



Département Informatique

RAPPORT DE PROJET DE FIN D'ETUDES

En vue de l'obtention du diplôme national de licence en

INGENIERIE DES SYSTEMES INFORMATIQUES

SYSTEME EMBARQUE ET INTERNET DES OBJETS

Sujet :

Détection d'anomalie de clôture

Elaboré par :

Afrah Chroudi et Chadlia el Amri

Soutenu le juin 2022 devant le jury composé de :

Président :

Examineur :

Encadrant académique : **Dr. Mejdi Ben Dkhil**

Année Universitaire : 2021/201

Code Sujet: -ISI22-011

DEDICACES

À cœur plein de volonté et de persévérance

Quand il y a l'envie de réaliser un rêve

Rien d'impossible pour arriver à nos desseins

Louange à Dieu Tout Puissant, qui m'a permis de voir ce jour tant attendu.

L'occasion m'a permis également de remercier fortement toutes les personnes qui m'ont aidée et m'ont donné leur encouragement, alors je dédie ce modeste travail :

À ma chère mère Leila

Tu as toujours été pour moi un exemple de la mère respectueuse, honnête, de la personne la plus généreuse, ce modeste travail est le fruit de tous les sacrifices que tu as déployés pour mon éducation et ma formation, Cela où je t'adresse mille mercis.

À mon cher père Radhwan

Je voudrais te remercier pour ta compréhension, ta confiance, j'exprime tout le respect que j'ai toujours eu pour toi.

À mon cher grand frère Seifeddine

Pour l'aide que tu m'as apporté, ton soutien, réconforté et encouragé. Aucune dédicace ne saurait exprimer l'amour l'estime que j'ai toujours eue pour toi. Je te souhaite beaucoup de bonheur et de prospérité.

À mon cher frère Salim

A tous les moments d'enfance passés avec toi mon frère. Puissent nos liens fraternels se consolider et se pérenniser encore plus. Je te souhaite une vie pleine de santé et de bonheur.

À ma chère sœur Zaineb

Pour l'amour qu'elle me réserve. Je t'aime ma sœur et je te souhaite une vie longue et heureuse.

À mes chers neveux Farès et Chiraz

Avoir un neveu est le plus beau cadeau qu'un frère puisse vous faire. Vos sourires, vos enthousiasmes et vos yeux brillants sont incomparables. Vous êtes apportés beaucoup de bonheur à notre famille. Je vous aime .

À mes chers(es) ami(es)

Milles merci à tous ceux que j'aime et qui m'aiment.

Afrah chroudi

DEDICACES

À cœur plein de volonté et de persévérance

Quand il y a l'envie de réaliser un rêve

Rien d'impossible pour arriver à nos desseins

Louange à Dieu Tout Puissant, qui m'a permis de voir ce jour tant attendu.

L'occasion m'a permis également de remercier fortement toutes les personnes qui m'ont aidée et m'ont donné leur encouragement, alors je dédie ce modeste travail :

À ma chère mère Dalinda

Tu as toujours été pour moi un exemple de la mère respectueuse, honnête, de la personne la plus généreuse, ce modeste travail est le fruit de tous les sacrifices que tu as déployés pour mon éducation et ma formation, Cela où je t'adresse mille mercis.

À mon cher père Trarek

Je voudrais te remercier pour ta compréhension, ta confiance, j'exprime tout le respect que j'ai toujours eu pour toi.

Chadlia elamri

REMERCIEMENTS

Nous tenons à exprimer tout d'abord nos sincères remerciements à **DIEU** qui nous a donné durant ces trois mois la volonté et le courage pour réaliser ce travail.

Nos plus vifs remerciements s'adressent aussi à notre encadreur **Mr. Mejdi Ben Dkhil** pour sa disponibilité, collaboration, pour sa compétence pédagogie et ses directives qu'il n'a cessé de nous prodiguer tout au long de ce projet.

Nous voudrions également remercier tout le cadre professoral pour la formation qu'ils nous ont donné durant ces trois années.

Enfin, nous remercions vivement les membres du jury qui ont accepté d'évaluer ce travail.

TABLE DES MATIERES

Introduction générale	1
CHAPITRE 1: Contexte générale du projet	3
Introduction	4
1.1 Cadre général du projet	4
1.2 Présentation du projet	4
<i>1.2.1 Problématique</i>	<i>4</i>
<i>1.2.2 Solution proposée</i>	<i>4</i>
1.3 Etude de l'existant :	5
<i>1.3.1 Solutions existantes :</i>	<i>5</i>
<i>1.3.2 Critique de l'existant</i>	<i>7</i>
1.4 Méthodologie de travail	8
<i>1.4.1 Méthode en Cascade</i>	<i>8</i>
<i>1.4.2 Les étapes de la méthode en cascade</i>	<i>8</i>
<i>1.4.3 Validation de choix :</i>	<i>8</i>
Conclusion	9
CHAPITRE 2: L'état de l'art	10
Introduction :	11
2.1 Machine Learning	11
<i>2.1.1 Les types d'apprentissage automatique :</i>	<i>11</i>
2.2 Détection d'anomalie :	13
<i>2.2.1 Les techniques de détection d'anomalies :</i>	<i>15</i>
<i>2.2.2 Les algorithmes de détection d'anomalies :</i>	<i>16</i>
<i>2.2.3 Comparaison des algorithmes les plus courants :</i>	<i>17</i>

2.3	Deep learning :	19
2.3.1	<i>Artificial Neural Network (ANN)</i> :	19
	Rétro-propagation et hyperparamètres :	22
2.3.2	<i>Convolutional Neural Network(CNN)</i> :	22
2.4	Transfer Learning :	25
	Conclusion.....	28
	CHAPITRE 3: Etude technique	29
	Introduction :	30
3.1	Environnement matériel :	30
3.2	Environnement logiciel :	30
3.2.1	<i>Anaconda</i> :	30
3.2.2	<i>Jupyter Notebook</i> :	31
3.2.3	<i>Python</i> :	31
3.2.4	<i>Scikit-learn</i> :	32
3.2.5	<i>Tensorflow</i> :	32
3.2.6	<i>Keras</i> :	33
3.2.7	<i>OpenCV</i> :	33
3.2.8	<i>Microsoft OfficeWord</i> :	34
	Conclusion.....	34
	CHAPITRE 4: réalisation.....	35
	Introduction	36
4.1	Prétraitement des données :	36
4.1.1	<i>Collection de la base de données</i> :	36
4.1.2	<i>Nettoyage des données</i> :	37
4.1.3	<i>Répartition de la base de données</i> :	40
4.1.4	<i>Techniques de prétraitement des données</i> :	41

4.2	Entrainement des modèles :.....	44
4.2.1	<i>Hyperparamètres liés à l'algorithme d'entraînement</i> :.....	44
4.2.2	<i>Paramètres d'optimisation</i> :.....	45
	Conclusion :.....	46
	CHAPITRE 5: résultats obtenus	47
	introduction :	48
5.1	Indicateurs de performance :.....	48
5.1.1	<i>Précision moyenne (Accuracy)</i> :.....	48
5.1.2	<i>Précision</i> :.....	49
5.1.3	<i>Rappel</i> :.....	49
5.1.4	<i>F1-score</i> :.....	49
5.1.5	<i>Matrice de confusion</i> :.....	49
5.2	Résultats obtenus et évaluation :.....	50
5.2.1	<i>Résultats obtenus par CNN</i> :.....	50
5.2.2	<i>Résultats obtenus par Xception</i> :.....	52
5.2.3	<i>Résultats obtenus par VGG16</i> :.....	52
5.2.4	<i>Résultats obtenus par ResNet50</i> :.....	53
5.3	Comparaison des performances des modèles :	54
	conclusion	54
	Conclusion générale	55
	Bibliographie	56

LISTE DES FIGURES

Figure 1.1: Architecture proposée pour la détection semi-supervisée d'anomalies dans les scènes de vidéosurveillance dans la nature.....	6
Figure 1.2: Architecture proposée pour la détection profonde d'anomalies par l'attention visuelle dans les vidéos de surveillance.....	7
Figure 2.1: Apprentissage supervisé	12
Figure 2.2: Apprentissage non-supervisé	12
Figure 2.3: Apprentissage par renforcement	13
Figure 2.4: Anomalie ponctuelle dans une série temporelle de consommation	14
Figure 2.5: Anomalie contextuelle dans une série temporelle de température.....	14
Figure 2.6: Anomalie collective correspondant à un arrêt de compteur	15
Figure 2.7: Exemples des fonctions d'activation.....	20
Figure 2.8: Single layer perceptron.....	21
Figure 2.9 : Multi Layer Perceptron	21
Figure 2.10 : l'architecture d'un Convolutional Neural Network-CNN	23
Figure 2.11 : Opération de convolution	23
Figure 2.12 : Opération de Pooling	24
Figure 2.13 : Fully Connected Layer	24
Figure 2.14 : Dropout.....	25
Figure 2.15 : Architecture du modèle Xception	26
Figure 2.16 : Architecture du modèle Xception	27
Figure 2.17 : Architecture du modèle Restnet50	28
Figure 4.1 : Exemples des images des clôtures	36
Figure 4.2: Redimensionnement d'une image de test.....	38
Figure 4.3: Histogramme de l'image de test	39
Figure 4.4: Histogramme de l'image test apres l'egalisation.....	39
Figure 4.5 : Application du filtre non-local.....	40
Figure 4.6 : Répartition de la base de données	41
Figure 4.7 : Les paramètres utilisés pour l'augmentation	43
Figure 4.8: Transformation des image test et validation	43
Figure 4.9 : L'application de l'augmentation.....	44

Figure 4.10 : La fonction " EarlyStopping "	45
Figure 5.1 : Matrice de confusion	50
Figure 5.2 : Résultats obtenus par CNN 3 layers	51
Figure 5.3 : Résultats obtenus par CNN 5 layers	51
Figure 5.4 : Résultats obtenus par Xception	52
Figure 5.5 : Résultats obtenus par VGG16	53
Figure 5.6 : Résultats obtenus par Resnet50	53

LISTE DES TABLEAUX

Tableau 2.1: Comparaison des algorithmes de machine learning	18
Tableau 5.1 : Comparaison des performances des modèles.....	54

LISTE DES ACRONYMES

ANN: Artificial Neural Network

CNN: Convolutional Neural Network

DL: Deep Learning

FC: Fully Connected

IA: Intelligence Artificielle

ML: Machine Learning

VGG: Visual Geometry Group

INTRODUCTION GÉNÉRALE

Au cours des dernières années, la télésurveillance est désormais une technique de sécurité incontournable pour assurer la sécurité contre les menaces qui ne cessent de s'accroître sous diverses formes, comme le vol qualifié, les accidents ou les activités illégales.

De nos jours, les caméras de surveillance sont couramment installées dans des lieux publics ou privés tels que les banques, les marchés commerciaux, les gares ferroviaires et routières, les rues bondées ou les zones à fort trafic, les maisons et les zones clôturées qui sont susceptibles de menaces à la sécurité, afin de garantir la protection. Traditionnellement, ces opérateurs sont chargés d'examiner les images de vidéosurveillance et de déterminer l'existence d'événements anormaux ou dangereux. Cependant, la détection manuelle de ces anomalies à partir de toutes les séquences capturées par une caméra de surveillance nécessite beaucoup de travail et de temps et une personne qui se consacre à cette mission.

Dans le cadre de l'automatisation de l'identification des anomalies à partir des vidéos de surveillance, des techniques d'apprentissage automatique y compris l'apprentissage profond, ont été utilisées pour détecter les objets et traiter les images.

Dans ce contexte, dans ce projet de fin d'études, nous nous intéressons à développer une approche d'apprentissage profond efficace et convenable qui permet de détecter les anomalies présentes dans les clôtures à partir des images capturées par une caméra de surveillances se basant sur des modèles CNN de l'apprentissage profond.

Le présent rapport est composé de cinq chapitres comme suit :

— Le premier chapitre intitulé « Le contexte général du projet » présentera le cadre général du projet, une brève description du projet, la problématique et la solution proposée, ainsi il s'articule autour de l'étude de quelques solutions techniques existantes et finalement la méthodologie adoptée.

— Le deuxième chapitre intitulé « état de l'art » sera consacré aux techniques de détection d'anomalies par apprentissage automatique.

— Le troisième chapitre intitulé « Étude technique » présentera l'environnement matériel et logiciel de la réalisation de ce projet.

— Le quatrième chapitre intitulé « Réalisation » présentera les différentes phases de la réalisation de notre approche y compris l'analyse et le prétraitement des images capturées par une caméra de surveillance ainsi que l'implémentions des différents algorithmes utilisés.

— Finalement, le cinquième chapitre intitulé « Résultats obtenus » présentera les différents résultats obtenus.

Ce rapport sera clôturé par une conclusion générale dans laquelle nous nous exposerons le bilan de notre travail et les perspectives de recherche.

CHAPITRE 1: CONTEXTE GENERALE DU PROJET

Plan

1. Cadre général du projet	4
2. Présentation du projet.....	4
3. Etude de l'existant	5
4. Méthodologie de travail.....	8

INTRODUCTION

Ce chapitre est composé de quatre parties. La première partie présentera le cadre général du projet, suivi d'une deuxième partie dans laquelle nous allons présenter notre projet ainsi que la problématique et la solution proposée. La troisième partie comporte l'étude de l'existant et nous allons finir par la méthodologie choisie au cours de la réalisation de ce projet.

1.1 Cadre général du projet

Ce stage est réalisé dans le cadre d'un projet de fin d'études pour l'obtention du diplôme national de licence en ingénierie des systèmes informatiques à l'Institut Supérieur des Sciences Appliquées et de Technologie de Sousse (ISSATSO) pour l'année universitaire 2021/2022. Ce stage a été effectué à l'ISSAT de Sousse au sein du département informatique. Le sujet est intitulé "détection d'anomalie de clôture à l'aide d'une caméra de surveillance".

1.2 Présentation du projet

1.2.1 Problématique

La télésurveillance est un système technique qui permet de contrôler à distance plusieurs lieux simultanément grâce à une caméra de surveillance. Les images obtenues par les caméras sont transmises sur un ensemble d'écrans pour être visionnées et analysées, puis archivées ou détruites. Cette surveillance a pour but de contrôler les conditions de sécurité et de sûreté de ces lieux. Généralement, ces images sont analysées par des personnes physiques, ce qui rend la tâche longue et coûteuse. De plus, l'efficacité de la caméra dépend de l'attention et de la réactivité du surveillant. Par conséquent, l'un des plus grands défis pour l'humanité aujourd'hui est de trouver une méthode de surveillance qui soit plus efficace et capable de détecter d'éventuels problèmes le plus rapidement possible avec une précision maximale.

1.2.2 Solution proposée

À cet égard, avec l'intervention de l'intelligence artificielle et du Deep Learning dans des nombreux domaines comme celui du traitement d'image, du traitement de vidéo et de la reconnaissance d'actions, ou encore de la détection d'anomalies, nous proposons une solution

basée sur le Deep Learning pour la détection d'anomalie de clôture à partir des images capturées par une caméra de surveillance.

