REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : /M2/2023

Mémoire

présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Intelligence Artificielle**

Détection de somnolence des conducteurs utilisant l'apprentissage profond

---

Par :

BABAHENINI MAHMOUD CHAKIB

Soutenu le 02 juillet 2023, devant le jury composé de :

YOUKANA Imene	MCB	Président
BABAHENINI Mohamed Chaouki	Professeur	Rapporteur
HIOUANI Rima	MAB	Examinateur

Remerciements

Je remercie tout d'abord le bon dieu pour m'avoir donnée le courage et la santé nécessaires afin d'accomplir ce travail.

Je souhaite exprimer ma profonde gratitude envers le Prof. BABAHENINI Mohamed Chaouki, mon encadreur de mémoire. Je le remercie sincèrement pour son accompagnement attentif, son suivi régulier, ses conseils inestimables et son soutien indéfectible tout au long de mon projet. Sa présence et son expertise ont joué un rôle inestimable dans ma réussite.

Enfin, je voudrais adresser ma gratitude et mes sincères remerciements à tous ceux qui nous ont apporté leur aide et leur soutien, que ce soit de près ou de loin, dans la réalisation de ce projet. J'aimerais également exprimer ma reconnaissance envers tous mes amis pour leur soutien inconditionnel.

Résumé

La somnolence chez les conducteurs est un état de fatigue excessive qui peut compromettre la sécurité sur la route. Elle est souvent causée par le manque de sommeil, les troubles du sommeil ou la prise de certains medicaments. La somnolence au volant peut entraîner une diminution de l'attention, des temps de réaction plus longs et une augmentation du risque d'accidents graves.Cependant, la détection de la somnolence chez les conducteurs peut s'avérer difficile, l'utilisation de traitement d'imagerie et Les méthodes d'apprentissage profond ont montré des résultats prometteurs dans le domaine de la détection de la somnolence.

Dans ce travail, nous avons construit un modèle YOLO-LSTM pour détecter automatiquement la somnolence à partir d'une caméra. L'ensemble de données (dataset) utilisé dans notre étude contient des images des conducteurs. Les résultats obtenus démontrent l'efficacité de notre modèle pour détection de la somnolence chez les conducteurs.

Mots-clés : Somnolence, Apprentissage profond, YOLO, LSTM, Dataset.

Abstract

Driver drowsiness is a state of excessive fatigue that can compromise safety on the road. It is often caused by lack of sleep, sleep disorders or the use of certain medications. Drowsiness at the wheel can lead to reduced attention, longer reaction times and an increased risk of serious accidents. However, detecting drowsiness in drivers can be difficult, and the use of imaging processing and deep learning methods have shown promising results in the field of drowsiness detection.

In this work, we have built a YOLO-LSTM model to automatically detect drowsiness from a camera. The dataset used in our study contains images of drivers. The results obtained demonstrate the effectiveness of our model for detecting drowsiness in drivers.

Keywords : Drowsiness, Deep learning, YOLO, LSTM, Dataset.

Table des matières

Introduction générale **1**

1 Techniques de détection de la somnolence chez les conducteurs **4**

1 Introduction 5

2 Problème de la somnolence lors de la conduite 5

2.1 Relation entre la somnolence et la conduite 5

2.2 Les facteurs influençant la somnolence 6

3 La reconnaissance faciale 6

3.1 Introduction aux expressions faciales 6

3.2 Définition de visage 7

3.3 Les étapes de reconnaissance faciales 8

3.3.1 Détection de visage 8

3.3.2 Extraction des caractéristiques du visage 9

4 Techniques de détection de la somnolence des conducteurs 9

4.1 Techniques basées sur le traitement d'images 10

4.1.1 Techniques de correspondance des modèles 10

4.1.2 Techniques basée sur le clignement des yeux 10

4.1.3 Technique basée sur le bâillement 11

4.2 Technique basée sur le signal EEG 13

4.3 Techniques basées sur les réseaux de neurones artificiels 13

5 Conclusion 13

2 Apprentissage profond : outils théoriques pour traité la somnolence 15

1 Introduction 16

2 Exploration de l'apprentissage automatique 16

2.1 Les phases de l'apprentissage automatique 16

2.2 Types d'apprentissage automatique 17

2.2.1 Apprentissage supervisé (Supervised learning) 18

2.2.2 Apprentissage non supervisé (Unsupervised learning) . . 19

2.2.3 Apprentissage par renforcement (Reinforcement learning) 20

3 Classification des méthodes d'apprentissage profond 21

3.1 Réseaux de neurones artificiels 21

3.2 Les fonctions d'activation 22

3.2.1 ReLu 22

3.2.2 Sigmoid 23

3.2.3 Softmax 23

3.3 Réseaux de neurones récurrents 24

3.4 Réseaux de neurones convolutionnels 24

3.4.1 L'architecture réseaux de neurones convolutionnels . . . 25

3.4.2 Architectures CNN populaires 27

3.5 Réseaux de mémoire à long terme et à court terme (LSTM) . . . 29

3.5.1 LSTM et RNN 31

4 Travaux connexes à la détection de la somnolence 32

5 Conclusion 35

3 Conception de l'architecture pour la détection de la somnolence 36

1 Introduction 37

2 Objectif 37

3 L'architecture générale 37

4 L'architecture détaillée 38

4.1 Préparation des données 39

4.1.1 Dataset pour YOLO 39

4.2 Chargement des données 41
4.3 La division de l'ensemble de données 41
4.4 L'architecture YOLOv4 42
4.4.1 Composants d'un détecteur d'objets YOLO : 42
4.4.2 Sélection de l'architecture YOLOv4 43
4.5 Détection 45
4.6 Évaluation du modèle YOLO 45
4.7 Dataset pour LSTM : 46
4.8 Détection de la somnolence 46
5 Conclusion 46

4 Implémentation de l'architecture et résultats 47

1 Introduction 48
2 Environnement de développement du matériel 48
3 Environnement de déloppement logiciel 48
3.1 Python 49
3.2 Google Colab 49
3.3 OpenCV 49
3.4 NumPy 49
3.5 Matplotlib 49
4 Préparation et prétraitement de dataset 50
4.1 Extraction des images par une vidéo 50
4.2 La création de labels 51
4.3 La division de Dataset 52
5 Construction le modèle YOLO 53
5.1 Télécharger pré-entraînée weights 53
5.2 Préparation des données 53
5.3 Entraînement le modèle YOLOV4 55
6 Réalisation notre modèle LSTM 55
6.1 Importer les bibliothèques et modules 55

6.2 Chargement de l'ensemble des données(Dataset) 56
6.3 Entraînement modèle LSTM 57
7 Résultats obtenus à la détection de la somnolence 58
7.1 Présentation des performances obtenu 58
7.1.1 YOLO 58
7.1.2 LSTM 60
8 Détection de la somnolence 62
9 Conclusion 62

Conclusion générale et perspectives **64**

Table des figures

1.1 Muscles faciaux et leur contrôle nerveux.[2] 7
1.2 Les différentes méthodes de détection de somnolence. 9
1.3 Illustration des contours supérieurs et inférieurs et la structure du bâillement. 12

2.1 Les principaux types d'apprentissage automatique.[35] 17
2.2 La classification et la régression.[50] 19
2.3 créer clusters à partir des données.[38] 20
2.4 Architecture des réseaux neuronaux artificiels.[26] 22
2.5 Réseaux de neurones récurrents. 24
2.6 A.Représentation par couche convolutive[37] 25
2.7 B.Représentation par couche convolutive[37] 26
2.8 Types pool.[6] 27
2.9 L'architecture VGGNet. 28
2.10 L'architecture ResNet. 28
2.11 Architecture d'une cellule LSTM.[5] 29
2.12 Les composants d'une cellule LSTM.[5] 30
2.13 Le modèle LSTM.[5] 31
2.14 LSTM et RNN.[18] 32

3.1 L'architecture générale. 38
3.2 L'architecture détaillée. 39
3.3 Création label. 40

3.4 Les classes dataset. 40
3.5 Labels des 5 classes. 41
3.6 Division de l'ensemble des données(Dataset). 42
3.7 Détecteur d'objets de l'architecture YOLO.[11] 43
3.8 Boîtes de délimitation pour la dimension et localisation prédiction.[46] . . 44

4.1 Extraire une séquence d'images d'une vidéo. 51
4.2 Création de labels. 52
4.3 Télécharger YOLOV4 pré-entraînée weights. 53
4.4 Fichier obj.names pour YOLOV4 54
4.5 Fichier obj.data pour YOLOV4. 54
4.6 Entraînement le modèle YOLOV4. 55
4.7 Courbe Loss et mAP. 58
4.8 Résultat du modèle YOLO. 59
4.9 Exemple de tests. 60
4.10 Accuracy. 61
4.11 Loss 61
4.12 Résultat de la détection 62

Liste des tableaux

2.1 Travaux connexes à la détection de la somnolence 34

4.1 Description des bibliothèques et modules utilisés. 56

Introduction générale

L'étude de la Fondation pour la sécurité routière de l'Association américaine des automobilistes a révélé que la fatigue du conducteur, responsable de 16% à 21% des accidents de la route, souligne l'importance de détecter les signes de fatigue, ce que la reconnaissance faciale pourrait contribuer à accomplir en évaluant les caractéristiques du visage, afin de prévenir les conséquences graves de la somnolence au volant telles que le ralentissement du temps de réaction, la diminution de l'attention et la baisse des capacités cognitives.[7]

Parmi les méthodes proposées[53] :

- Méthode basé sur l'état de conduite du véhicule, Ramesh et al. ont utilisé un capteur pour détecter l'état de mouvement du volant en temps réel pour déterminer le degré de fatigue du conducteur.
- Méthode de vision, Grâce à la vision artificielle, grâce a mesuré la taille et la position de la pupille à l'aide d'une lumière infrarouge de différentes longueurs d'onde. ont extrait la forme géométrique de la bouche à l'aide de la vision.Cette technologie présente l'avantage de fournir des informations visuelles non invasives qui ne sont pas affectées par d'autres éléments externes tels que l'état de conduite du véhicule, les caractéristiques de conduite individuelles et l'environnement routier.
- Méthode du signal EEG, Wang Fei et al. ont utilisé la fusion d'informations pour intégrer les indications physiologiques aux conditions de conduite afin d'identifier la fatigue du conducteur en rassemblant le signal EEG du sujet et les données associées à la manipulation du volant

Compte tenu des risques importants liés à la somnolence, la détection de ce phénomène chez les conducteurs est cruciale pour la sécurité routière. Notre approche se concentre sur l'utilisation d'algorithmes d'apprentissage profond, plus précisément YOLO (You Only Look Once) avec LSTM , pour utiliser les données de reconnaissance faciale afin de détecter les signes de somnolence du conducteur.

En exploitant les avancées du apprentissage profond, notre méthode vise à identifier les indicateurs de somnolence, tels que les yeux fermés ou le bâillement, à partir des images du visage des conducteurs. Notre modèles de réseaux de neurones convolutifs pour extraire les caractéristiques discriminantes et réaliser la détection en temps réel. Cette approche permet d'alerter les conducteurs en cas de signes de somnolence, contribuant ainsi à réduire les risques d'accidents liés à la fatigue au volant.

Les autres parties du memoire sont organisées de la manière suivante :

Le premier chapitre : est divisé en deux sections distinctes :

- La première section se focalise sur la façon dont la somnolence des conducteurs affecte la sécurité routière, elle étudie les facteurs qui contribuent à la somnolence au volant. L'objectif est de faire connaître les problèmes de somnolence chez les conducteurs.
- La deuxième section aborde spécifiquement la reconnaissance faciale utilisant la vision par ordinateur pour détecter la somnolence des conducteurs. Elle étudie les différentes étapes et les principaux conceptions, tels que la détection des visages, l'extraction des caractéristiques faciales. L'objectif est de présenter les fondements de la technique de vision par ordinateur utilisée pour la détection de la somnolence.

Le deuxième chapitre : se concentre sur les concepts fondamentaux de l'apprentissage automatique et de l'apprentissage profond, en mettant les principes des réseaux de neurones artificiels. Les concepts de rétropropagation, l'apprentissage supervisé et non supervisé, ainsi que les architectures couramment utilisées, telles que les réseaux de neurones convolutifs (CNN) et les réseaux de neurones récurrents (RNN), réseaux de mémoire à long terme et à court terme (LSTM), donnent quelques exemples.Architectures

populaires de CNN.

Le troisième chapitre : Est consacré à la conception du système que nous proposons pour la détection de la somnolence et qui correspond aux architectures générales et détaillées.

Le quatrième chapitre : Nous aborderons l'implémentation de l'architecture et les outils utilisés dans ce projet et comment mettre en œuvre le système d'apprentissage en profond. L'évaluation empirique et les résultats sont également présentés dans ce chapitre.

Chapitre 1

Techniques de détection de la somnolence chez les conducteurs

1 Introduction

La somnolence au volant est un problème grave qui peut arriver à tout conducteur, quel que soit son âge ou son expérience. Lorsque nous sommes fatigués et que nous conduisons, notre vigilance diminue, nos temps de réaction s'allongent et notre jugement se dégrade. Cela peut entraîner des accidents dangereux, voire mortels, sur la route.

La somnolence au volant est désormais reconnue comme un problème de santé publique par les milieux spécialisés, au vu des données d'accidentologie. On estime que l'endormissement au volant est à l'origine de 20% des accidents de la route.Cela peut être dû à des facteurs tels que le manque de sommeil, des maladies qui rendent les gens somnolents pendant la journée, ou des médicaments qui affectent le système nerveux central.[36]

Dans ce chapitre, nous aborderons problème de la somnolence lors de la conduite, ainsi que les techniques de détection de la somnolence chez les conducteurs. Nous étudierons également l'utilisation des techniques de détection de la somnolence. L'objectif de ce travail est de sensibiliser les conducteurs aux dangers de la somnolence au volant et de les aider à prendre des mesures pour éviter les accidents liés à cet état.

2 Problème de la somnolence lors de la conduite

2.1 Relation entre la somnolence et la conduite

La somnolence est une perte de conscience qui se manifeste par une somnolence et des difficultés à rester éveillé, mais la personne peut être réveillée par des stimuli faciles. Elle peut être due à un manque de sommeil, à la drogue, à l'alcool ou à un problème nerveux. La plupart des gens se sentent somnolents parce qu'ils sont fatigués, ce qui peut être à la fois mental et physique. D'autre part, si vous faites quelque chose pendant longtemps, comme conduire, il sera difficile de maintenir un niveau de performance élevé. Aussi, pour arrêter ou réduire le nombre d'accidents, le niveau de sommeil d'un conducteur

doit toujours être vérifié.

2.2 Les facteurs influençant la somnolence

Plusieurs facteurs contribuent à la somnolence au volant, parmi se facteur on cite :

— Être éveillé pendant plus de 17 heures ;

— Le manque de sommeil : un manque de sommeil accumulé ;

— Conduire entre 2 et 5 heures du matin et entre 13 et 15 heures, lorsqu'il est important de favoriser la somnolence ;

— Présence de troubles du sommeil non traités, tels que l'apnée du sommeil.

— Conduite au-dessus de la limite de vitesse. Les vitesses élevées exigent un traitement rapide des informations et une adaptation visuelle plus fréquente. Ce stress accru entraîne une plus grande fatigue et donc une baisse de la vigilance.

3 La reconnaissance faciale

3.1 Introduction aux expressions faciales

L'expression faciale est un aspect important de la communication et du comportement non verbaux, qui est un changement visuellement perceptible du visage par l'activation (volontaire) ou moins d'un ou de plusieurs des 44 muscles faciaux (250 000 expressions possibles) de l'expression faciale, déjà étudiée par Darwin et Duchenne de Boulogne au 18ème siècle,a joué un rôle important dans l'étude de l'émotion depuis les travaux de Sylvan TomkinsSixties Ses élèves Paul Ekman et Caroll Izard soutiennent l'idée de nombres limités émotions de base liées aux expressions faciales automatiques et universelles et congénital.

Dans la seconde moitié du dix-huitième siècle, le neurologue Duchenne de Boulogne mène une série d'expériences sur l'expression faciale. Utilisez la photographie et stimulation électrique des muscles faciaux pour accentuer les mouvements associés pour exprimer des émotions. En particulier, notez que le sourire exprime un véritable bonheur diffèrent du sourire volontaire par la contraction localisée du muscle orbicularis oculiautour des

yeux .[2]

La figure ci-dessous 1.1 montre les moscles faciaux.

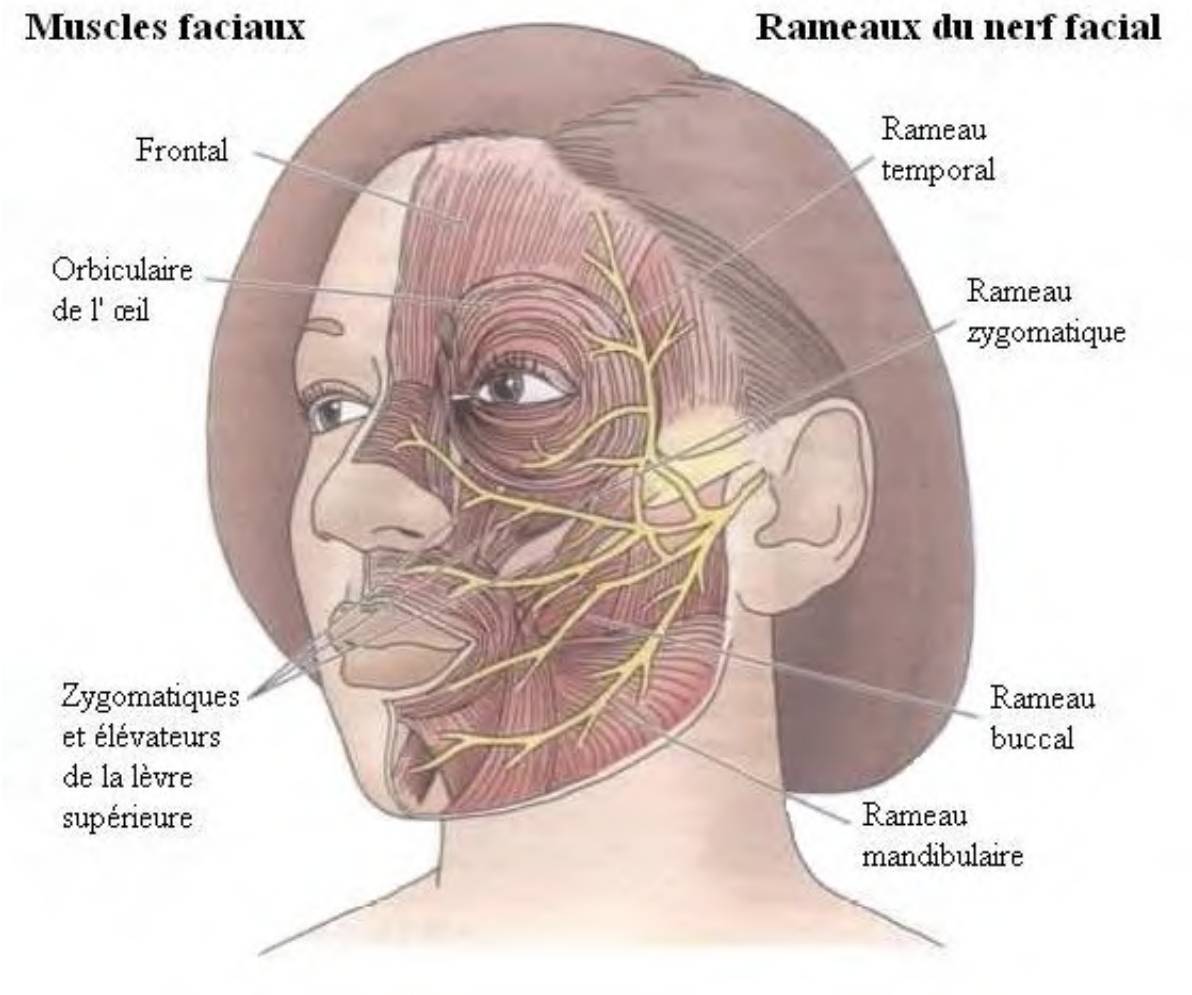


FIGURE 1.1 – Muscles faciaux et leur contrôle nerveux.[2]

3.2 Définition de visage

Le visage peut être défini comme : Une structure tridimensionnelle avec une configuration "externe" - le contour du visage modelé par les proéminences osseuses et souligné par les cheveux - dans laquelle s'inscrit la configuration "interne" crée par un ensemble de traits.Il existe également des traits particuliers tels que les cheveux, les lunettes, la couleur de la peau, etc.

