

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

الجمهورية الجزائرية الديمقراطية الشعبية

MINISTRY OF HIGHER EDUCATION
AND SCIENTIFIC RESEARCH

HIGHER SCHOOL IN APPLIED SCIENCES
--T L E M C E N--



المدرسة العليا في العلوم التطبيقية
École Supérieure en
Sciences Appliquées

وزارة التعليم العالي والبحث العلمي

المدرسة العليا في العلوم التطبيقية
-تلمسان-

Mémoire de fin d'étude

Pour l'obtention du diplôme d'Ingénieur

Filière : Automatique

Spécialité : Automatique

Présenté par :

SEKKIL Hicham Mohamed

MEBROUKI Mahmoud

Thème

**L'utilisation de l'IA dans les systèmes
embarqués pour le développement des
options automatiques des voitures
intelligentes**

Soutenu publiquement, le 26 / 09 / 2021, devant le jury composé de :

M. Mustapha Anwar BRAHAMI	Maitre de conférences B	ESSA. Tlemcen	Président
M. Hicham MEGNAFI	Maitre de conférences A	ESSA. Tlemcen	Directeur de mémoire
Melle. Imane NEDAJR	Maitre de conférences B	ESSA. Tlemcen	Co-Directeur de mémoire
M. Ghouti ABDELLAOUI	Maitre de conférences B	ESSA. Tlemcen	Examineur 1
M. Zohir KARAOUZANE	Maitre assistant A	ESSA. Tlemcen	Examineur 2

Année universitaire : 2020 / 2021



Mémoire de fin d'étude

Pour l'obtention du diplôme d'Ingénieur

Filière : Automatique

Spécialité : Automatique

Présenté par :

SEKKIL Hicham Mohamed

MEBROUKI Mahmoud

Thème

L'utilisation de l'IA dans les systèmes embarqués pour le développement des options automatiques des voitures intelligentes

Soutenu publiquement, le 26 / 09 / 2021, devant le jury composé de :

M. Mustapha Anwar BRAHAMI	Maître de conférences B	ESSA. Tlemcen	Président
M. Hicham MEGNAFI	Maître de conférences A	ESSA. Tlemcen	Directeur de mémoire
Melle. Imane NEDAJR	Maître de conférences B	ESSA. Tlemcen	Co-Directeur de mémoire
M. Ghouti ABDELLAOUI	Maître de conférences B	ESSA. Tlemcen	Examineur 1
M. Zohir KARAOUZANE	Maître assistant A	ESSA. Tlemcen	Examineur 2

Année universitaire : 2020 / 2021

بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Remerciement

Nous remercions, en premier lieu, Allah, le tout puissant, de nous avoir permis et accorder la volonté, la patience et le courage pour réaliser ce travail.

Au nom d'ALLAH le clément et le miséricordieux.

*Nous tenons tout d'abord à adresser nos plus vifs remerciements au directeur de ce mémoire, monsieur **Hicham MEGNAFI**, pour sa patience, sa disponibilité et surtout ses judicieux conseils, qui ont contribué à alimenter nos réflexions dans l'élaboration de ce mémoire et surtout pour son soutien technique qui est son domaine de créativité.*

*Nous présentons nos remerciements, les plus sincères à mademoiselle **Imane NEDJAR**, codirecteur de ce mémoire, pour son soutien tout au long de nos travaux. Nous tenons à exprimer notre profonde gratitude de nous avoir fait bénéficier de son expérience scientifique avec des qualités humaines indéniables.*

De même, nous souhaitons remercier les différents membres constituant le jury de mémoire pour avoir accepté de donner de leur temps pour évaluer notre travail.

Enfin, nous voudrions remercier nos familles et nos proches pour leurs soutiens sans failles qu'ils nous ont apportés depuis toujours. Nous remercions également tous nos amis et tous ceux qui ont été impliqués d'une manière ou d'une autre dans la réussite de ce travail.

Dédicace

Je dédie ce travail à ceux que j'aime le plus :

À qui je dois tout, pour son soutien, ses conseils, ses sacrifices et tout son amour, que j'admire dans cette vie... Mon très cher père Hocine.

À celle qui m'a toujours accompagné, avec son cœur, son amour, ses prières et toute sa présence, ma lumière dans les obscurités et mon amour... ma précieuse mère Kheira CHERRATI.

À qui je sais qu'elle ne cessera jamais de m'aimer, de se consacrer à moi, et de me donner du courage... Ma sœur adorée Nada.

À qui je tiens la tête haute sur leurs épaules, la source de ma force et de ma confiance... mes frères bien-aimés, Abdeljalil et Abdenour.

À mon trésor dans cette vie... mes deux cousins, Mohamed Labyad et Siraj Fahian.

À ceux sans qui je ne serais pas complet... mes amis, Hicham, Adem, Abdelkarim, Amine, Mohamed, Fayçal, Oussama et Ramzi.

À certaines qui ont une place toute particulière et précieuse... ChahraZad et Sarah.

À mon binôme Mahmoud, pour sa patience et sa compréhension tout au long de ce projet.

À tous ceux qui me sont chers et que j'ai omis de citer.

À tous ceux qui ont contribué de près ou de loin à la réalisation de ce travail.

À vous tous... Merci... D'être vous.

Hicham.

Dédicace

Du fond du cœur je dédie ce travail à

*Mes chers parents que nulle dédicace ne puisse exprimer mes sincères sentiments
et mon éternelle gratitude, pour leur patience illimitée, leur encouragement continu,
leur aide, en témoignage de mon profond amour et respect pour leurs grands sacrifices.*

Mes frères et ma belle-sœur, pour leur amour et leur soutien

Mes chers grands parents et toute ma famille.