1.3 Etude de l'existant :

L'étude de l'existant est une étape inévitable avant le lancement de n'importe quel projet. En effet, il est nécessaire d'avoir élaboré une analyse totale des projets déjà existants qui représentent des solutions au problème proposé. Cette étude préalable permet d'identifier les points faibles et les points forts des solutions actuelles.

1.3.1 Solutions existantes :

Parmi les solutions proposées nous pouvons citer :

— Détection automatique d'anomalie dans les grillages à l'aide des réseaux antagonistes génératifs (GAN) :

Dans ce projet, un drone a été utilisé pour surveiller les sections du désert où il n'y a pas de points de contrôle. Puis, enregistrer ces vidéos pour les analyser et construire leur propre base de données. En effet, cette approche est constituée de quatre étapes principales:

- Étape 1 : Vérifier les différentes vidéos enregistrées par le drone et déterminer laquelle d'entre elles permettent d'obtenir les données nécessaires pour entraîner l'algorithme.
- Étape 2 : Extraction d'un modèle 3D constitué d'un nuage de points à partir de la vidéo sélectionnée, ce modèle sera extrait à l'aide de la technique "Structure from Motion (SfM)" qui est un outil permettant de localiser les clôtures.
- Étape 3 : Création d'un masque 2D à partir du modèle 3D qui sera complété par le mapping des trames originales qui composent la vidéo. Ce masque sera utilisé pour diviser et obtenir des échantillons normaux de 32×32 pixels, ces échantillons seront utilisés à la fois pour l'apprentissage et le test du réseau de neurones.
- Étape 4 : Finalement, ces données sont utilisées comme des entrées pour entraîner l'algorithme GANanomaly, qui permettra la détection automatique des anomalies dans les clôtures.

Cette approche a atteint une AUROC de 0.67. [1]

— Détection semi-supervisée d'anomalies dans les scènes de vidéosurveillance dans la nature :

Dans ce projet, on utilise une approche de détection d'anomalies dans des scènes de vidéosurveillance basée sur un algorithme d'apprentissage semi-supervisé.

Dans la proposition, les caractéristiques de la scène sont générées via un réseau 3D à convolution temporelle (T-C3D) basé sur une architecture qui se compose de 4 étapes comme il est indiqué dans la Figure ci-dessous.

Dans la première étape, l'extracteur des caractéristiques est préformé avec le jeu de données cinétique à grande échelle. Ensuite, les caractéristiques résultantes sont entrées dans un classificateur d'apprentissage à instances multiples (MIL). Dans la troisième étape, il s'agit de la création d'un modèle avec une nouvelle fonction de perte de classement proposée. Dans la dernière étape, on amplifie l'écart entre les résultats de classement vidéo anormaux et les résultats de classement vidéo normaux.

Trois bases de données ont été utilisées dans ce projet : 'UCF-Crime', 'GBA' et 'Web Dataset'.

Cette approche a atteint une AUC de 80,36 %.[2]

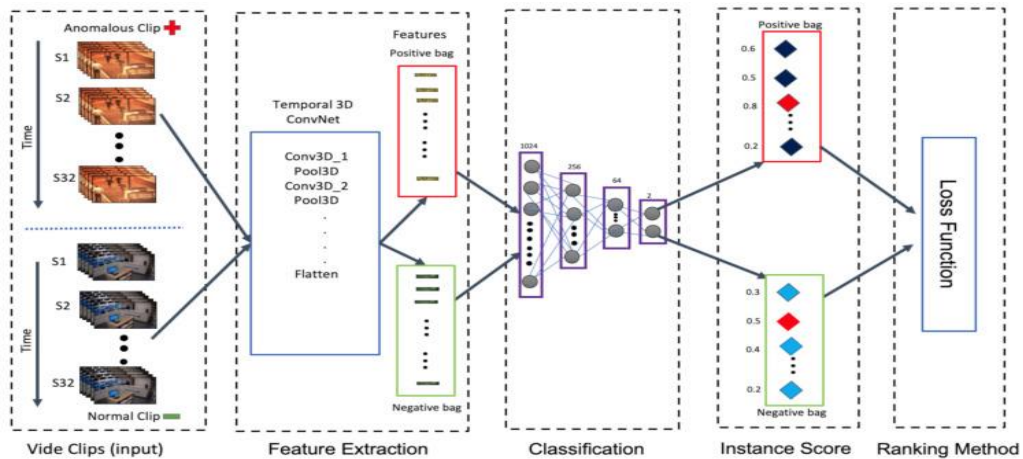


Figure 1.1: Architecture proposée pour la détection semi-supervisée d'anomalies dans les scènes de vidéosurveillance dans la nature

— Détection profonde d'anomalies par l'attention visuelle dans les vidéos de surveillance :

Ce travail utilise une méthode pour apprendre le comportement d'anomalie dans la vidéo en trouvant une région d'attention à partir d'informations spatio-temporelles, contrairement au plein cadre. Il s'agit d'une soustraction de fond robuste (BG) pour extraire le mouvement, indiquant l'emplacement des régions d'attention. Les régions résultantes sont finalement introduites dans un réseau de neurones convolutifs tridimensionnel (3D CNN). Concrètement, en tirant parti de la C3D (Convolution 3 dimensions) pour exploiter complètement une relation spatio-temporelle. Un réseau de convolution profond est développé pour distinguer les événements normaux et anormaux. Ce système est formé et testé par rapport à un ensemble de données d'anomalies 'UCFCrime' à grande échelle pour valider son efficacité.

En termes de précision, le modèle d'attention visuelle proposé peut obtenir des précisions de 99,25%.[3]

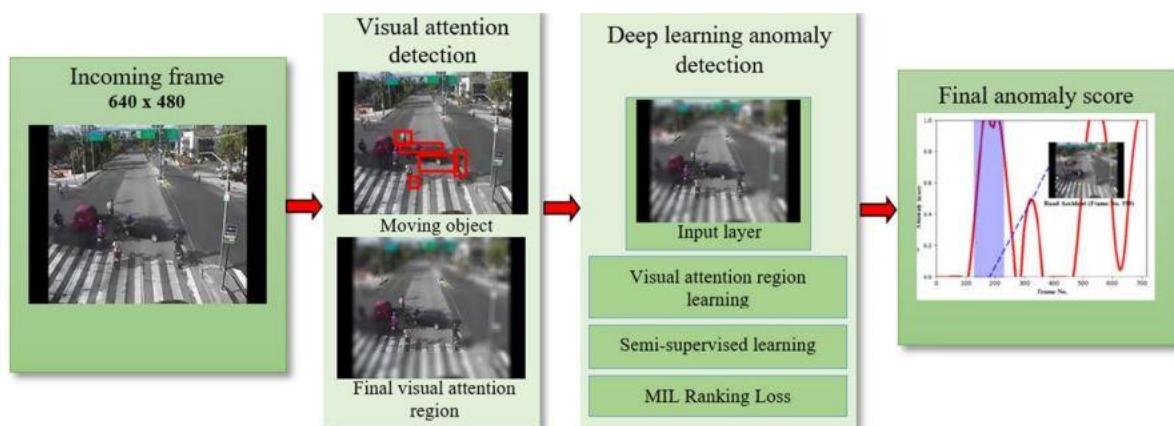


Figure 1.2: Architecture proposée pour la détection profonde d'anomalies par l'attention visuelle dans les vidéos de surveillance

1.3.2 Critique de l'existant

Après l'analyse des résultats de ces trois approches, nous avons constaté que les solutions existantes présentaient les faiblesses suivantes :

- La plupart de ces solutions sont dédiées pour la sécurité en général. Elles ne sont pas spécifiées pour notre approche pour la détection d'anomalie de clôture.
- Les solutions qui sont dédiées pour la sécurité des clôtures donnent des valeurs de précision très faibles d'où elles ne sont pas très efficaces.

1.4 Méthodologie de travail

1.4.1 Méthode en Cascade

La méthode de travail utilisée lors de l'accomplissement de ce projet est la méthode en cascade. Le modèle en cascade est une approche linéaire et séquentielle du cycle de vie du développement logiciel (SDLC) qui est populaire dans l'ingénierie logicielle et le développement des produits. [4]

Cette méthode se base sur la décomposition des processus séquentiels au sein d'un projet de manière descendante où chaque phase est effectuée une seule fois, Les sorties de chaque phase antérieure sont intégrées comme entrées de la phase suivante.

1.4.2 Les étapes de la méthode en cascade

Dans notre cas le modèle en cascade divise les processus de développement en six phases:

- Étape 1 : Étape des exigences : La première étape constituée d'une définition et une description des exigences de projet (une description grossière du projet),
- Étape 2 : Étape de l'analyse : les besoins sont définis de façon détaillée. En effet cette étape permet de créer un cahier des charges (état de l'art), un plan de projet et une budgétisation du projet etc....
- Étape 3 : Conception : c'est la phase de la planification ou l'élaboration d'une architecture logicielle ainsi qu'un plan de construction détaillé du logiciel (préparer L'architecture des modèles)
- Étape 4 : Mise en œuvre : lors de cette étape, le produit est réalisé suivant les spécifications reçues (implémentations des modèles).
- Étape 5 : Validation : Le résultat de cette phase est un produit testé et vérifié selon les exigences du projet
- Étape 6 : Mise en service : Finalement, la livraison et le déploiement du produit.

1.4.3 Validation de choix :

La méthode en cascade repose sur le fait que les membres d'équipes suivent strictement une séquence d'étapes et ne progressent pas tant que la phase précédente n'est pas terminée. Cette structure est mieux adaptée aux petits projets dont les livrables sont faciles à définir dès

le départ dans une petite équipe ce qui est le cas de notre projet. En effet cette méthode peut nous offrir plusieurs avantages dont on cite :

- **une structure claire** : la méthode en cascade offre un ensemble d'étapes clairement définies et une structure simple, ce qui nous permet d'obtenir des résultats plus complets et plus aboutis.
- **l'objectif final est fixé dès le départ**: la méthode en cascade consiste à fixer dès le début un objectif ce qui nous permet d'éviter de se perdre dans les détails à mesure que le projet avance.
- **un transfert efficace des informations** : cette méthode priorise l'accès aux informations afin que les membres d'équipes nouvellement embauchées puissent se mettre à niveau rapidement en cas de besoin.

CONCLUSION

Dans ce chapitre, nous avons détaillé les éléments qui concernent le contexte général du projet, en particulier le cadre général du projet, le problème conduisant à sa mise en œuvre, l'étude des solutions existantes et la méthodologie adoptée. Par la suite, nous passerons au chapitre de l'état de l'art.

CHAPITRE 2: L'ETAT DE L'ART

Plan

1. Machine Learning	11
2. Détection d'anomalie	13
3. Deep learning	19
4. Transfer Learning	25

INTRODUCTION :

Dans cette partie, nous allons nous focaliser sur les concepts du Machine Learning, la détection d'anomalie, le Deep Learning, les réseaux de neurone artificiels ANN, l'un des algorithmes les plus performants du Deep Learning les Convolution Neural Network ou CNN et le Transfert Learning.

2.1 Machine Learning

L'apprentissage automatique(ou machine Learning) est une méthode d'analyse de données qui automatise la création de modèles analytiques. Il s'agit d'une branche de l'intelligence artificielle basée sur l'idée que les systèmes peuvent apprendre des données, identifier des modèles et prendre des décisions avec une intervention humaine minimale.

Semblable à la façon dont le cerveau humain acquiert des connaissances et des compréhensions, l'apprentissage automatique s'appuie sur des entrées, telles que des données d'entraînement ou des graphiques de connaissances, pour comprendre les entités, les domaines et les liens entre eux. Une fois les entités sont définies, l'apprentissage en profondeur peut commencer

L'apprentissage automatique est une technologie en plein essor car elle profite à tous les types d'entreprises dans tous les secteurs. Les applications de machine learning sont illimitées. Du domaine de santé aux services financiers, des transports à la sécurité et du marketing, cette technologie est capable d'aider chaque type d'entreprise à s'adapter et à progresser de manière agile. L'apprentissage automatique peut même résoudre les problèmes de la vision par ordinateur et par les détections d'anomalies

2.1.1 Les types d'apprentissage automatique :

Il existe différents types d'apprentissage automatique : le supervisé, le non-supervisé et celui par renforcement.

- **Apprentissage supervisé** : les méthodes supervisées utilisent des données étiquetées pour apprendre dans but de construire un modèle prédictif.

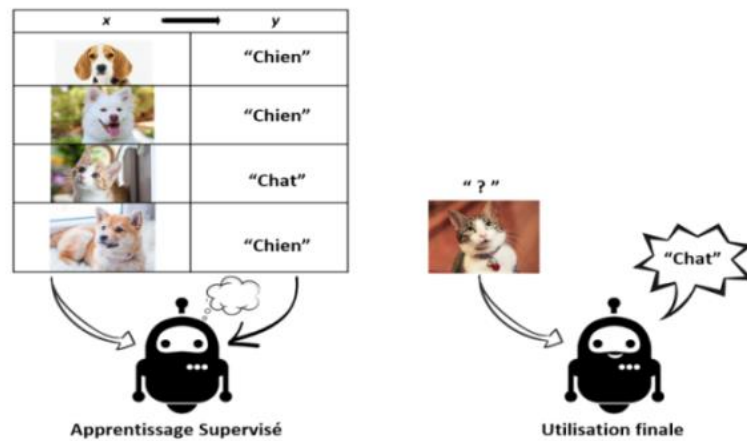


Figure 2.1: Apprentissage supervisé

- **Apprentissage non-supervisé** : ce type d'algorithmes utilise des données non-étiquetées pour apprendre et elles sont utilisées pour trois tâches principales : le regroupement, l'association et la réduction de la dimensionnalité

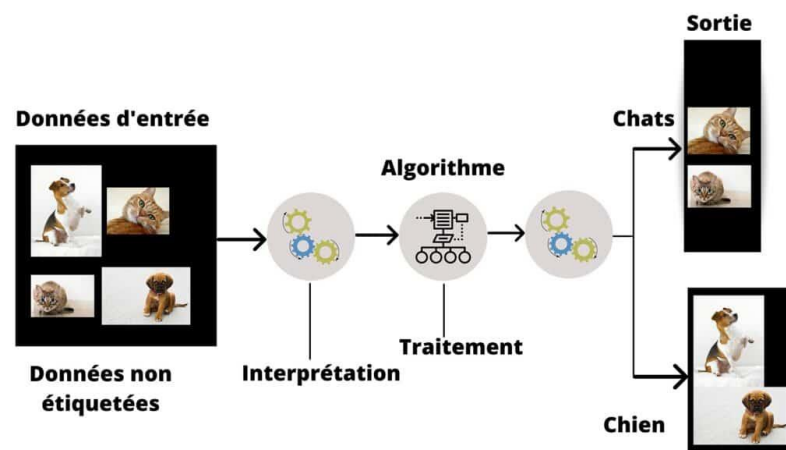


Figure 2.2: Apprentissage non-supervisé

- **Apprentissage par renforcement** : les méthodes par renforcement permettent l'agent d'apprendre avec l'expérience en interagissant avec son environnement.