3.3 Les étapes de reconnaissance faciales

L'étape cruciale du système est l'extraction des caractéristiques faciales, elle consiste à identifier les points d'intérêt qui décrivent les expressions faciales Cependant, il y a trois paramètres à considérer comme hypothèses nécessaires pour notre application [2] :

- Le visage du sujet doit être directement devant la caméra ;
- Le sujet commence toujours avec une expression faciale neutre lors de la phase d'initialisation ;
- Les conditions d'éclairage doivent être stables.

Toute modification de l'un de ces paramètres peut avoir une influence sur les résultats de notre système.

3.3.1 Détection de visage

Dans cette étape, l'objectif est d'extraire le visage à partir de l'image. La détection de visage peut être réalisée en utilisant des méthodes basées sur la détection du visage de la couleur de la peau, la forme de la tête ou en détectant les différentes caractéristiques du visage. Cependant, cette étape devient plus complexe l'image contient plusieurs objets de visage ou un fond non uniforme qui peut perturbant le segmentation précise du visage. La réussite de cette étape dépend de la qualité des images acquises.[21] :

- **Méthodes basées sur les connaissances :** Ces méthodes basées sur des règles tentent de représenter les connaissances sur les caractéristiques d'un visage. Par exemple, dans une image, un visage est souvent caractérisé par la présence de deux yeux symétriques, un nez et une bouche. Ces règles décrivent les relations entre les différentes caractéristiques faciales.
- **Méthodes basées sur les caractéristiques invariantes :** Ces méthodes utilisent des caractéristiques structurelles telles que les traits faciaux, la texture et la couleur de peau, qui restent présentes même lorsque la pose, le point de vue ou les conditions d'éclairage varient.
- **Méthodes basées sur l'apprentissage :** Les modèles sont entrainés à partir

appris ici à partir d'une série d'images d'apprentissage qui permettent de capturer la diversité de l'apparence d'un visage. Ces méthodes utilisent des techniques telles que l'apprentissage automatique et l'analyse statistique pour déterminer les caractéristiques appropriées des images de visage et de non-visage.

3.3.2 Extraction des caractéristiques du visage

Nous pouvons le faire de deux manières diverses : La première consiste à extraire des régions entières du Visage, souvent réalisée avec une approche de reconnaissance universelle. La deuxième méthode dite de reconnaissance locale, extrait quelques points de différence des zones caractéristiques du visage, telles que la bouche et le nez.[39]

4 Techniques de détection de la somnolence des conducteurs

Trouver des approches pour détecter la somnolence au volant est crucial pour améliorer la sécurité routière. Les scientifiques utilisent trois approches différentes pour détecter la somnolence : les techniques de traitement d'images, les techniques de réseaux de neurones artificiels et les techniques d'EEG (électroencéphalographie). La Figure 1.2 présente les différentes méthodes utilisées dans ces approches.

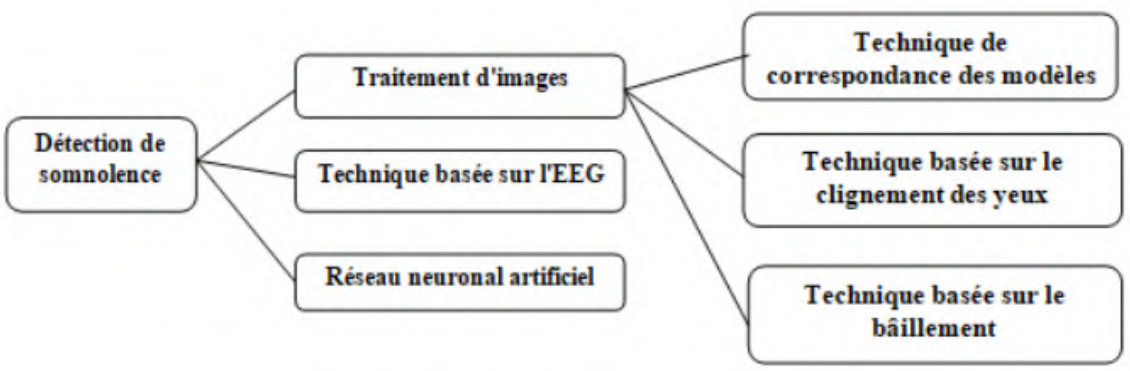


FIGURE 1.2 – Les différentes méthodes de détection de somnolence.

4.1 Techniques basées sur le traitement d'images

Des symptômes utiles peuvent être observés dans les systèmes de surveillance du visage du conducteur, comme la fatigue et la distraction. Ces symptômes se répartissent en trois catégories :

4.1.1 Techniques de correspondance des modèles

La technique a été construite à l'aide de la technologie Augment, ce qui donne au système plusieurs modules qui fonctionnent simultanément. Parce que l'œil est la zone la plus importante du visage où les signes de fatigue apparaissent, on peut alors utiliser ces états, c'est à dire que si le conducteur ferme les yeux pendant un certain temps, le système déclenche une alarme. Ce système peut également être formé pour créer des modèles ouverts et fermés de l'œil du conducteur.[33]

4.1.2 Techniques basée sur le clignement des yeux

Cette technique est basée sur les signes de somnolence et les symptômes comportementaux du micro-sommeil visibles sur le visage et autour du visage et des yeux. Avec une fermeture des paupières plus longue, les changements dans le schéma des clignements des yeux (moins de clignements spontanés, clignements plus longs et plus visibles) sont mesurés pour détecter la somnolence du conducteur. Lorsque le conducteur se sent somnolent, ses yeux clignotent et son apparence de paupière est différente des situations normales, la somnolence peut alors facilement reconnaissable.[29]

Un système qui utilise un modèle de couleur de peau dans l'espace colorimétrique HSI pour détecter les visages. Utilisation des contours pour la localisation des yeux, et la mise en correspondance dynamique pour le suivi des yeux. L'état des yeux peuvent alors déduire l'état du conducteur. Si les yeux sont fermés pendant cinq frames consécutifs, Le conducteur est supposé être fatigué.[24]

Les systèmes de détection de somnolence du conducteur sont généralement basés sur la détection et l'emplacement des visages et des yeux, ils calculent le rapport d'aspect (Eye Aspect Ratio EAR) en fonction de l'espacement des paupières pour déterminer si

les yeux sont ouverts ou fermés.[13]

L'une des mesures les plus couramment utilisées pour détecter la somnolence est le PERCLOS (Percentage of Eye Closure), qui a été développé pour évaluer les variations visibles dans le mouvement des paupières. Cette mesure repose sur le calcul du pourcentage de fermeture de l'œil au fil du temps et reflète les fermetures lentes des paupières plutôt que les clignements.[51][12] Le PERCLOS peut être obtenu par l'équation ci-dessous :

$$f_{\text{PERCLOS}} = \frac{N_{\text{fermie}}}{N_{\text{ferméetouvert}}} \times 100\% \tag{1.1}$$

— Nferméetouvert : représente le nombre total d'ouvertures et de fermetures des yeux dans une période

— Nfermie : représente le nombre d'images des yeux fermés à un moment donné.

4.1.3 Technique basée sur le bâillement

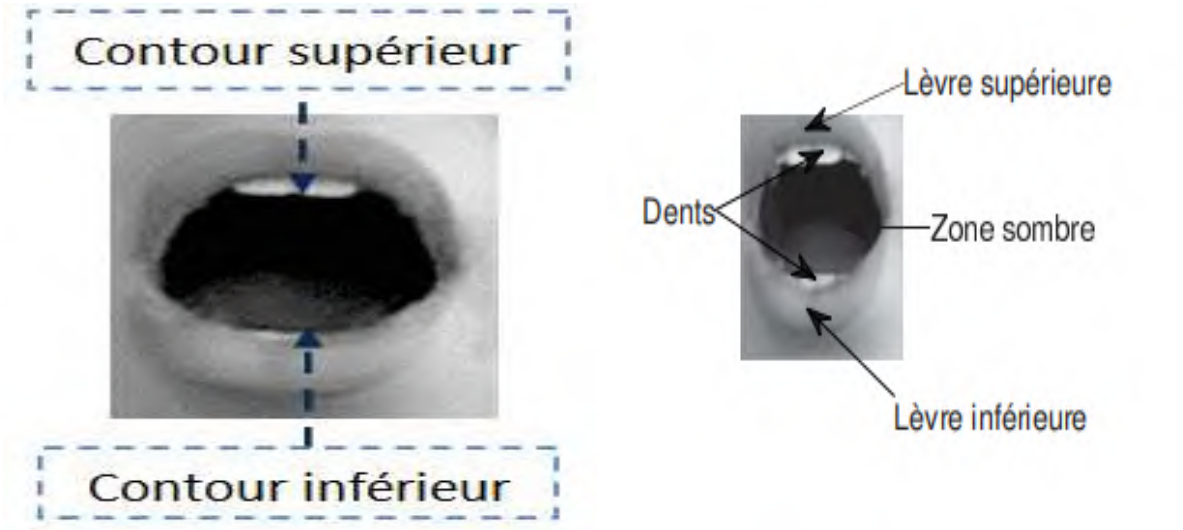
Il s'agit d'une technique non invasive de détection en temps réel de la somnolence du conducteur par le bâillement. Ce système se compose de plusieurs étapes y compris la détection et le suivi du visage, On suppose que le bâillement est modélisé avec une grande ouverture verticale de la bouche. On peut reconnaître les bâillements en fonction du taux d'ouverture de la bouche et des changements de quantité dans la zone du contour de la bouche. Le détecteur de contours que nous proposons est le modèle original et se base sur la morphologie de la bouche lors du bâillement Dans ce cas, la bouche a une grande surface sombre avec une forme pseudo-circulaire, comme le montre la figure 3.20. Cette surface est délimitée par des zones de peau correspondant aux lèvres inférieure et supérieure et éventuellement à des parties de dents. Ainsi, nous pouvons exploiter la grande variation des intensités entre la zone sombre du bâillement et la peau des lèvres (ou les dents) pour construire le détecteur du contour de la grande ouverture de la bouche. Comme pour le détecteur du contour de l'iris.[8] Nous considérons uniquement

les pixels x de l'image de la bouche susceptibles d'appartenir à la grande ouverture de celle-ci. Pour chaque pixel x, nous déterminons le voisinage de n pixels au-dessus et de n pixels en dessous de x, puis calculons la différence entre x et ces n pixels au-dessus et en dessous. Le nombre de voisins n est défini par un cinquième du nombre de colonnes dans l'image de la bouche.[8]

— **Contour supérieur :** Si au moins $(n-1)$ voisins de grande valeur donnent une différence supérieure au seuil th_{sup} et si au moins $(n-1)$ voisins bas produisent une différence inférieure à un seuil th_{inf} , alors on conclut que le pixel x appartient au contour supérieur de l'ouverture de la bouche et réglez-le sur 1.

— **Contour inférieur :** Si nous avons au moins $(n-1)$ voisins hauts qui donnent une différence inférieure à th_{inf} et d'au moins $(n-1)$ voisins bas qui donnent une différence supérieure à th_{sup} , nous concluons que le pixel x appartient au contour inférieure de l'ouverture de la bouche et le fixons le 1.

La figure 1.3 illustre les contours supérieur et inférieur de la bouche pendant le bâillement.



Contours supérieur et inférieur Structure de bâillement

FIGURE 1.3 – Illustration des contours supérieurs et inférieurs et la structure du bâillement.

4.2 Technique basée sur le signal EEG

C'est une technique qui servent à mesurer l'activité électrique du cerveau par des électrodes placées sur le cuir chevelu, ces électrodes seront capturés a travers d'un casque que le conducteur doit porter pendant la conduite. Cette méthode est basée sur l'analyse du spectre de puissance et l'algorithme FastICA.[25]

4.3 Techniques basées sur les réseaux de neurones artificiels

Dans cette technologie, les neurones sont utilisés pour détecter la somnolence du conducteur. Plusieurs chercheurs effectuent des recherches dans le domaine de l'optimisation de la détection de la somnolence des conducteurs à l'aide d'un réseau neuronal fictif. Les personnes qui sont fatiguées présentent divers comportements visuels qui sont facilement discernables en raison de changements dans les caractéristiques faciales telles que :les yeux, la tête et le visage. La technique proposée est basée [17] sur la création d'un réseau neuronal artificiel pour détecter le sommeil en utilisant ces signaux visuels.

5 Conclusion

En conclusion, la somnolence au volant est un problème majeur qui peut entraîner des accidents graves sur la route. Les conducteurs peuvent présenter divers symptômes de somnolence, tels que des yeux lourds, des bâillements fréquents, une difficulté à se concentrer, etc. Il est important de prendre des mesures pour détecter la somnolence des conducteurs en temps réel afin de prévenir les accidents.

L'utilisation de la technologie pour détecter la somnolence des conducteurs est en constante évolution. Les techniques d'analyse des expressions faciales, en particulier celles basées sur les réseaux de neurones convolutifs (CNN), sont de plus en plus utilisées pour détecter la somnolence des conducteurs en temps réel.

Cependant, il est important de noter que ces techniques doivent encore être améliorées pour améliorer leur précision et leur fiabilité. En outre, il est important de sensibiliser les conducteurs aux risques de conduire en état de somnolence et de les encourager

à prendre des mesures préventives, telles que faire des pauses régulières et se reposer suffisamment avant de prendre la route.

En fin de compte, la sécurité routière est l'affaire de tous, et la prévention de la somnolence au volant est un aspect essentiel de cette sécurité.

Chapitre 2

Apprentissage profond : outils théoriques pour traité la somnolence

1 Introduction

La détection de la somnolence chez les conducteurs et un sujet de préoccupation majeur dans le domaine de la sécurité routière. Les accidents causés par les conducteurs fatigués ou somnolents peuvent avoir des conséquences dramatiques. Cependant, il est parfois difficile de détecter les signes de somnolence chez les conducteurs, et cela peut se révéler très dangereux. c'est là qu'intervient l'intelligence artificielle. des chercheurs se sont penchés sur l'utilisation de l'IA pour détecter la somnolence chez les conducteurs.

L'apprentissage automatique et l'apprentissage profond sont considérés comme des sous-domaines de l'IA.récemment suscité beaucoup d'intérêt.L'apprentissage automatique permet aux ordinateurs de développer des modèles d'apprentissage par eux-mêmes, sans aucune programmation, à partir de gros ensembles de données.La couche immédiatement inférieure est occupée par l'apprentissage profond, est l'une des nombreuses approches du l'apprentissage automatique qui a connu un grand succès dans ces dernières années. Ces approches d'IA ont démontré un potentiel important dans plusieurs domaines.

Ce chapitre donne un aperçu des méthodes d'apprentissage profond que nous utiliserons pour développer notre travail.Nous nous concentrerons principalement sur l'architecture des réseaux de neurones convolutifs (CNN), en abordant d'abord l'apprentissage automatique et les différents types d'algorithmes qu'il utilise. Ensuite, nous aborderons les algorithmes d'apprentissage en profond. Nous fournissons également un aperçu des architectures CNN les plus courantes.

2 Exploration de l'apprentissage automatique

2.1 Les phases de l'apprentissage automatique

De manière générale, les algorithmes d'apprentissage automatique se séparent en plusieurs phases, qui sont les suivantes :

1. Collecte et préparation des données : il s'agit de collecter les données nécessaires

pour entraîner le modèle d'apprentissage automatique et le préparer pour l'analyse.

2. Sélection de modèles d'apprentissage automatique : Il existe différents types de modèles d'apprentissage automatique, chacun présentant des avantages et des inconvénients en fonction du type de données et de la tâche.
3. Entraînement du modèle : le modèle est entraîné sur les données d'apprentissage pour voir les motifs et les relations entre les variables.
4. Évaluation du modèle : Une fois le modèle formé, il doit être évalué sur des données de test pour mesurer sa précision et sa capacité à généraliser à de nouvelles données.
5. Utilisation du modèles : Une fois que le modèle a été évalué et validé, il peut être utilisé pour faire des des prévisions ou des prédictions sur de nouvelles données.

2.2 Types d'apprentissage automatique

L'apprentissage automatique est un domaine assez vaste. La figure ci-dessous 2.1 montre les types d'apprentissage automatique.

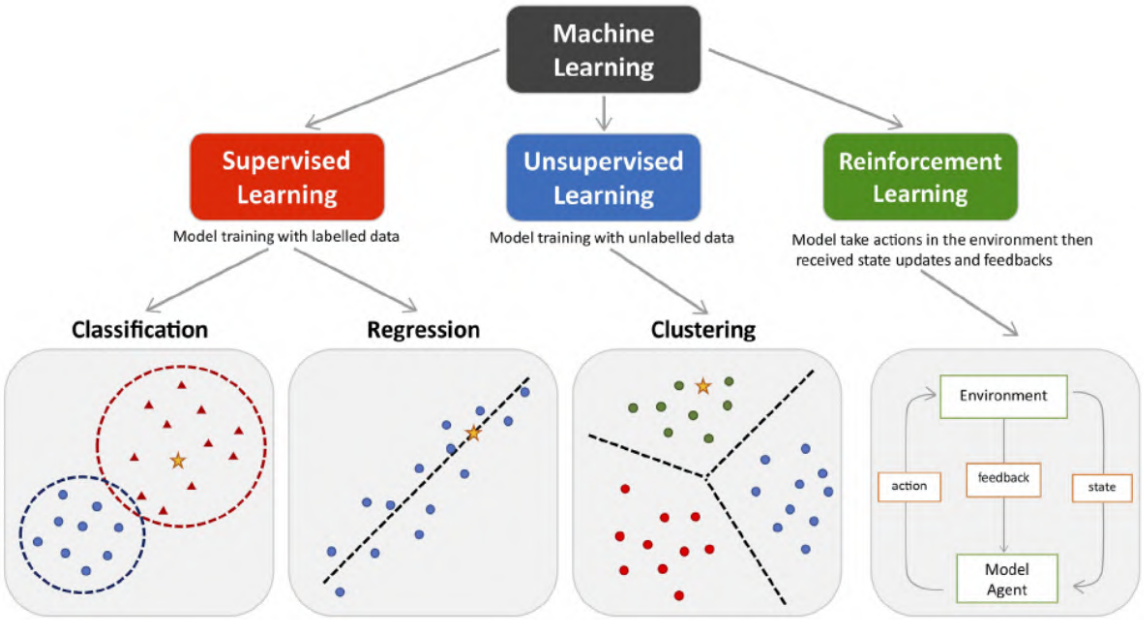


FIGURE 2.1 – Les principaux types d'apprentissage automatique.[35]

2.2.1 Apprentissage supervisé (Supervised learning)

L'apprentissage supervisé est une approche d'apprentissage automatique qui implique par la création d'un algorithme capable d'apprendre à prédire. Cela est possible grâce à un entraînement.