Tous mes amis et tous ce qui m'ont aidé de près ou de loin dans la réalisation de ce travail.

Mahmoud.

Résumé

L'objectif de cette étude est d'utiliser les techniques d'intelligence artificielle afin de développer des options automatiques pour les voitures intelligentes. Nous avons développé trois applications différentes telles que le système de détection de la fatigue et de la distraction, le démarreur de voiture à reconnaissance faciale et un système de reconnaissance des panneaux de signalisation. Nous avons utilisé la vision par ordinateur dans le processus de développement pour sa facilité et sa technique non exigeante en termes d'instruments, comme l'apprentissage profond (Deep Learning) qui est la technique la plus précise et en constante évolution pour la vision par ordinateur. Nous avons ensuite implémenté les modèles CNN formés dans un Raspberry Pi pour que le système embarqué puisse être placé dans de vraies voitures et tester les applications en temps réel.

Nous avons expliqué le processus de création de chaque application étape par étape, depuis les informations théoriques et les réalisations de l'état de l'art jusqu'à nos applications, la formation, les tests et la mise en œuvre.

Mots clés : Intelligence Artificielle, Vision par Ordinateur, Apprentissage Profond, CNN, Détection de la Fatigue, Reconnaissance Faciale.

Abstract:

The use of AI in embedded systems for the development of automatic options for smart cars

The objective of this study is to use Artificial Intelligence technics in order to develop smart options for intelligent cars. We developed three different applications such as driver's fatigue and distraction detection system, face recognition car starter and a traffic signs recognition system. We use computer vision in the development process for its easiness and non-instrument demanding technique so as Deep Learning being the ever growing and most accurate technique for computer vision. We then implemented the trained CNN models into a Raspberry Pi for the embedded system to be placed in real cars and test the applications in real-time.

We explain the process for creating each applications step by step from theoretic information and state of the art realizations to our applications, training testing and implementation.

Keywords: Artificial Intelligence, Computer Vision, Deep Learning, CNNs, Fatigue detection, Face Recognition.

ملخص:

هذه الدراسة تهدف إلى استخدام تكنولوجيا الذكاء الاصطناعي لتطوير تطبيقات ذكية لسيارات ذكية. طورنا ثلاث تطبيقات مختلفة مثل نظام رصد التعب وتشتت انتباه السائق، نظام تشغيل السيارة بالتعرف على الوجوه ونظام التعرف على إشارات المرور. استخدمنا الرؤية الحاسوبية في عملية التطوير لسهولة التي لا تتطلب استخدام أجهزة مكلفة ومشتتة. كما أن تقنية التعلم العميق هي تقنية دائمة التطور والأكثر دقة لرؤية الحاسوب. بعد ذلك قمنا بتطبيق نماذج سي إن إن المدربة في راسبري باي للنظام المضمن ليتم وضعه في السيارات الحقيقية واختبار التطبيقات في الوقت الحقيقي.

شرحنا عملية إنشاء كل تطبيق خطوة بخطوة بدءاً من المعلومات النظرية ومروراً بالجوانب التقنية، وصولاً إلى تطبيقاتنا واختبارات التدريب وتطبيقاتها.

الكلمات المفتاحية: الذكاء الاصطناعي، رؤية الحاسوب، التعلم العميق، شبكة الخلايا العصبية التلافيفية، رصد التعب، التعرف على الوجه.

Table des matières

Liste des acronymes	xi
Table des figures	xiii
Introduction Générale	1
Chapitre 1 : Introduction aux systèmes avancés d'aide à la conduite	3
1.1 Introduction.....	3
1.2 Systèmes avancés d'aide à la conduite	4
1.2.1 Définition.....	4
1.2.2 Évolution des systèmes d'aide à la conduite.....	5
1.2.3 Niveaux d'automatisation de la conduite.....	9
1.2.4 Les systèmes d'aide à la conduite les plus courants	10
A. Aide au stationnement	11
B. L'assistant à la direction et au guidage	11
C. Avertissement de changement de voie.....	12
D. L'assistant au freinage d'urgence.....	12
E. Régulateur de vitesse (adaptatif)	12
F. Détecteur de fatigue.....	13
G. La reconnaissance des panneaux de signalisation.....	13
1.2.5 Les Risques associés aux systèmes avancés d'aide à la conduite	13
1.2.6 Système d'aide à la conduite proposé.....	15
1.3 Conclusion	15
Chapitre 2 : État de l'art des systèmes d'assistance à la conduite.....	16
2.1 Introduction.....	16
2.2 Reconnaissance de visages.....	16

2.3	Module de détection de fatigue du conducteur.....	17
2.3.1	Analyse de d'état des yeux.....	18
2.3.2	Bâillements fréquents	21
2.4	Module de reconnaissance des panneaux de signalisation	22
2.4.1	Localisation et la détection des panneaux routiers	23
A.	Modélisation colorimétrique	23
B.	Modélisation géométrique.....	24
C.	Méthodes avec apprentissage	25
2.4.2	Classification	27
A.	Méthodes d'apprentissage basées sur des caractéristiques extraits manuellement	27
B.	Méthodes d'apprentissage profond (Deep Learning)	28
2.5	Systèmes embarqués	29
2.5.1	Définition.....	30
2.5.2	Caractéristique.....	30
A.	Spécification de domaine d'application.....	30
B.	Temps-réel	30
C.	Réactivité	31
D.	Criticité, Fiabilité	31
E.	L'autonomie	31
F.	La consommation énergétique.....	32
2.5.3	Architecture d'un système embarqué	32
2.5.4	Les différentes applications des systèmes embarquées	33
A.	L'utilisation des systèmes embarqués dans les réseaux cellulaires (IOT)	33
B.	L'émergence de l'intelligence artificiel dans les systèmes embarqués	36
2.6	Conclusion	37

