paramètres au sein de cette activation. Il y a plusieurs façons de faire cette mise en commun, comme prendre la moyenne ou le maximum, ou une combinaison linéaire prise par des neurones dans le bloc. Par exemple, la Figure 2.13 montre le max Pooling qui revient de prendre la valeur maximale de la sélection sur une fenêtre 2×2 .

Il existe plusieurs types de Pooling comme le mean Pooling et le sum Pooling mais le plus utilisé est le max Pooling car il est rapide à calculer et immédiat et permet de simplifier efficacement l'image.

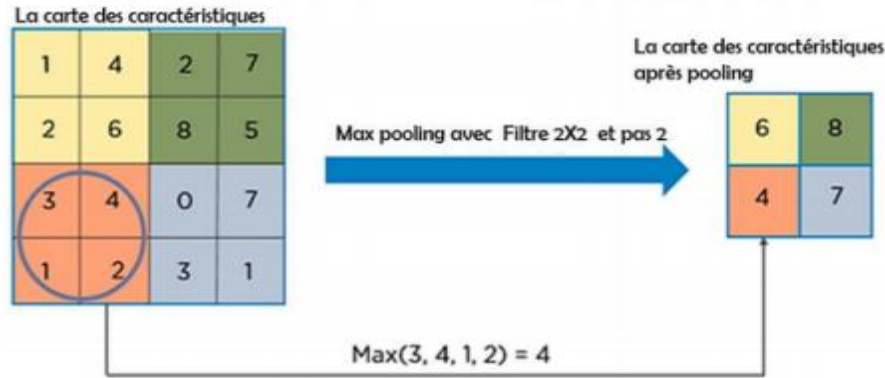


FIGURE 2. 13: MAX POOLING SUR UNE FENETRE 2 x2

✓ Couche totalement connectée

Enfin, après les couches de convolution et pooling, le raisonnement de haut niveau dans le réseau neuronal se fait via des couches totalement connectées.

Dans les réseaux de neurones convolutifs, chaque couche agit comme un filtre de détection pour la présence de caractéristiques spécifiques ou de motifs présents dans les données d'origine. Les premières couches d'un réseau convolutifs détectent des caractéristiques qui peuvent être reconnues et interprétées facilement. Les couches ultérieures détectent de plus en plus des caractéristiques plus abstraites. La dernière couche du réseau convolutifs est capable de faire une classification ultra-spécifique en combinant toutes les caractéristiques spécifiques détectées par les couches précédentes dans les données d'entrée.

Les couches totalement connectées font les mêmes tâches que celles des ANN standard et tenteront de produire des notes de classe à partir des activations, pour les utiliser pour la classification. Il est également suggéré d'utiliser Relu entre ces couches pour améliorer les performances.

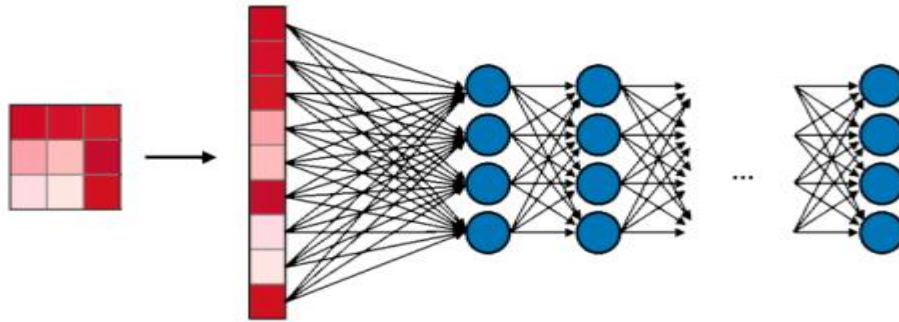


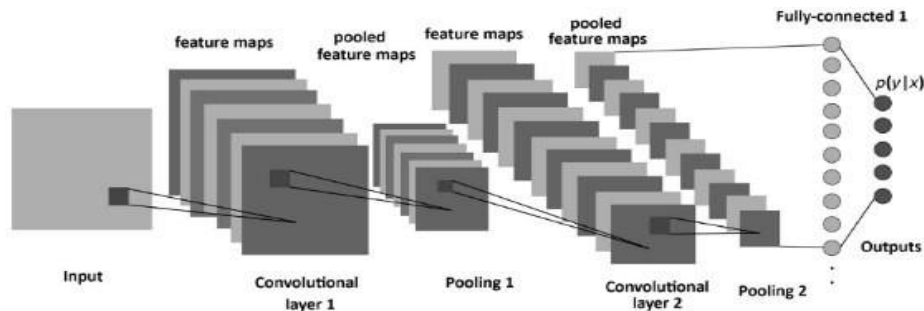
FIGURE 2. 14: EXEMPLE D'OPERATION ENTIEREMENT CONNECTE

✓ Couche de correction (Relu)

C'est une couche pour améliorer l'efficacité du traitement en intercalant entre les couches de traitement une couche qui va opérer une fonction mathématique (fonction d'activation) sur les signaux de sortie.

✓ Couche de sortie (output layer)

La couche de sortie contient l'étiquette qui est sous forme codée comme le montre-la figure suivante :

FIGURE 2. 15: EXEMPLE MONTRANT L'ETIQUETTE CODEE DE LA COUCHE DE SORTIE
CNN

3.5. Algorithme proposé

3.5.1. Approche proposée

Dans cette section, on présente les détails de notre approche proposée. En premier lieu, une approche d'apprentissage en profondeur basée sur le réseau de neurone convolutive et autre approche d'apprentissage automatique appliquée à la reconnaissance de feu.

3.5.2. Etude comparative

Dans cette partie, on s'intéresse à la comparaison des deux types d'apprentissage profond et automatique.

Tableau 2. 3: Comparaison selon les caractéristiques entre les deux types d'apprentissages.