L'apprentissage supervisé se divise généralement en deux catégories principales : la classification et la régression.

Classification

Le processus de création d'un modèle pour aider à diviser les données en groupements de catégories distincts, la simplification des données et la prédiction sont dans la catégorie de categorisation. Ces ensembles de données peuvent être résumés en utilisant la méthode de classification, qui aide également à identifier les relations clés et la structure de l'ensemble de données. Si des classes distinctes de choses sont déterminées à exister, elles peuvent être identifiées, leurs qualités peuvent être énumérées, et l'information peut être structurée et récupérée plus efficacement. Cela facilite également l'attribution de nouveaux objets au système.[19]

Regression

La régression est une forme de technique de modélisation prédictive dans laquelle nous essayons de trouver une relation significative entre une variable dépendante et une ou plusieurs variables indépendantes également appelées variables cibles. Il existe différents types de techniques de régression : linéaire, logistique, polynomiale, Ridge, Lasso et bien d'autres.[3]

La figure ci-dessous 2.2 montre la classification et la régression.

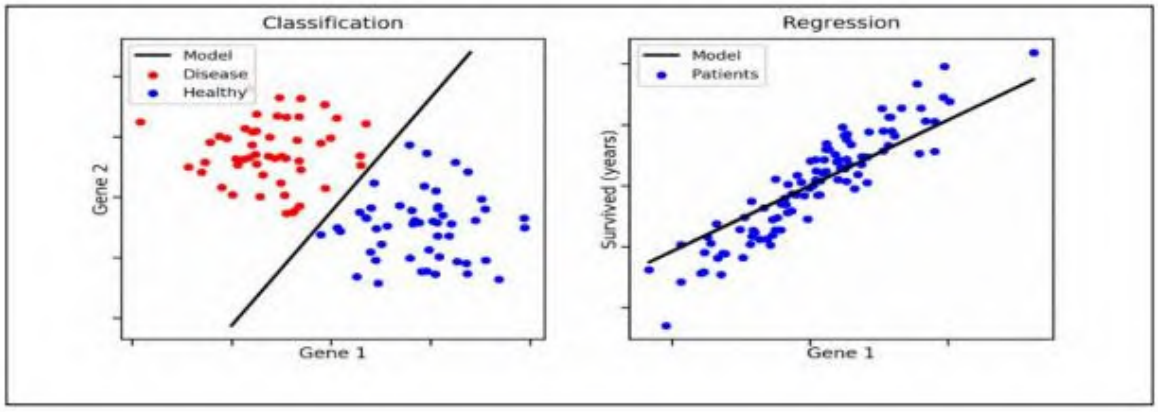


FIGURE 2.2 – La classification et la régression.[50]

2.2.2 Apprentissage non supervisé (Unsupervised learning)

Apprentissage non supervisé lorsque les données ne sont pas étiquetées. On dispose donc de données d'entrée dont on ne connaît pas les sorties associées. L'ensemble de données et le but du système sont d'identifier les caractéristiques communes dans les données d'entraînement.

L'apprentissage non supervisé consiste principalement en des algorithmes de regroupement (clustering).

Clustering

L'analyse de cluster est une technique utilisée dans l'extraction de données et l'apprentissage automatique pour organiser des éléments comparables en clusters. K-means clustering est une approche fréquemment utilisée pour l'analyse de cluster où le but est de diviser un ensemble d'objets en clusters K de telle sorte que le total des distances carrées entre les objets et leur moyenne de cluster attribuée est minimisé[38].

La figure ci-dessous 2.3 montre la création clusters à partir des données.

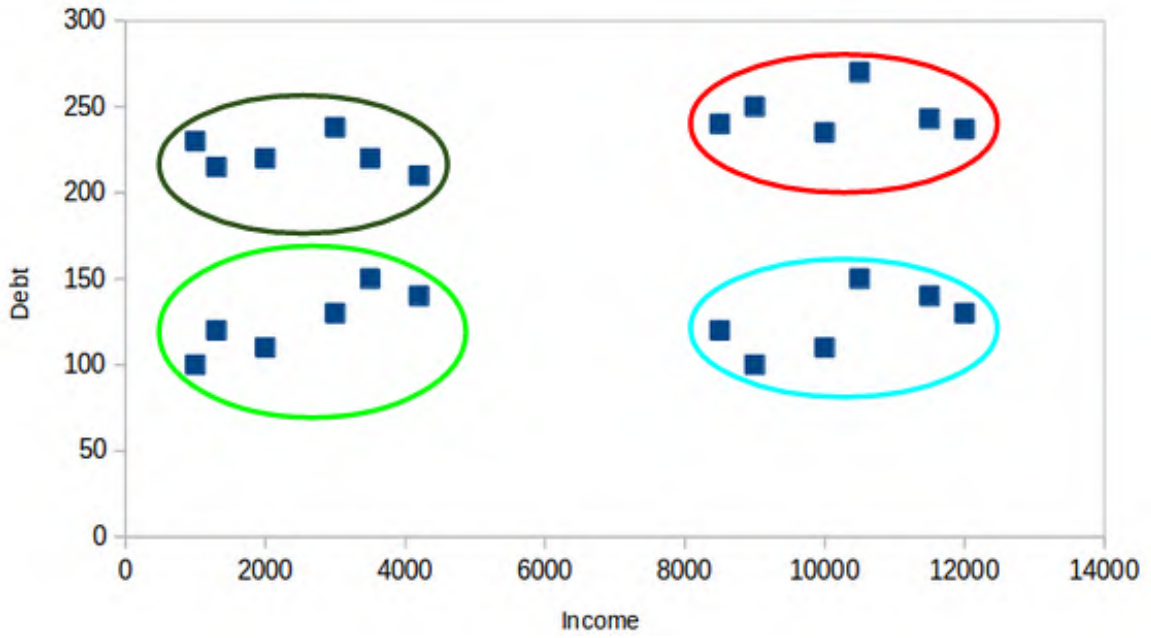


FIGURE 2.3 – créer clusters à partir des données.[38]

Le cluster hiérarchique et le cluster k-means sont deux stratégies courantes dans le domaine de l'apprentissage non supervisé utilisé pour regrouper les points de données en catégories distinctes. Alors que le k-means cluster divise les données en un nombre spécifié de clusters, le clustering hiérarchique développe une structure hiérarchisée semblable à un arbre pour décrire les relations entre les groupes.[38]

2.2.3 Apprentissage par renforcement (Reinforcement learning)

L'apprentissage par renforcement se concentre sur la manière dont un agent autonome interagit avec son environnement pour apprendre à choisir les comportements optimaux en vue d'atteindre ses objectifs. Il utilise les activités réalisées par l'agent dans l'environnement pour évaluer et récompenser son comportement, en prenant en compte les conséquences de ses actions. l'apprentissage par renforcement trouve des applications variées, telles que l'ordinateur jouant aux échec contre un humain, l'apprentissage de la reconnaissance de phrases parlées et l'apprentissage de la catégorisation de nouvelles formations astronomiques.[16]

3 Classification des méthodes d'apprentissage profond

3.1 Réseaux de neurones artificiels

Le réseau neuronal artificiel est dérivé des réseaux neuronaux biologiques qui définissent la structure du cerveau humain. Les réseaux neuronaux artificiels, comme le cerveau humain, ont des neurones en plusieurs couches qui sont reliés les uns aux autres. Ces neurones sont appelés nœuds. Dans ANN, les dendrites provenant des réseaux neuronaux biologiques représentent les entrées, les noyaux cellulaires les nœuds, les synapses les poids et les axons représentent la sortie.

Les ANN sont des modèles statistiques non linéaires qui démontrent une relation complexe entre les entrées et les sorties afin de découvrir un nouveau modèle. Les réseaux neuronaux artificiels sont utilisés pour un large éventail d'applications, notamment la reconnaissance d'image, la réception de la parole, la traduction automatique et le diagnostic médical.

Le fait que ANN apprenne à partir d'échantillons de données est un avantage important. L'application la plus typique d'ANN est pour l'approximation de fonctions aléatoires. Avec ces types de technologies, on peut arriver à des solutions qui spécifient la distribution de manière rentable. ANN peut également offrir un résultat de sortie basé sur un échantillon de données plutôt que l'ensemble de données complet. Les ANN peuvent être utilisés pour améliorer les méthodes d'analyse de données existantes en raison de leurs capacités de prédiction élevées[26]. La figure ci-dessous 2.4 illustre L'architecture des réseaux neuronaux artificiels.

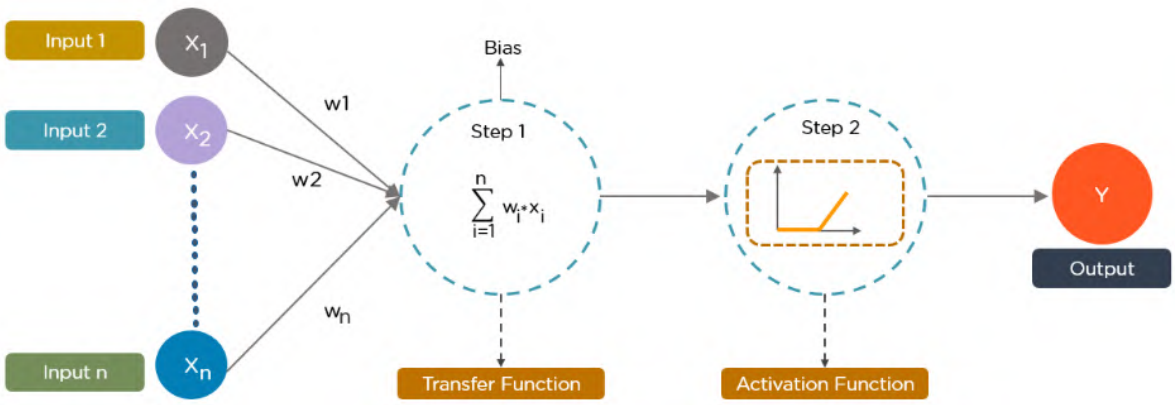


FIGURE 2.4 – Architecture des réseaux neuronaux artificiels.[26]

3.2 Les fonctions d'activation

3.2.1 ReLu

Le ReLU, ou unité d'activation linéaire rectifiée, est l'un des rares marqueurs du changement dans l'apprentissage profond. Il est facile à utiliser, mais il est bien supérieur aux fonctions d'activation telles que sigmoid ou tanh. La formulede ReLu est :

$$f(x) = \ f(x) = max(0, x) \tag{2.1}$$

La fonction ReLU et sa dérivée sont toujours monotones. Si la fonction reçoit une entrée négative, elle renvoie 0 ; cependant, si la fonction obtient une valeur positive x, elle retourne cette valeur. En conséquence, la sortie a une plage de 0 à l'infini.[1] Leaky ReLU est une forme modifiée de la fonction ReLU. Au lieu de définir la valeur de la fonction ReLU comme zéro lorsque x est négatif, elle la définit comme une composante linéaire extrêmement petite Composante linéaire de x.Mathématiquement, on peut l'écrire comme [43]

$$f(x) = \ 0.01x, x < 0 \tag{2.2}$$

$$f(x) = \ x, x >= 0 \tag{2.3}$$

3.2.2 Sigmoid

C'est la fonction d'activation la plus utilisée puisqu'elle est fonction non linéaire. La fonction sigmoïde transforme la valeurs dans la plage de 0 à 1. Il peut être défini comme :

$$f(x) = 1/e^{(-x)} \tag{2.4}$$

La fonction sigmoïde est continuellement différenciable et Une fonction en forme de S. Le dérivé de la La fonction est :

$$f'(x) = 1 - sigmoid(x) \tag{2.5}$$

De plus, la fonction sigmoïde n'est pas symétrique autour de zéro, ce qui implique que les signes des valeurs de sortie de tous les neurones seront les mêmes. Cette situation peut être améliorée par Utilisation de la fonction sigmoïde.[43]

3.2.3 Softmax

La fonction softmax est une partie de la mathématique qui prend un vecteur de nombres réels et les transforme en un vector de probabilités qui s'ajoutent tous à 1. Chacun des valeurs du vecteur original est changé en un nombre entre 0 et 1. La formule de la fonction softmax est présentée ci-dessous :

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{2.6}$$

Comme indiqué ci-dessus, la fonction softmax peut fonctionner avec un vecteur z de longueur K.Pour chaque valeur en z, la fonction softmax applique la fonction exponentielle standard à la valeur. Il la divise ensuite par la somme des exposants de chaque valeur en z.[9]

3.3 Réseaux de neurones récurrents

Les réseaux neuronaux récurrents (RNN) 2.5 sont un type de réseaux de neurones couramment utilisés dans l'apprentissage en profond. (Deep Learning). Les RNN utilisent leurs anciennes sorties comme nouvelles entrées et conviennent au traitement de données séquentielles. [48]. RNN peuvent être déployés et entraînés au fil du temps en utilisant la rétropropagation standard ou en utilisant une variante de rétropropagation appelée rétropropagation dans le temps (BPTT).[40]

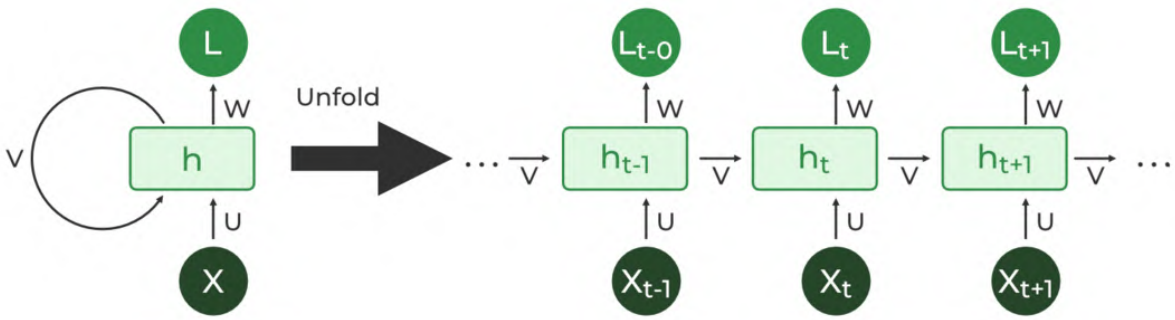


FIGURE 2.5 – Réseaux de neurones récurrents.

3.4 Réseaux de neurones convolutionnels

Les réseaux de neurones convolutifs désignent une sous-catégorie de réseaux de neurones Cependant,Les CNN sont conçus pour traiter les images en entrée.Un réseau neuronal convolutif (CNN) est utilisé dans de nombreux domaines, y compris la reconnaissance, la classification, la détection, la prédiction et d'autres fonctions.La capacité d'apprendre automatiquement un large et un nombre abstrait de caractéristiques lors de la formation parallèle d'un groupe de données dans les limites d'un problème de modélisation prédictive spécifique est le plus important avantage des réseaux neuronaux convolutifs qui ont été introduits[32].

3.4.1 L'architecture réseaux de neurones convolutionnels

Le réseaux de neurones convolutionnel (CNN) est composée de trois types de couches :la couche de **convolution**, la couche de **pooling** et la couche de **full connected**

La couche de convolution :

C'est la partie du CNN qui effectue les calculs pour obtenir des informations utiles à partir des données de l'image. Chaque couche convolutive est composée de noyaux, qui sont des filtres pouvant être appris. Une image colorée peut être considérée comme une grille tridimensionnelle de pixels. Un filtre, quant à lui, est un réseau bidimensionnel de poids qui est déplacé sur les champs réceptifs de l'image pour voir si une caractéristique est présente ou non. Un filtre est généralement appliqué à une image sous la forme d'une matrice 3x3, mais la taille peut changer. Le produit en points des poids des pixels d'origine et des poids du filtre est alors calculé. Le filtre est ensuite déplacé de n pixels, ce qui est également appelé "stride", et le produit en points des pixels qui sont entrés et des poids du filtre est trouvé. Ce processus, également appelé "convolution", se poursuit jusqu'à la fin de la matrice lue. Le résultat de chaque produit de points est une grille appelée carte d'activation ou carte de caractéristiques. Le filtre a été utilisé pour extraire les informations sur les caractéristiques de l'image et les placer sur la carte des caractéristiques[37]. Les figures 2.6 2.7 montront les Représentation par couche convolutive dans les réseaux de neurones convolutionnel (CNN).

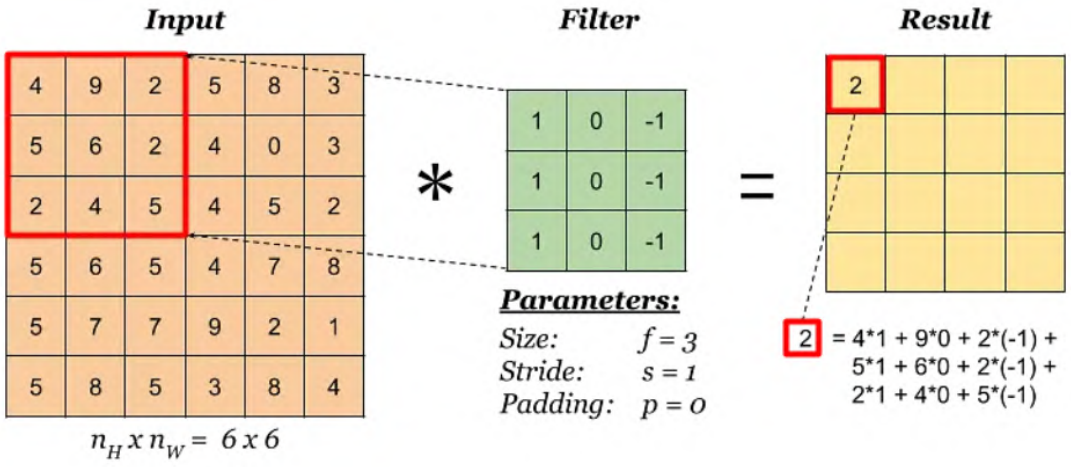


FIGURE 2.6 – A.Représentation par couche convolutive[37]

Comme vous pouvez le voir dans l'image ci-dessus, le noyau ou le filtre a une dimension de 3x3 et un sous-matrice de la même taille que l'image est extrait. Le produit de la matrice est enregistré dans la matrice de sortie. Le filtre avance alors d'un pixel, puisque le pas est de 1 dans l'exemple précédent, et le calcul du produit de points est répété.