Chapitre 3 : L'apprentissage profond.....	38
3.1 Introduction.....	38
3.2 Définition.....	39
3.3 Historique.....	39
3.4 Réseaux de neurones artificiels.....	40
3.4.1 Modélisation d'un neurone artificiel	41
3.4.2 Principe de fonctionnement.....	42
3.4.3 Développement d'un réseau de neurones.....	42
A. Choix d'une base de données	42
B. Choix d'une architecture adéquate.....	43
C. Apprentissage du réseau :	43
3.4.4 Les couches des réseaux neuronaux	43
Types de couches fréquemment utilisées	44
3.4.5 Les types des réseaux de neurones.....	44
A. Réseaux de neurones monocouche (perceptron simple)	44
B. Réseaux de neurones multicouches (perceptron multicouches MLP)	45
C. Réseaux de neurones récurrents (<i>Recurrent Neural Network-RNN</i>)	45
D. Réseaux de neurones à connexions locales	46
3.5 Réseaux de neurones convolutifs.....	47
3.5.1 Le principe de fonctionnement	47
3.5.2 Les couches du réseau de neurone convolutif	48
A. Couche de convolution.....	49
B. Couche de Pooling.....	50
C. Couche entièrement connectée.....	51
3.5.3 Les fonctions d'activation	51

A.	ReLU (Rectified Linear Units) :.....	51
B.	Sigmoid :.....	52
C.	SoftMax :.....	52
3.5.4	Les architectures du réseau de neurones les plus courants.....	53
A.	VGG.....	53
D.	ResNet	54
E.	Inception- V1	54
F.	Xception.....	54
G.	MobileNet-v1.....	55
3.6	Conclusion	55
Chapitre 4 : Résultats Expérimentaux et Réalisation.....		56
4.1	Introduction.....	56
4.2	Outils utilisés.....	57
4.2.1	Hardware	57
A.	Raspberry Pi 3 Model B+.....	57
B.	Pi Camera V2.....	60
C.	Afficheur LCD I2C 20*4.....	61
D.	Autres Composants.....	62
E.	Schéma de câblage.....	63
4.2.2	Software	63
4.3	Les applications du système.....	65
4.3.1	Démarrreur avec reconnaissance faciale	66
A.	Préparation des données	67
B.	Entraînement du modèle.....	68
C.	Test et implémentation.....	69

4.3.2	Système de détection de la fatigue	72
A.	Préparation des données	72
B.	Entraînement du modèle.....	74
C.	Test et implémentation.....	75
D.	Système de détection des distractions.....	77
4.3.3	Détecteur de panneaux de signalisation	79
A.	Préparation des données	80
B.	Entraînement du modèle.....	84
C.	Test et implémentation.....	87
4.4	Conclusion	90
Conclusion Générale		91
Références.....		93

Liste des acronymes

IA	Intelligence artificiel
OMS	Organisation mondiale de la santé
ONISR	Observatoire National Interministériel de la Sécurité Routière
ADAS	Advanced Driver Assistance Systems
NCAP	New Car Assesement Program
ABS	Anti-lock Braking System
ESP	Electronic Stability Program
ACC	Adaptative Cruise Control
AEB	Automatic emergency braking
HLDI	Highway Loss Data Institute
SAE	Society of Automotive Engineers
SVM	Support Vecteur Machine
CNN	Convolutional Neural Network
MLP	Multi Layer Perceptron
LDA	Linear Discriminant Analysis
EAR	Eye Aspect Ratio
MAR	Mouth Aspect Ratio
PERCLOS	PERcentage of eye CLOSure
FOM	Frequency Of Mouth
EEG	Electroencéphalogramme
HOG	Histogram of Oriented Gradients
ACF	Aggregate Channel Features

MPP	Max pooling positions
RNN	Recurrent Neural Network
ANN	Artificial Neural Network
ReLU	Rectified Linear Unit
RFID	Radio frequency Identification
PDA	Personal Digital Assistant
IOT	Internet Of Things
IIOT	Internet des objets industriels
RTOS	Real Time Operating System

Table des figures

Figure 1-1 : Quelques systèmes d'assistance à la conduite [3].....	5
Figure 1-2 : Evolution du nombre de décès sur les routes françaises depuis 1960, associée à la disponibilité de systèmes d'aide à la conduite [14].....	8
Figure 1-3 : Les niveaux d'automatisation de la conduite [16].....	9
Figure 2-1 : Architecture d'un système embarqué.....	32
Figure 3-1 : L'Analogie Entre le Neurone Biologique et le Neurone Artificiel.....	41
Figure 3-2 : Modèle d'un neurone artificiel.....	42
Figure 3-3 : Schéma de perceptron simple.....	45
Figure 3-4 : Schéma de perceptron multicouches.....	45
Figure 3-5 : Schéma de Réseaux de neurones récurrents.....	46
Figure 3-6 : Schéma de Réseaux de neurones à connexions locales.....	46
Figure 3-7 : Schéma d'un réseau de neurone convolutif.....	48
Figure 3-8 : Schéma d'une couche de convolution avec un noyau de taille 3*3 et un pas de 2.....	49
Figure 3-9 : Illustration d'une opération de convolution.....	50
Figure 3-10 : Fonctionnement d'une opération Max pooling de taille 2*2 et d'un pas de 2.....	50
Figure 3-11 : La fonction d'activation ReLu.....	51
Figure 3-12 : La fonction d'activation Sigmoid.....	52
Figure 3-13 : La fonction d'activation SoftMax.....	53
Figure 4-2 : Les ports de Raspberry Pi 3 model B+.....	58
Figure 4-3 : Les différents pins de Raspberry Pi et ses nombres.....	60
Figure 4-4 : Raspberry Pi Camera Module v2.....	60
Figure 4-5 : Spécifications et taille du module d'affichage LCD I2C 20x4 avec sac à dos.....	62
Figure 4-6 : Schéma de câblage du system réel et sous le logiciel Fritzing.....	63
Figure 4-7 : Logos des outils utilisés dans cette étude.....	65
Figure 4-8 : Organigramme de processus de démarreur avec reconnaissance faciale.....	66
Figure 4-9 : Exemple des photos d'entraînement augmentée avec les fonctions utilisées sur la base de données.....	68
Figure 4-10 : Les courbes de précision et d'erreur du MobileNetV2 pendant 2 époques d'apprentissage.....	69