Caractéristiques	Apprentissage en profondeur	Apprentissage automatique
Robustesse	Oui	Non
Quantité de la base de données	Enorme	Moyenne
Type de la base de données	Statique	Dynamique
Taille de la base de données	Dizaines de Go	Quelques Go
Apprentissage : temps de traitement	Très élevé	Peu élevé
Mémoire	Grande capacité	Moyenne capacité
Ressource	Nécessité des GPU	CPU classique
Intervention humaine	Oui	Non

Pour classer un modèle dans un processus d'apprentissage automatique, un classifieur est utilisé. Ce classifieur utilise les caractéristiques d'un objet pour identifier la classe à laquelle il appartient. Par exemple, si un objet est une voiture, le classificateur est formé pour identifier sa classe en lui fournissant des données d'entrée et en leur attribuant une étiquette qui ne nécessite pas une énorme base de données et des performances hardware hautes gammes. Par opposition à l'apprentissage en profondeur qui nécessite une énorme quantité de données ce qui oblige une grande capacité de mémoire et un temps de traitement et d'exécution assez élevé ce qui est non adaptable avec notre situation qui vise le temps réel. Finalement, on a opté le choix de machine Learning pour l'implémentation au niveau du

système embarqué choisi puisque on a essayé de tester notre prototype sur une carte Raspberry Pi 4 Model B (Partie facultative).

3.6. Conclusion

Dans ce chapitre, on a présenté en premier lieu l'intelligence artificielle, ses domaines d'applications, un abstrait sur les deux types d'apprentissages : machine Learning et Deep Learning, leurs particularités. On a effectué par la suite une étude comparative entre les deux approches proposées.

Chapitre 3 :

Réalisation pratique et validation

4.1. Introduction

Au début, on commence par l'initialisation du système, la mise en œuvre de l'environnement de travail, la réalisation et le développement des deux modèles d'apprentissage.

Une fois les modèles sont prêts, on implémente le modèle choisi dans la carte Raspberry pi puis on passe à la phase de test et validation qui va prouver le concept réalisé et valoriser l'efficacité de la plateforme de supervision qui facilite cette tâche.

4.2. Initialisation de système

4.2.1. Choix de logiciel

Au cours de développement de notre projet on a présenté les outils logiciels employés dans notre projet détaillés en fonction de leurs tâches dans le tableau ainsi que leurs définitions citées par la suite.

Tableau 3. 1: Utilité des logiciels par rapport au projet

Logiciel	Fonction par rapport au projet
Classificateur cascade	<ul style="list-style-type: none"> • Extraction les caractéristiques spécifiques de la base de données. • Entraînement du modèle.
Anaconda	Environnement du développement du modèle machine Learning et Deep Learning pour la détection en temps réel.
Google colab	Environnement de développement Python accessible en ligne.
ThingsBoard	Supervision du système.
Kaggle	Offre une base de données open source.

Anaconda

Anaconda est une distribution libre et open source des langages de programmation Python et R appliqué au développement d'applications dédiées à la science des données et à l'apprentissage automatique, qui vise à simplifier la gestion des paquets et de déploiement.



FIGURE 3. 1 : LOGO ANACONDA

Google colab

Google Colab ou Colaboratory est un service cloud, offert par Google (gratuit), basé sur Jupyter Notebook et destiné à la formation et à la recherche dans l'apprentissage automatique en se basant sur le langage python . Cette plateforme permet d'entraîner des modèles de Machine Learning directement dans le cloud. Sans donc avoir besoin d'installer quoi que ce soit sur notre ordinateur à l'exception d'un navigateur.



FIGURE 3. 2 : LOGO GOOGLE COLAB

Classificateur cascade

C'est un logiciel qui assure une méthode de détection d'un objet efficace proposée par Paul Viola et Micheal Jones. Il s'agit d'une approche basée sur l'apprentissage automatique d'où la fonction cascade est formée à partir d'un grand nombre d'image positive et négatives.



FIGURE 3.3: LOGO DU CLASSIFICATEUR CASCADE

ThingsBoard

ThingsBoard est une plateforme IoT open source destiné à la collecte, au traitement, à la visualisation et à la gestion des appareils. Il respecte tous les protocoles IoT standard tels que CoAP, MQTT et HTTP aussi rapidement que les déploiements cloud et sur site.



FIGURE 3. 4: LOGO DE LA PLATEFORME THINGSBOARD

Kaggle

Kaggle est une plateforme en ligne dédiée aux compétitions de science des données. Elle offre aux scientifiques des données du monde entier la possibilité de collaborer sur des projets de science des données, de résoudre des problèmes complexes et de participer à des compétitions en utilisant des ensembles de données fournis par des entreprises, des organisations gouvernementales et des chercheurs.



FIGURE 3.5: LOGO DE LA PLATEFORME KAGGLE

4.2.2. Langage de programmation

Python

Python est un langage de programmation open source. Il s'agit d'un langage de programmation interprété, qui ne nécessite donc pas d'être compilé pour fonctionner. Un programme interpréteur permet d'exécuter le code Python sur n'importe quel ordinateur.

Css

Les feuilles de style en cascade, généralement appelées CSS (Cascading Style Sheets), forment un langage informatique qui décrit la présentation des documents HTML et XML. Les standards définissant CSS sont publiés par le World Wide Web Consortium.

Html

HTML signifie « HyperText Markup Language » qu'on peut traduire par langage de balises pour l'hypertexte ». Il est utilisé afin de créer et de représenter le contenu d'une page web et sa structure.

XML

XML, pour extensible Markup Language (langage de balisage extensible), est un langage de balisage généraliste recommandé par le W3C (World Wide Web Consortium) comme le HTML. XML est un sous-ensemble du langage SGML.

4.2.3. Outils et bibliothèques utilisés

Open CV

Open CV (Open Source Computer Vision Library) est une bibliothèque libre de logiciels de vision et d'apprentissage automatique initialement développé par Intel et spécialisé dans le traitement d'image en temps réel.