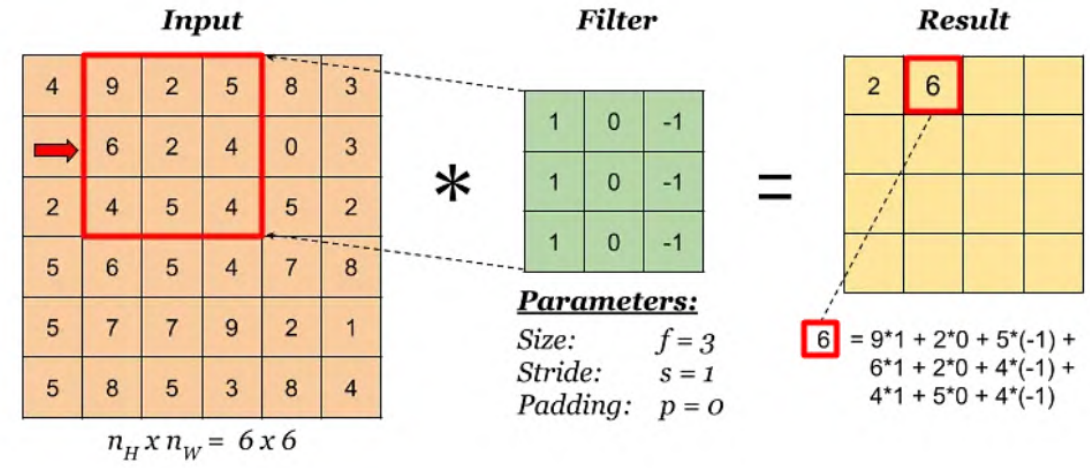


FIGURE 2.7 – B.Représentation par couche convolutive[37]

Il continue à se déplacer autour de la grille de la même manière, créant une carte de caractéristiques. La valeur du pas peut être n'importe quel nombre entier, et un pas plus grand diminue la matrice de sortie.

La couche de pooling

La couche de pooling est souvent placée entre deux couches de convolution. Effectue une opération de pooling, qui est un processus de sous-échantillonnage. Elle consiste réduire la taille des images tout en préservant leurs caractéristiques importantes. Pour ce faire, l'image est découpée en cellules régulières de 2 × 2 pixels qui ne se chevauchent pas, ou en cellules de 3 × 3 pixels qui sont séparées par un pas de 2 pixels . Ensuite, nous appliquons le processus de pooling. Les résultats obtenus sont les mêmes que les entrées, mais plus petits. La couche de pooling réduit le nombre de paramètres et de calculs dans le réseau. Cela permet d'améliorer les performances du réseau et d'éviter le surapprentissage.[34].

Deux types courants de couches de pooling comme montre la figure 2.8, sont pooling

max et pooling average, où l'on prend respectivement la valeur maximale et la valeur moyenne.Le pooling max est plus souvent utilisé que le pooling average. La couche pooling n'a pas de paramètres à apprendre. L'intuition de ce que fait pooling max est que le grand nombre signifie qu'une caractéristique peut être détectée[10].

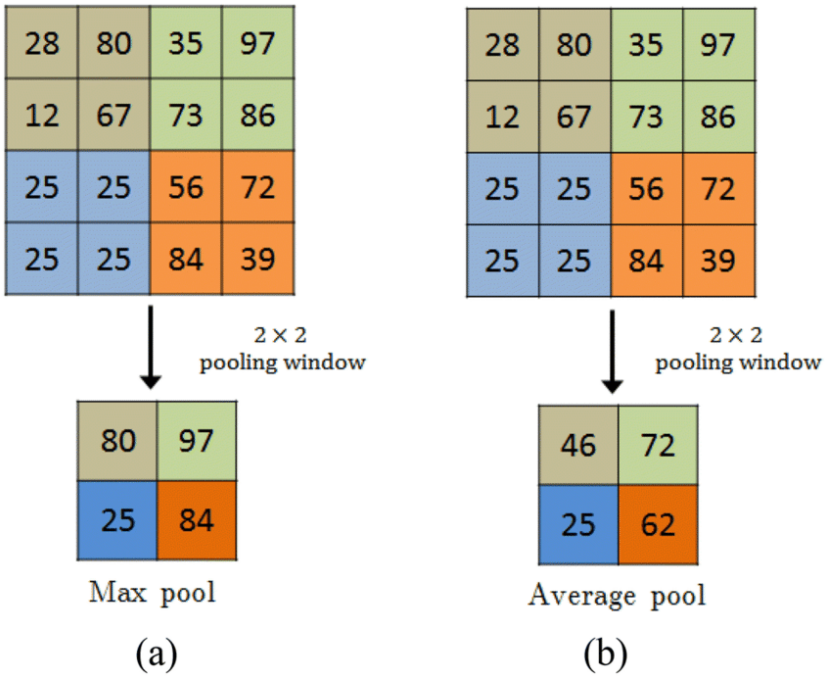


FIGURE 2.8 – Types pool.[6]

La couche de full connected

La couche fully-connected est toujours la dernière couche d'un réseau neuronal, qu'il s'agisse d'un réseau convolutif ou non, et n'est donc pas typique de CNN.

Ce type de couche reçoit un vecteur en entrée et crée un nouveau vecteur en sortie.[34]

3.4.2 Architectures CNN populaires

L'architecture VGGNet

VGGNet est l'architecture CNN développée par Karen Simonyan, Andrew Zisserman et al. à l'Université d'Oxford. VGGNet comme montre la figure ci-dessous 2.9, est un CNN à 16 couches avec jusqu'à 95 millions de paramètres et entraîné sur plus d'un milliard d'images (1000 classes). Il peut prendre de grandes images d'entrée de 224 x 224 pixels

pour lesquelles il dispose de 4096 caractéristiques convolutives.

Le modèle VGG CNN est efficace sur le plan des calculs et sert de base solide pour de nombreuses applications en vision par ordinateur en raison de son applicabilité à de nombreuses tâches, y compris la détection d'objets[28].

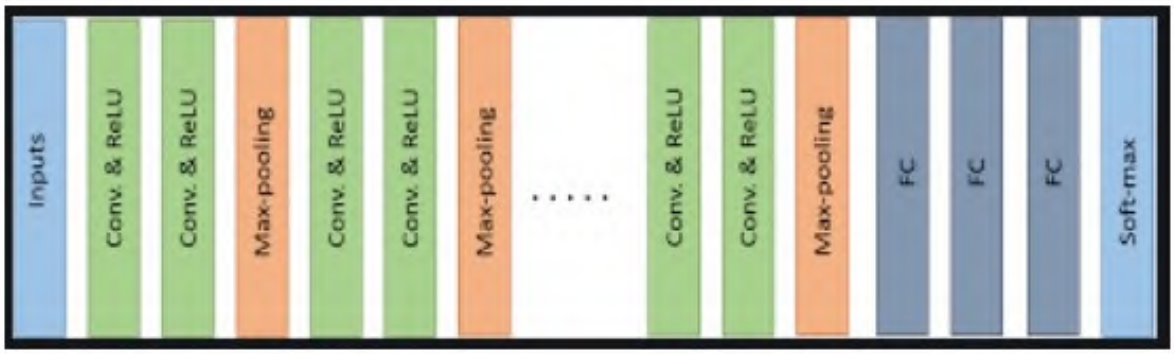


FIGURE 2.9 – L'architecture VGGNet.

L'architecture ResNet

Kaiming He et al. développé un modèle de réseau de neurones convolutifs appelé ResNet, qui se distingue par ses connexions à saut unique et l'utilisation de la normalisation par lots (Batch Normalization). Contrairement à d'autre architectures, ResNet ne comporte pas de couches entièrement connectées la fin du réseau. Cependant, son principal inconvénient réside dans la complexité de son évaluation en raison du grand nombre de paramètres. Jusqu'à présent, ResNet est considéré comme un modèle de réseau CNN de pointe et est souvent le choix privilégié dans l'utilisation de ConvNets dans la pratique[23]. La figure ci-dessous illustre 2.10 l'architecture ResNet.

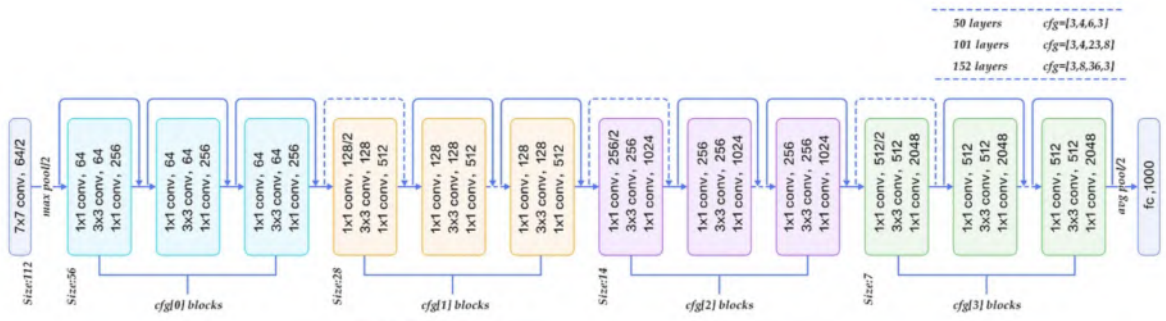


FIGURE 2.10 – L'architecture ResNet.

Inception

Il s'agit une architecture qui exploite les modules Inception. Ce module contient des filtres de tailles différentes (1x1, 3x3 et 5x5) qui permettent capturer des informations spatiales. Ensuit, un filtre Concat qui rend cela possible pour concaténer les résultats du filtre. De plus, afin de réduire la densité de connexion, un pool moyen global dans le dernier couche au lieu d'un couche entièrement connecté.

Ces ajustements de paramètres ont conduit à une réduction significative du nombre de paramètres.[45]

3.5 Réseaux de mémoire à long terme et à court terme (LSTM)

Réseaux à mémoire à long terme et à court terme (LSTM), comme illustré dans la figure 2.11 ci-dessous, est une architecture de réseau neuronal récursif capable d'apprendre les dépendances à long terme. Ce modèle a été développé pour traiter les problèmes de gradient disparaissant et considéré comme une architecture de réseau neuronal profond au fil du temps. Le principal composant de la couche de mémoire à long terme est la cellule de mémoire.[15].

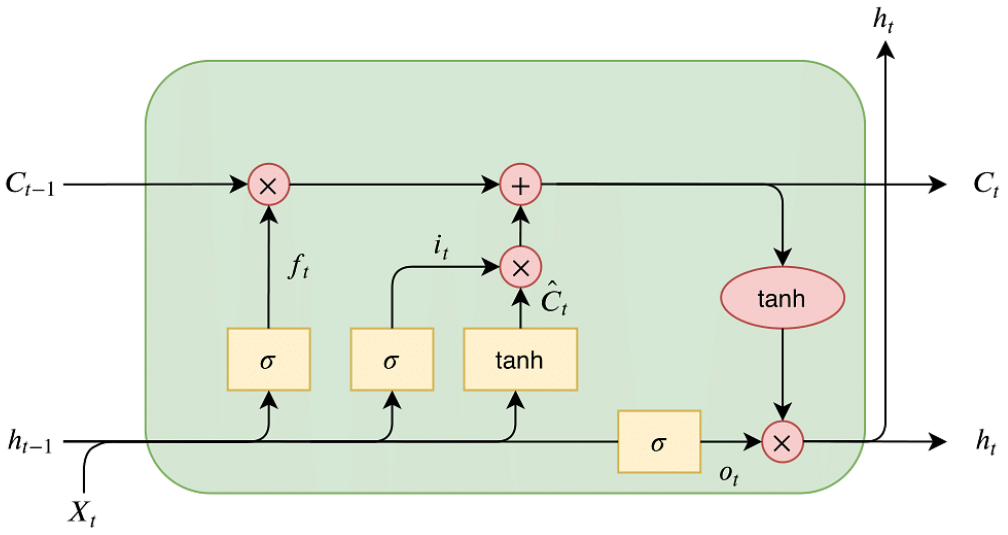


FIGURE 2.11 – Architecture d'une cellule LSTM.[5]

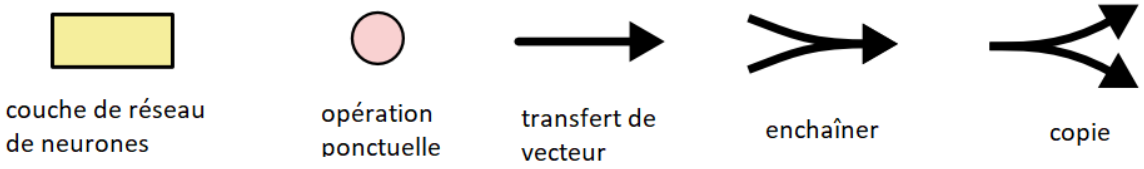


FIGURE 2.12 – Les composants d'une cellule LSTM.[5]

Dans la figure 2.12 ci-dessus, chaque ligne relie la sortie d'un nœud aux entrées d'autres nœuds. Les cercles roses indiquent des processus ponctuels, tels que l'addition de vecteurs, tandis que les boîtes jaunes indiquent les niveaux appris du réseau neuronal. Lorsque deux lignes se rejoignent, elles n'en forment plus qu'une. Lorsqu'une ligne bifurque, ses informations sont copiées et les copies vont à des endroits différents[4].
Une cellule de mémoire se compose de quatre éléments principaux : une porte d'entrée, un neurone avec reconnexion, une porte d'oubli et une porte de sortie. porte de sortie. Les équations suivantes montrent le fonctionnement pas à pas d'une couche de cellules de mémoire pour des séries temporelles d'entrée sous la forme suivante $X = (x1, x2, x3, ..., xn)$, cellules mémoires cachées $H = (h1, h2, h3, ..., hn)$. [15]

$$\begin{aligned} i_t &= \sigma\left(x_t U^i + h_{t-1} W^i\right) \\ f_t &= \sigma\left(x_t U^f + h_{t-1} W^f\right) \\ o_t &= \sigma\left(x_t U^o + h_{t-1} W^o\right) \\ \tilde{C}_t &= \tanh\left(x_t U^g + h_{t-1} W^g\right) \\ C_t &= \sigma\left(f_t * C_{t-1} + i_t * \tilde{C}_t\right) \\ h_t &= \tanh\left(C_t\right) * o_t \end{aligned} \tag{2.7}$$

Le signe dans ce calcul est considéré comme une multiplication par éléments, et en refusant les termes de biais, on peut montrer comment la couche cachée est calculée à un instant ht. Dans les calculs ci-dessus :

— i, f, o sont appelés les portes d'entrée, d'oubli et de sortie, en respectant.

— W^i, W^f, W^o les poids reliant la couche de récurrence à t-1 à la couche cachée à l'instant t.

— U^i, U^f, U^o les poids qui relient la couche cachée au temps t-1 à la couche récursive au temps t.

Dans le modèle présenté dans figure 2.13 ci-dessus, la porte de sortie est la dernière partie d'une cellule LSTM. Cependant, ce n'est qu'une partie de l'ensemble du processus. Il y a d'autres éléments à prendre en compte avant que le réseau LSTM puisse produire les prédictions souhaitées.[5]

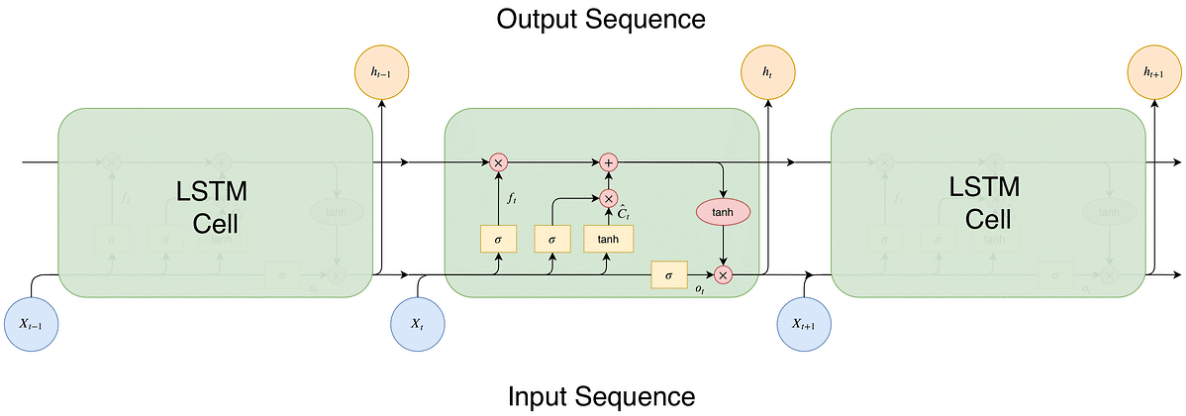


FIGURE 2.13 – Le modèle LSTM.[5]

3.5.1 LSTM et RNN

Bien que les RNN se sont avérés efficaces pour prédire des séquences, il peut être difficile d'apprendre les dépendances à long terme. Cela est principalement dû au problème de l'explosion et de la disparition du gradient, qui se produit lorsque le gradient se propage à travers de nombreuses couches d'un réseau récurrent[52].

Dans la figure 2.14 présentée ci-dessous montre les architectures LSTM et RNN.

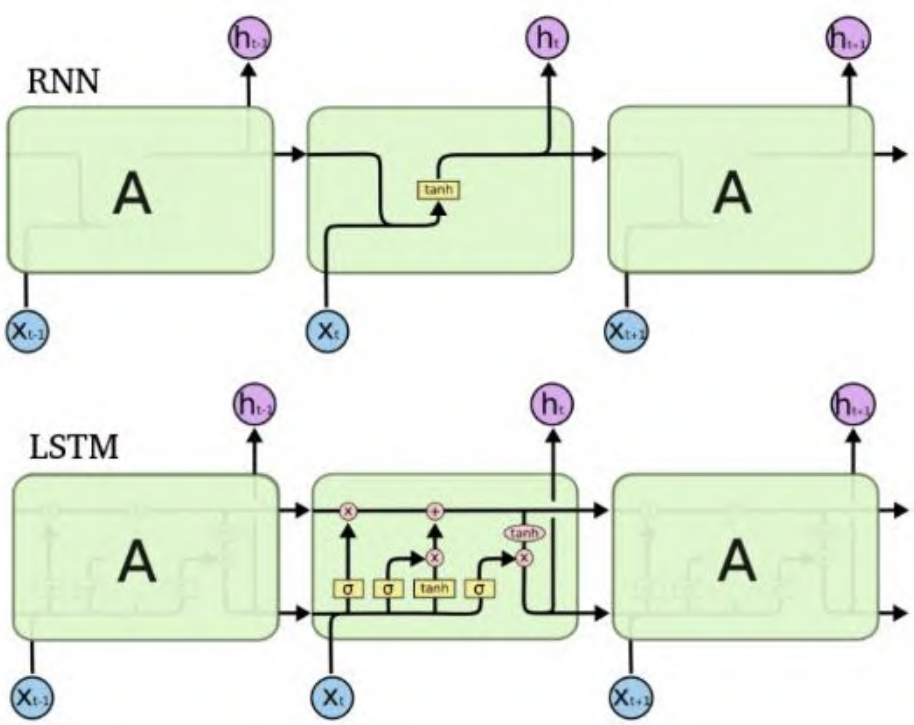


FIGURE 2.14 – LSTM et RNN.[18]

4 Travaux connexes à la détection de la somnolence

Il existe plusieurs travaux de recherche et du développement liés à la détection de la somnolence chez les conducteurs. Voisi quelques exemples d'emplois dans ce domaine :

— **Vitalile et al** [30]. ont développé un système de détection des signes de somnolences chez les conducteurs basé sur une caméra infrarouge. Ils ont mis en place un algorithme qui exploite le phénomène des pupilles brillantes. Cet algorithme permet de détecter et de suivre les yeux du conducteur. Lorsque la somnolence est détectée, le système émet un avertissement sous forme d'un message d'alarme. Le système a été évalué sur des images frontales de neuf sujets présentant des structures faciales présentant plusieurs rapports de fermeture des yeux et de regard dans des conditions d'obscurité totale et d'éclairage ambiant. Les résultats préliminaires ont montré des résultats prometteurs avec une précision suffisante pour entre des yeux complètement fermés, à moitié fermés et complètement ouverts.