Figure 4-11 : Reconnaissance faciale avec l'architecture de CNN MobileNetV2 avec une personne non-reconnue.	70
Figure 4-12 : Circuit de démarreur de la voiture commandé par le Raspberry Pi.	71
Figure 4-13 : Exemples de données dans YawDD.....	72
Figure 4-14 : Prétraitement des données, redimensionnement et mise à l'échelle des gris.....	73
Figure 4-15 : La détection de l'état des yeux et de la bouche avec EAR et MAR.....	73
Figure 4-16 : Les courbes de précision et l'erreur du modèle MobileNetV2 pendant 10 époques d'apprentissage pour yeux et de bouche à gauche et à droite respectivement.	74
Figure 4-17 : Un exemple des images classifiées correctement avec le model MobileNetV2.....	75
Figure 4-18 : L'organigramme de décision de la détection de la fatigue.	76
Figure 4-19 : Comparaison entre le détecteur de visage frontal de dlib et Haar.....	78
Figure 4-20 : Distraction du conducteur détectée avec le message d'alerte à l'écran.	79
Figure 4-21 : Les deux étapes de la détection des panneaux de signalisation.....	79
Figure 4-22 : Le processus de l'application de classification des panneaux de signalisation.	80
Figure 4-23 : Un exemple d'image de chacune des 43 classes de l'ensemble de données.	81
Figure 4-24 : La étiquettes et le nombre de chaque classe dans la base de données.....	82
Figure 4-25 : Exemple d'une image prétraitée.	83
Figure 4-26 : Exemple d'une photo augmentée avec les fonctions utilisées sur la base de données.	83
Figure 4-27 : Résumé des couches du réseau de neurones.	85
Figure 4-28 : Les courbes de précision et d'erreur du modèle pendant 5 époques d'apprentissage.	86
Figure 4-29 : Le processus d'alimentation du modèle en images à classer.....	87
Figure 4-30 : Exemples de différents panneaux de signalisation classés.....	88
Figure 4-31 : L'étiquette du panneau de signalisation prédite affichée à l'écran.....	89
Figure 4-32 : Un exemple de la simulation faite.....	90

Introduction Générale

L'intelligence a toujours été un concept passionnant (mais peu clair) pour l'homme. Pendant de nombreuses années, il a essayé de la simuler dans des machines et de recréer une intelligence artificielle. En effet, les scientifiques ont réussi à le faire en quelque sorte, même si ces machines ne peuvent pas encore exprimer une réelle intelligence, elles peuvent néanmoins accomplir des tâches impressionnantes et étonnantes, de l'automatisation d'usines entières à la détection de maladies, en passant par les robots et les voitures autonomes, et ainsi de suite, partout dans nos vies.

Un domaine qui a récemment adopté l'IA et qui a connu une croissance spectaculaire au cours des dernières années est l'industrie automobile et le développement de voitures intelligentes pour automatiser et faciliter de nombreuses tâches pour le conducteur afin d'améliorer l'expérience de conduite.

Dans ce projet, nous visons à développer un système intelligent en utilisant l'IA et les systèmes embarqués pour développer des applications intelligentes pour les voitures. Ce mémoire va discuter du concept de ces systèmes, de l'IA en détails et de nos résultats expérimentaux avec le projet final.

Ce travail est consacré à :

- Mettre en place un environnement d'apprentissage profond pour créer des applications d'IA.
- Apprenez à préparer des ensembles de données convenant à différents réseaux neuronaux.
- Obtenez une meilleure compréhension et manipulation des systèmes embarqués.
- Implémenter des modèles d'apprentissage profond dans un Raspberry Pi.

- Utiliser ces connaissances pour ajouter des fonctionnalités intelligentes aux automobiles.

Notre plan de travail est donc le suivant :

- **Chapitre 1 :**

Ce chapitre sert d'introduction aux systèmes d'aide à la conduite, en plus de notre système à développer.

- **Chapitre 2 :**

Dédié à la discussion de l'état de l'art pour chaque application parmi celles que nous développerons dans ce projet, afin de savoir quelles méthodes sont les meilleures et d'argumenter notre choix.

- **Chapitre 3 :**

Ce chapitre comportera une description détaillée de l'intelligence artificielle, de l'apprentissage profond et des réseaux neuronaux convolutifs.

- **Chapitre 4 :**

Le noyau de ce mémoire où nous présentons notre travail, les outils utilisés (matériels et logiciels), le processus de développement, les résultats expérimentaux et l'implémentation dans le système embarqué.