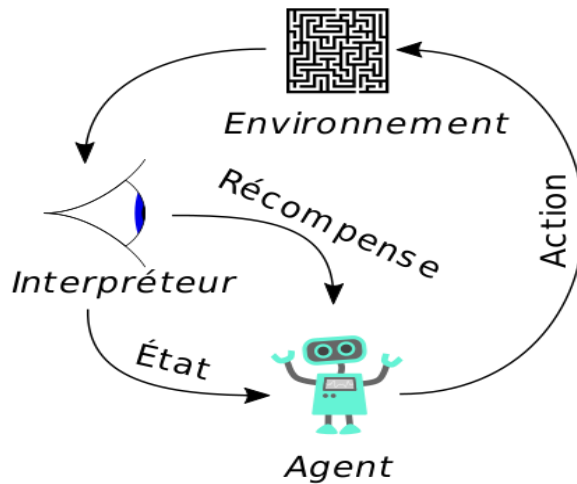


Figure 2.3: Apprentissage par renforcement

2.2 Détection d'anomalie :

La détection des anomalies, également connue sous le nom d'analyse des valeurs aberrantes, est une exploration progressive de données qui identifie les points de données, les événements et/ou les observations qui s'écartent du comportement normal d'un ensemble de données. La détection d'anomalies est fondamentalement importante et suscite l'intérêt de plusieurs chercheurs dans des divers domaines d'applications, notamment la détection d'intrusions, la détection de fraudes, la détection d'anomalies dans l'imagerie médicale, la détection de dommages industriels, la détection d'anomalies dans les données de capteurs, le traitement d'image ou vidéo etc...

Généralement, il existe une classification générale des anomalies qui s'applique dans plusieurs domaines d'application et qui peut être divisée en trois types principaux :

- **Anomalies de point :** correspondent à un point de données considéré comme valeur aberrante car il est suffisamment différent ou éloigné de l'ensemble des données comme indiqué dans la figure ci-dessous :

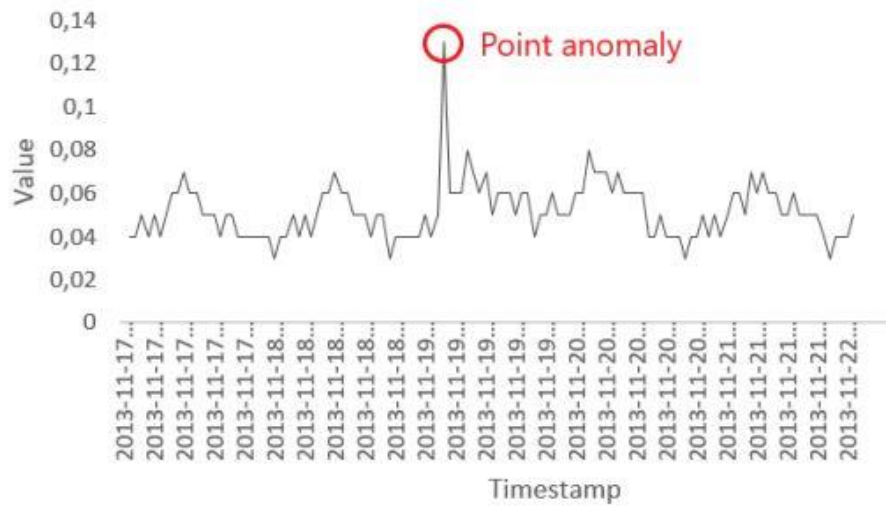


Figure 2.4: Anomalie ponctuelle dans une série temporelle de consommation

- **Anomalies contextuelles :** correspondent à un point de données différent ou éloigné des autres points de données mais dans un contexte spécifique la figure suivante présente une anomalie contextuelle dans une série temporelle de température mensuelle.

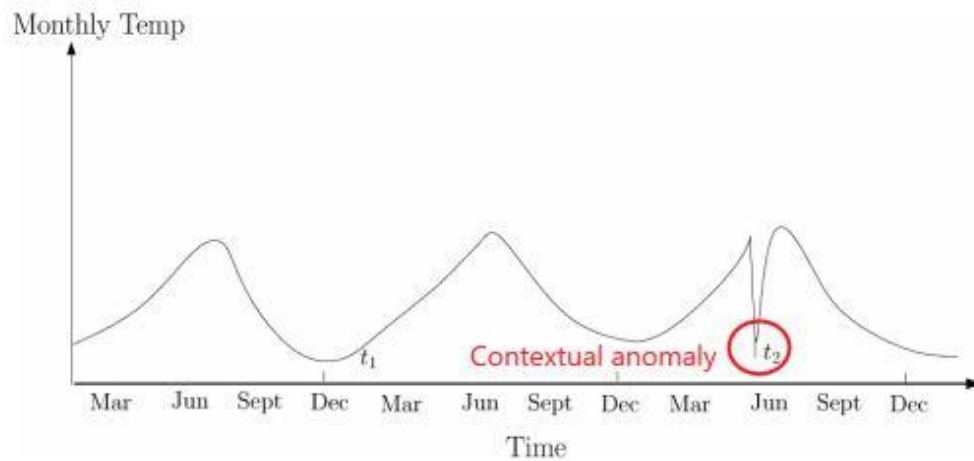


Figure 2.5: Anomalie contextuelle dans une série temporelle de température

- **Anomalies collectives** : correspondent à une collection d'observations qui est différente de l'ensemble des données.

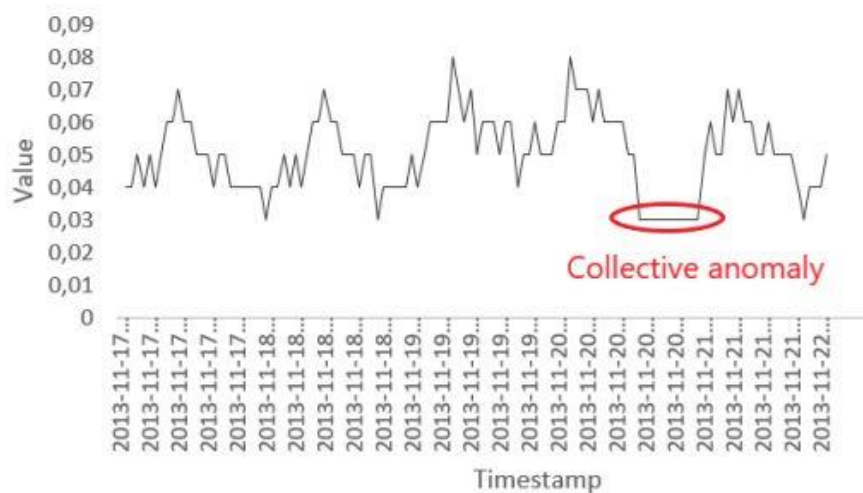


Figure 2.6: Anomalie collective correspondant à un arrêt de compteur

2.2.1 Les techniques de détection d'anomalies :

Plusieurs techniques de détection d'anomalies ont été développées spécifiquement pour certains domaines d'application, tandis que d'autres sont plus génériques. Mais généralement toutes ces techniques appartiennent à l'une de ces 3 catégories :

- **Détection supervisée des anomalies** : les modèles supervisés requièrent un ensemble de formation étiqueté contenant à la fois des échantillons normal et anormaux pour construire le modèle prédictif et apprendre à identifier les anomalies dans les nouvelles données.
- **Détection non supervisée d'anomalies** : Tout comme les techniques d'apprentissage automatique non supervisées, la détection d'anomalies non supervisée traite les données non étiquetées. En effet ces techniques ne nécessitent pas de données d'entraînement mais supposent que le comportement normal est le plus fréquent.
- **Détection semi supervisé d'anomalies** : cette technique utilise un ensemble de données d'apprentissage normal et étiqueté pour construire un modèle représentant un comportement normal. Ils utilisent ensuite ce modèle pour détecter les anomalies en testant la probabilité que le modèle génère une instance rencontrée.

2.2.2 Les algorithmes de détection d'anomalies :

Plusieurs algorithmes d'apprentissage automatique peuvent être utilisés pour la détection d'anomalies et dépendent de la taille de l'ensemble de données et du type de problème. Parmi ces algorithmes on peut citer :

K-NearestNeighbor(KNN) : k-NN est un algorithme de ML supervisé fréquemment utilisé pour les problèmes de classification (parfois aussi des problèmes de régression). Son principe est de calculer les distances approximatives entre différents points sur les vecteurs d'entrée, puis attribue le point non étiqueté à la classe de ses K plus proches voisins. Lorsqu'il s'agit d'une détection d'anomalies, k-NN peut adopter une approche non supervisée. En effet, il n'y a pas d'apprentissage réel impliqué dans le processus de la détection d'anomalie et il n'y a pas d'étiquetage prédéterminé des valeurs aberrantes ou non aberrante dans l'ensemble de données, au lieu de cela, il est entièrement basé sur des valeurs seuils.

DecisionTree (DT) :L'algorithme arbre de décision (DecisionTree) appartient à la famille des algorithmes d'apprentissage supervisé. Contrairement à d'autres algorithmes d'apprentissage supervisé, l'algorithme d'arbre de décision peut également être utilisé pour résoudre des problèmes de régression et de classification. Les arbres de décision utilisent plusieurs algorithmes pour prendre la décision de diviser un nœud en deux ou plusieurs sous-nœuds. La création de sous-nœuds augmente l'homogénéité des sous-nœuds résultants.

Support Vector Machine (SVM) : La machine à vecteurs de support (SVM) est un algorithme supervisé généralement utilisé pour la classification. Les SVM utilisent des hyperplans dans un espace multidimensionnel pour diviser les points de données en classes. L'hyper paramètre nu est le seuil (pourcentage) pour les valeurs aberrantes qu'on doit choisir manuellement. SVM est généralement appliqué lorsqu'il y a plus d'une classe impliquée dans le problème. Cependant, dans la détection d'anomalies, il est aussi utilisé pour les problèmes contentent une seule classe. Le modèle est formé pour apprendre les cas normaux et peut identifier si des données inconnues appartiennent à cette classe ou représentent une anomalie.

K-Means : K-means est un algorithme d'apprentissage automatique non supervisé. Il s'agit du clustering (ou regroupement) basé sur la distance. Il divise l'ensemble de données en plusieurs clusters. Dans cette méthode, K points aléatoires sont sélectionnés comme centroïdes dans un jeu de données. Ensuite, les éléments sont disposés aux centroïdes les plus

proches en calculant la distance. Le processus est répété pour obtenir des distances optimales entre les données d'échantillon et les centroïdes.

Isolation Forest : Isolation Forest (ou La forêt d'isolement) est un algorithme d'apprentissage non supervisé qui identifie les anomalies en isolant les valeurs aberrantes dans les données. La forêt d'isolement est basée sur l'algorithme de l'arbre de décision. Il isole les valeurs aberrantes en sélectionnant de manière aléatoire une caractéristique de l'ensemble de caractéristiques donné, puis en sélectionnant de manière aléatoire une valeur de fractionnement entre les valeurs maximale et minimale de cette caractéristique. Ce partitionnement aléatoire des caractéristiques produira des chemins plus courts dans les arbres pour les points de données anormaux, les distinguant ainsi du reste des données.

Fuzzy C-Means (FCM) :Fuzzy C-means est une méthode de clustering, qui accorde une pièce de données à appartenir à deux clusters ou plus. Il est plus utilisé dans applications pour lesquelles la classification stricte des données n'est pas significative ou difficile à résoudre.

DBSCAN : DBSCAN est un algorithme d'apprentissage non supervisé basé sur le principe de densité. Il est capable de découvrir des clusters dans des grands ensembles de données spatiales en examinant la densité locale des points de données et affiche généralement de bons résultats lorsqu'il est utilisé pour la détection d'anomalies. Les points qui n'appartiennent à aucun cluster obtiennent leur propre classe «-1 » afin qu'ils soient facilement identifiables. Cet algorithme gère bien les valeurs aberrantes lorsque les données sont représentées par des points de données non discrets.

2.2.3 Comparaison des algorithmes les plus courants :

Les expériences ont démontré que les méthodes d'apprentissage supervisé surpassent de manière significative les méthodes non supervisées si les données de test ne contiennent aucune attaque inconnue. Parmi les méthodes supervisées, les meilleures performances sont obtenues par les méthodes non linéaires, telles que SVM, et les méthodes basées sur des règles telles que l'arbre de décision. Les techniques non supervisées telles que K-Means et isolation Forest ont obtenu des meilleures performances par rapport aux autres techniques, grâce à leurs capacités à détecter efficacement toutes les classes d'attaques. En effet chacune des techniques a ses avantages et ses inconvénients. Il n'y a pas d'approche qui fait mieux que les autres pour tous les types de problèmes.

Le tableau ci-dessous présente une comparaison entre les algorithmes les plus courants. La comparaison résume les avantages et les inconvénients de chacun.

Tableau 2.1: Comparaison des algorithmes de machine learning

Algorithme	Technique	Avantage	Inconvénients
KNN	Supervisé	<ul style="list-style-type: none"> -Utile pour construire des modèles qui impliquent des données non standard, tels que les textes. -robuste aux données d'entraînement bruitées. -simple à mettre en œuvre. 	<ul style="list-style-type: none"> -Sensible au choix de la fonction de similarité qui est utilisé pour comparer les instances. -technique coûteux en calcul
DecisionTree	Supervisé	<ul style="list-style-type: none"> -nécessite moins d'efforts pour la préparation des données lors du prétraitement. - ne nécessite pas de normalisation des données. - Capable de gérer à la fois les données numériques et catégoriques. 	<ul style="list-style-type: none"> -Un petit changement dans les données peut entraîner un changement important dans la structure de l'arbre de décision, provoquant une instabilité. -Risque d'overfittingoù les apprenants créent des arbres trop complexes qui ne généralisent pas bien les données -Les calculs peuvent devenir très complexes
SVM	supervisé	<ul style="list-style-type: none"> -Peut traiter des données de très grande dimension. -Fonctionne relativement bien lorsque les classes sont bien séparées. -Fonctionne bien avec des données même non structurées et semi-structurées telles que les textes, des images et des arbres. 	<ul style="list-style-type: none"> -Exige beaucoup de mémoire -N'est pas performante avec des jeux de données volumineux -Un temps d'entraînement très long.

K-Means	non-supervisé	<ul style="list-style-type: none"> -complexité faible -très facile à mettre en œuvre -s'adapte facilement aux nouveaux exemples 	<ul style="list-style-type: none"> -Traite uniquement les données numériques. -Sensible au bruit et aux valeurs aberrantes. -Les clusters sont sensibles à l'affectation initiale des centroïdes.
Isolation Forest	non-supervisé	<ul style="list-style-type: none"> -Moins d'effort de calcul. -Exigence faible de mémoire. -Fonctionne bien même avec un petit ensemble des données. 	<ul style="list-style-type: none"> -Problème de détection d'anomalie local ce qui affecte la précision de l'algorithme.
FCM	non-supervisé	<ul style="list-style-type: none"> -Permet à un point de données d'être dans plusieurs clusters. -Offre une représentation plus naturelle du comportement. 	<ul style="list-style-type: none"> -Il faut définir c, le nombre de clusters. -Les clusters sont sensibles à l'affectation initiale des centroïdes.
DBSCAN	non-supervisé	<ul style="list-style-type: none"> -Ne nécessite pas de préciser le nombre de clusters dans les données d'entrées. -Robuste aux valeurs aberrantes. 	<ul style="list-style-type: none"> -Ne gère pas les ensembles de données de grande dimension d'une façon efficace -les résultats sont sensibles aux choix des paramètres.