— **Mallah et al** [31]. ont développé un système résistant à la lumière. Ils ont utilisé l'algorithme Haar pour la détection d'objets et le classificateur de visage intégé aux bibliothèques OpenCV. Les régions oculaires sont dérivées à partir de la région faciale en utilisant les facteurs anthropométriques. Ensuite, ils détectent les paupières pour mesurer le degré de fermeture des yeux.

— **Tian et Qin** [47],ont développé un système visant à détecter l'état des yeux du conducteur, qui est un facteur clé pour identifier l'attention visuelle du conducteur. Ce système propose en temps réel pour détecter l'état des yeux du conducteur. Le système combine plusieurs techniques de traitement d'image de base qui permettent d'obtenir de meilleures performance que d'autres techniques arithmétiques. Et ils ont différentes méthodes pour détecter l'état des yeux le jour et la nuit. Tout d'abord, le système combine la correspondance de la couleur de la peau et la projection verticale dans la journée et applique l'intensité de la lumière et la projection verticale la nuit pour détecter et suivre la région du visage, puis la valide par la symétrie géométrique du visage ; deuxièmement, ils ont appliqué la projection horizontale pour détecter la position des yeux et la valider par certaines caractéristiques faciales. Enfin, une nouvelle fonction de complexité est utilisée pour juger de l'état des yeux, et la fréquence de fermeture des yeux est enregistrée.

Techniques	Caractéristiques	Avantages	Limites
Vitalile et al [30]	-Clignement des yeux comme indicateur de somnolence. -S'appuie sur des techniques de traitement d'image.	-Utilise des mesures directes des mouvements oculaires pour évaluer la somnolence.	-La détection basée uniquement sur les mouvements oculaires ne pas sur tous les signes de somnolence. -Nécessite des équipements de suivi oculaire appropriés
Mallah et al [31]	-Exploite la reconnaissance faciale pour détecter la somnolence. -Utilise des algorithmes d'analyse d'images pour extraire et évaluer les caractéristiques faciales associées à la fatigue.	-Les caractéristiques faciales utilisées, telles que la fermeture des yeux et les mouvements lents, sont souvent des signes visibles de la somnolence.	-La détection basée sur les caractéristiques faciales peut être sensible aux variations individuelles, -Nécessite des techniques de reconnaissance faciale, ce qui peut être complexe et nécessiter des ressources de calcul importantes.
Tian et Qin [47]	-Utilise des techniques de traitement d'image pour suivre les mouvements oculaires.	-Cette méthode applique différentes méthodes arithmétiques pour détecter l'état des yeux du conducteur, de jour comme de nuit	-Les résultats ne sont pas bons lorsque la tête du conducteur est inclinée ou qu'il y a une grande partie d'autre chose avec une couleur de peau.

TABLE 2.1 – Travaux connexes à la détection de la somnolence

5 Conclusion

Dans ce chapitre, nous avons exploré les bases fondamentales des méthodologies d'apprentissage automatique ainsi que les bases de l'apprentissage profond. Nous avons ensuite plongé plus dans les réseaux de neurones récurrents (RNN), le réseaux neuronal convolutionnels (CNN) et réseaux de mémoire à long terme et à court terme (LSTM), en analysant leurs différentes couches et architectures. Nous avons également examiné de près quelques exemples bien connus de CNN, tels que ResNet, VGGNet et l'inception. En conclusion, nous avons dressé un aperçu complet les avancées les plus récentes dans le domaine. Le chapitre suivantse focalisera sur notre système de détection de la somnolence, en détaillant ses composants essentiels et en évaluant ses performances de manière approfondie.

Chapitre 3

Conception de l'architecture pour la détection de la somnolence

1 Introduction

Dans ce chapitre, nous allons discuter de notre architecture de réseau de neurones convolutifs et LSTM pour la détection de la somnolence. Tout d'abord, nous commencerons par présenter l'architecture générale de notre modèle basé sur les réseaux de neurones convolutifs et LSTM. Ensuite, nous allons explorer plus en détail notre système en décrivant son architecture détaillée.

2 Objectif

L'objectif de la détection de la somnolence est d'identifier les signes de fatigue chez les conducteurs afin de prévenir les risques d'accidents de la route. Cette détection permet de surveiller l'état d'éveil et de vigilance des conducteurs, en particulier lorsqu'ils sont au volant pendant de longues périodes ou dans des conditions propices à la somnolence.

L'identification précoce de la somnolence permet de prendre des mesures appropriées, telles que des pauses régulières, des alertes sonores ou visuelles, ou même une recommandation de repos, pour éviter les accidents causés par la fatigue au volant. L'objectif ultime est d'améliorer la sécurité routière en réduisant les risques liés à la somnolence des conducteurs.

3 L'architecture générale

Notre système suivra certaines étapes afin de construire un modèle d'apprentissage profond pour la détection de la somnolence. Le système commence par collecter l'ensemble de données, appliquer un prétraitement et en le divisant. Ensuite, nous introduisons l'ensemble de données divisé dans le modèle YOLO, ce qui permet d'obtenir un modèle précis. Notre modèle peut alors détecter les differentes classes de dataset. Par la suite, nous procédons à l'entraînement de notre modèle LSTM afin de déterminer le niveau de fatigue du conducteur. En combinant ces deux modèles YOLO et LSTM pour détecter

la somnolence, comme le montre la figure suivante 3.1.

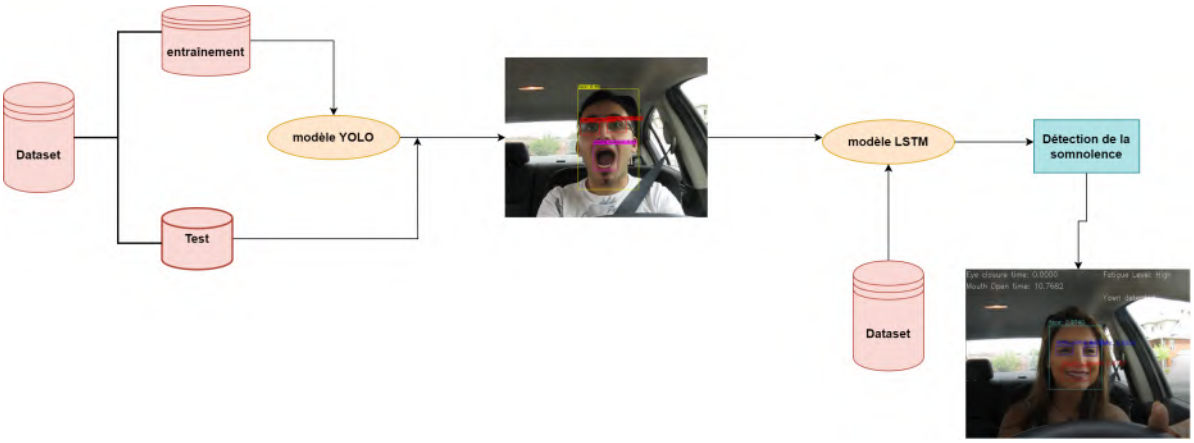


FIGURE 3.1 – L'architecture générale.

4 L'architecture détaillée

Notre système de détection de **la somnolence** repose sur l'utilisation de l'apprentissage profond et suit un processus en plusieurs étapes pour construire notre modèle. En premier lieu, nous extrayons les données à partir de **vidéos** et les soumettons à un **prétraitement** pour les préparer de manière appropriée. Par la suite, ces données prétraitées sont fournies en entrée au modèle **YOLO**. Grâce à cela, le modèle est en mesure de **détecter l'état de visage, des yeux et de la bouche du conducteur**, qu'ils soient ouverts ou fermés. Par la suite, nous procédons à l'entraînement de notre **modèle LSTM** afin de déterminer le niveau de fatigue du conducteur (faible, moyen ou élevé). En combinant ces deux modèles YOLO et LSTM pour **détecter la somnolence**, comme présenté dans la figure 3.2 ci-dessous.

Cette approche nous permet de bénéficier des avantages des deux modèles. Le modèle YOLO est employé pour détecter l'état des yeux et de la bouche du conducteur, tandis que le modèle LSTM est entraîné à évaluer le niveau de fatigue du conducteur.

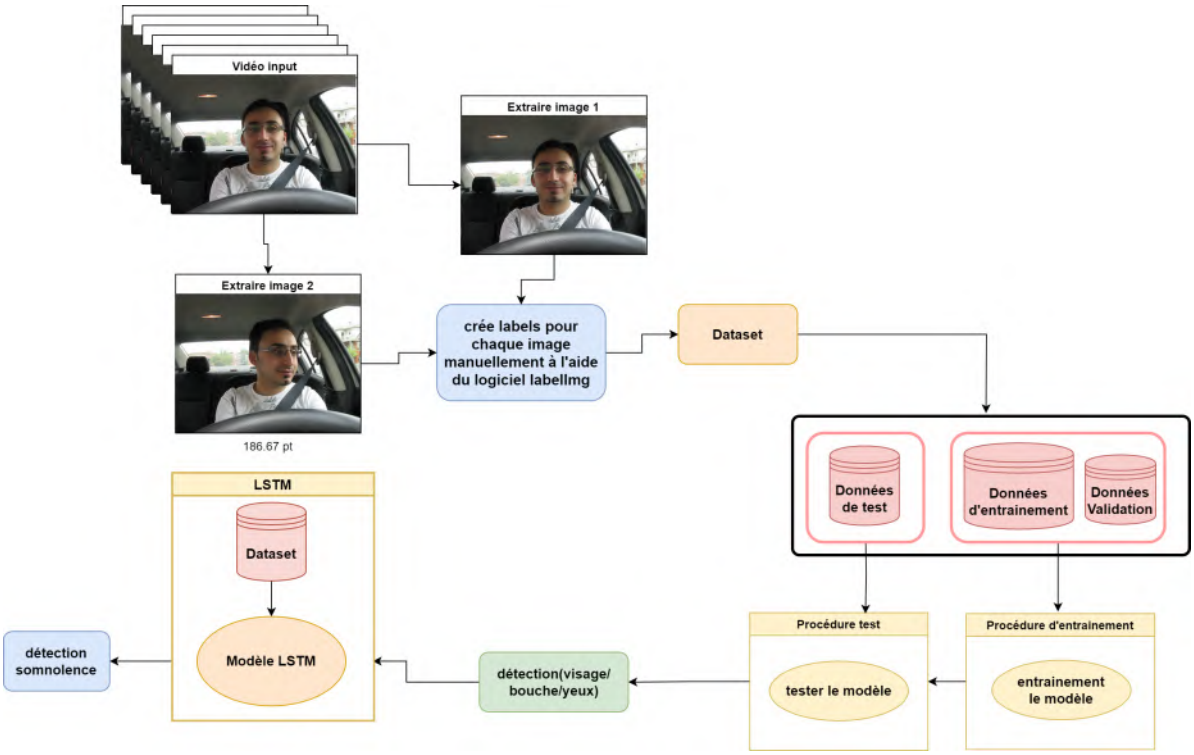


FIGURE 3.2 – L'architecture détaillée.

4.1 Préparation des données

La première étape essentielle avant d'utiliser les données dans les modèles d'apprentissage profond est la préparation des données.

4.1.1 Dataset pour YOLO

Nous allons utiliser des vidéos pour extraire des images et pour cela nous avons téléchargé des vidéos sur la somnolence.

Creation labels

Après avoir extrait les images et pour créer des labels pour chaque image, on a utilisé le logiciel labelImg, comme montre la figure 3.3.

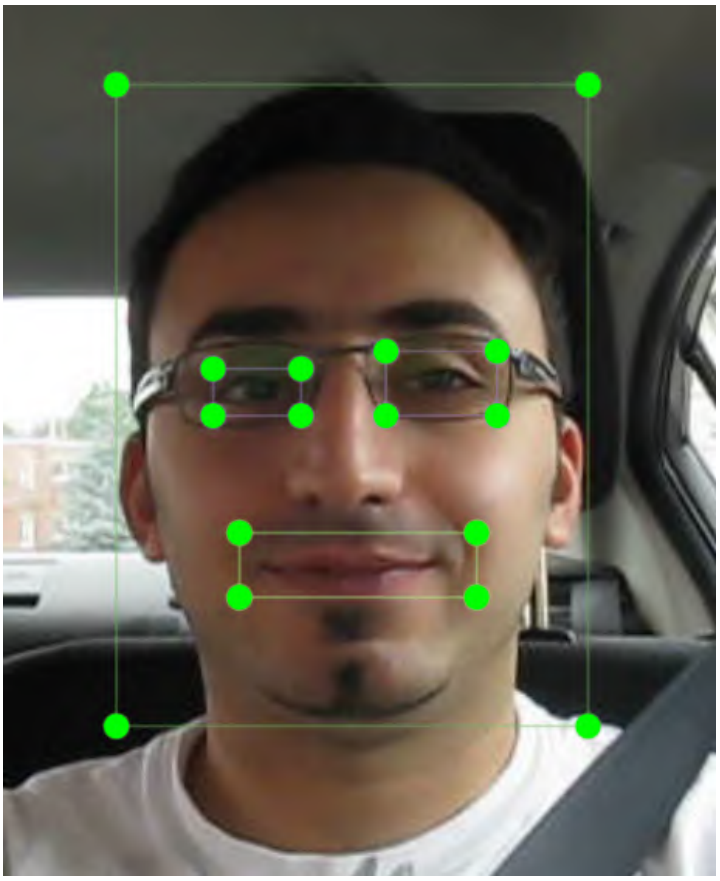


FIGURE 3.3 – Création label.

Les classes dataset

Chaque label a ses propres caractéristiques faciales et ses yeux (fermés ou ouverts) et sa bouche (fermée ou ouverte).

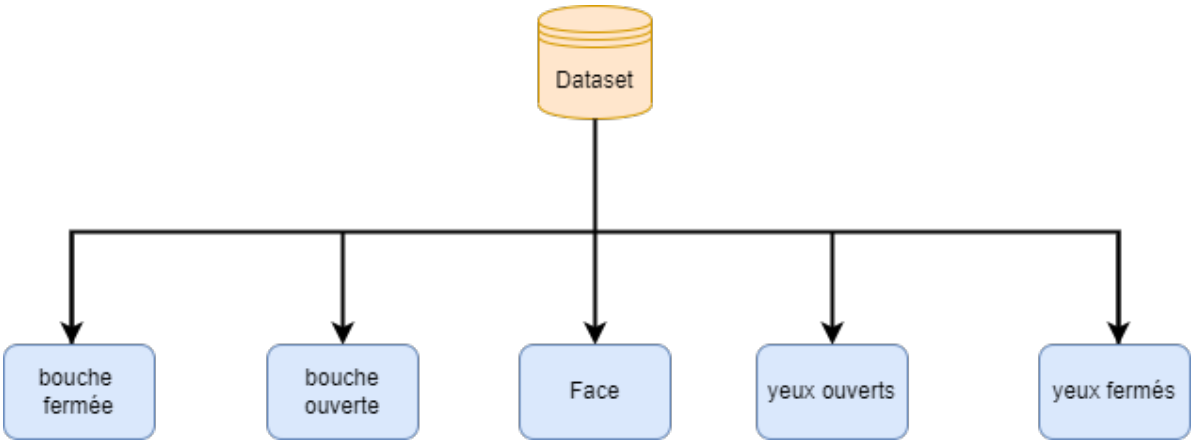


FIGURE 3.4 – Les classes dataset.

4.2 Chargement des données

Étiquetage des données (Data labeling) : Les données ont été étiquetées en 5 classes dans le format YOLO.

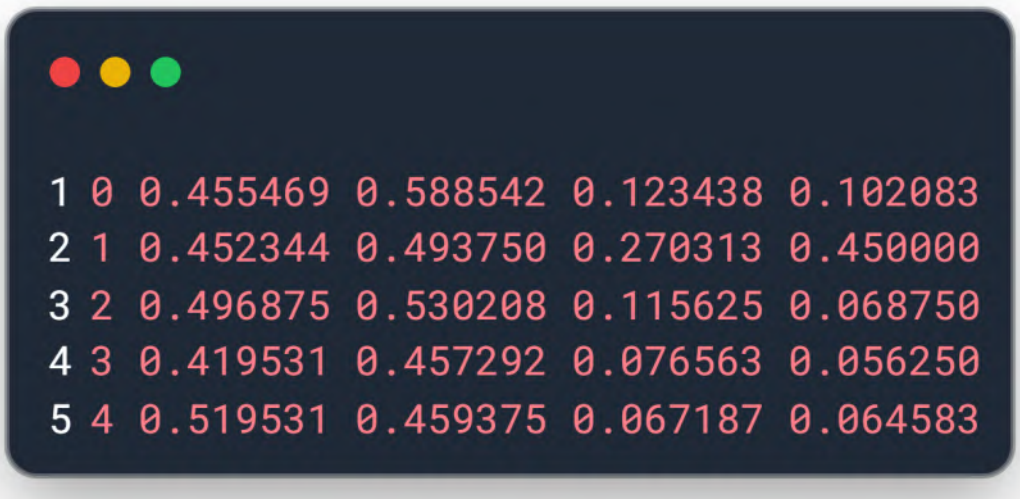


FIGURE 3.5 – Labels des 5 classes.

Chaque ligne a le format suivant : classe, centre de x, centre de y, largeur, hauteur. où nous disposons de 5 classes, comme le montre la figure ci-dessus 3.5.

4.3 La division de l'ensemble de données

Une approche courante pour diviser un ensemble des données (dataset) consiste à les séparer en ensemble d'entraînement et de test. Le rapport optimal d'échantillons répartis dans chaque ensemble peut varier selon le problème spécifique. Cependant, la répartition est de 80% pour l'entraînement et de 20% pour les tests, comme présenté dans la figure ci-dessous 3.6.

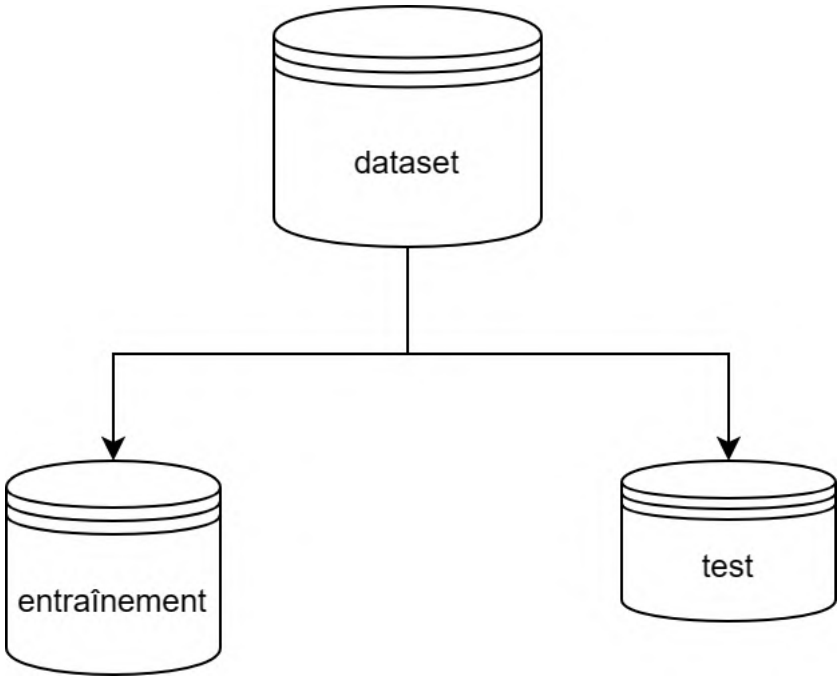


FIGURE 3.6 – Division de l'ensemble des données(Dataset).