FIGURE 3. 5: LOGO D'OPEN CV

Keras

Keras est une bibliothèque open source écrite en python, La bibliothèque Keras permet d'interagir avec les algorithmes de réseaux de neurones profonds et d'apprentissage automatique, notamment Tensorflow.

Flask

Flask est un micro Framework open-source de développement web en Python. Flask a pour objectif de garder un noyau simple mais extensible. Il n'intègre pas de système d'authentification, pas de couche d'abstraction de base de données, ni d'outil de validation de formulaires. Pandas

Pandas

Pandas est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données.

Matplotlib

Matplotlib est une bibliothèque destinée à tracer et visualiser des données sous formes de graphiques.

Pillow

C'est une bibliothèque d'imagerie python qui ajoute des capacités de traitement d'images à l'interpréteur python. Elle fournit une prise en charge étendue des formats de fichier, une représentation interne efficace et des capacités de traitement d'image assez puissantes.

Tensorflow

Tensorflow est une bibliothèque de Machine Learning, il s'agit d'une boîte à outils permettant de résoudre des problèmes mathématiques extrêmement complexes avec aisance.

Cette bibliothèque permet notamment d'entraîner et d'exécuter des réseaux de neurones pour la classification de chiffres écrits à la main, la reconnaissance d'image...



FIGURE 3. 7: LOGO TENSORFLOW

4.3. Détection d'incendie par Machine Learning

4.3.1. Base de données utilisée

La base de données utilisée dans nos approches est « Fire-dataset » du Kaggle. L'ensemble de données a été créé par l'équipe du NASA Space Apps Challenge en 2018 dont l'objectif est de développer un modèle capable de distinguer les images contenant du feu et les images réguliers. De sorte que tout le problème était une classification binaire entre deux classes. Les données sont divisées en 2 dossiers, le dossier des images de feu contient 755 images de feu, l'autre dossier avait des images négatives qui contiennent 244 images qui ne contiennent pas de la fumée.

4.3.2. Préparation du fichier XML

La plupart des classificateurs cascades existants en version libre contiennent les caractéristiques du visage humain tel que les fichiers `haarcascade_eye.xml`, `haarcascade_frontalface_alt.xml` et `haarcascade_frontalface_default.xml` qui sont tous utilisés pour la détection de visage et des traits humains, alors il est nécessaire de préparer un fichier xml qui contient les vecteurs caractéristiques du fumée extraite à partir de la base de données utilisées à l'aide du classificateur cascade installé dans notre poste.

Tout d'abord, on commence à créer un dossier pour notre classificateur qui contient les deux sous dossiers à l'intérieur. L'un doit être pour les images positives contenant du feu et l'autre doit être pour les images négatives qui ne contiennent pas du feu. Puis on définit la taille de tampon de pré-calcul, la largeur, la hauteur de l'échantillon et le type de la fonction utilisée pendant l'entraînement (Haar ou LBP).

Dans notre cas, on a utilisé la fonction Haar pour des raisons de précision. Après que l'opération est terminée, un fichier XML apparaît dans le dossier du classificateur créé. Ce fichier d'extension XML contient les caractéristiques qui spécifient la classe feu.

La figure 3.8 présente le contenu du fichier XML effectué.

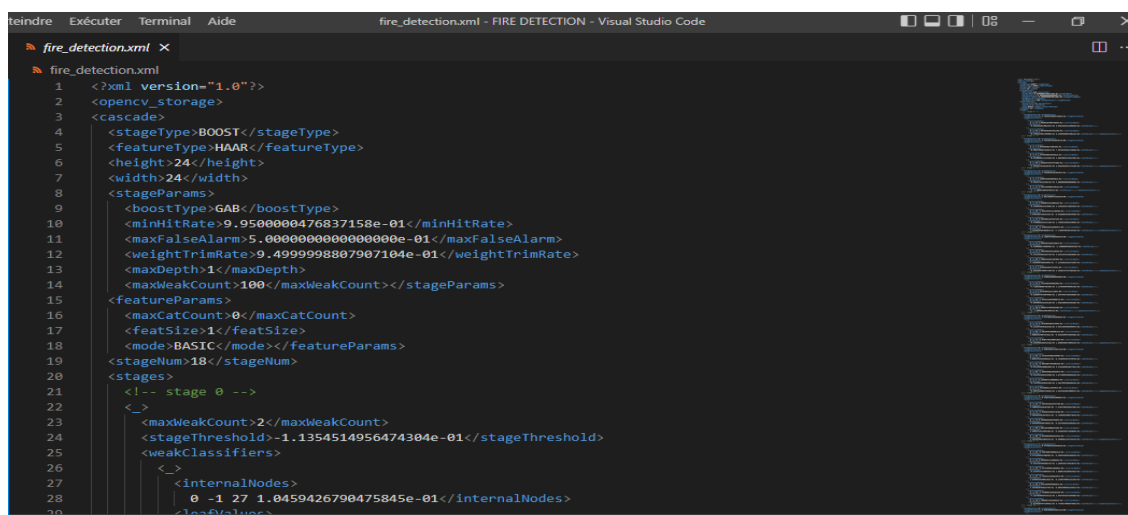


FIGURE 3. 8: CONTENU DU FICHER XML

4.3.3. Traitement de la vidéo

Le traitement de la vidéo est fait de la façon suivante :

- ❖ Lancement du Webcam.
- ❖ Dimensionnement de l'image.
- ❖ Comparaison de l'image extraite du Webcam avec les images de la base de données.
- ❖ Identification de l'image qui contient le feu.

Remarque : Le dimensionnement de l'image est fait pour qu'on puisse comparer l'image extraite du Webcam avec les images de la base de données et par suite faire la vérification d'existence de feu.

4.3.4. Exécution

Dans cette partie, on commence à développer le code qui va lancer le flux vidéo et détecter la présence du feu qui est faite à l'aide du classificateur cascade.

On a créé une boucle qui, après le lancement de la vidéo, fait la transformation d'images obtenues en niveau de gris et identifie l'existence de feu en temps réel bien représenté dans la figure 3.9.