Chapitre 1

Introduction aux systèmes avancés d'aide à la conduite

1.1 Introduction

Le nombre d'usagers de la route a considérablement augmenté au cours des dernières décennies. De ce fait, de nombreux défis se sont imposés concernant la sécurité routière afin d'éviter le grand nombre de pertes humaines et matérielles résultant d'accidents de la circulation. L'Organisation mondiale de la santé, dans son rapport sur la sécurité routière, a constaté que les accidents de la route entraînent environ 1,25 million de décès par an [1]. De plus, c'est la première cause de décès chez les jeunes de 15 à 29 ans.

Par ailleurs, en prenant en compte les coûts médicaux et sociétaux (premiers secours, convalescence, etc.), les coûts matériels (dommages des véhicules, etc.), les frais généraux (frais d'expertise, de justice, etc.) mais aussi la perte de production future des victimes, une estimation du coût d'un tué sur la route a été donnée par l'Observatoire National Interministériel de la Sécurité Routière (ONISR). Celle-ci s'élevait à 1,2 million d'euro approximativement en 2011 [2]. Au total, le coût direct de l'insécurité routière en France a été estimé à 23

MD€ pendant la même année. De ce fait, l'insécurité routière représente un cout qui pèse lourdement sur la collectivité. Pour toutes ces raisons, la sécurité routière est une priorité en matière de santé publique en Europe.

Grâce aux nombreuses avancées technologiques et aux nouvelles normes de sécurité visant à rendre la conduite de plus en plus sûr, les constructeurs automobiles ont développé bon nombre de solutions ayant pour objectif l'amélioration de la sécurité routière. De ces solutions on s'intéresse aux systèmes d'aide à la conduite (*Advanced Driver Assistance System : ADAS*).

1.2 Systèmes avancés d'aide à la conduite

Aujourd'hui, de nombreux systèmes avancés d'aide à la conduite (ADAS) ont la capacité d'assister le conducteur de diverses manières pour réduire la fatigue au volant et améliorer la sécurité.

L'émergence de ces nouvelles technologies est principalement due aux progrès réalisés dans de nombreux domaines, notamment : l'électronique, les communications, le traitement du signal et de l'image, l'automatisation, ainsi que La baisse des coûts de l'électronique (capteurs, réseaux de télécommunication, ordinateurs), il est possible d'intégrer des fonctions de plus en plus avancées dans les véhicules.

1.2.1 Définition

Les ADAS sont des systèmes conçus pour être intégrés dans les véhicules afin de renforcer la quantité et la qualité des informations transmises au conducteur et de l'assister voire se substituer à lui lorsqu'il se met dans des situations potentiellement accidentogènes.

La figure 1.1 montre quelques exemples des systèmes avancés d'aide à la conduite.



Figure 1-1 : Quelques systèmes d'assistance à la conduite [3].

1.2.2 Évolution des systèmes d'aide à la conduite

Les constructeurs sont dans un premier temps intervenus en matière de sécurité secondaire qui a pour but de réduire les dommages causés aux passagers du véhicule lors d'un accident. Des progrès ont été enregistrés en matière de résistance, ainsi que d'absorption du choc de la caisse, de même que des coussins gonflables (airbags) ont été intégrés aux véhicules. Le développement de ces équipements a été encouragé par la création en 1997 de l'Euro NCAP (en anglais « European New Car Assessment Program », programme d'évaluation européenne des automobiles). Cet organisme indépendant a pour fonction principale d'effectuer des essais de choc (crash tests) afin de fournir aux consommateurs une évaluation de la sûreté des voitures les plus vendues en Europe.

Les constructeurs sont ensuite intervenus sur la sécurité primaire (dite préventive ou active) qui regroupe l'ensemble des équipements destinés à éviter qu'un accident se produise. C'est à cette catégorie qu'appartiennent actuellement la plupart des systèmes d'aide à la conduite. L'arrivée de l'électronique dans l'automobile a permis d'ouvrir de nouvelles possibilités en la matière.

En 1978 Bosch introduit le système d'anti blocage des freins ABS (Anti-lock Braking System) qui permet de conserver le contrôle directionnel du véhicule en empêchant les roues de bloquer lors d'un freinage intensif [4].

L'objectif de l'ABS est de maintenir le pouvoir directionnel du véhicule lors du freinage. L'ABS se démocratise ensuite progressivement avant de devenir obligatoire sur tous les véhicules neufs de grande série commercialisés en Europe à partir de 2004.

Quelques années plus tard, l'introduction Le programme électronique de stabilisation ESP (Electronic Stability Program) a marqué un grand pas en avant pour le développement des ADAS.

C'est un système électronique antidérapage qui corrige la trajectoire en agissant sur le système de freinage ainsi que sur le couple moteur [5].

Ainsi, il détecte les éventuels dérapages du véhicule et aide le conducteur à garder le contrôle en appliquant des forces de freinage sur une ou plusieurs roues en exploitant la fonction ABS. S'il détecte une trajectoire anormale, il s'enclenche automatiquement sans intervention du conducteur en utilisant le système de freinage pour ramener la voiture sur sa trajectoire.

Suite à son introduction dans le marché en 1995 sur les Mercedes Classe S et BMW série 7, l'analyse des données d'accidentologie dans différents pays a démontré l'apport considérable en termes de sécurité routière [6] [7].

L'année 1996 a vu l'émergence de l'assistance au freinage d'urgence qu'est conçu comme un complément de l'ABS, clairement pour compenser l'incompétence du conducteur. En effet, il déclenche automatiquement la pleine puissance de freinage lorsque le conducteur enfonce rapidement la pédale de frein et se restreint ensuite, de peur de bloquer une roue. L'électronique analyse instantanément l'intention et supplée ensuite le conducteur

trop timoré, incapable d'assurer un freinage maximal jusqu'au bout. Le but est ici de raccourcir la distance d'arrêt pour éviter un accident ou en réduire la gravité.