4.4 L'architecture YOLOv4

YOLOv4 est la quatrième version de la famille de modèles You Only Look Once. YOLOv4 fait de la détection en temps réel une priorité et conduit l'entraînement sur un seul GPU[44]. YOLOv4 fonctionne deux fois plus vite qu'EfficientDet avec des performances comparables, et qu'il améliore le mAP et le FPS de YOLOv3 de 10% et 12%, respectivement, sur l'ensemble de données MS COCO.Les caractéristiques ci-dessus donnent des résultats de pointe : 43,5% mAP (65,7% mAP50) dans l'ensemble de données MS COCO à une vitesse en temps réel d'environ 65 FPS[42]

4.4.1 Composants d'un détecteur d'objets YOLO :

Une architecture complète de détecteur d'objets 3.7 se compose généralement de trois parties[42] :

— Backbone(Dorsale)

— Neck

— Head(Tête)

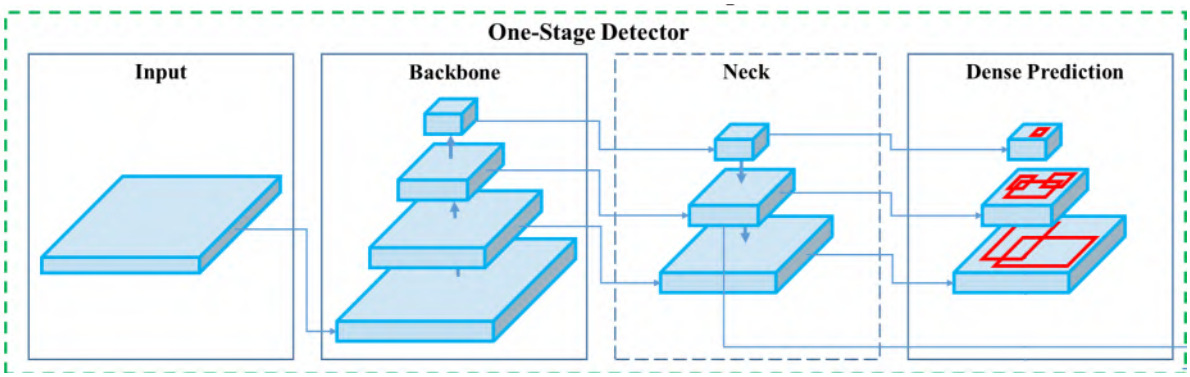


FIGURE 3.7 – Détecteur d'objets de l'architecture YOLO.[11]

— **Backbone :** un backbone est utilisé comme un extracteur de caractéristiques pré-entraîné avec des tâches de classification sur un ensemble de données ImageNet. Les détecteurs d'objets formés et testés sur GPU utilisent un backbone plus lourd (par exemple, VGG, ResNet, DenseNet et Darknet-19/53), tandis que, pour le CPU, un backbone plus léger (par exemple, SqueezeNet, MobileNet, etc.) est utilisé. Il y a toujours un compromis entre la vitesse et la précision. Les backbones plus légers sont plus rapides mais moins précis que les plus lourds.

— **Neck :**YOLO neck combine et mélange les représentations des couches ConvNet avant de les transmettre à la tête de prédiction head. Neck est un sous-ensemble du sac de spécialités, il recueille essentiellement des cartes de caractéristiques(feature) provenant de différentes étapes de l'épine dorsale. En termes simples, il s'agit d'un agrégateur de caractéristiques(feature). neck du pipeline de détection d'objets sera discuté plus en détail dans les sections suivantes.

— **Head(tête) :** Head est utilisée pour prédire les coordonnées des boîtes englobantes et des étiquettes de la classe des objets.

4.4.2 Sélection de l'architecture YOLOv4

Réseau backbone YOLOv4 :Formation des caractéristiques(Feature)

Le réseau de base d'un détecteur d'objets est généralement pré-entraîné à l'aide de la classification ImageNet. Le pré-entraînement signifie que les poids du réseau ont déjà été ajustés pour identifier les caractéristiques pertinentes d'une image, mais ils s'ajusteront

à la nouvelle tâche de détection d'objet.

Les auteurs ont considéré les réseaux de base suivants pour le détecteur d'objets YOLOv4.

— CSPResNext50.

— CSPDarknet53.

— EfficientNet-B3.

En matière de classification d'images, EfficientNet surpasse les autres réseaux de taille comparable. Cependant, les auteurs de YOLOv4 supposent que les autres réseaux peuvent être plus performants en matière de détection d'objets et décident de les expérimenter tous. Sur la base de leur intuition et des résultats expérimentaux (c'est-à-dire BEAUCOUP de résultats expérimentaux), le réseau YOLOv4 final met en œuvre CSPDarknet53 pour le réseau backbone.

YOLOv4 Neck : Agrégation des caractéristiques(Feature)

L'étape suivante de la détection d'objets consiste à mélanger et à combiner les caractéristiques(features) formées dans backbone ConvNet afin de préparer l'étape de détection.

YOLOv4 Head :L'étape de la détection

YOLOv4 implémente le même head YOLO que YOLOv3 pour la détection avec les étapes de détection basées sur l'ancrage, et trois niveaux de granularité de détection.

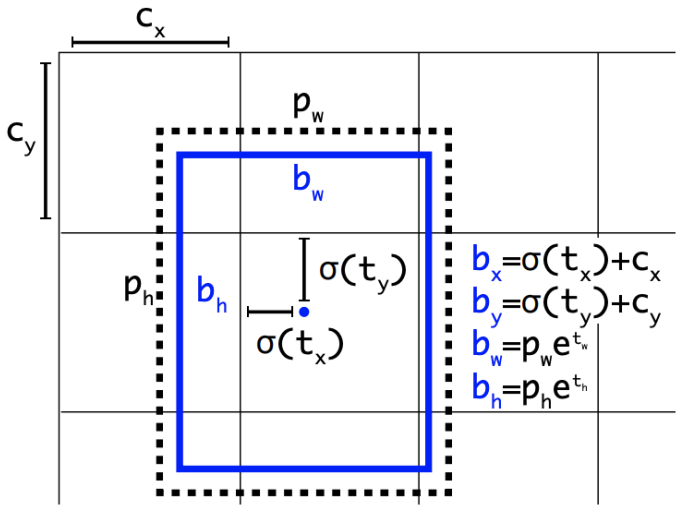


FIGURE 3.8 – Boîtes de délimitation pour la dimension et localisation prédiction.[46]

4.5 Détection

Après une phase d'entraînement approfondie,le modèle YOLOv4 est maintenant capable de détecter avec une grande précision les différentes classes associées à la somnolence. Pour évaluer les performances du modèle YOLOv4, nous effectuerons des tests en utilisant la partie du dataset spécifiquement réservée à cet effet. Cette étape nous permettra de vérifier l'efficacité du modèle.

4.6 Évaluation du modèle YOLO

Afin d'évaluer les performances du modèle, nous devons calculer certaines mesures telles que la précision(precision), le rappel(recall) et le score F1 ont été utilisées. Quatre facteurs sont utilisés pour définir les mesures de performance[41] :

1. **True(vrai) positive (TP) :**Le modèle a prédit l'étiquette et les correspondances correctes conformément à la vérité.
2. **True(vrai) negative (TN)** Le modèle ne prédit pas l'étiquette et ne fait pas partie de la vérité.
3. **False(faux) positive (FP) :** Le modèle a prédit l'étiquette, mais cela ne fait pas partie de la vérité.
4. **False(faux) negative (FN) :** Le modèle ne prédit pas d'étiquette, mais il fait partie de la vérité.

Pour calculer les mesures de précision, le rappel (recall) et le F1 score, on fait[27] :

— **précision :** La précision est la capacité du classificateur à identifier de manière unique les objets pertinents.. Elle correspond à la proportion de prédictions positives correctes et s'exprime par la formule :

$$\text{Precision } = \frac{TP}{TP + FP} = \frac{TP}{\text{all detections}} \tag{3.1}$$

— **rappel(recall) :**Le rappel est une métrique qui mesure la capacité d'un classificateur à trouver tous les cas pertinents (c'est-à-dire toutes les vérités fondamen-

tales). Il s'agit du pourcentage de vrais positifs trouvés parmi toutes les vérités base et est défini comme suit :

$$\text{Recall } = \frac{TP}{TP + FN} = \frac{TP}{\text{all truths}} \tag{3.2}$$

— **F1 score :** Le score F1 est la moyenne harmonique de la précision et du rappel(recall).

$$F_1 \text{ score } = 2\left(\frac{\text{Precision } \times \text{ Recall}}{\text{Precision } + \text{ Recall}}\right) \tag{3.3}$$

4.7 Dataset pour LSTM :

Nous avons utilisé un ensemble de données (dataset) qui se focalise sur la prédiction du niveau de fatigue. Ce dataset est stocké dans un fichier Excel que nous avons utilisé pour notre étude :

— FatigueDetect : un fichier Excel contenant des données sur la fermeture des yeux et les bâillements pour différencier les niveaux de fatigue (faible, moyen, élevé).

4.8 Détection de la somnolence

La détection de la somnolence repose sur la combinaison des modèles YOLO et LSTM. Le modèle YOLO est utilisé pour détecter le visage, l'état des yeux et de la bouche, tandis que le modèle LSTM évalue le niveau de fatigue du conducteur (faible, moyen, élevé). En combinant ces deux modèles, nous sommes en mesure de détecter la somnolence.

5 Conclusion

Au cours de ce chapitre, nous avons présenté en détail notre approche pour la conception de modèle utilisant l'apprentissage profond avec YOLO et LSTM. Dans le prochain chapitre, nous fournirons une description des environnements et outils de développement utilisés dans la mise en oeuvre de notre méthodologie proposée.

Chapitre 4

Implémentation de l'architecture et résultats

1 Introduction

Le développement de notre conception qui met en avant notre méthodologie complète pour la création d'un système d'apprentissage profond dédié à la détection de la somnolence. Notre approche s'appuie sur l'utilisation de l'architecture YOLOv4 et de la technique LSTM, comme proposé dans le chapitre 3.
Nous décrivons en détail les différentes étapes impliquées dans le processus de développement, allant de la collecte et la préparation des données d'entraînement. Nous expliquons également la méthodologie utilisée pour l'entraînement du modèle, en mettant l'accent sur les techniques de pré-traitement des données.
De plus, nous présentons les résultats obtenus à travers une évaluation approfondie de notre système.

2 Environnement de développement du matériel

Nous avons généré nos résultats sur une machine qui possèdent les caractéristiques suivantes :

- **Type de processeur :** 11th Gen Intel(R) Core(TM) i5-11300H @ 3.10GHz 3.11 GHz.
- **Capacité de la RAM :** 8 Go.
- **Disque dur :** 512 GO (SSD)
- **Système d'exploitation :** Windows10 Professionnel 64 bits

3 Environnement de déloppement logiciel

Nous avons utilisé l'environnement pycharm avec le langage de programmation python, nous avons également utilisé différentes bibliothèques et des outils qui nous aides à réaliser notre programme.

3.1 Python

Python est un langage de programmation interprété de haut niveau, polyvalent et facile à apprendre. Il prend en charge les structures de données de haut niveau, une approche simple mais efficace de la programmation orientée objet, et une syntaxe élégante qui permet aux programmeurs de s'exprimer de manière claire et concise.[49]

3.2 Google Colab

La formation d'un modèle d'apprentissage en profondeur peut nécessiter une charge de travail importante du CPU/GPU, ce qui nous a amenés à utiliser la plate-forme cloud Google Colab. Colaboratory est un projet de recherche Google créé pour aider à diffuser l'enseignement et la recherche sur l'apprentissage automatique. Il s'agit d'un environnement de notebook Jupyter qui ne nécessite aucune configuration pour s'utiliser et s'exécute entièrement sur le cloud.[20]

3.3 OpenCV

OpenCV (Open Source Computer Vision Library) est une bibliothèque logicielle open source de vision par ordinateur et d'apprentissage automatique. OpenCV a été conçu pour fournir une infrastructure commune aux applications de vision par ordinateur et pour accélérer l'utilisation de la perception automatique dans les produits commerciaux.

3.4 NumPy

NumPy est un module Python permettant de manipuler les tableaux. Il contient des outils pour manipuler un objet tableau multidimensionnel de haute performance. C'est le module Python le plus important pour le calcul scientifique.[22]

3.5 Matplotlib

Matplotlib est une bibliothèque de visualisation de données en Python. Elle dispose de tous les outils nécessaires pour créer des graphiques et de visualisations de haute

qualité. Matplotlib permet de créer de nombreux types de graphiques, comme des graphiques linéaires, des graphiques en barres, des graphiques circulaires, des graphiques de dispersion, des graphiques en boîte, des graphiques en nuages de points, des graphiques en surface, des graphiques en 3D, et bien d'autres encore.[14]

4 Préparation et prétraitement de dataset

4.1 Extraction des images par une vidéo

```
import numpy as np
import cv2
from glob import glob
from os import path
path = 'C:/Users/Mahmoud/Downloads/F/main/YOLO/obj1'
def save_frame(video_path, save_dir, gap=10):
    name = video_path.split("/")[-1].split(".")[0]
    save_path = os.path.join(save_dir, name)
    create_dir(save_path)
    cap = cv2.VideoCapture(video_path)
    idx = 0
    while True:
        ret, frame = cap.read()
        if ret == False:
            cap.release()
            break
         # Enregistre le premier frame
        if idx == 0:
            cv2.imwrite("{}{}.png".format(save_path, idx), frame)
        else:
        # Enregistre le frame toutes les 'gap' images
            if idx % gap == 0:
                cv2.imwrite("{}{}.png".format(save_path, idx), frame)
        if idx==5000:
if _name_ == "_main_":
```

```
# Chemin d'acces aux videos
    video_paths = glob("C:/Users/Mahmoud/Downloads/F/main/YOLO/G.avi")
    # Repertoire de sauvegarde des frames
    save_dir = "C:/Users/Mahmoud/Downloads/F/main/YOLO/obj1"
    for path in video_paths:
        save_frame(path, save_dir, gap=1)
```

La figure 4.1 ci-dessous montre l'extraction des images par une vidéo.

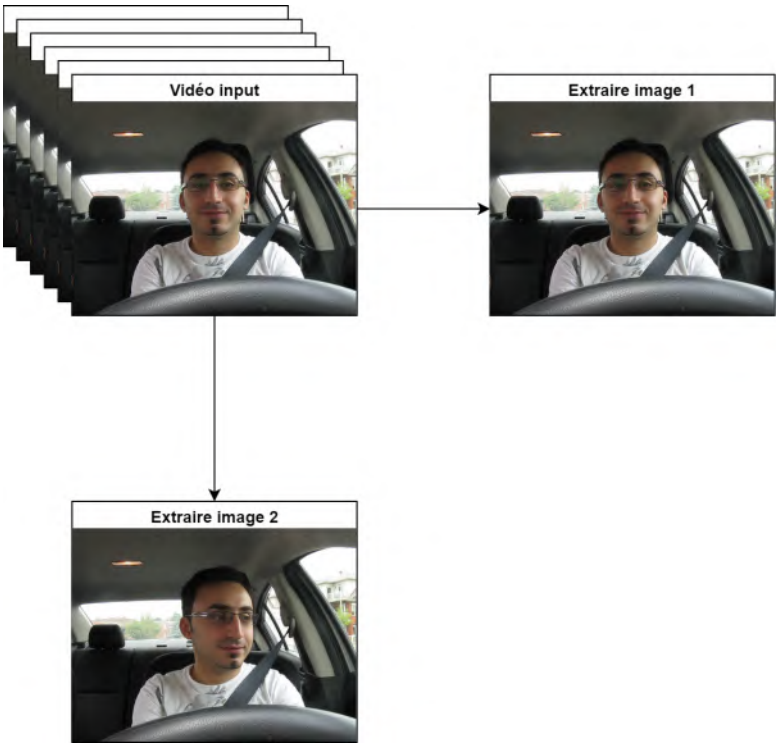


FIGURE 4.1 – Extraire une séquence d'images d'une vidéo.

4.2 La création de labels

On a utilisé le logiciel labelImg, comme le montre la figure 4.2, pour créer des labels pour chaque image, et pour cela on ajoute les étiquettes ou des classes aux boîtes de délimitation pour indiquer ce que chaque objet représente.

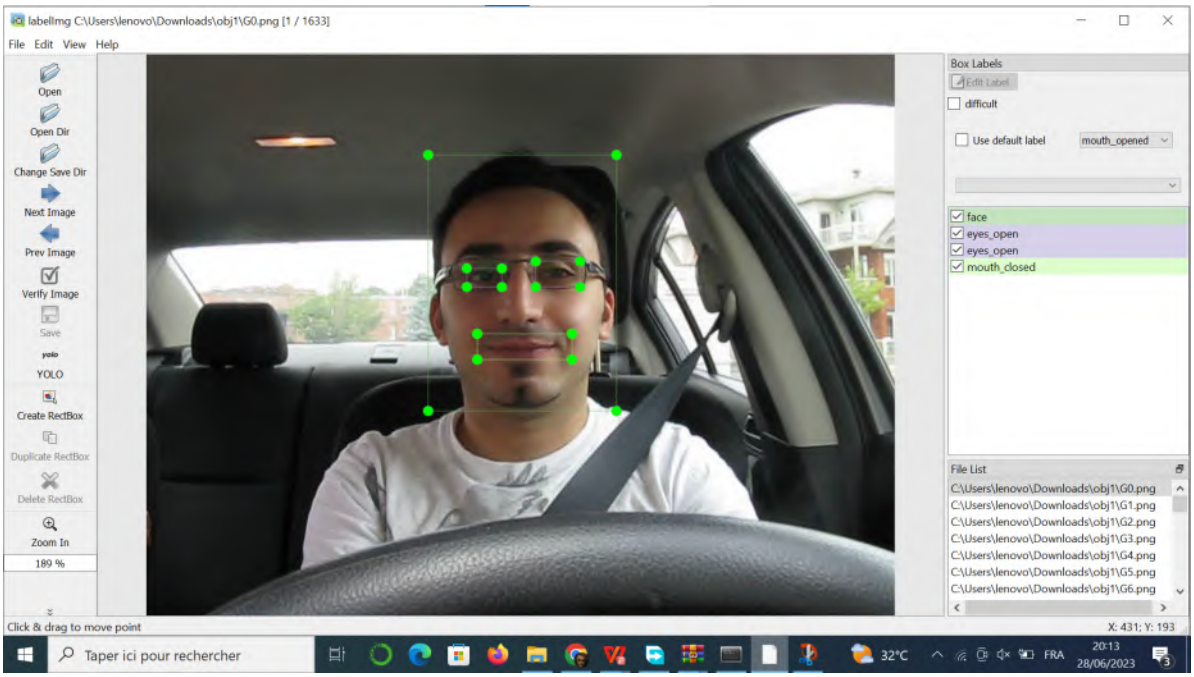


FIGURE 4.2 – Création de labels.