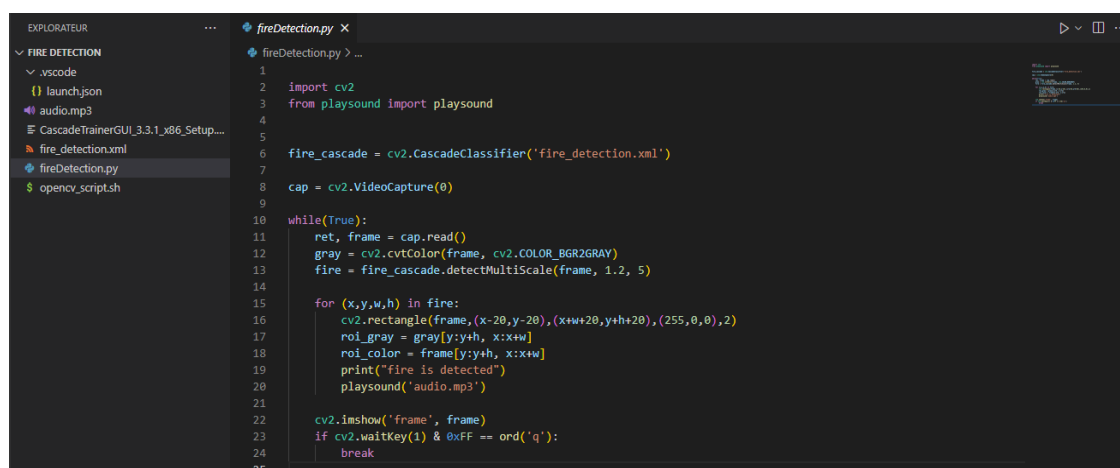


FIGURE 3. 9: CODE DE DETECTION DU FEU A PARTIR D'UN FLUX VIDEO EN TEMPS REEL

4.4. Détection d'incendie par Deep Learning

Dans cette partie, on a construit une architecture de réseau de neurone convolutive à partir de la même base de données.

Vu que le Deep Learning nécessite une grande quantité de données pour garantir la précision, On a utilisé la technique d'augmentation de data ou DA (Data Augmentation) pour l'agrandir et la rendre plus volumineuse.

4.4.1. Augmentation du data

L'augmentation des données est une technique qui consiste à agrandir artificiellement le nombre de données en créant de nouvelles données. Cela se fait en appliquant différents types de transformations aux données originales. Ces transformations comprennent une série d'opérations dans le domaine de la manipulation d'images telles que le zoom, le pivotement, le redimensionnement et bien d'autres transformations.

Cette technique est utilisée pour réduire le risque de sur apprentissage sur des données d'image, elle permet également d'améliorer les performances de notre modèle et d'avoir une bonne capacité de généralisation.

La figure 3.10 montre un exemple de data augmentation appliquée sur une image.

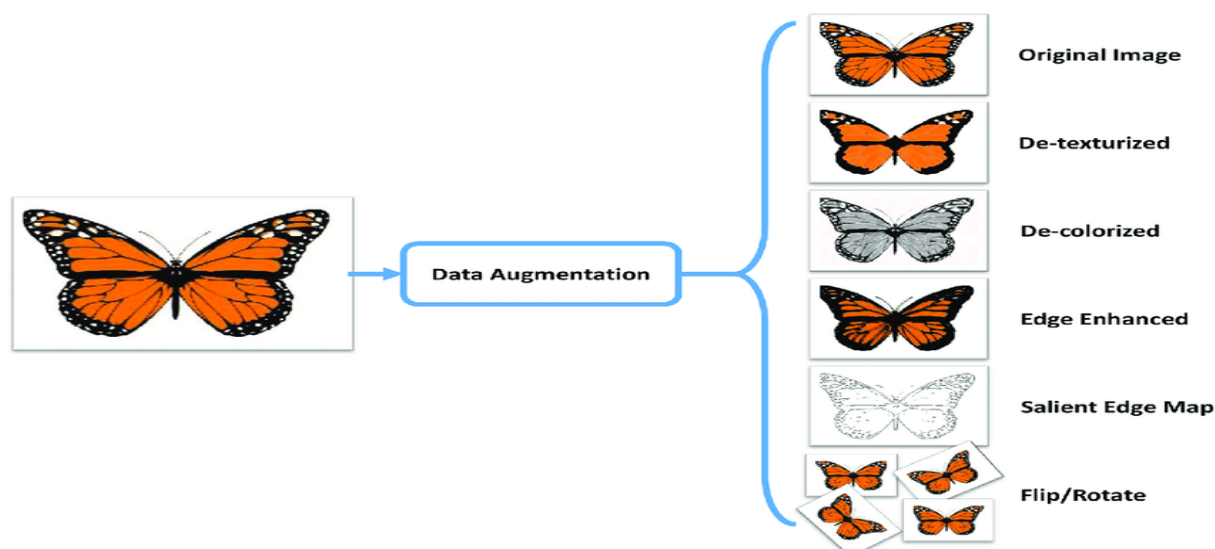


FIGURE 3. 10: APPLICATION DU DATA AUGMENTATION

Dans notre modèle, on a choisi d'appliquer les techniques d'augmentation suivante : Rotation, Zoom et Contraste.

Voici la partie du code expliquant celui-ci :

```
[ ] data_augmentation = keras.Sequential([
    keras.layers.experimental.preprocessing.RandomContrast(0.3),
    keras.layers.experimental.preprocessing.RandomRotation(0.2),
    keras.layers.experimental.preprocessing.RandomZoom(0.5)
])
```

FIGURE 3. 11: TECHNIQUES DE DATA AUGMENTATION UTILISEES

➤ Effet de d'augmentation de data

Les figures ci-dessous montrent l'effet de data augmentation sur un échantillon d'image du data set.