Les premiers systèmes d'assistance au parking ont investi le marché au milieu des années 90s [8]. A l'époque ils se basaient uniquement sur des capteurs ultrason de proximité pour détecter les obstacles entourant le véhicule. Initialement, ces systèmes s'enclenchaient en marche arrière et émettaient une alerte à l'approche d'un obstacle. Plus tard, une caméra de recul a été introduite afin de faciliter encore plus la tâche de stationnement.

Le régulateur de vitesse adaptatif (ou ACC pour Adaptive Cruise Control) est un dispositif évolué de régulation de vitesse qui permet de conserver une distance préétablie avec le véhicule qui le précède. Un capteur installé à l'avant du véhicule mesure la distance et la vitesse relative du véhicule qui précède. Mercedes-Benz a été le premier constructeur à proposer un régulateur du genre, sur sa Classe S d'année-modèle 1999 [9].

L'arrivée des systèmes de navigation portables autour des années 2004 par exemple, a eu un impact sur la sécurité. En aidant le conducteur à s'orienter, ces dispositifs ont le potentiel de réduire sa charge de travail et lui permettent de se concentrer sur la tâche primaire de conduite ce qui réduit le risque d'accidents liés à l'inattention [10].

Le premier système de freinage d'urgence autonome AEB (Automatic emergency braking) fait son apparition sur le Volvo XC 60 en 2008 [11]. Comme son nom l'indique, ce dispositif peut déclencher automatiquement l'action de freinage lorsque ses capteurs détectent un véhicule, un piéton ou un autre obstacle.

Aux États-Unis, le HLDI (Highway Loss Data Institute HLDI) a comparé les taux de réclamations d'assurance par année de véhicule assuré pour les véhicules Volvo S60 et XC60 avec AEB à basse vitesse standard avec des véhicules comparables sans AEB. Les Volvo avec AEB à basse vitesse ont enregistré 18% moins de réclamations pour collision, qui couvraient les dommages au véhicule du conducteur responsable [12].

Dans une étude similaire au Royaume-Uni, les modèles Volvo XC60 équipés d'un AEB à basse vitesse standard ont enregistré 6 % de sinistres en moins pour leurs propres dommages [13].

The AEB system, after being tested in China in a realistic scenario and in different weather, light and speed conditions, proved that it can reduce deaths by 13.2%, and reduce injuries by 9.1%. Based on China policy, the market penetration rate of smart vehicles equipped with AEB is expected to be 34.0% in 2025 and 60.3% in 2030 [11].

Toutes ces avancées ont un impact sur les chiffres de la sécurité routière. Par exemple, Une étude réalisée par Page et al en 2011 estime que sur la période 2000-2010, les avancées en matière de sécurité ont contribué à sauver 16% des victimes passagers et 11% de l'ensemble des victimes (passagers + piétons). [2]

La figure 1.2 montre l'évolution du nombre de décès sur les routes françaises depuis 1960, associée à la disponibilité de systèmes d'aide à la conduite.

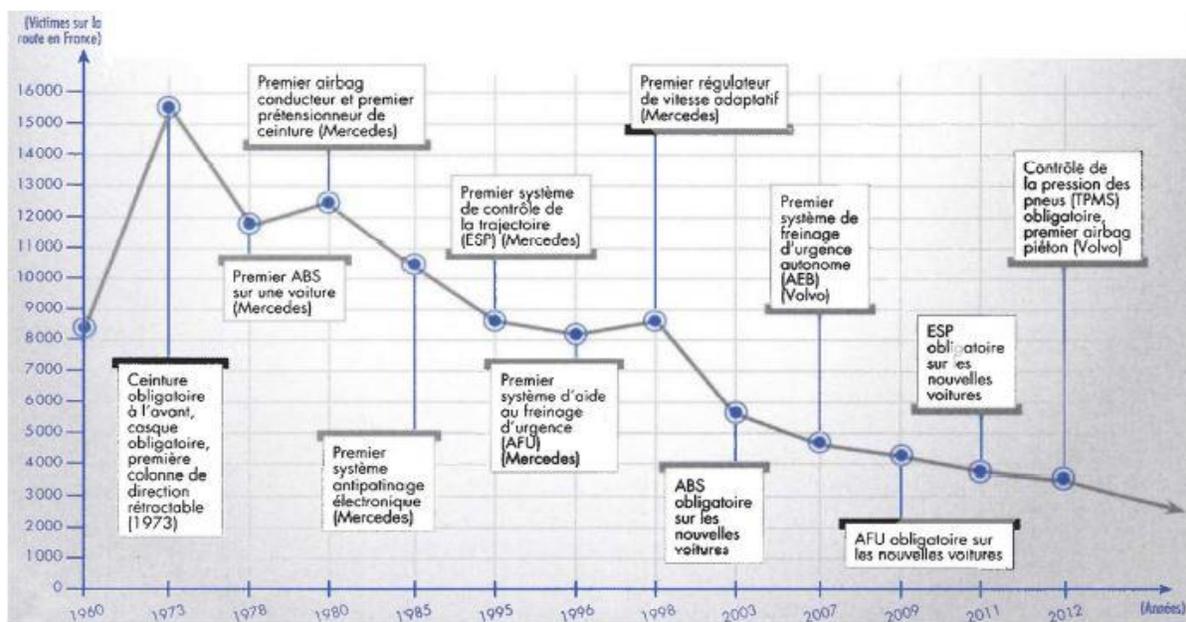


Figure 1-2 : Evolution du nombre de décès sur les routes françaises depuis 1960, associée à la disponibilité de systèmes d'aide à la conduite [14].