2.3 Deep learning :

Le Deep learning ou apprentissage profond est l'une des technologies principales du Machine Learning. C'est une discipline qui permet d'analyser les données non structurées telles que des images, des vidéos et du texte. Il repose sur un ensemble d'algorithmes qui sont capables de simuler les actions du cerveau humain grâce à des réseaux de neurones artificielles. Cette dernière, désigne une technique d'intelligence artificielle qui donne à une machine la capacité de voir et ainsi permet l'analyse et l'interprétation d'images ou vidéos.

2.3.1 Artificial Neural Network (ANN) :

Un réseau de neurones artificiels (ANN) est un système de technologie de l'information basé sur une conception à l'origine inspirée du cerveau humain. Ces réseaux émulent un

réseau neuronal biologique, ils sont composés de milliards de neurones qui établissent entre eux des liaisons et former des réseaux plus ou moins complexes pour traiter des informations en parallèle conçues pour résoudre des problèmes qui s'avèrent impossibles ou difficiles selon les normes humaines ou statistiques. Haykin [5] décrit ANN comme une combinaison massivement parallèle de simple unité de traitement qui peut acquérir des connaissances de l'environnement par un processus d'auto-apprentissage et de stocker les connaissances dans ses connexions. Parmi les réseaux de neurones les plus populaires on cite le réseau de neurones convolutifs (CNN).

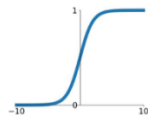
Les composantes principales du réseau de neurones sont :

- **Entrées :** Peuvent être entrées du système complet ou peuvent provenir d'autres neurones.
- **Poids :** c'est une variable qui est appliquée à l'entrée de chacun des neurones. Son objectif est de calculer une donnée de sortie.
- **Biais :** c'est une valeur numérique qui est ajoutée une fois que le poids est appliqué à l'entrée. son objectif est de varier le seuil d'enclenchement du neurone.
- **Fonction d'activation :** L'un des paramètres les plus importants du modèle CNN est la fonction d'activation. C'est une fonction mathématique appliquée à un signal en sortie d'un neurone artificiel. Son objectif est d'aider les réseaux de neurones à apprendre et à décider quelles informations du modèle doivent être déclenchées dans le sens direct et lesquelles ne doivent pas être transmises à la fin du réseau. Il existe plusieurs fonctions d'activation parmi lesquelles on peut citer : Sigmoid, LeakyReLU, tanh, ELU etc...

Activation Functions

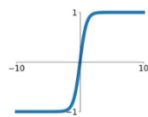
Sigmoid

$$\sigma(x) = \frac{1}{1+e^{-x}}$$



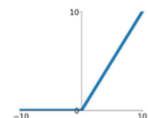
tanh

$$\tanh(x)$$



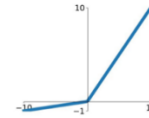
ReLU

$$\max(0, x)$$



Leaky ReLU

$$\max(0.1x, x)$$



Maxout

$$\max(w_1^T x + b_1, w_2^T x + b_2)$$

ELU

$$\begin{cases} x & x \geq 0 \\ \alpha(e^x - 1) & x < 0 \end{cases}$$

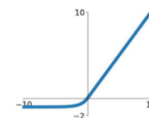


Figure 2.7: Exemples des fonctions d'activation

— **Sortie** : Valeur résultat ou sortie du réseau.

Single Layer Perceptron :

Single Layer Perceptron est un réseau de neurones composé d'entrées multiples et d'une seule sortie et chaque entrée possède un poids comme il est indiqué dans la figure suivante :

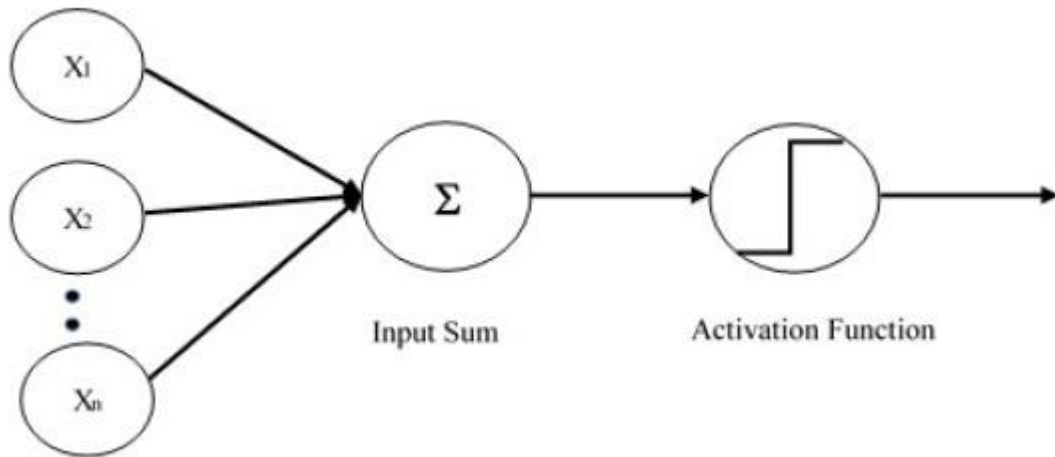


Figure 2.8:Single layer perceptron

Multi-Layer Perceptron :

Multi-Layer Perceptron est un réseau de neurones qui possède une couche d'entrée, une couche de sortie et des couches intermédiaires appelées couches cachées comme il est indiqué dans la figure suivante :

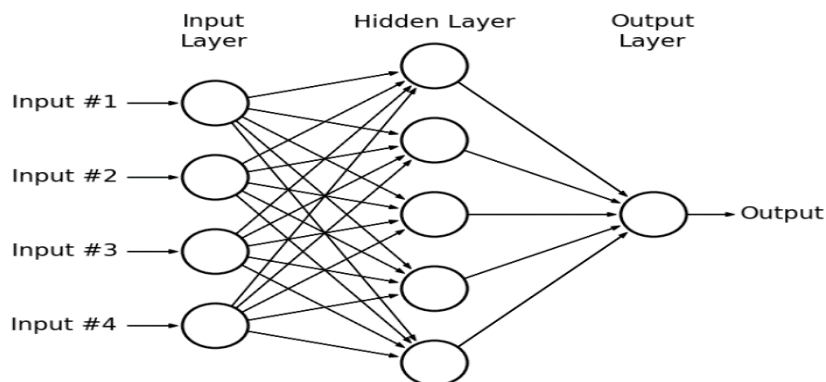


Figure 2.9:Multi Layer Perceptron

Rétro-propagation et hyperparamètres :

- **Rétro-propagation** : c'est une méthode de réglage qui consiste à entraîner des réseaux neuronaux artificiels, elle est utilisée en conjonction avec une méthode d'optimisation de descente en gradient pour réduire la différence entre les sorties du réseau et les sorties désirées. Ainsi, il permet de propager l'erreur à travers le réseau dans le sens opposé de la propagation des activations.
- **Descente gradient** : c'est un algorithme utilisé pour entraîner le réseau de neurones artificiels, calculer le gradient et corriger les poids.
- **Batch size** : c'est un hyperparamètre de la descente gradient qui fait référence au nombre d'exemples entraînés en une itération.
- **Epochs**: est un hyperparamètre qui définit le nombre de fois durant lesquelles l'algorithme d'apprentissage fonctionnera à travers l'ensemble de données d'entraînement.

2.3.2 Convolutional Neural Network(CNN) :

Le réseau de neurones convolutionnels (CNN) est l'un des réseaux de neurones profonds les plus populaires. Il prend ce nom de l'opération linéaire mathématique entre les matrices appelées convolution. Ce type de réseau a été inspiré du cortex visuel des vertébrés. Le CNN a une excellente performance dans les problèmes d'apprentissage automatique. En particulier les applications qui traitent des données d'image, telles que le plus grand ensemble de données de classification d'image (Image Net), la vision par ordinateur, et dans le traitement du langage naturel (NLP).

L'Architecture d'un Convolutional Neural Network-CNN :

Le réseau de neurones convolutionnels (CNN) est composé des multiples couches telles que les couches convolutives, les couches de pooling et les couches « Fullyconnected » (FC). De plus, il existe deux paramètres plus importants qui sont la couche « dropout » et la fonction d'activation.[6]

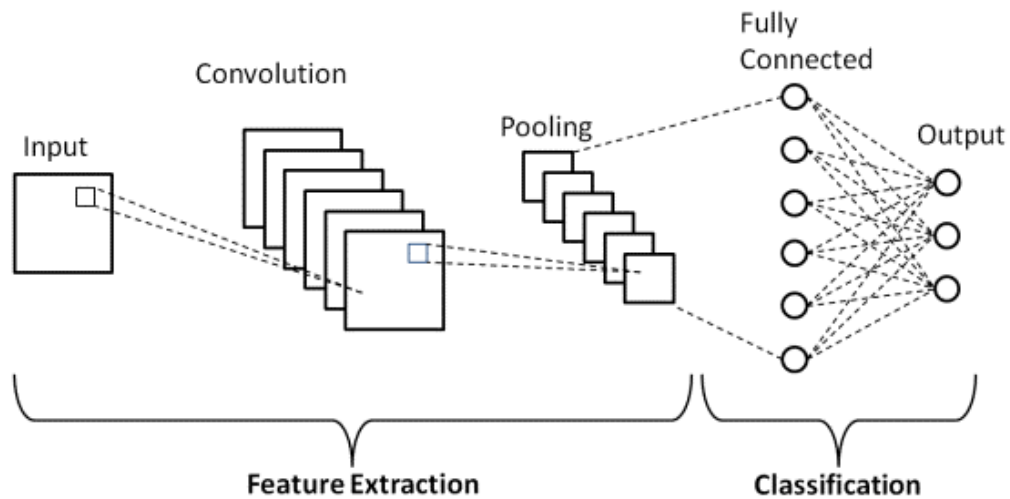


Figure 2.10: l'architecture d'un Convolutional Neural Network-CNN

- **Les couches convolutives :** c'est la première couche du CNN qui est utilisée pour extraire les différentes caractéristiques des images d'entrée. L'opération mathématique de convolution dans cette couche est effectuée entre l'image d'entrée et un filtre d'une taille particulière $M \times M$. Le principe de cette opération est de faire glisser le filtre sur l'image d'entrée et de calculer le produit scalaire de convolution entre la feature et chaque portion de l'image d'entrée.

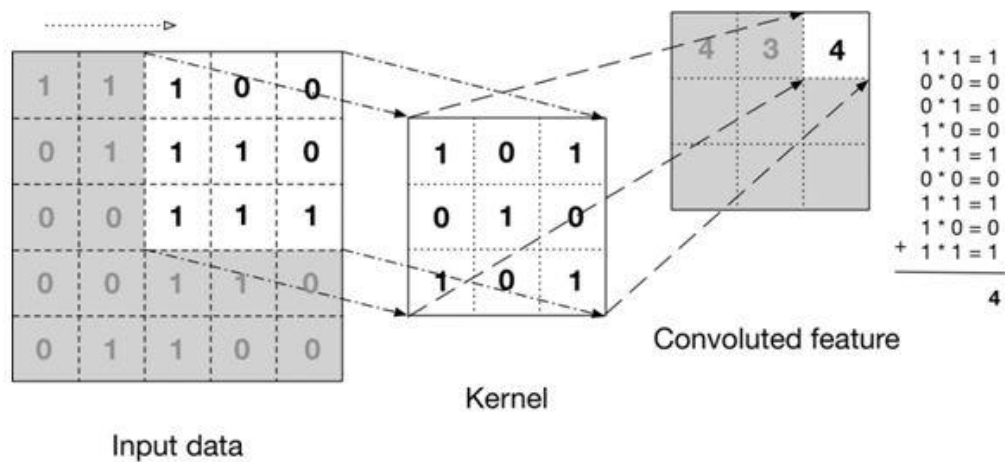


Figure 2.11: Opération de convolution

- **Les couches de pooling:** Ce type de couche est spécifié pour réduire la taille de la carte des caractéristiques convoluées afin de réduire les coûts de calcul utilisant plusieurs types d'opérations telles que le Max-Pooling et l'Averge-Pooling.

On prend un exemple, pour chaque groupe de 4 pixels, le pixel ayant la valeur maximale (Max Pooling, le plus populaire) ou la moyenne des pixels (Average pooling)[7].

La figure ci-dessous clarifie l'énoncé.

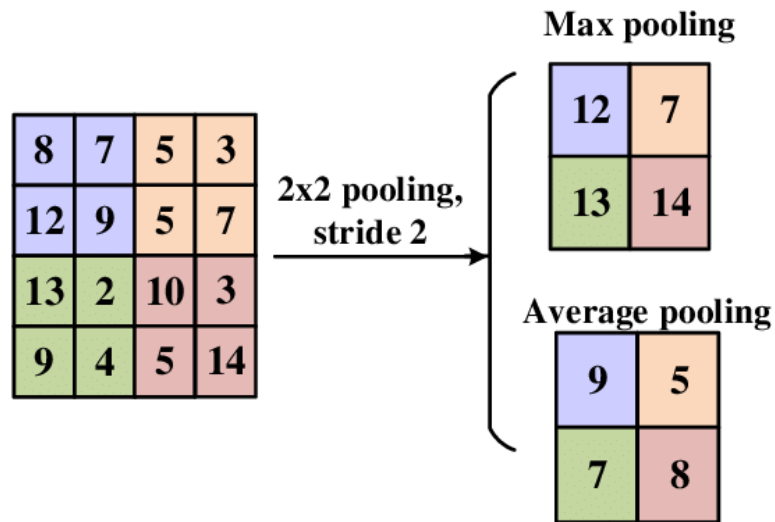


Figure 2.12: Opération de Pooling

- **FullyConnected Layer (FC)** : La couche entièrement connectée (FullyConnected), est la dernière couche du CNN. Elle est utilisée pour connecter les neurones entre deux couches différentes.

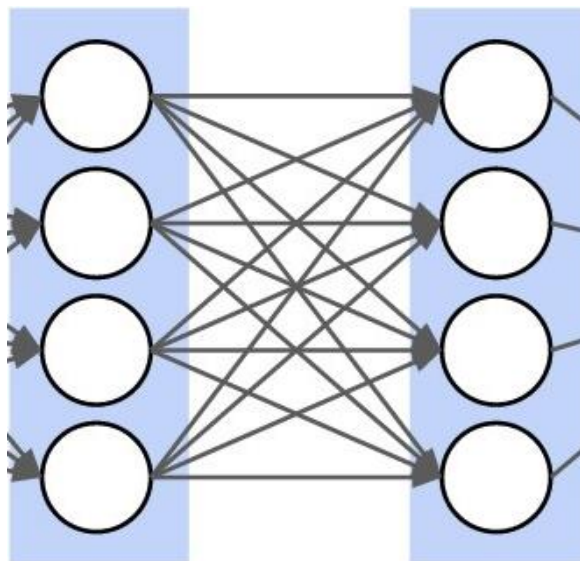


Figure 2.13: Fully Connected Layer

Dropout : est une technique permettant de réduire l'overfitting lors de l'entraînement d'un modèle. Le terme "Dropout" fait référence à la suppression de neurones dans les couches d'un modèle d'apprentissage en profondeur. En effet, elle permet la désactivation temporaire de certains neurones du réseau, ainsi que toutes ses connexions entrantes et sortantes.