FIGURE 3. 12: IMAGE ORIGINALE

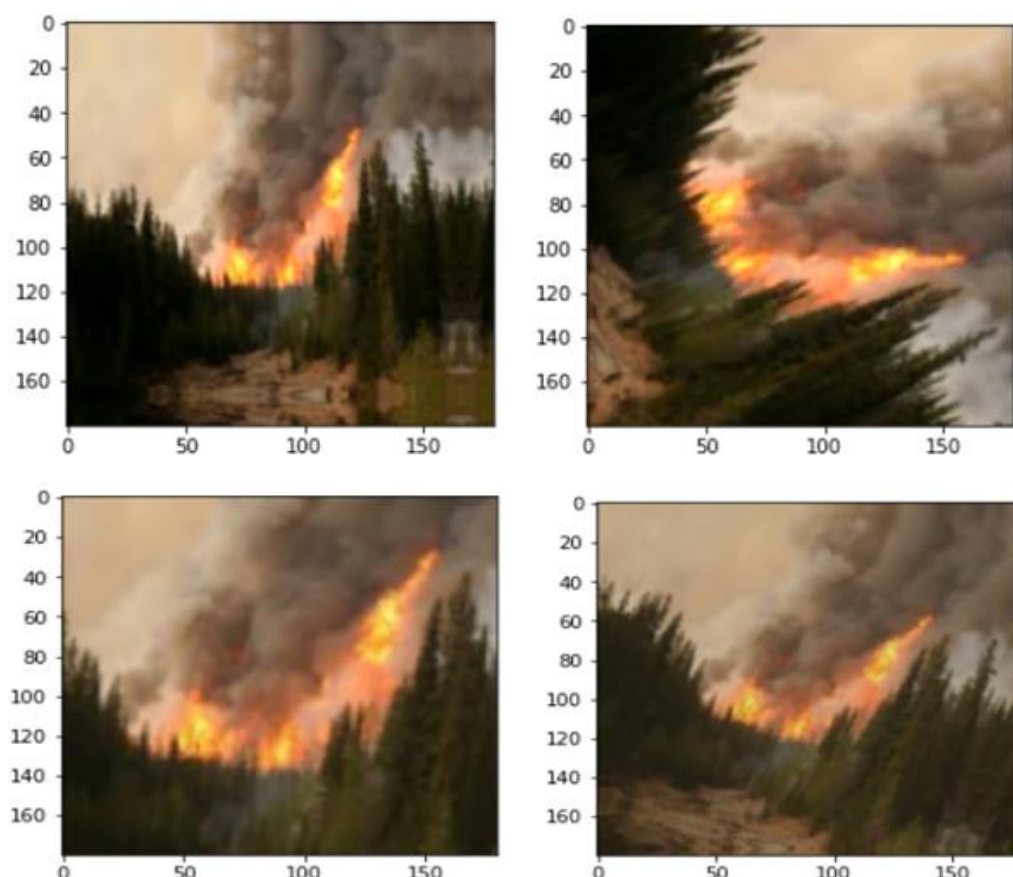


FIGURE 3. 13: EFFET DU DATA AUGMENTATION SUR L'IMAGE ORIGINALE

4.4.2. Architecture de modèle CNN

Dans notre modèle, on utilise 1844 images dans le test et 7374 images dans cette classification de type binaire.

Notre modèle est composé de trois couches de convolution, trois couches de max pooling et deux couches fully connected.

L'image en entrée est de taille 180*180, l'image passe d'abord à la première couche convolutive qui est composée de 64 filtres de taille 3*3 suivi d'un max pooling, la deuxième couche est composée de 32 filtres de taille 3*3 et la troisième couche qui est formée de 16 filtres de taille 3*3.

Chacune de nos couches de convolution est suivie d'une fonction d'activation qui est pour la première couche est relu et pour les deux autres couches est soft max.

Après cette première couche de convolution, des features map de taille 64*64 seront créés suivi d'une couche de max pooling.

Les 64 features maps qui sont obtenus auparavant. Ils sont donnés en entrée de la deuxième couche de convolution qui est composé de 32 filtres, une fonction soft max est appliquée sur la couche de convolution, ensuite on applique un max pooling pour réduire la taille de l'image

ainsi la quantité de paramètres et de calcul. A la sortie de cette couche, on aura 32 features maps de taille 32*32.

On répète la même chose avec la troisième couche de convolution qui est composée par 16 filtres et suivie d'une fonction d'activation soft max.

Après ces trois couches de convolution, on utilise un réseau de neurone composé de trois couches fully connected.

Layer (type)	Output Shape	Param #
sequential (Sequential)	(180, 180, 3)	0
conv2d (Conv2D)	(None, 180, 180, 64)	1792
max_pooling2d (MaxPooling2D)	(None, 90, 90, 64)	0
conv2d_1 (Conv2D)	(None, 90, 90, 32)	18464
max_pooling2d_1 (MaxPooling2D)	(None, 45, 45, 32)	0
conv2d_2 (Conv2D)	(None, 45, 45, 16)	4624
max_pooling2d_2 (MaxPooling2D)	(None, 22, 22, 16)	0
dropout (Dropout)	(None, 22, 22, 16)	0
flatten (Flatten)	(None, 7744)	0
dense (Dense)	(None, 10)	77450
dense_1 (Dense)	(None, 1)	11

Total params: 102,341		
Trainable params: 102,341		
Non-trainable params: 0		

FIGURE 3. 14: ARCHITECTURE DU MODELE CNN

4.4.3. Résultats obtenus et discussion

Afin de montrer les résultats obtenus par notre modèle, on illustre dans ce qui suit les résultats en termes de précision et d'erreur.