1.2.3 Niveaux d'automatisation de la conduite

Il est naturel de penser que tous les travaux de recherche et développement investis jusqu'à présent aboutiront à un système de conduite entièrement autonome. Dans le même temps, la définition de niveaux intermédiaires d'automatisation est devenue indispensable pour répondre aux besoins de coordination de tous les développements, fournissant ainsi un cadre pour la spécification et les exigences techniques du système d'automatisation.

La Society of Automotive Engineers (SAE International) a publié en 2014 une classification en 5 niveaux d'automatisation de la conduite, le niveau 0 correspondant à une conduite complètement manuelle et le niveau 5 à un rêve idéal où le véhicule serait capable de naviguer de façon complètement autonome pour toutes les missions et dans tous les environnements [15].

La figure suivante indique les 5 niveaux d'automatisation de la conduite :

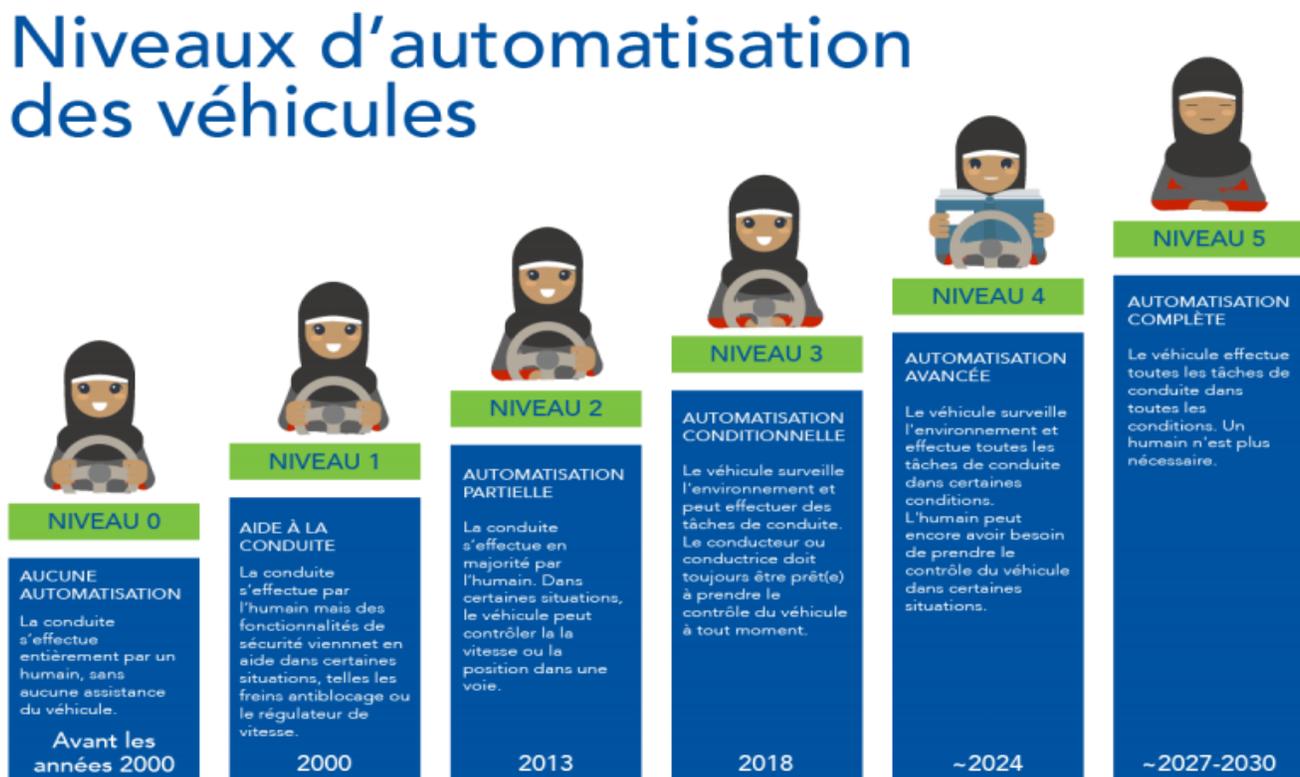


Figure 1-3 : Les niveaux d'automatisation de la conduite [16].

On va expliquer ces différents niveaux :

- **Niveau 0** : le conducteur humain opère seul le véhicule.
- **Niveau 1** : le système automatisé du véhicule assiste de temps en temps le conducteur humain dans certaines tâches de la conduite.
- **Niveau 2** : le système automatisé du véhicule peut exécuter seul certaines tâches sous le contrôle du conducteur.
- **Niveau 3** : le système automatisé peut aussi bien exécuter seul certaines tâches et assurer seul le contrôle de son environnement. Mais le conducteur doit être en mesure de reprendre le contrôle du véhicule à tout moment.
- **Niveau 4** : le système automatisé peut aussi bien exécuter certaines tâches et assurer le contrôle de son environnement sans que le conducteur n'ait besoin de reprendre le contrôle du véhicule.
- **Niveau 5** : le système automatisé peut effectuer toutes les tâches que l'humain peut réaliser, dans n'importe quelles circonstances.