4.3 La division de Dataset

À cette étape, nous procédons à la division de nos données en deux ensembles distincts : l'ensemble d'entraînement (train) et l'ensemble de test (test). La répartition se fait de la manière suivante : 80% des données sont attribuées à l'ensemble d'entraînement, tandis que les 20% restants sont réservés à l'ensemble de test.

```
import glob, os

dataset_path = '/content/darknet/config/ob1'

# Pourcentage d'images utiliser pour l'ensemble de test
percentage_test = 20

# Cree train.txt et test.txt
file_train = open('/content/darknet/config/train.txt', 'w')
file_test = open('/content/darknet/config/test.txt', 'w')

# Remplir les fichiers train.txt et test.txt
counter = 1
```

```
index_test = round(100 / percentage_test)
for pathAndFilename in glob.iglob(os.path.join(dataset_path, "*.png"))
                                        :
    title, ext = os.path.splitext(os.path.basename(pathAndFilename))
    if counter == index_test:
        counter = 1
        file_test.write("/content/darknet/config/ob1" + "/" + title +
                                                '.png' + "\n")
    else:
        file_train.write("/content/darknet/config/ob1" + "/" + title +
                                                 '.png' + "\n")
        counter = counter + 1
```

5 Construction le modèle YOLO

5.1 Télécharger pré-entraînée weights

```
1 !wget https://github.com/AlexeyAB/darknet/releases/download/darknet_yolo_v4_pre/yolov4.137
```

FIGURE 4.3 – Télécharger YOLOV4 pré-entraînée weights.

5.2 Préparation des données

Définition du fichier "obj.names". Dans ce fichier, nous spécifions les noms des classes, comme le montre la figure 4.4.

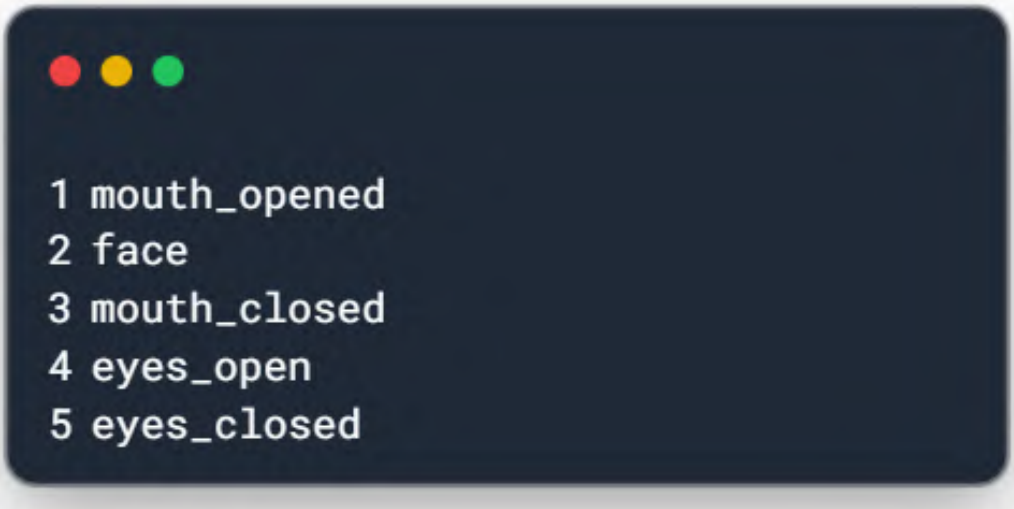


FIGURE 4.4 – Fichier obj.names pour YOLOV4

Définition du fichier "obj.data", dans ce fichier, nous spécifions le chemin d'accès à notre dataset, le fichier"obj.names" et le nombre des classes, comme le montre la figure 4.5.

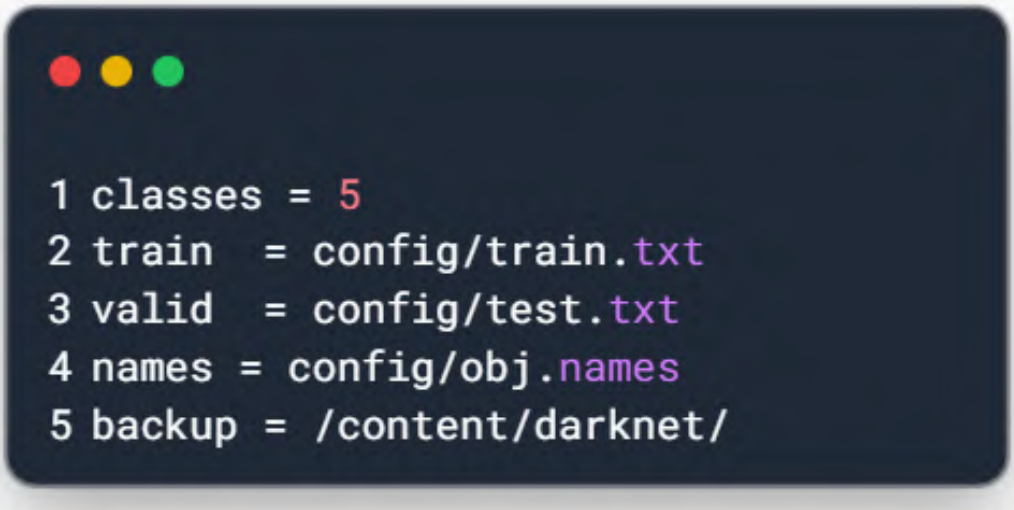


FIGURE 4.5 – Fichier obj.data pour YOLOV4.

5.3 Entraînement le modèle YOLOV4

```
1 !chmod +x ./darknet
2 !./darknet detector train config/obj.data config/yolov4.cfg config/yolov4.conv.137
```

FIGURE 4.6 – Entraînement le modèle YOLOV4.

6 Réalisation notre modèle LSTM

6.1 Importer les bibliothèques et modules

Afin de créer des réseaux de neurones profond à l'aide de Keras, nous commençons par importer les différentes bibliothèques et modules nécessaires, comme indiqué dans le code et expliqué dans le Tableau 4.1 ci-dessous.

```
import numpy as np
import pandas as pd
import cv2
from keras.models import Sequential
from keras.layers import Dense
from keras.layers import LSTM
from tensorflow import keras
from sklearn.model_selection import train_test_split
```

Bibliothèques/modules	Descriptions
pandas	Pandas est une bibliothèque open-source populaire en Python qui fournit des structures de données et des outils d'analyse de données performants.
Dense	Est utilisé pour instancier une couche dense.
LSTM	Cela permet d'ajouter une couche LSTM à un modèle Keras existant et de configurer ses paramètres pour l'entraînement et la prédiction.
Adam	Optimiseur qui implémente l'algorithme Adam. L'optimisation d'Adam est une méthode de descente de gradient stochastique basée sur l'estimation adaptative des moments du premier et du second ordre.
sklearn	sklearn est une bibliothèque puissante et populaire d'apprentissage en Python, offrant une gamme d'outils et d'algorithmes pour développer, entraîner et évaluer des modèles d'apprentissage de manière efficace.

TABLE 4.1 – Description des bibliothèques et modules utilisés.

6.2 Chargement de l'ensemble des données(Dataset)

Dans cette étape, nous allons utiliser dataset à partir d'un fichier Excel. Ensuite, nous diviserons les données en ensembles de dataset pour l'entraînement et le test, En résumé, ce code charge les données (dataset) à partir d'un fichier Excel.

```
# importer dataset
df=pd.read_excel('C:/Users/Mahmoud/Downloads/F/main/LSTM/LSTM_200.xlsx
                                        ')
# Extraire values
values = df.values
values=values.astype('float32')
```

```
#Extraire les caracteristiques et les variables cibles
X=values[:,:-3]
Y=values[:,400:]
# Fractionner les donnees en ensembles entrainement et de test
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.
                                    15, random_state=4)
#Remodeler les donnees pour le modele LSTM
x_train = x_train.reshape((x_train.shape[0], 1, x_train.shape[1]))
x_test= x_test.reshape((x_test.shape[0], 1, x_test.shape[1]))
print(x_train.shape, y_train.shape,  x_test.shape, y_test.shape)
```

6.3 Entraînement modèle LSTM

Ce code ci-dessous présente comment effectuer l'entraînement d'un modèle LSTM. Il illustre les étapes nécessaires pour entraîner efficacement un modèle LSTM.

```
#creation de modele
model = Sequential()
model.add(LSTM(200, input_shape=(x_train.shape[1],x_train.shape[2])))
model.add(Dense(150, activation='relu'))
model.add(Dense(150, activation='relu'))
model.add(Dense(3, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='SGD',
                                    metrics=['accuracy'])
model.fit(x_train, y_train, epochs=80,batch_size=64,verbose=1)
#Sauvegarder le modele
model.save('model_dense300')
```

7 Résultats obtenus à la détection de la somnolence

7.1 Présentation des performances obtenu

7.1.1 YOLO

Pour l'entraînement, comme le montre la figure 4.7, nous avons effectué 10000 itérations de notre modèle YOLO. nous avons obtenu deux courbes la premiere pour "mAP" et la second pour "Loss".

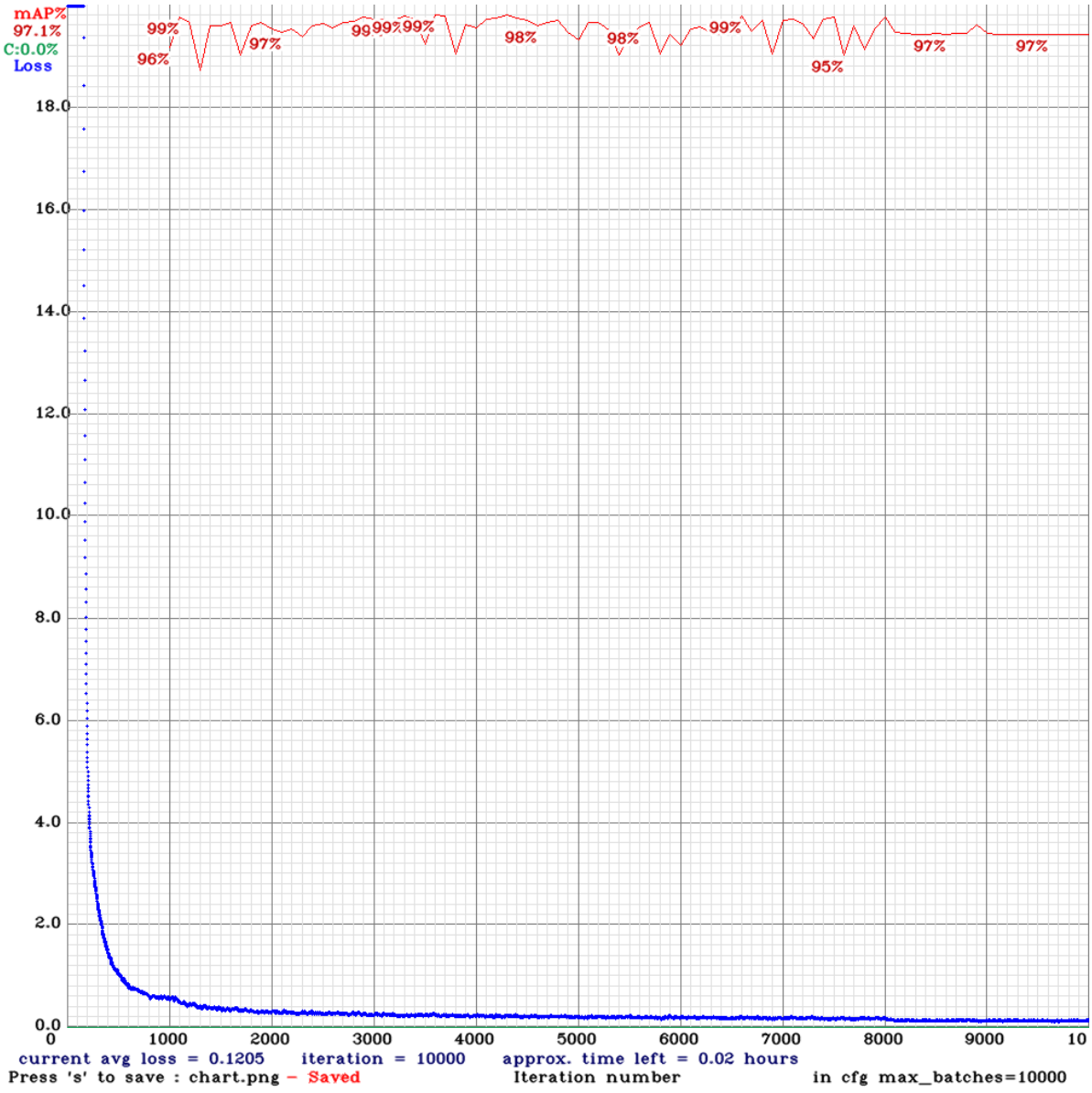


FIGURE 4.7 – Courbe Loss et mAP.

Pour l'évaluation des performances, le modèle a livré des résultats pour les classes, comme le montre la figure 4.8.
la classe "face", "mouth-closed" et "mouth-opened" on a une valeur de précision/rappel de "1" parce qu'ils sont faciles à détecter et indiquant la forte capacité du modèle à reconnaître et à interpréter avec précision, En outre "eyes-open" présente des valeurs de précision/rappel de "0.97", par contre la classe "eyes-closed" présente des valeurs inferieures que les autres classes en précision/rappel.

```
class_id = 0, name = mouth_opened, ap = 100.00%
class_id = 1, name = face, ap = 100.00%
class_id = 2, name = mouth_closed, ap = 100.00%
class_id = 3, name = eyes_open, ap = 97.53%
class_id = 4, name = eyes_closed, ap = 87.92%

 for conf_thresh = 0.25, precision = 0.99, recall = 0.97, F1-score = 0.98
 for conf_thresh = 0.25, TP = 544, FP = 7, FN = 18, average IoU = 83.66 %

 IoU threshold = 50 %, used Area-Under-Curve for each unique Recall
 mean average precision (mAP@0.50) = 0.970908, or 97.09 %
```

FIGURE 4.8 – Résultat du modèle YOLO.

Pour toutes les classes, les résultats du précision, rappel "recall" et F1 score obtenus par mon modèle mettent en lumière ses performances.
On a obtenu un score de "0,99" pour la précision, ce qui est excellent, et pour rappel "recall" on a de "0.97" et enfin on a F1 score qui a atteint une valeur impressionnante de "0.98", Ce résultat F1 score démontre l'efficacité du modèle dans diverses classes.

Le modèle a obtenu un résultat de 0,97 en mean average precision(mAP), ce qui démontre sa performance élevée et sa précision remarquable dans le prédictions. De plus, nous obtenons une valeur de "0,1205" pour la moyenne loss (average loss), ce qui est illustré par notre courbe.

Résultats du modèle sur les données test

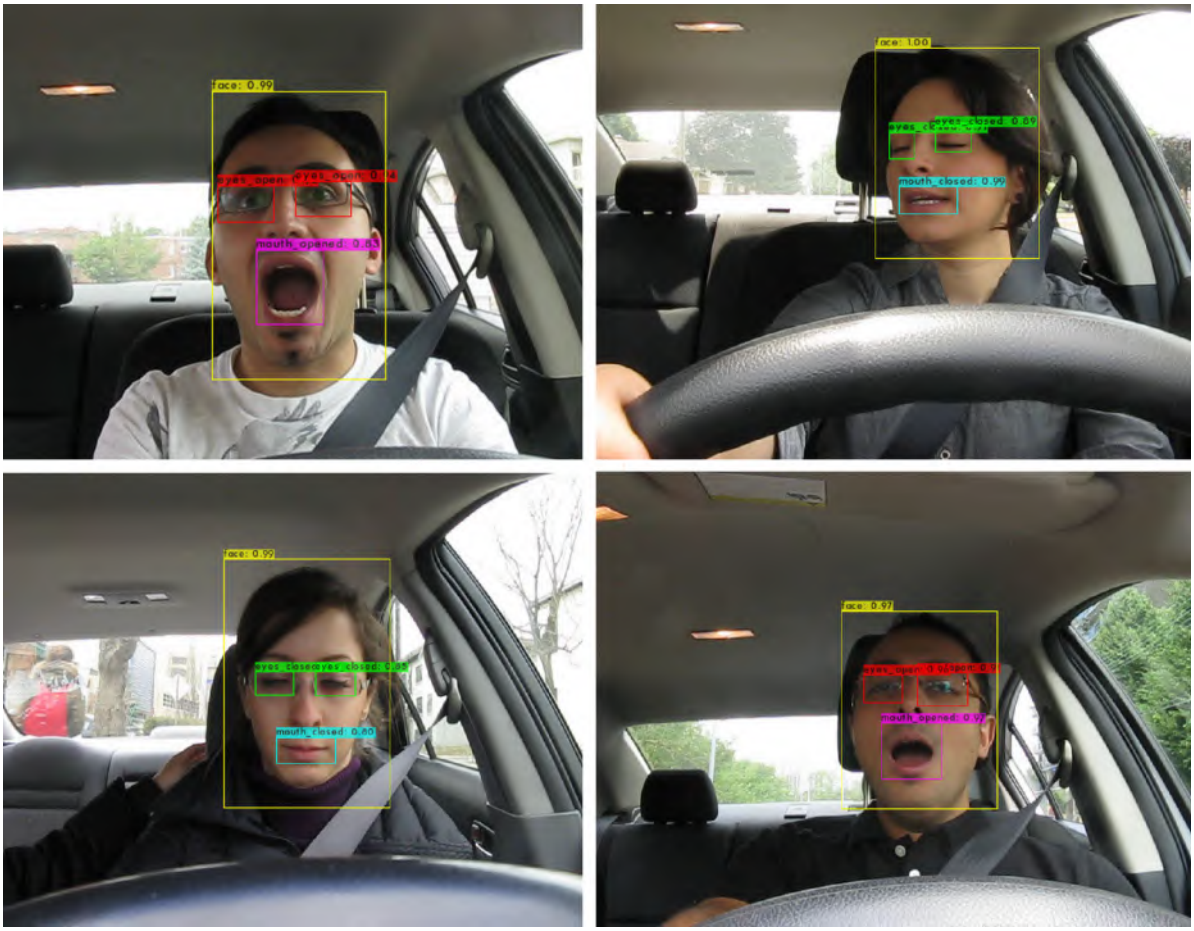


FIGURE 4.9 – Exemple de tests.

7.1.2 LSTM

Pour suivre l'évolution des performances de notre modèle LSTM en apprentissage profond tout au long de l'entraînement, nous avons créé :

— Un graphe ("accuracy") sur l'ensemble de données d'entraînement ("train-acc") pour les différentes époques d'entraînement.

— Un graphe ("loss") sur l'ensemble de données d'entraînement ("train-loss") au cours des époques d'entraînement.