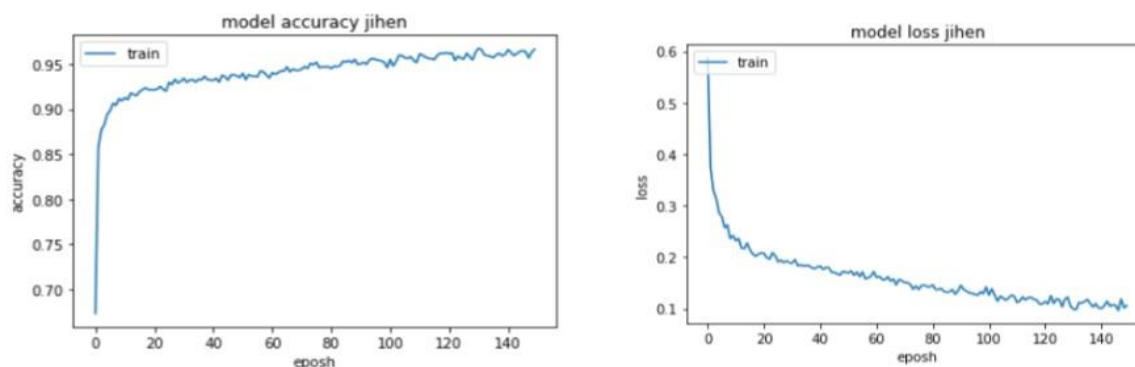


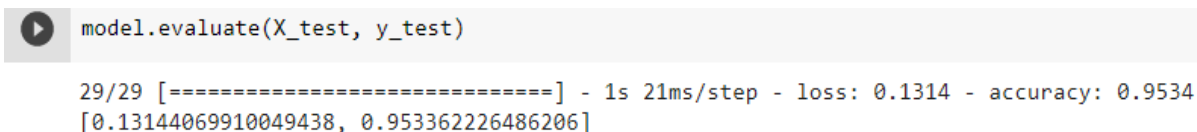
FIGURE 3. 15: PRECISION ET ERREUR

Après l'analyse des résultats obtenus, On constate les remarques suivantes :

D'après la figure 3.15 la précision de l'apprentissage augmente avec le nombre d'époque, ceci reflète qu'à chaque époque le modèle apprend plus d'informations. Si la précision est diminuée alors on aura besoin de plus d'information pour faire apprendre notre modèle et par conséquent on doit augmenter le nombre d'époque et vice versa.

❖ Evaluation du modèle

La meilleure précision obtenue est 0.9534 tandis qu'on a minimisé l'erreur à 0.1314 montré par la figure 3.16.



```

model.evaluate(X_test, y_test)

29/29 [=====] - 1s 21ms/step - loss: 0.1314 - accuracy: 0.9534
[0.13144069910049438, 0.953362226486206]

```

FIGURE 3. 16: EVALUATION DU MODELE

4.5. Codage de la partie localisation et suivi du système via GPS


Notre système va assurer la supervision de l'espace alors il est judicieux de le localiser et de suivre son déplacement à chaque instant.

4.5.1. Programmation de module GPS

Les photos suivantes contient deux capture de code python développés pour cette partie :

- ❖ le premier permet de localiser le système en temps réel.
- ❖ le deuxieme permet d'envoyer les données au plateforme Thingsboard.

🔧 Code python pour obtenir la localisation



```

GNU nano 5.4 map.py
'''
https://electroniqueamateur.blogspot.com/2021/01/module-gps-neo-6mv2-et-raspberry-pi.html
'''
import os
import time
import sys
import json
import random
import paho.mqtt.client as mqtt
import serial
import pynmea2
import string
from pubnub.pnconfiguration import PNConfiguration
from pubnub.pubnub import PubNub
from pubnub.exceptions import PubNubException

pnChannel = "raspi-tracker";
pnconfig = PNConfiguration()
pnconfig.subscribe_key = "sub-c-7c979a0b-e5d0-4396-8861-7bf10f98e099"
pnconfig.publish_key = "pub-c-980a9414-e983-469f-af23-3be47c5e7278"

pnconfig.ssl = False
pubnub = PubNub(pnconfig)
pubnub.subscribe().channels(pnChannel).execute()
ser = serial.Serial('/dev/ttyAMA0', 9600)
# Thingsboard platform credentials
THINGSBOARD_HOST = 'demo.thingsboard.io'
ACCESS_TOKEN = 'KHbeoxfDNK6aPe8rLHfT'

while True:
    donneesBrutes = ser.readline().decode('utf-8')
    if donneesBrutes[0:6] == "$GPRMC":
        donneesTraitees = pynmea2.parse(donneesBrutes)
        print( "Latitude: " + str(donneesTraitees.latitude) + " Longitude: " + str(donneesTraitees.longitude) )
        #lat = donneesTraitees.latitude
        #lng= donneesTraitees.longitude
        #try:
            #envelope = pubnub.publish().channel(pnChannel).message({'lat':lat,'lng':lng}).sync()
            #print("publish timetoken: %d" % envelope.result.timetoken)
        #except PubNubException as e:
            #handle_exception(e)

```

🔧 Code python pour le traqueur GPS en temps réel

```

GNU nano 5.4                                gps.py *
'''
import os
import time
import sys
import json
import random
import paho.mqtt.client as mqtt
import serial
import pynmea2

ser = serial.Serial('/dev/ttyAMA0',9600)
# Thingsboard platform credentials
THINGSBOARD_HOST = 'demo.thingsboard.io'
ACCESS_TOKEN = 'KHbeoxF0MKgaPz0d1HFT'

INTERVAL = 5
sensor_data = {'latitude':0,'longitude':0,'Latitude (tuple)':0,'Longitude (tuple)':0,'Vitesse':0,'Date':0}
next_reading = time.time()
client = mqtt.Client()
client.username_pw_set(ACCESS_TOKEN)
client.connect(THINGSBOARD_HOST,1883,60)
client.loop_start()
while True:
    donneesBrutes = ser.readline().decode('utf-8')
    if donneesBrutes[0:6] == "$GPRMC":
        donneesTraitees = pynmea2.parse(donneesBrutes)
        print( "Latitude: " + str(donneesTraitees.latitude) + " Longitude: " + str(donneesTraitees.longitude) )
        #sensor_data ['Latitude'] = str(donneesTraitees.latitude)
        sensor_data ['Latitude'] = donneesTraitees.latitude
        sensor_data ['Longitude'] = donneesTraitees.longitude
        #sensor_data ['Latitude (tuple)'] = donneesTraitees.latitude
        #sensor_data ['Longitude (tuple)'] = donneesTraitees.longitude
        #sensor_data ['Vitesse'] = donneesTraitees.speed_string('kph')
        #sensor_data ['Date'] = donneesTraitees.date_string('%d/%m/%Y')
        client.publish('v1/devices/me/telemetry',json.dumps(sensor_data),1)
        #client.publish('v1/devices/me/attributes', json.dumps(sensor_data()), 1)
        next_reading += INTERVAL
        sleep_time = next_reading-time.time()
        if sleep_time >0:
            time.sleep(sleep_time)
except KeyboardInterrupt:
    # pass
client.loop_stop()
client.disconnect()