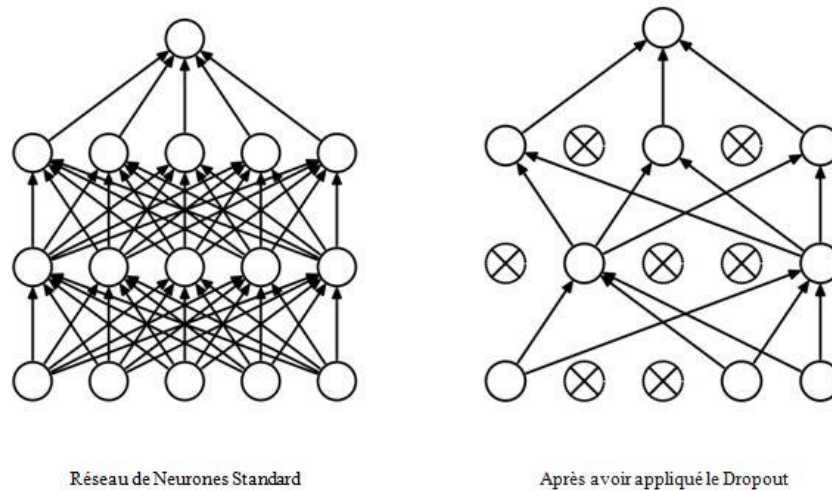


Figure 2.14 :Dropout

2.4 Transfer Learning :

Le Transfer Learning, ou apprentissage par transfert correspondant à l'ensemble des méthodes qui permettent de transférer les connaissances acquises à partir de la résolution de problèmes donnés pour traiter un autre problème. En effet, le Transfer Learning nous permet de développer rapidement des modèles performants et résoudre efficacement des problèmes complexes. Bien qu'il s'agisse d'une technique émergente, l'apprentissage par transfert est déjà utilisé dans divers domaines de l'apprentissage automatique tel par exemple dans le Computer Vision. Il s'exprime généralement par l'utilisation de modèles pré-entraînés. Un modèle pré-entraîné est un modèle qui a été entraîné sur un grand ensemble de données d'apprentissage pour résoudre un problème similaire à celui que nous voulons résoudre. Par conséquent, en raison du coût de calcul d'entraînement de tels modèles, il est plus courant d'importer et d'utiliser des modèles basés sur le Transfer Learning.

En faite, il existe plusieurs modèles pour le Transfer Learning, dont on cite les modèles testés dans notre travail :

Xception : Ce modèle a été proposé par François Chollet le créateur et mainteneur de la bibliothèque Keras. Le Xception est une extension de l'architecture d'inception qui remplace le modèle d'inception standard (un modèle de reconnaissance d'images). En effet, c'est un réseau de neurones convolutionnel profond de 71 couches, il comporte 36 couches convolutives qui forment la base d'extraction de caractéristiques du réseau. Ces 36 couches sont structurées en 14 modules, qui ont tous des connexions résiduelles linéaires autour d'eux, à l'exception du premier et du dernier module. Comme une entrée, il prend une image en couleurs de taille 208×176 px et la classifie dans une classe. Par conséquent, cette approche permet de réduire l'utilisation des ressources lors du calcul matriciel, sans modifier le nombre de paramètres.[8]

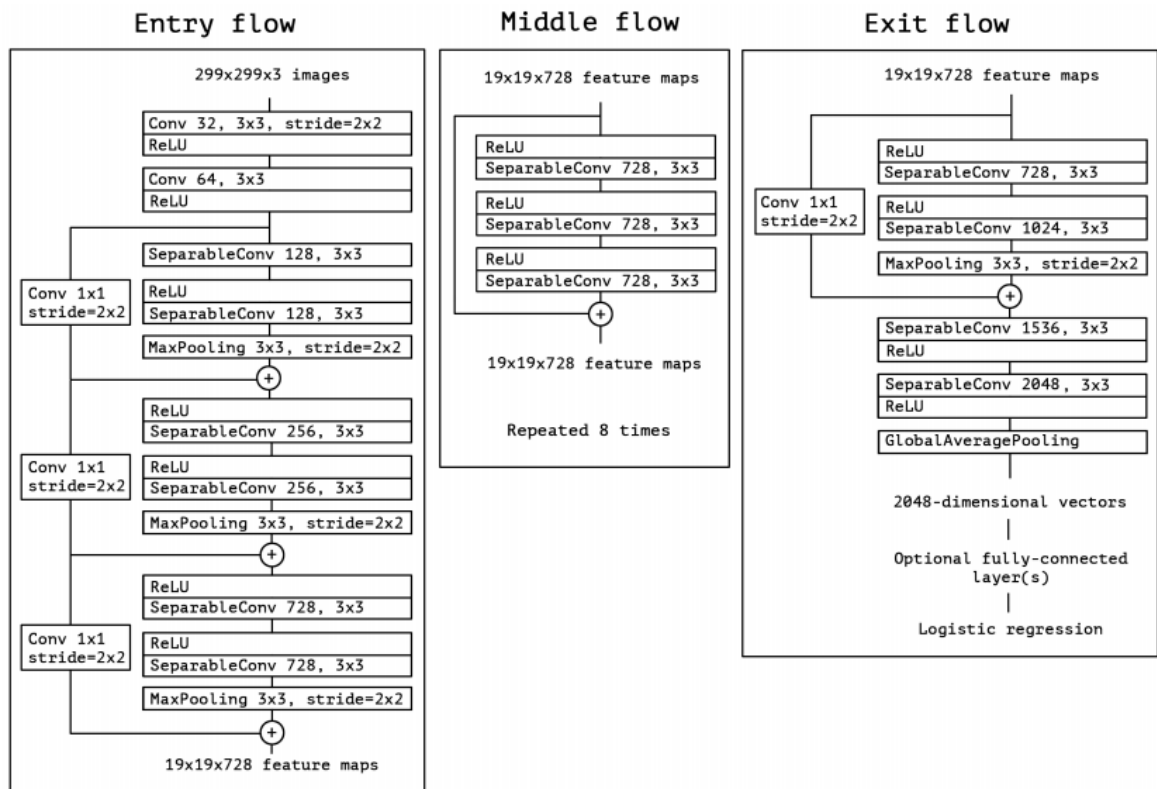


Figure 2.15: Architecture du modèle Xception

VGG16 : VGG16 est un modèle de réseau neuronal convolutif proposé par K. Simonyan et A. Zisserman de l'Université d'Oxford dans l'article ' Very DeepConvolutional Networks for Large-Scale Image Recognition '. Le modèle atteint une précision de test de 92,7% dans le top 5 dans ImageNet (un ensemble de données de plus de 14 millions d'images appartenant à 1000 classes). Ce modèle est l'un des algorithmes les plus populaires pour la classification d'images. Il a été entraîné sur 1,2 millions d'images de 1000 catégories différentes en utilisant un GPU Nvidia Titan Black pendant plusieurs semaines. Il fonctionne avec un système imbriqué de 3*3 couches convolutives empilées les unes sur les autres.

Il se compose de 16 couches convolutionnelles : 13 couches convolutionnelles et trois couches entièrement connectées (fully-connected). Il prend en entrée une image encodée en RVB et de taille 224 x 224 px. Ce modèle est assez simple à comprendre et à l'utiliser pour le Transfer Learning.[9]

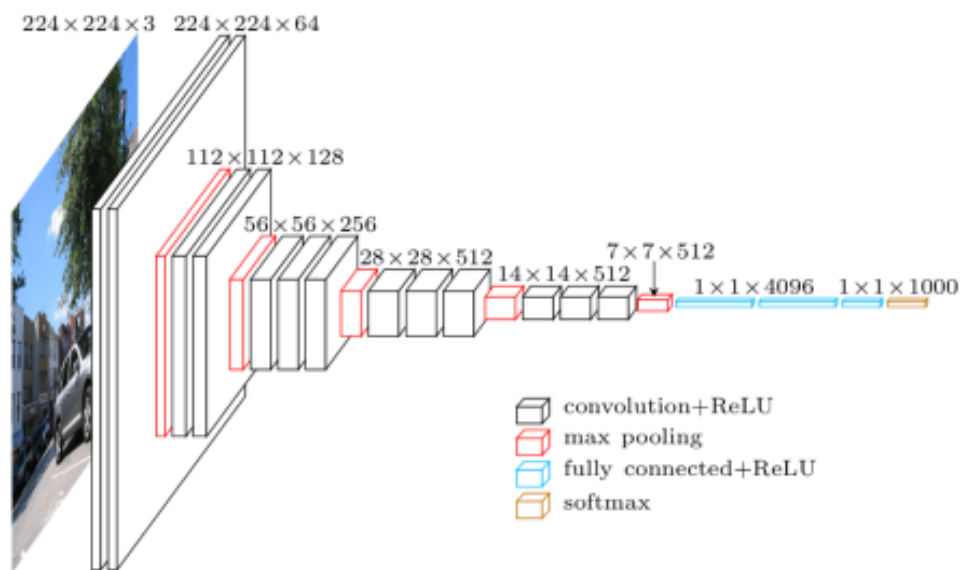


Figure 2.16: Architecture du modèle Xception

Resnet50 : Restnet(Residual Networks)est un réseau de neurones innovant qui a été introduit pour la première fois par Kaiming He, Xiangyu Zhang, Shaoqing Ren et Jian Sun dans leur article de recherche sur la vision par ordinateur de 2015 intitulé "Apprentissage résiduel profond pour la reconnaissance d'images ".Ce modèle a connu un succès immense

en 2015, il a remporté la première place à la compétition ILSVRC qui est une compétition annuelle de vision par ordinateur, avec une erreur de 3,57 %. En effet, ResNet a de nombreuses variantes qui fonctionnent sur le même concept mais ont des nombres de couches différents. Il est utilisé pour désigner la variante qui peut fonctionner avec 50 couches de réseau de neurones. Ce dernier prend une image d'entrée ayant une hauteur, et une largeur multiple de 32 et 3 pour la largeur de canal (ex : 224 x 224 x 3).[10]

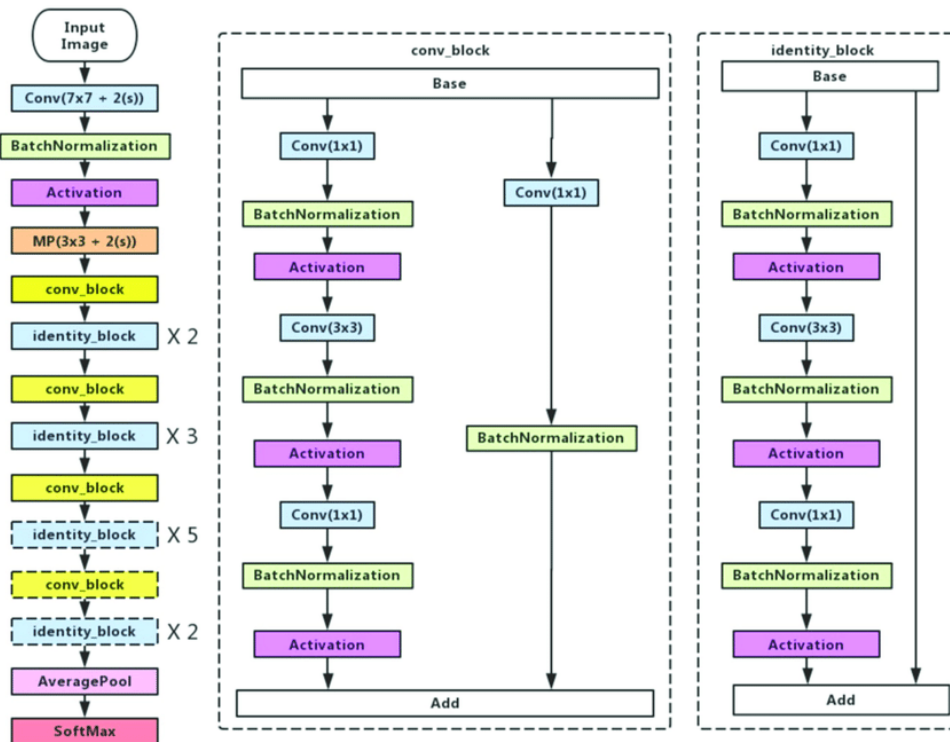


Figure 2.17 : Architecture du modèle Restnet50

CONCLUSION

Tout au long de ce chapitre, nous avons commencé par découvrir les concepts d'apprentissage automatique. Ensuite, la notion de détection d'anomalie et à la fin nous avons focalisés sur le Deep Learning et le Transfer Learning, les réseaux de neurone artificiels ANN et les CNN. Maintenant, nous pouvons commencer la partie de l'étude technique.

CHAPITRE 3: ETUDE TECHNIQUE

Plan

1. Environnement matériel :30
2. Environnement logiciel :30

INTRODUCTION :

Dans ce chapitre, nous arrivons à la phase d'achèvement du projet. Afin d'accomplir cette tâche avec succès, nous devons choisir les outils nécessaires à utiliser. Ce chapitre sera consacré, tout d'abord, à la présentation de l'environnement de travail technique.

3.1 Environnement matériel :

Tout au long de notre projet, nous avons utilisé un ordinateur bureautique avec les caractéristiques suivantes :

- Processeur : Intel® Core™ i5-10400F@2.90GHZ
- RAM : 16GB
- GPU : NVIDIA GeForce GTX 1660
- Disque Dur : 250 GB SSD
- Système d'exploitation : Windows 10 Pro 64-bit

3.2 Environnement logiciel :

Lors de l'élaboration du projet, nous avons utilisé les outils et les librairies suivants :

3.2.1 Anaconda :

Anaconda [11] 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 (traitement de données à grande échelle, analyse prédictive, calcul scientifique), qui vise à simplifier la gestion des paquets et de déploiement



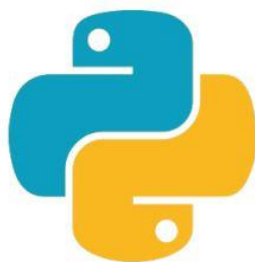
3.2.2 Jupyter Notebook :

JupyterNotebook [12] est une application Web Open Source permettant de créer et de partager des documents contenant du code (exécutable directement dans le document), des équations, des images et du texte. Avec cette application il est possible de faire du traitement de données, de la modélisation statistique, de la visualisation de données, du Machine Learning, etc. Elle est disponible par défaut dans la distribution Anaconda.



3.2.3 Python :

Le langage Python [13] est un langage de programmation open source multi-plateformes et orienté objet grâce à sa simplicité, compatibilité et ses bibliothèques spécialisées Python s'utilise pour de nombreuses situations comme le développement logiciel, l'analyse de données, ou la gestion d'infrastructures. Ce langage est très utilisé dans le domaine de machine learning et data science car il offre des dizaines de librairies de fonctions. Les plus utilisées en data-science sont Pandas, Numpy (tableaux multi-dimensionnels), Scipy (calcul scientifique), Matplotlib (tracé graphique) et Seaborn (représentation graphique multifenêtrage), Scikit-Learn (pour faire de l'apprentissage proprement dit) et Keras qui permet d'attaquer la bibliothèque TensorFlow (réseau de neurones et CNN pour Convolutional Neural Network pour le traitement d'images).