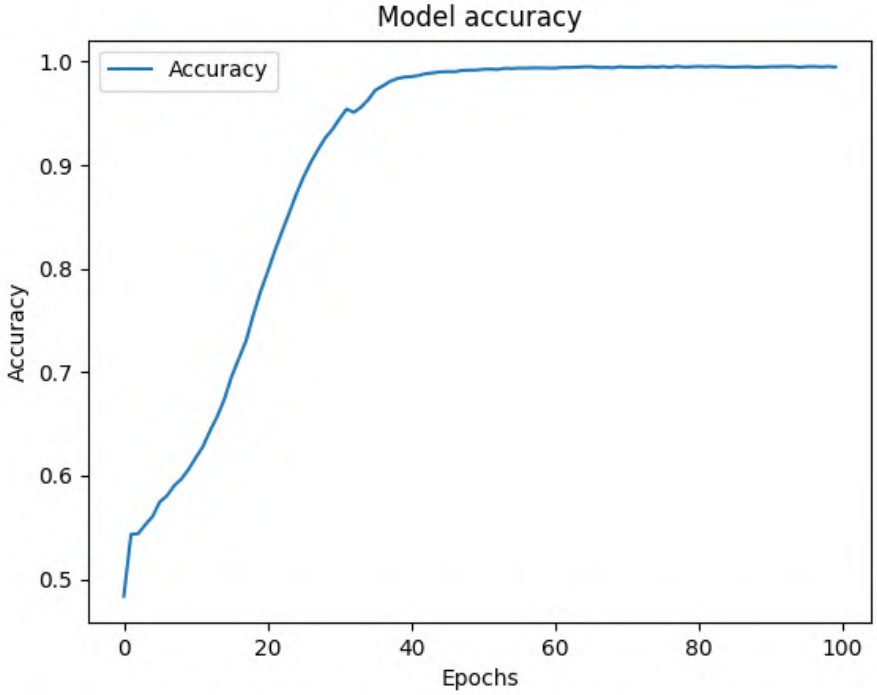


FIGURE 4.10 – Accuracy.

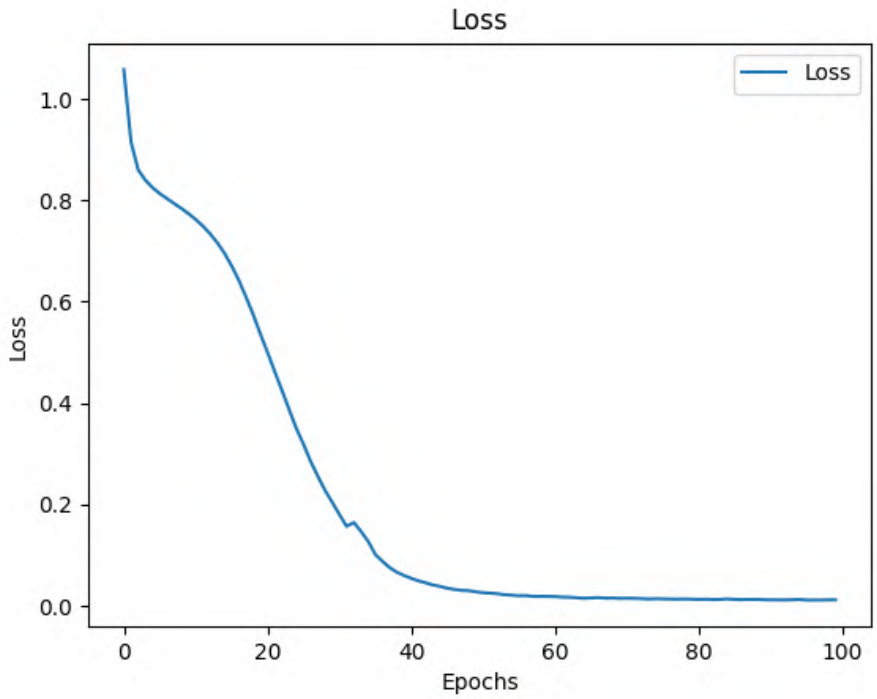


FIGURE 4.11 – Loss

8 Détection de la somnolence

On a capturé une image d'un conducteur à partir d'une vidéo, sur laquelle nous avons appliqué des cadres de détection pour le visage, la bouche et les yeux. Cette image nous permet d'observer le temps de fermeture des yeux (eye closure) ainsi que l'ouverture de la bouche (mouth open). De plus, nous avons évalué le niveau de fatigue du conducteur classé comme élevé (high), moyen (medium) ou faible (low) à partir des yeux fermés et de la bouche ouverte, nous pouvons obtenir une estimation approximative de son état de vigilance. Nous avons également détecté les bâillements (yawn detected) et la fermeture des yeux (eye closure) dans cette image. Ces informations sont précieuses pour évaluer le degré de vigilance du conducteur et prendre les mesures nécessaires afin d'assurer sa sécurité.

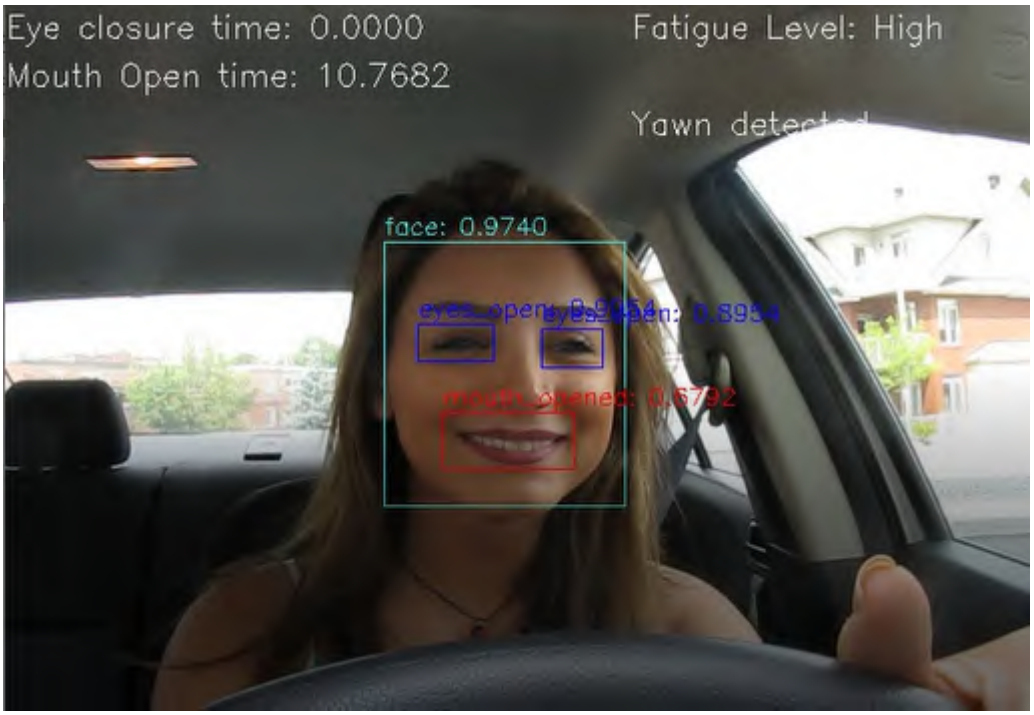


FIGURE 4.12 – Résultat de la détection .

9 Conclusion

Dans ce chapitre, nous avons fourni une vue d'ensemble et une explication complète de l'architecture de notre modèle. Nous avons également abordé les différents outils

et progiciels essentiels pour la mise en œuvre de notre système. En outre, nous avons détaillé les résultats de nos expériences , en soulignant les performances remarquables obtenues dans la détection de la somnolence.

Conclusion générale et perspectives

L'intelligence artificielle (IA) a ouvert de nouvelles perspectives dans divers domaines, y compris la détection de la somnolence. Grâce à l'apprentissage profond, l'IA joue un rôle central dans la révolution des pratiques liées à la détection de la somnolence. En exploitant la puissance de l'IA, cette étude vise à améliorer l'exactitude des méthodes de détection de la somnolence et obtenir de meilleurs résultats dans la prévention des incidents liés à la somnolence.

L'intelligence artificielle (IA) joue un rôle essentiel dans l'amélioration de la précision de la détection de la somnolence, notamment grâce à l'utilisation de l'architecture YOLO-LSTM. Cette architecture combine les avantages de YOLO pour la détection du visage, des yeux et de la bouche, et ceux de LSTM pour la prédiction de la fatigue basée sur la fermeture des yeux et les bâillements.

En utilisant YOLO, notre système est capable de localiser et de suivre les traits du visage, y compris les yeux et la bouche, ce qui permet de détecter les signes de somnolence tels que les clignements des yeux fréquents. En parallèle, en utilisant LSTM, le système analyse les séquences temporelles de fermeture des yeux et de bâillements, qui sont des indicateurs importants de la fatigue.

Les résultats obtenus ont clairement démontré l'efficacité de notre modèle pour la détection la somnolence. en YOLO nous avons obtenu un score de précision 97% et en LSTM nous avons obtenu 96%.

En conclusion, notre résultat met en évidence l'impact remarquable de l'IA et de l'apprentissage profond dans la détection de la somnolence. Grâce à la conception et à la mise en œuvre d'un système basé sur YOLO-LSTM.

Perspectives

Cependant, cette recherche pourrait être améliorée sur certains points que nous considérons comme des perspectives pour nos travaux :

Affiner le modèle : En perfectionnant le modèle YOLO pré-entraîné spécifiquement pour la détection de la somnolence, il est possible d'obtenir des améliorations significatives dans les résultats, en effectuant l'entraînement sur un ensemble de données plus vaste. avec des annotations détaillées.

Élargir l'ensemble de données : Bien que l'ensemble de données que nous avons utilisé pour l'entraînement du modèle, l'ajout d'une plus grande variété d'images de conducteurs dans différentes situations de conduite, avec des annotations spécifiques pourrait renforcer la capacité du modèle à généraliser et à détecter efficacement la somnolence. Cette expansion de l'ensemble de données contribuerait à améliorer la robustesse de notre système et à garantir sa performance optimale dans une gamme plus étendue de scénarios réels de conduite.

Bibliographie

[1] Rectified linear unit (relu), 2019. URL https://deepchecks.com/glossary/rectified-linear-unit-relu/.

[2] Faiza Abdat. Reconnaissance automatique des émotions par données multimodales : expressions faciales et signaux physiologiques. *Université de Metz, France*, 2010.

[3] Priyal Agarwal. Regression analysis, 2021. URL https://www.analyticsvidhya.com/blog/2021/05/regression-analysis-real-time-portugal-2019-election-results/.

[4] Shivanshu Aggarwal. Understanding lstm networks, 2015. URL https://colah.github.io/posts/2015-08-Understanding-LSTMs/.

[5] Shivanshu Aggarwal. The ultimate guide to building your own lstm models, 2023. URL https://www.projectpro.io/article/lstm-model/832.

[6] Pratik Ahamed, Soumyadeep Kundu, Tauseef Khan, Vikrant Bhateja, Ram Sarkar, and Ayatullah Mollah. Handwritten arabic numerals recognition using convolutional neural network. *Journal of Ambient Intelligence and Humanized Computing*, 11, 11 2020. doi:10.1007/s12652-020-01901-7.

[7] Ali Mansour Al-Madani, Ashok T Gaikwad, Vivek Mahale, Zeyad AT Ahmed, and Ahmed Abdullah A Shareef. Real-time driver drowsiness detection based on eye movement and yawning using facial landmark. In *2021 International Conference on Computer Communication and Informatics (ICCCI)*, pages 1–4. IEEE, 2021.

[8] Nawal Alioua. *Extraction et analyse des caractéristiques faciales : Application à l'hypovigilance chez le conducteur*. PhD thesis, INSA de Rouen ; Université Mohammed V-Agdal (Rabat, Maroc). Faculté des sciences, 2015.

[9] Talha Ashar. How to implement the softmax function in python. URL `https://www.educative.io/answers/how-to-implement-the-softmax-function-in-python`.

[10] Nebojša Bačanin Džakula et al. Convolutional neural network layers and architectures. In *Sinteza 2019-International Scientific Conference on Information Technology and Data Related Research*, pages 445–451. Singidunum University, 2019.

[11] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4 : Optimal speed and accuracy of object detection. *arXiv preprint arXiv :2004.10934*, 2020.

[12] David F Dinges and Richard Grace. Perclos : A valid psychophysiological measure of alertness as assessed by psychomotor vigilance. *US Department of Transportation, Federal Highway Administration, Publication Number FHWA-MCRT-98-006*, 1998.

[13] Wenhui Dong and Xiaojuan Wu. Fatigue detection based on the distance of eyelid. In *Proceedings of 2005 IEEE International Workshop on VLSI Design and Video Technology, 2005.*, pages 365–368. IEEE, 2005.

[14] Daniel M Faes. Use of python programming language in astronomy and science. *arXiv preprint arXiv :1807.04806*, 2018.

[15] Mehrdad Farahani, Marzieh Farahani, Mohammad Manthouri, and Okyay Kaynak. Short-term traffic flow prediction using variational lstm networks. *arXiv preprint arXiv :2002.07922*, 2020.

[16] S Gangadhar and R Shanta. Chapter 8-machine learning. *Handbook of Statistics*, 38 :197–228, 2018.

[17] Er Manoram Vats1and Er Anil Garg. Detection and security system for drowsy driver by using artificial neural network technique. *International Journal of Applied Science and Advance Technology*, 1(1) :39–43, 2012.

[18] Bernard GIUSIANO. *Deep Learning 2.* PhD thesis, 2020.

[19] Allan David Gordon. *Classification.* CRC Press, 1999.

[20] Teddy Surya Gunawan, Arselan Ashraf, Bob Subhan Riza, Edy Victor Haryanto, Rika Rosnelly, Mira Kartiwi, and Zuriati Janin. Development of video-based emotion recognition using deep learning with google colab. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 18(5) :2463–2471, 2020.

[21] OUAMANE Hanane. *Identification de reconnaissance faciale avec des expressions.* PhD thesis, université mohamed khider, 2012.

[22] Charles R Harris, K Jarrod Millman, Stéfan J Van Der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, Julian Taylor, Sebastian Berg, Nathaniel J Smith, et al. Array programming with numpy. *Nature*, 585(7825) :357–362, 2020.

[23] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[24] Wen-Bing Horng, Chih-Yuan Chen, Yi Chang, and Chun-Hai Fan. Driver fatigue detection based on eye tracking and dynamic template matching. In *IEEE International Conference on Networking, Sensing and Control, 2004*, volume 1, pages 7–12. IEEE, 2004.

[25] Tzyy-Ping Jung, Colin Humphries, Te-Won Lee, Scott Makeig, Martin McKeown, Vicente Iragui, and Terrence J Sejnowski. Extended ica removes artifacts from electroencephalographic recordings. *Advances in neural information processing systems*, 10, 1997.

[26] Debasish Kalita. An overview and applications of artificial neural networks, 2022. URL `https://www.analyticsvidhya.com/blog/2022/03/an-overview-and-applications-of-artificial-neural-networks-ann/`.

[27] Kiprono Elijah Koech. Confusion matrix for object detection., 2020. URL `https://towardsdatascience.com/confusion-matrix-and-object-detection-f0cbcb634157`.

[28] Ajitesh Kumar. Different types of cnn architectures explained : Examples, 2023. URL `https://vitalflux.com/different-types-of-cnn-architectures-explained-examples/`.

[29] Amol M Malla, Paul R Davidson, Philip J Bones, Richard Green, and Richard D Jones. Automated video-based measurement of eye closure for detecting behavioral microsleep. In *2010 annual international conference of the IEEE engineering in medicine and biology*, pages 6741–6744. IEEE, 2010.

[30] Amol M Malla, Paul R Davidson, Philip J Bones, Richard Green, and Richard D Jones. Automated video-based measurement of eye closure for detecting behavioral microsleep. In *2010 annual international conference of the IEEE engineering in medicine and biology*, pages 6741–6744. IEEE, 2010.

[31] Amol M Malla, Paul R Davidson, Philip J Bones, Richard Green, and Richard D Jones. Automated video-based measurement of eye closure for detecting behavioral microsleep. In *2010 annual international conference of the IEEE engineering in medicine and biology*, pages 6741–6744. IEEE, 2010.

[32] Mehdi Mohammadi, Ala Al-Fuqaha, Sameh Sorour, and Mohsen Guizani. Deep learning for iot big data and streaming analytics : A survey. *IEEE Communications Surveys & Tutorials*, 20(4) :2923–2960, 2018.

[33] M Mohammedi, Juba Mokrani, et al. *Proposition d'un protocole de détection de somnolence au volant pour une conduite sûre.* PhD thesis, université Abderrahmane Mira-Bejaia, 2020.

[34] Kimia Nadjahi Pascal Monasse. Découvrez les différentes couches d'un cnn, 2022. URL https://openclassrooms.com/fr/courses/4470531-classez-et-segmentez-des-donnees-visuelles/5083336-decouvrez-les-differentes-couches-dun-cnn.

[35] Junjie Peng, Elizabeth C Jury, Pierre Dönnes, and Coziana Ciurtin. Machine learning techniques for personalised medicine approaches in immune-mediated chronic inflammatory diseases : applications and challenges. *Frontiers in pharmacology*, 12 : 720694, 2021.

[36] P Philip. Somnolence et conduite automobile : un enjeu de santé publique encore ignoré. *Médecine du Sommeil*, 1(3) :29–32, 2005.

[37] projectpro. Introduction to convolutional neural networks architecture, 2023. URL https://www.projectpro.io/article/introduction-to-convolutional-neural-networks-algorithm-architecture/560#mcetoc_1fs0joo1hg.

[38] Sharma Pulkit. The most comprehensive guide k-means clustering. *Analytics Vidhya*, 2019.

[39] Siouane Mouhssin Saada Fawzi. *Reconnaissance des expressions faciales en temps réel.* PhD thesis, Université Blida, 2020.

[40] Tim Jones Samaya Madhavan. Deep learning architectures, 2017. URL https://developer.ibm.com/articles/cc-machine-learning-deep-learning-architectures/.

[41] Deval Shah. Mean average precision (map) explained., 2022. URL https://www.v7labs.com/blog/mean-average-precision.

[42] Aditya Sharma. Achieving optimal speed and accuracy in object detection (yolov4)., 2022. URL https://pyimagesearch.com/2022/05/16/achieving-optimal-speed-and-accuracy-in-object-detection-yolov4.

[43] Sagar Sharma, Simone Sharma, and Anidhya Athaiya. Activation functions in neural networks. *Towards Data Sci*, 6(12) :310–316, 2017.

[44] Jacob Solawetz. What is yolov4 ? a detailed breakdown., 2020. URL `https://blog.roboflow.com/a-thorough-breakdown-of-yolov4/`.

[45] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[46] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet : Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020.

[47] Zhichao Tian and Huabiao Qin. Real-time driver's eye state detection. In *IEEE International Conference on Vehicular Electronics and Safety, 2005.*, pages 285–289. IEEE, 2005.

[48] Yassine Touzani. Deep learning : les réseaux de neurones récurrents, 2021. URL `https://datavalue-consulting.com/deep-learning-reseaux-neurones-recurrents-rnn/`.

[49] Raphael Vallat. Pingouin : statistics in python. *J. Open Source Softw.*, 3(31) :1026, 2018.

[50] Daniel Vieira and Joao Paixao. Vector field neural networks. *arXiv preprint arXiv :1905.07033*, 2019.

[51] WW Wierwille. Overview of research on driver drowsiness definition and driver drowsiness detection. In *Proceedings : International Technical Conference on the Enhanced Safety of Vehicles*, volume 1995, pages 462–468. National Highway Traffic Safety Administration, 1995.

[52] Tian. Yan, Zhang. Kaili, Li. Jianyuan, Lin. Xianxuan, and Yang. Bailin. Lstm-based traffic flow prediction with missing data. *Neurocomputing*, 318 :297–305, 2018.

[53] Zuopeng Zhao, Nana Zhou, Lan Zhang, Hualin Yan, Yi Xu, Zhongxin Zhang, et al. Driver fatigue detection based on convolutional neural networks using em-cnn. *Computational intelligence and neuroscience*, 2020, 2020.