```

4.5.2. Exécution

- ✚ L'exécution du programme de localisation par GPS sera en tapant la simple commande `< python map.py>` qui nous donne à chaque instant l'altitude et la longitude du système présenté par la suite dans la partie test et validation.
- ✚ l'exécution du programme `python gps.py` par la commande `<python gps.py>` permet d'envoyer les données illustrés du premier programme vers la plateforme Thingsboard pour assurer le suivi du système en temps réel.

4.5.3. Réalisation réelle

Le montage suivant montre le cablage du module GPS neo et du module caméra pi avec la carte raspberry pi 4.

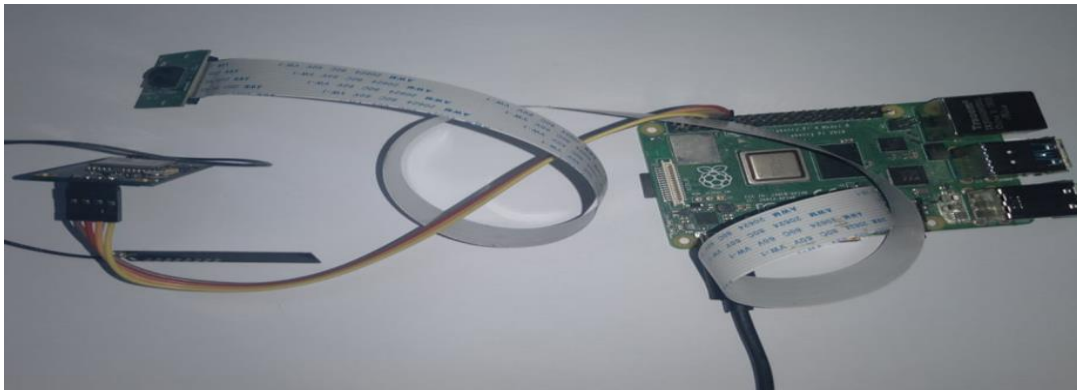


FIGURE 3. 17: BRANCHEMENT DU RASPBERRY PI AVEC LE CAMERA DU PI ET LE GPS NEO

4.6. Supervision du système via la plateforme Thingsboard

La plateforme thingsboard réalisé comporte deux parties essentielles : une partie qui représente le flux vidéo en temps réel et l'autre partie définie la localisation exacte du système en assurant le suivi de son déplacement sur une carte map programmé sur des coordonnées géographique de arbitraire. Les étapes de création sont brièvement cités par la suite :

- Création d'un nouveau dispositif.
- Choix du protocole de communication (MQTT dans notre projet).
- Création d'un nouveau dashboard.
- Création de deux widgets : une pour le module caméra et l'autre pour la carte map.
- Création d'un nouveau card html.

- Programmation du plateforme pour la réception des données.

Le contenu du dashboard est représenté par les figures suivantes.

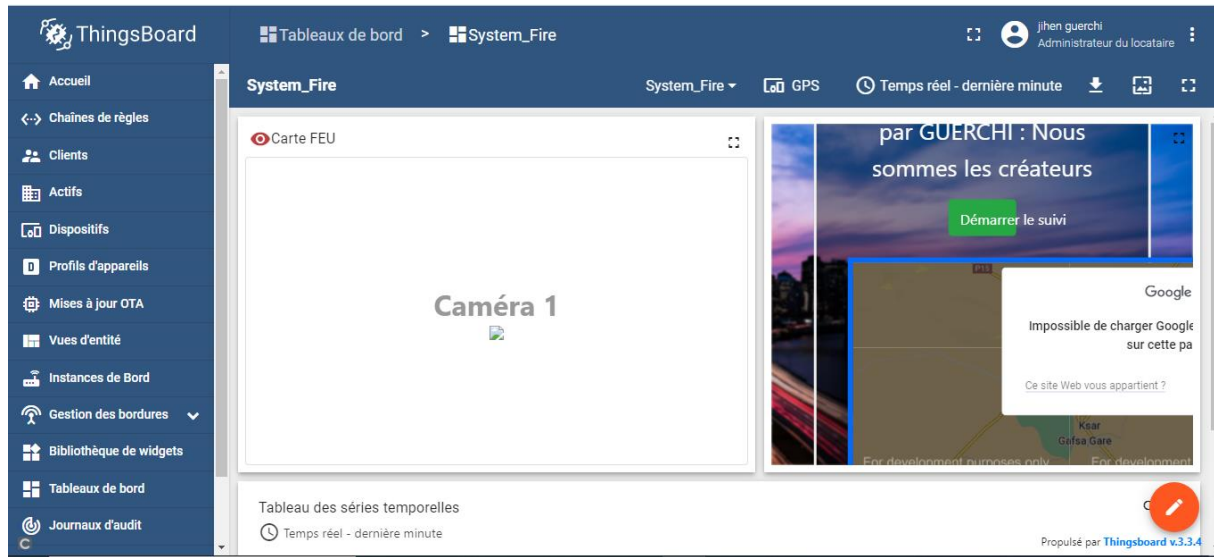


FIGURE 3. 18: DASHBOARD DE SUPERVISION

4.7. Test et validation

Une fois le modèle de détection et de geo localisation est bien implementé dans la carte Raspberry pi on passe à l'étape de test et validation.