3.2.4 Scikit-learn :

scikit-learn [14] est une bibliothèque libre Python destinée à l'apprentissage automatique. Elle est développée par de nombreux contributeurs notamment dans le monde académique par des instituts français d'enseignement supérieur et de recherche comme Inria(Institut national de recherche en informatique et en automatique).Elle propose dans son Framework de nombreuses bibliothèques d'algorithmes à implémenter, clé en main. Ces bibliothèques sont à disposition notamment des data scientists.Elle comprend notamment des fonctions pour estimer des forêts aléatoires, des régressions logistiques, des algorithmes de classification, et les machines à vecteurs de support. Elle est conçue pour s'harmoniser avec d'autres bibliothèques libres Python, notamment NumPy et SciPy.



3.2.5 Tensorflow :

TensorFlow [15] est une bibliothèque open source, permettant d'exécuter des applications de machine learning et de Deep Learning. Cet outil dédié à l'apprentissage automatique a été développé par Google, et il est fortement utilisé dans le domaine de l'intelligence artificielle (IA). Ainsi, des professionnels comme des novices peuvent créer des modèles de machine learning ou de Deep Learning pour optimiser les capacités de leur matériel.



3.2.6 Keras :

Keras [16] est une bibliothèque open source de prototypage rapide de modèles de Deep Learning écrit en Python, à la portée des débutants en IA, elle s'articule autour d'une API de haut niveau supportant différentes librairies de réseaux de neurones artificiels récurrents ou convolutifs, comme Tensorflow, Microsoft Cognitive Toolkit, PlaidML ou Theano.

L'objectif de Keras est d'offrir un cadre pour développer au plus vite des réseaux de neurones artificiels. Initiée en 2015, cette technologie repose sur le travail de François Chollet, un développeur de Google. Elle s'inscrit dans le cadre du projet Oneiros (pour Open-ended Neuro-Electronic Intelligent Robot Operating System).



3.2.7 OpenCV :

OpenCV [17] est une bibliothèque graphique initialement développée par Intel. Elle est spécialisée dans le traitement d'images, que ce soit pour de la photo ou de la vidéo. Les algorithmes d'OpenCV permettent d'appliquer divers traitements sur les images pour y faciliter la détection d'éléments précis.



3.2.8 Microsoft OfficeWord :

Word [18] est l'un des logiciels de traitement de texte les plus utilisés dans le monde. Appartenant à la suite bureautique de Microsoft, Word permet de rédiger et mettre en forme des documents textes.



CONCLUSION

Dans ce chapitre, nous avons présenté l'environnement de développement, les différents logiciels et les bibliothèques utilisés.

CHAPITRE 4: REALISATION

Plan

1. Prétraitement des données	36
2. Entraînement des modèles	44

INTRODUCTION

Ce chapitre se répartit comme suit : la première partie sera consacrée pour les prétraitements de l'ensemble de données et la deuxième partie présentera l'implémentation des différents modèles CNN.

4.1 Prétraitement des données :

4.1.1 Collection de la base de données :

La collection et l'identification des données est une étape nécessaire pour la bonne performance des algorithmes de l'apprentissage automatique ou profond.

Dans ce projet nous avons collecté 118 images des clôtures à partir de Google.

Ces images sont catégorisées en deux types, des images des clôtures normales et d'autres anormales.

Donc notre base de données contient 118 images réparties sous deux classes :

- Une première classe contient 99 des images de clôtures normales.
- Une deuxième classe contient 19 des images de clôtures anormales.



Normal



Anormal



Normal



Anormal

Figure 4.1: Exemples des images des clôtures

4.1.2 Nettoyage des données :

Puisque nous avons collecté notre base de données de Google, toutes les images sont différentes en termes de taille, qualité et caractéristiques ce qui peut conduire à des mauvais résultats en entravant le processus d'entraînement de modèle. Donc il est nécessaire d'appliquer certaines transformations sur ces images pour qu'elles soient prêtes à être utilisées. A cet égard notre première étape dans ce projet a été le prétraitement ou le nettoyage des images.

Le but du nettoyage des images est d'améliorer la qualité des images afin que nous puissions mieux les analyser. Grâce au nettoyage, nous pouvons supprimer les distorsions indésirables et améliorer certaines fonctionnalités nécessaires pour notre application.

Dans ce projet, on a commencé par le redimensionnement des images pour qu'elles s'adaptent aux contraintes du modèle CNN de transfert Learning. Puis, on a appliqué une égalisation d'histogrammes des images pour améliorer le contraste des images. Finalement, on a appliqué le filtre non-local Means pour supprimer les bruits des images.

Avant d'appliquer tous ces transformations sur l'ensemble des images, on a choisi au hasard une image de la base de données pour le test dans le but d'évaluer chaque étape du nettoyage des images.

Redimensionnement des images :

Le but de cette étape est de changer les tailles des images en 229×229 pour qu'elles s'adaptent aux contraintes du modèle CNN sans changer ses attributs. Pour le redimensionnement des images on a utilisé la fonction `resize()` du OpenCV .

Tout d'abord nous avons réalisé la transformation sur l'image du test et on a obtenu le résultat suivant :

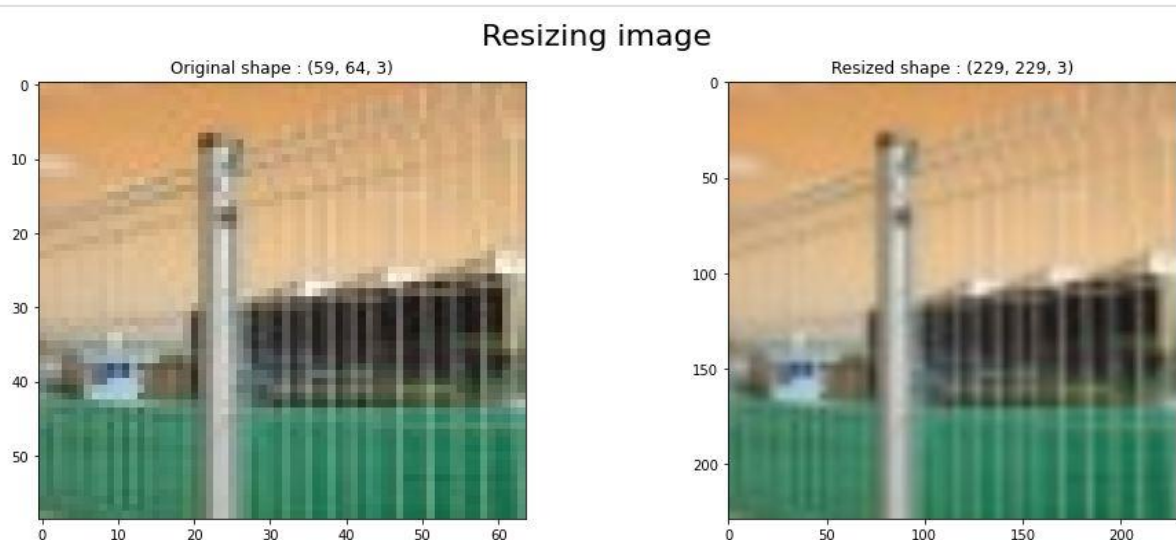


Figure 4.2: Redimensionnement d'une image de test

On peut remarquer dans la figure 4.2 que le redimensionnement en 229×229 a déformé la photo initiale du test. D'ailleurs, toutes les autres images seront déformées. A cet égard nous allons utiliser des méthodes basées sur les histogrammes de ces images pour prétraiter au mieux nos données.

Égalisation d'histogrammes des images:

L'égalisation d'histogrammes est une technique de traitement d'image utilisée pour améliorer le contraste des images en utilisant les histogrammes. En effet, l'histogramme d'une image est une courbe statistique qui représente la distribution de ses pixels selon leur intensité.

Toutes nos images sont représentées sous la forme RGB. Avec ce type de codage il est difficile de réaliser l'égalisation d'histogrammes. Donc, on a transformé l'image du test en YUV, ce système de codage est créé depuis une source RVB. Il est codé en 3 composantes : Y représente la luminance (informations de luminosité), U et V représentent des données de chrominance (informations de couleur).

On a obtenu l'histogramme de l'image du test suivant :

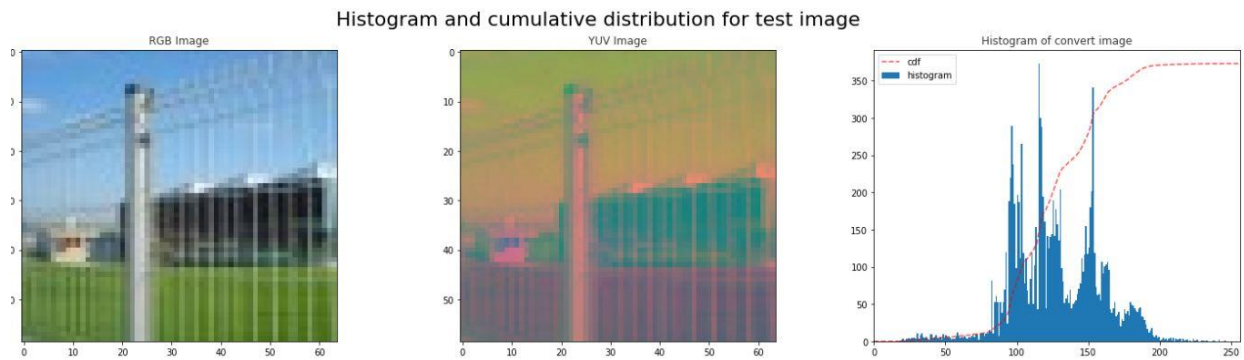


Figure 4.3: Histogramme de l'image de test

Depuis l'histogramme on constaté des pics importants au centre d'où la luminosité est plus élevée dans ces parties de l'image, et pour avoir une bonne égalisation il est nécessaire de répartir la lumière dans tout le spectre de l'image.

En effet, l'intérêt de convertir l'image dans l'espace colorimétrique YUV est de pouvoir agir sur le canal "luminance" (Y) indépendamment des autres canaux de chrominance. Donc on peut réaliser l'égalisation sur ce seul canal Y.

Nous avons testé l'égalisation avec OpenCV et on a obtenu le résultat suivant :

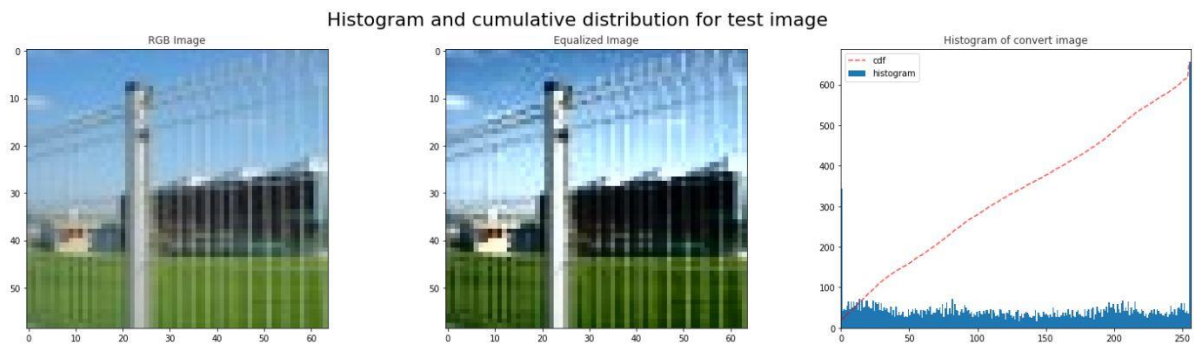


Figure 4.4: Histogramme de l'image test apres l'egalisation

Nous avons constaté que l'histogramme couvre l'ensemble du spectre et la CDF est constante et linéaire donc l'image résultante est bien égalisée.

⇒ Après avoir appliqué ce prétraitement sur une image de test, on a obtenu des résultats satisfaisants c'est pourquoi il sera appliqué à l'ensemble des images.

Application du filtre non-local Means:

Dés qu'on a collecté les images de Google, on n'a pas pu trouver des images parfaites avec des qualités meilleures or la plupart de ces images sont bruitées c'est à dire comportent des pixels atypiques qui peuvent fausser la détection de features.

En vue des résultats plus précis, il est nécessaire de remédier les bruits présents dans ces images. Donc dans cette étape on a décidé d'utiliser un filtre qui permet de supprimer les bruits en préservant les importants features. Il existe plusieurs types de filtres : les filtres linéaires (ex : le filtre Gaussien...) non linéaires (ex : le filtre médian) etc ...

Dans ce projet on a choisi le filtre non-local Means qui représente l'un des meilleurs filtres pour débruiter les images.

Après avoir appliqué non-local Means sur une image de test on a obtenu le résultat suivant :

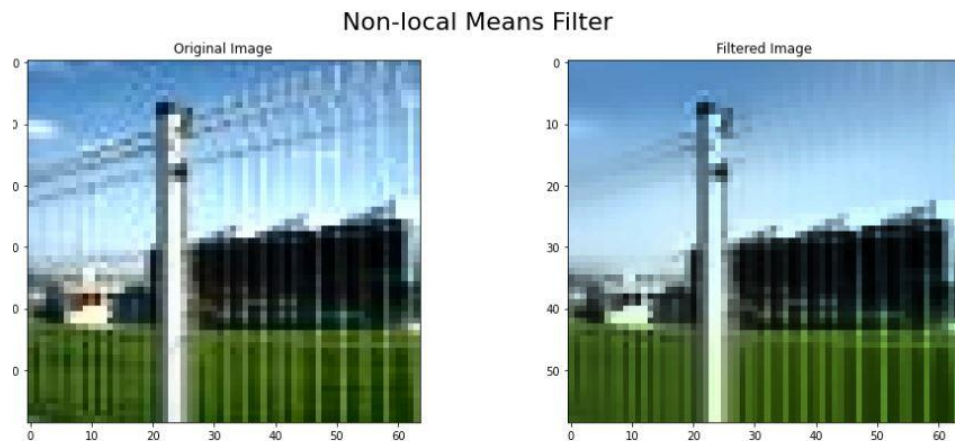


Figure 4.5: Application du filtre non-local

On peut simplement remarquer la disparition des bruits sans perdre les features importantes.

⇒ Les images sont prêtes pour être utilisées dans les modèles d'apprentissage.

4.1.3 Répartition de la base de données :

Dans cette étape on va diviser la base de données en trois catégories : 'Train' , 'Test', et 'Validation'.

Nous avons utilisé 90% des images pour la phase d'apprentissage/train, 25% pour la phase de Test et un sous-ensemble de validation qui représente 15% du train dans le but d'éviter le sur-apprentissage (Overfitting).

La répartition finale de la base de données est comme suit:

- 89 des images pour l'entraînement.
- 11 images pour le Test.
- 18 images pour la validation.

La figure suivante représente la répartition de la base de données :

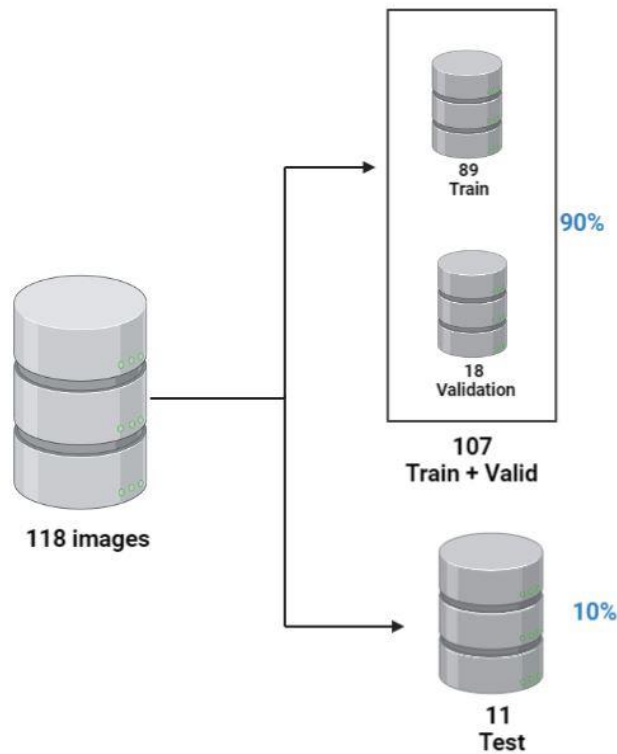


Figure 4.6: Répartition de la base de données

4.1.4 Techniques de prétraitement des données :

Le prétraitement des données est inévitable pour la construction du modèle au cours de l'apprentissage automatique. En effet, il correspond aux étapes appliquées sur les images avant qu'elles ne soient utilisées dans le modèle pour l'entraînement dans le but d'améliorer les résultats obtenus par les modèles entraînés. Parmi les étapes du pré-traitement on a déjà mentionné l'importance de redimensionner les images afin qu'elles s'adaptent à l'exigence de la taille d'images d'entrée de différents modèles CNN par exemple le modèle Xception qui nécessite des images de taille 299×299 comme entrées.

En effet, il existe plusieurs techniques pour améliorer les performances des modèles CNN et parmi les techniques les plus efficaces le 'data augmentation /l'augmentation des données'.

L'augmentation des données :

C'est un ensemble des techniques visant à augmenter artificiellement la quantité des images en générant des nouveaux échantillons à partir des échantillons existants. Il s'agit notamment d'apporter des petites modifications aux caractéristiques des images (taille, position, zoom, etc.) d'où la création des nouvelles images avec des caractéristiques différentes. En effet, cet ensemble de techniques permet d'améliorer la performance du modèle. D'ailleurs cette opération est très importante lorsque la taille de la base de données est très petite, ce qui est le cas de notre projet, le nombre d'images total est de 118, Ce petit nombre d'images peut être insuffisant pour bien entraîner les modèles CNN. En appliquant l'augmentation, des nouvelles données vont être générées d'où plus échantillons sont fournis pour l'apprentissage qui mène à améliorer la performance du modèle. A cet égard, nous avons utilisé cette technique à l'aide de la classe ImageDataGenerator de la bibliothèque Keras.

ImageDataGenerator:

ImageDataGenerator représente un moyen rapide et facile d'augmenter nos images. Il fournit une multitude de techniques d'augmentation différentes telles que la standardisation, la rotation, les décalages, les retournements, le changement de luminosité et bien d'autres. L'une de ces avantages les plus importants est sa capacité à produire une augmentation d'image en temps réel et éviter le sur-apprentissage. De plus, il peut générer dynamiquement des images e lorsque que le modèle est encore en phase d'entraînement. de plus il garantit que le modèle reçoit de nouvelles variations des images à chaque époque. Mais il ne renvoie que les images transformées et ne les ajoute pas aux images d'origine. En outre, l'utilisation d'ImageDataGenerator nécessite moins de consommation du mémoire grâce à sa capacité de changer les images par lots (Epochs). [19]

La figure suivante représente les paramètres utilisés pour l'augmentation :

```
train_datagen = ImageDataGenerator(rescale=1./255,
                                   rotation_range=40,
                                   width_shift_range=0.2,
                                   height_shift_range=0.2,
                                   shear_range=0.2,
                                   zoom_range=0.2,
                                   channel_shift_range=20,
                                   horizontal_flip=True,
                                   )
```

Figure 4.7 : Les paramètres utilisés pour l'augmentation

- **Rescale (1. /255)** : il s'agit de transformer chaque valeur de pixel de la plage [0,255] en [0,1]. Le but de cette opération est de traiter toutes les images de la même manière puisque certaines images ont une plage de pixels élevée, d'autres une plage de pixels faible.
- **Rotation_range(40)** : permet de pivoter l'image au hasard entre 0 et 40 degrés.
- **Width_shift_range (0.2)** : permet de décaler l'image au hasard vers la gauche ou la droite (décalages horizontaux) de 20%.
- **Height_shift_range (0.2)** : fonctionne de la même manière que width_shift_range mais le décalage se fait verticalement (vers le haut ou vers le bas).
- **Shear_range (0.2)** : Moyens de cisailier l'image de 20%.
- **Zoom-range (0.2)** : permet de zoomer au hasard à l'intérieur des images. Dans notre cas d'agrandir l'image plus au moins 20%.
- **Channel_shift_range(20)** : permet décaler au hasard chaque canal de l'image.
- **Horizontal_flip(True)** : permet de retourner essentiellement les lignes et les colonnes horizontalement.

Ces transformations sont seulement appliquées sur les images de la phase d'entraînement.

Pour les images de Test et validation, nous n'appliquons que l'option "rescale=1./255" comme il est indiqué dans la figure :

```
val_datagen = ImageDataGenerator(rescale=1./255)
```

Figure 4.8: Transformation des image test et validation

La figure suivante présente l'application de l'augmentation sur une image d'entraînement :

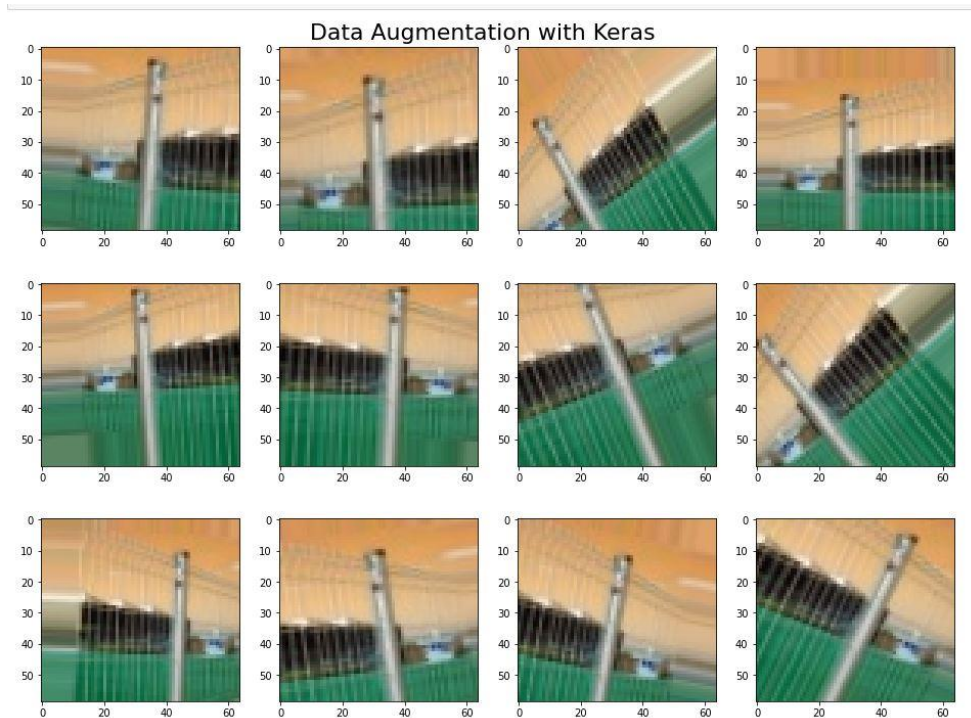


Figure 4.9: L'application de l'augmentation

4.2 Entraînement des modèles :

Dans cette phase nous avons entraîné trois réseaux de neurones de la bibliothèque Keras qui sont pré-entraînés sur la grande base de données "ImageNet" sont : Xception, VGG16, et ResNet50 en appliquant la méthode de transfert d'apprentissage (Transfer Learning).

4.2.1 Hyperparamètres liés à l'algorithme d'entraînement :

Un hyperparamètre correspond à un paramètre dont la valeur est utilisée pour contrôler la phase d'apprentissage.

— **Learning_rate (0.001)** : Le taux d'apprentissage est un facteur qui définit la rapidité avec laquelle un réseau met à jour ses paramètres.

— **Epochs (100)** : est un hyperparamètre qui définit le nombre de fois durant lesquelles l'algorithme d'apprentissage fonctionnera à travers l'ensemble des données d'entraînement.

— **Batch_size (32)** : correspond au nombre d'images entraînées en une itération.

4.2.2 Paramètres d'optimisation :

Optimizer :

C'est un algorithme ou une méthode utilisée pour minimiser une fonction d'erreur ou pour maximiser l'efficacité de la production. Nous avons utilisé <Adam > comme un optimizer. [20]

Fonction de perte :

La fonction de perte est une fonction qui évalue la différence entre les prédictions faites par le réseau de neurones et les valeurs réelles des observations utilisées lors de l'apprentissage. Pour la classification des deux classes nous avons utilisé "categorical_crossentropy".

Callbacks :

Un rappel est un ensemble de fonctions à appliquer à des étapes données de la procédure d'apprentissage. Nous pouvons utiliser des rappels pour obtenir une vue sur les états internes et les statistiques du modèle pendant l'entraînement. Pour pouvoir surveiller les performances des métriques à certains moments de l'exécution d'entraînement et pour effectuer une action qui peut dépendre de ces performances dans les valeurs métriques. [21]

Durant l'entraînement du réseau de neurones nous avons utilisé la fonction " EarlyStopping " de callbacks qui permet d'arrêter l'entraînement du modèle à certaines conditions.[22]

```
EarlyStopping= keras.callbacks.EarlyStopping(monitor='val_loss',  
                                             patience=10,  
                                             mode='min',  
                                             verbose=1,  
                                             restore_best_weights=True)
```

Figure 4.10: La fonction " EarlyStopping "

CONCLUSION :

Dans ce chapitre nous avons préparé les bases de données, ainsi que les techniques de prétraitement, par la suite nous avons implémenté la phase de l'apprentissage par l'entraînement de cinq modèles. Donc le chapitre suivant sera consacré pour la discussion des résultats obtenus par l'apprentissage de ces modèles.

CHAPITRE 5: RESULTATS OBTENUS

Plan

1. Indicateurs de performance	48
2. Résultats obtenus et évaluation	50
3. Comparaison des performances des modèles	54

INTRODUCTION :

Ce chapitre sera consacré aux différentes métriques d'évaluation ainsi qu'aux résultats obtenus par l'apprentissage de chaque modèle entraîné.

5.1 Indicateurs de performance :

Après l'implémentation de notre modèle il est nécessaire de l'évaluer. Donc l'étape suivante consiste à déterminer l'efficacité du en utilisant des métriques. De cette façon, les performances du modèle peuvent être améliorées en ajustant les hyper paramètres ou en ajustant les caractéristiques de l'ensemble de données d'entrée. Ces métriques nous permettent aussi de comparer plusieurs modèles et choisir le meilleur modèle pour résoudre notre problème.

5.1.1 Précision moyenne (Accuracy) :

La précision est l'une des métriques d'évaluation les plus simples à utiliser et à mettre en œuvre il s'agit d'une mesure qui décrit généralement les performances du modèle dans toutes les classes. Elle est définie comme le nombre de prédictions correctes divisé par le nombre de prédictions total. Or le nombre de prédictions correctes est égal à la somme du nombre de vrais positifs "True positive (TP)" et de vrais négatifs "True Negative (TN)", ainsi que le nombre de prédiction total est égal à la somme du nombre de vrais positifs, de vrais négatifs, de faux positifs "False Positive (FP)" et de faux négatifs "False Negative (FN)".[23]

$$\text{Accuracy} = \frac{\text{nombre de prédictions correctes}}{\text{nombre de prédictions total}} \quad (5.1)$$

Or:

nombre de prédictions correctes : True positives (TP) + True negatives (TN)

nombre de prédictions total : True positives (TP) + True negatives (TN) + False positives (FP) + False negatives (FN)

- True positives(TP) : un résultat où le modèle prédit correctement la classe positive
- Truenegatives(TN) : un résultat où le modèle prédit correctement la classe négative
- False positives(FP) : un résultat où le modèle prédit incorrectement la classe négative
- False negatives(FN) : un résultat où le modèle prédit incorrectement la classe négative

5.1.2 Précision :

La précision est calculée comme le rapport entre le nombre d'échantillons positifs correctement classés et le nombre total d'échantillons classés comme positifs :

$$\text{Précision} = \frac{TP}{TP+FP} \quad (5.2)$$

5.1.3 Rappel :

Il s'agit du rapport entre le nombre correctement prédit comme positifs et le nombre réellement positifs. Le rappel est calculé avec la formule suivante :

$$\text{Rappel} = \frac{TP}{TP+FN} \quad (5.3)$$

5.1.4 F1-score :

La métrique F1-score utilise une combinaison de précision et de rappel. En fait, le score F1 est la moyenne harmonique des deux. La formule des deux est :

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Rappel}}{\text{Precision} + \text{Rappel}} \quad (5.4)$$

Il est plus efficace pour évaluer les systèmes de classification binaire, qui classent les exemples en "positif" ou "négatif". [24]

5.1.5 Matrice de confusion :

La matrice de confusion n'est pas exactement une mesure de performance, mais une sorte de base sur laquelle d'autres mesures évaluent les résultats. Il s'agit d'une visualisation tabulaire des prédictions du modèle par rapport au résultat réel. Chaque ligne de la matrice de confusion représente les instances d'une classe prédite et chaque colonne représente les instances d'une classe réelle.[25]

		Predicted Class	
		Positive	Negative
Actual Class	Positive	True Positive (TP)	False Negative (FN) Type II Error
	Negative	False Positive (FP) Type I Error	True Negative (TN)

Figure 5.1: Matrice de confusion

5.2 Résultats obtenus et évaluation :

Comme il est mentionné dans le chapitre précédent, nous avons réalisé l'apprentissage de quatre différents réseaux de neurones sur notre base de données. Dans cette partie nous allons présenter les différents résultats obtenus par l'apprentissage de ces 4 modèles (CNN, Xception, VGG16 et Resnet50) et l'évaluer selon les différentes métriques d'évaluation pour aboutir à une comparaison entre les différents résultats.