Pour exécuter le modèle de détection il suffit de taper la commande suivante <python fire.py> Dans le terminal du raspberry pi.

Une fois un feu est détecté un message<fire is detected>apparaître dans le terminal du pi montré par la figure 3.18.

```

pi@raspberrypi: ~/Desktop/fire detection
(env) pi@raspberrypi:~/Desktop/fire detection$ python fire.py
playsound is relying on another python subprocess. Please use 'pip install pygob
ject' if you want playsound to run more efficiently.
[ WARN:0@1.369] global /tmp/pip-wheel-efxaz4j7/opencv-python_bedc0fac27944da0921
e079da44d32bf/opencv/modules/videoio/src/cap_v4l.cpp (889) open VIDEOIO(V4L2:/de
v/video0): can't open camera by index
* Serving Flask app '__name__' (lazy loading)
* Environment: production
WARNING: This is a development server. Do not use it in a production deployme
nt.
Use a production WSGI server instead.
* Debug mode: off
* Running on all addresses (0.0.0.0)
WARNING: This is a development server. Do not use it in a production deployme
nt.
* Running on http://127.0.0.1:5000
* Running on http://192.168.72.240:5000 (Press CTRL+C to quit)
192.168.72.189 - - [13/Apr/2022 03:18:18] "GET / HTTP/1.1" 200 -
192.168.72.189 - - [13/Apr/2022 03:18:19] "GET /video_feed HTTP/1.1" 200 -
fire is detected
Exception in thread Thread-3:
playing sound using playsound
Traceback (most recent call last):
  File "/usr/lib/python3.9/threading.py", line 954, in _bootstrap_inner

```

FIGURE 3. 19: EXECUTION DU MODELE



FIGURE 3. 22: COORDONNEES DU SYSTEME EN EFFECTUANT DES TESTS

Dans ce dernier chapitre, on a décrit, en premier lieu, l'architecture du système et les outils de développement utilisé dans notre projet. Dans la deuxième partie, on a bien présenté les deux modèles développés basés sur Machine et Deep Learning puis on a développé la partie géo localisation et le traqueur GPS en temps réel qui seront supervisé dans une plateforme IOT en ligne. Enfin, on a implémenté le modèle choisi dans une carte Raspberry pi pour

effectuer des tests et valider finalement le concept et donner une valeur ajoutée à notre mini projet.

Conclusion générale et perspective

L'objectif de notre projet était d'assurer une supervision efficace. L'idée générale était de créer une nouvelle technique pour détecter les incendies de forêt en supervisant l'espace forestier plus efficacement que les méthodes traditionnelles ayant plusieurs limites et qui ne sont pas adaptées aux grands espaces.

Bibliographiques

- [1]. « Evolution des incendies des forêts en Tunisie », l'EFFIS et l'ONAGRI Tunisienne, aout 2021, (consulté en mars 2022)
- [2].«Deep learning with python», CHOLLET Francois, 2017, (Consulté en Mars)
- [3].«Deep learning with Tensorflow», ZACCONE Giancarlo, MD REZAUL Karim et MENSRAWY Ahmed, 2017, (Consulté en Mars)
- [4].« Planification et optimisation de trajectoire d'un robot manipulateur à 6 ddl par des techniques neuro_floues », DJOKHRAB Ala eddine, 2015, (Consulté en Mars)
- [5].Réseaux de neurones, PARIZEAU Marc, Université Laval, 2004, (Consulté en Mars)
- [6]. Thèse doctorat, Génie électrique, Traitement de signal et d'image, Apprentissage Profond et Traitement d'Image pour la Détection de Fumée, L'Ecole Nationale Supérieure de Tunis en cotutelle avec l'université de Toulon, Rabeb kaabi, 2016, (Consulté en Avril)
- [7].O'SHEA Keiron, NASH Ryan. An introduction to convolutional neural networks. 2015, (Consulté en Avril)
- [8].WANG Peng, XU Jiaming, XU Bo, LIU Chenglin, ZHANG Heng, WANG Fangyuan HAO Hongwei. Semantic clustering and convolutional neural networks for short text categorization.2015, (Consulté en Mai).

Annexe 1 : Installation de quelques librairies nécessaires

Installation du paho-mqtt : pip install paho-mqtt

Installation du keras : pip install keras

Installation du GPS : pip install gps

Installation de l'OS : pip install os-sys

Installation du pandas : pip install pandas

Installation du matplotlib : pip install matplotlib

Installation du pillow: pip install pillow

Installation du h5py: pip install h5py

Installation du threaded: pip install threaded

Installation de l'utils : pip install utils

Installation du tensorflow : sudo -H pip3 install tensorflow-2.2.0-cp37-cp37m-linux_armv7l.whl wrapt --upgrade --ignore-installed

Résumé

L'objectif de ce projet est de réaliser un système embarqué se basant sur les techniques de l'intelligence artificielle à savoir le Machine Learning et le Deep Learning implémenté sur une carte de type Raspberry pi et en utilisant le langage de programmation python. Ce système permet de reconnaître le feu dans un flux vidéo en temps réel, de localiser le modèle via GPS et de traquer sa trajectoire pendant son déplacement ainsi que sa supervision via une plateforme en ligne.

Mots clés : Python, Système embarqué, Intelligence artificielle, Machine Learning, Deep Learning, Plateforme.

Abstract

The objective of this project is to develop an embedded system based on Artificial intelligence techniques, specifically Machine Learning and Deep Learning, implemented on a Raspberry Pi board using the Python programming language. This system is capable of real-time recognition of fire in a video stream, locating the source using GPS, tracking its trajectory during movement, and providing supervision through an online Platform..

Keywords : Python, Embedded system, Artificial intelligence, Machine learning, Deep learning, Platform.