5.2.1 Résultats obtenus par CNN :

CNN 3 Layers :

Tout d'abord on a commencé par la construction d'un modèle CNN à 3 Layers. Après l'entraînement de ce modèle, on est passé à l'étape de l'évaluation. En fait, l'évaluation du modèle CNN 3 layers, nous permet de savoir une idée sur la performance de ce modèle. On a utilisé la précision moyenne (accuracy) comme une métrique d'évaluation du modèle. Les résultats sont représentés dans la figure suivant il s'agit d'un courbe des précisions en fonctions des 'Epoche' pour les 2 phases : validation et entraînement

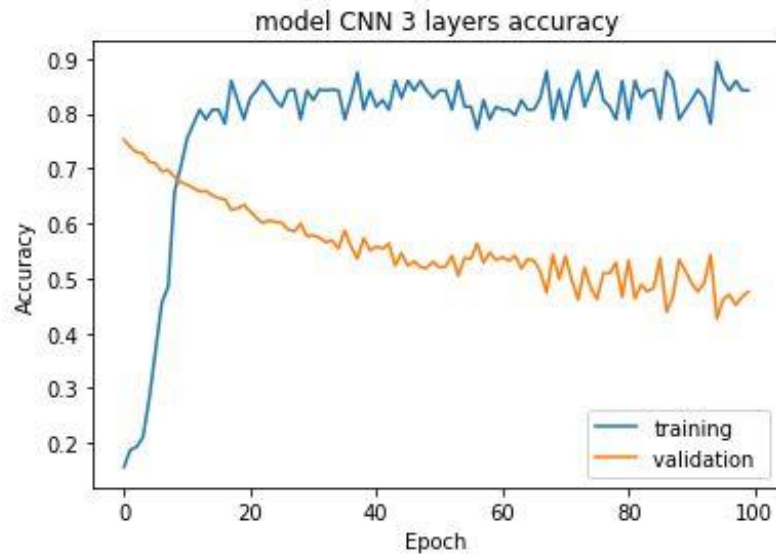


Figure 5.2:Résultats obtenus par CNN 3 layers

On peut constater d'après la courbe que plus que le nombre des Epoch augmente plus la précision diminue de la validation et la précision de l'entraînement (training) augmente.

CNN 5 Layers :

Dans le but d'améliorer le modèle précédant, on a ajouté 2 autres couches pour construire un modèle CNN à 5 Layers. On a obtenu les résultats suivants :

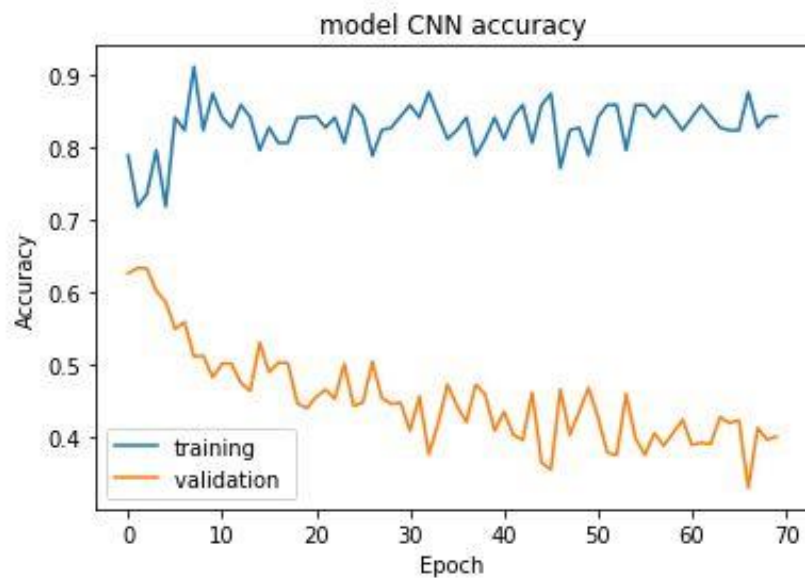


Figure 5.3 : Résultats obtenus par CNN 5 layers

On peut constater qu'en augmentant le nombre de couche d'un modèle CNN, la précision moyenne du modèle s'améliore. D'ailleurs, les résultats obtenus par le modèle à 5 couches sont mieux que le résultat obtenu par celui à 3 couches.

5.2.2 Résultats obtenus par Xception :

Après l'apprentissage du modèle Xception, nous avons évalué ce modèle sur un ensemble d'images de test. Cette évaluation est représentée sous la forme d'une courbe des précisions moyenne en fonction des Epoches. Le résultat est illustré dans la figure suivante :

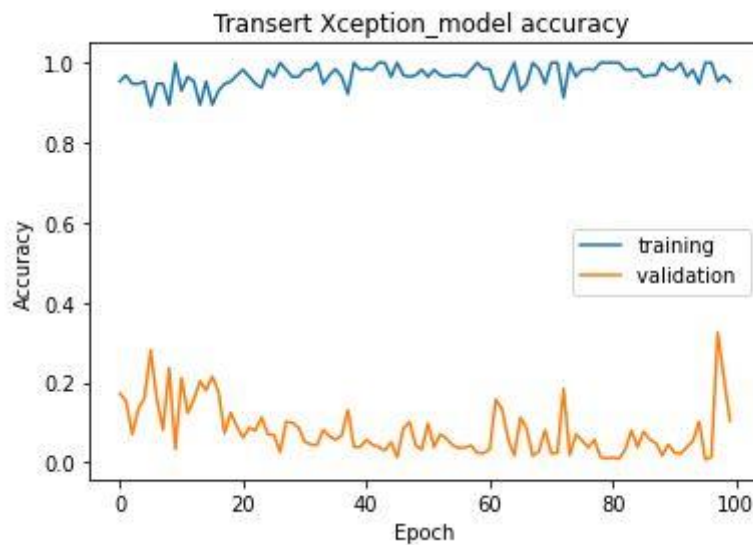


Figure 5.4: Résultats obtenus par Xception

On peut remarquer que ce modèle atteint de performance maximale en termes de précision moyenne lorsque on a augmenté l'Epoch, il a atteint une précision maximale de 40% pour la validation et 99% pour l'apprentissage.

5.2.3 Résultats obtenus par VGG16 :

La figure ci-dessous présente les résultats de l'apprentissage de classification par le modèle VGG16 :

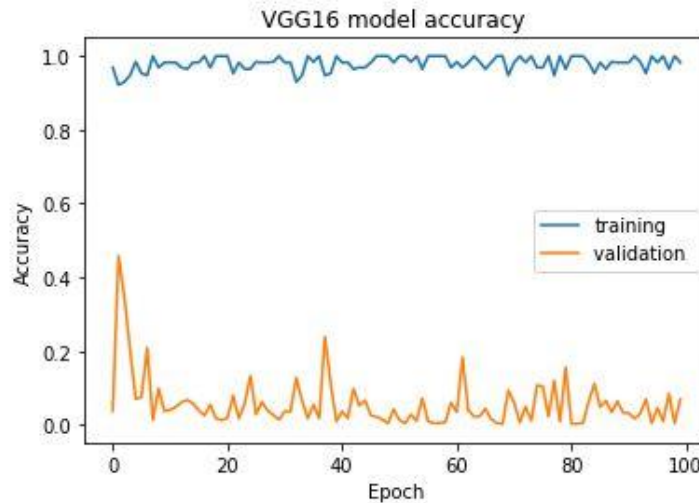


Figure 5.5 : Résultats obtenus par VGG16

Il s'agit de 2 courbes : première courbe représente la précision en fonction des Epoches pour la phase de traitement et une deuxième courbe pour la validation. Cette représentation nous permet de choisir le meilleur hyper paramètre 'Epoch'.

5.2.4 Résultats obtenus par ResNet50 :

De même manière d'évaluation des modèles précédents, on a utilisé les courbes de précision en fonction des Epoches pour évaluer ce modèle. Les résultats de l'apprentissage de classification par le modèle ResNet50 sont représentés dans la figure ci-dessous :

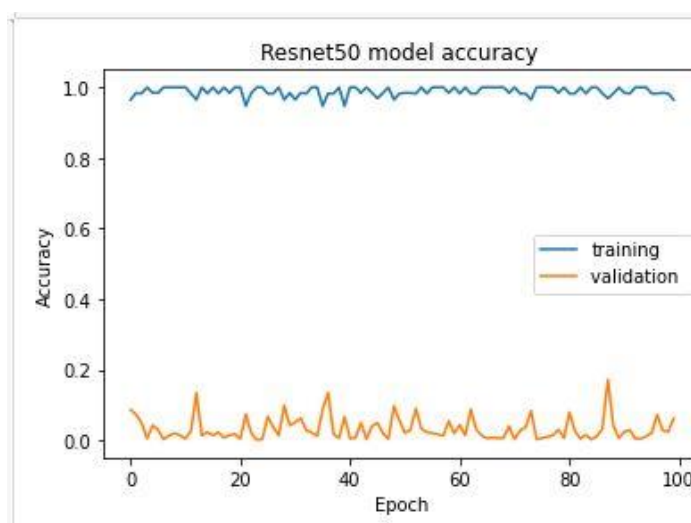


Figure 5.6: Résultats obtenus par Resnet50

5.3 Comparaison des performances des modèles :

La comparaison des résultats obtenus par les différents modèles est illustrée dans le tableau suivant.

Tableau 5.1 :Comparaison des performances des modèles

Modèle	Accuracy(%)
ResNet50	99
VGG16	98
Xception	97
CNN	92

Nous pouvons interpréter d'après la figure de comparaison ci-dessus que le modèle ResNet50 a eu la meilleure performance parmi les 4 modèles, la précision moyenne, la précision est atteinte 99%. Ce modèle est suivi respectivement par les modèles VGG16 (98%) , Xception (97%) et CNN (92%).

CONCLUSION

Nous avons présenté dans ce chapitre les différentes métriques d'évaluation puis nous avons évalué les résultats obtenus par les six modèles CNN (CNN à 3 layers, CNN à 5 Layers, Xception, VGG16, ResNet50).Par la suite, nous avons terminé par une comparaison des performances entre les différents modèles.

CONCLUSION GENERALE

Ce rapport est une présentation du travail réalisé durant notre stage de fin d'études pour l'obtention de notre diplôme national de licence en ingénierie des systèmes informatique au sein de l'ISSAT Sousse.

Le but de ce projet était de construire un modèle capable de détecter d'anomalie des clôtures en se basant sur l'apprentissage profond. Ces modèles peuvent aider les particuliers à évaluer la sécurité de leurs propriétés publiques et privées.

Ce projet compris principalement deux parties, une première partie consacrée pour la description de l'état de l'art et une deuxième partie pour l'implémentation de la solution.

Ce travail nous a permis d'approfondir nos compétences dans le domaine l'intelligence artificielle surtout le domaine de l'apprentissage profond et d'acquérir un savoir non négligeable. De plus, nous avons eu l'occasion d'acquérir une expérience en gestion des projets.

Nous pouvons envisager plusieurs améliorations qui entrent dans le cadre des perspectives de notre projet, en effet il est possible d'ajouter des nouvelles fonctionnalités à notre application.

En fin, nous souhaitons que notre effort donne satisfactions aux membres du jury, à notre encadreur et à tous nos enseignants, qui ne cessent de nous donner la formation adéquate.

BIBLIOGRAPHIE

- [1] A. G. PEREZ, «Automatic detection of anomalies in the wire fence,» Madrid, 2021.
- [2] P. Spagnolo, «Semi-Supervised Anomaly Detection in Video-Surveillance Scenes in the Wild» 2021. Address : <https://www.mdpi.com>
- [3] K. M. A. A. & A. P. J. D. Nasaruddin Nasaruddin, «Deep anomaly detection through visual attention in surveillance videos,» 2020. Address: <https://journalofbigdata.springeropen.com>
- [4] (). Address: <https://blog-gestion-de-projet.com>
- [5] s. haykin, «neural networks: a comprehensive foundation,» prentice hall, new jersey, 1999.
- [6] M. Gurucharan, «BASIC CNN ARCHITECTURE: EXPLAINING 5 LAYERS OF CONVOLUTIONAL NEURAL NETWORK,» 2020. Address: <HTTPS://WWW.UPGRAD.COM>.
- [7] G. B, «Convolutional neural network,» 2020. Address : <https://datascientest.com>
- [8] F. Chollet, «Xception : Deep Learning with Depthwise Separable Convolutions,» 2017. Address: <https://openaccess.thecvf.com>
- [9] M. u. Hassan, « VGG16 – Convolutional Network for Classification and Detection,» 2018. Address : <https://neurohive.io/en/popular-networks/vgg16/>.
- [10] G. Boesch, «Deep Residual Networks (ResNet, ResNet50),» 2022. Address : <https://viso.ai/deep-learning/resnet-residual-neural-network>
- [11] (). Address : [https://fr.wikipedia.org/wiki/Anaconda_\(distribution_Python\)](https://fr.wikipedia.org/wiki/Anaconda_(distribution_Python))
- [12] (). Address : <https://www.lebigdata.fr/jupyter-notebook>.

- [13] (). Address: <https://www.lebigdata.fr/python-langage-definition>.
- [14] (). Address: <https://fr.wikipedia.org/wiki/Scikit-learn>.
- [15] (). «TensorFlow : tout savoir sur la plateforme de deep learning de Google,» Address: <https://www.journaldunet.fr/web-tech/guide-de-l-intelligence-artificielle/1501861-tensorflow-tout-savoir-sur-la-plateforme-de-deep-learning-de-google>
- [16] (). Address: <https://opencv.org/>.
- [17] ().Address: https://simple.wikipedia.org/wiki/Microsoft_Word
- [18] (). «Image Data Preprocessing,». Address: <https://keras.io/api/preprocessing/image/>.
- [19] Musstafa. «Optimizers in Deep Learning,». 2021. Address: <https://medium.com/mllearning-ai/optimizers-in-deep-learning>
- [20] M. Cloney, «Keras Callbacks — Monitor and Improve Your Deep Learning,,» 2018. Address:<https://medium.com/singlestone/keras-callbacks-monitor-and-improveyour-deep-learning>
- [21] (). «Early Stopping,». Address : https://keras.io/api/callbacks/early_stopping/.
- [22] (). «Accuracy (error rate)» . Address: <https://deepai.org/machine-learning-glossary-and-terms/accuracy-error-rate>
- [23] () .Address: <https://www.educative.io/answers/what-is-the-f1-score>.
- [24]D.Democracy.«AjayKulkarni,».2020.Address:<https://www.sciencedirect.com/topics/engineering/confusion-matrix>

Résumé

Dans ce projet, nous avons créé un modèle basé sur l'apprentissage profond. Ce dernier est mis en œuvre sur une base de données des images des clôtures qui nous avons collectés par l'implémentation des trois réseaux de neurones convolutifs Xception, ResNet50, VGG16 et CNN. Par cette technique, Le meilleur résultat a été obtenu en utilisant ResNet50 qui avait la valeur de performance la plus élevée observée avec une précision de 99%.

Mots-clés : Détection d'anomalie, Apprentissage profond, réseaux de neurones convolutifs.

Abstract

In this project, we created a model based on deep learning. This last one is implemented on a database of images of the fences which we have collected by the implementation of the three convolutional neural networks Xception, ResNet50, VGG16 and CNN. By this technique, the best result was obtained by using ResNet50 which had the value highest performance observed with 99% accuracy.

Keywords : Anomaly detection , Deep learning , convolutional neural networks