1.2.4 Les systèmes d'aide à la conduite les plus courants

Il est impossible d'imaginer une voiture moderne sans système d'aide à la conduite. Dans un avenir proche, ils évolueront vers des systèmes complexes qui transféreront de plus en plus les tâches et les responsabilités de conduite aux véhicules. Selon le système et l'état de développement, la participation active des conducteurs deviendra de moins en moins, et ne sera finalement plus nécessaire. Nous vous présentons ici les systèmes d'aide à la conduite les plus courants mais aussi les plus innovants, et vous expliquons à quelles situations chaque système est adapté.

A. Aide au stationnement

Les aides au stationnement peuvent vous aider à manœuvrer, vous garer et quitter le parking et vous protéger des dommages causés par le stationnement. L'assistant utilise les informations des capteurs à ultrasons et de caméra. Les assistants de stationnement utilisent généralement des bourdonnements pour indiquer la distance par rapport aux objets à proximité afin d'aider le conducteur. La caméra de recul est un peu plus détaillée : elle montre en plus la zone derrière la voiture et complète la ligne recommandée comme direction lors des manœuvres de stationnement.

À l'avenir, le système d'aide à la conduite sera entièrement responsable du stationnement et effectuera indépendamment toutes les opérations de direction, de freinage et d'accélération nécessaires. Avec ce type de système automatisé, l'assistant peut conduire votre voiture dans un espace de stationnement exigu en appuyant sur un bouton et la retirer à nouveau si nécessaire.

B. L'assistant à la direction et au guidage

Malheureusement, les embouteillages, la circulation lente et les voies étroites sont monnaie courante, surtout sur les autoroutes. Cela signifie que pendant une longue période, vous devez garder une main ferme et conduire patiemment le long de la route. L'assistant de direction et de guidage peut vous aider dans cette situation - mais aussi à grande vitesse - à guider le véhicule vers le centre de la voie, sans avoir à toucher le volant.

La caméra enregistre les marquages au sol, de sorte que le système d'aide à la conduite se localise en fonction des limites de sa propre voie.

C. Avertissement de changement de voie

L'avertissement de sortie de voie peut vous aider à changer de voie en toute sécurité et vous aide à regarder dans le rétroviseur et à regarder par-dessus l'épaule. Les capteurs surveillent la zone à côté et derrière la voiture et couvrent les angles morts notoires. Si une autre voiture s'approche de votre véhicule dans un angle mort ou s'approche dangereusement et que le conducteur la rate, le système d'aide à la conduite le reconnaîtra et vous avertira de changer de voie. Par exemple, de tels avertissements peuvent être émis par des symboles clignotants dans le rétroviseur, ou même par des vibrations du volant ou des impulsions de direction.

D. L'assistant au freinage d'urgence

L'assistance au freinage avertit le conducteur en cas de danger, c'est-à-dire avant qu'une collision ne se produise. Des capteurs mesurent la vitesse actuelle et la distance par rapport au véhicule qui vous précède ou à l'obstacle. Le système émet des affichages visuels ainsi que des signaux d'avertissement sonores.

Avant le freinage d'urgence, le système d'aide à la conduite calcule la force de freinage optimale et la pression de freinage requise. Selon le système, cette pression calculée indépendamment est traduite en freinage automatique par l'assistant ou relâchée uniquement lorsque la pédale de frein est enfoncée.

E. Régulateur de vitesse (adaptatif)

Le régulateur de vitesse vous permet de sélectionner la vitesse de croisière souhaitée. Cela se fait souvent en combinaison avec le régulateur de distance. Une fois activé, ce système d'aide à la conduite maintient en permanence la vitesse que vous avez sélectionnée, offrant un confort accru sur les longs trajets. Le régulateur de vitesse peut adapter la vitesse de manière électronique à l'environnement et corriger la puissance du moteur en conséquence.

F. Détecteur de fatigue

Ce système d'aide à la conduite permet au conducteur de conduire, de détecter simultanément tout signe de fatigue et d'émettre des avertissements pour l'avertir de faire attention.

Cela se fait d'une part à l'aide d'une caméra installée en haut du pare-brise qui filme la route. Un ordinateur de bord analyse les images en temps réel pour identifier tout comportement qui pourrait indiquer des signes potentiels de fatigue du conducteur, comme une déviation des marquages au sol ou une perte de vitesse. D'autre part, la caméra Infra, qui analyse le visage du conducteur et tente de détecter tout signe de fatigue, comme les bâillements fréquents ou la fermeture ponctuelle des yeux.

Si l'un des signes de fatigue mentionnés apparaît, le système alerte le conducteur par des vibrations dans le volant ou des signaux visuels ou sonores.

G. La reconnaissance des panneaux de signalisation

Lorsque vous êtes coincé dans un trafic très lent, il peut souvent être difficile de prêter pleinement attention à toutes les situations. Planification d'itinéraire, trafic à venir, limites de vitesse et ce qui se passe à côté et derrière vous. Ce qui pose problème au conducteur car il doit garder toutes ces choses à l'esprit.

L'une des solutions proposées à ce problème est le système de reconnaissance des panneaux de signalisation, qui aide à soulager la pression sur le conducteur en utilisant un système de caméras qui scanne les feux de circulation et affiche les données traitées sur l'écran.

1.2.5 Les Risques associés aux systèmes avancés d'aide à la conduite

Avant d'utiliser toute nouvelle technologie, il doit passer un test pour connaître l'étendue de son application en plus de connaître les risques potentiels et leur impact sur les utilisateurs. Il n'en va pas autrement dans les

systèmes d'aide à la conduite. Cependant, le risque est nul vis-à-vis de ces systèmes du fait de la complexité des situations de conduite à traiter ainsi que des objectifs de coûts de production limités et des cycles de développement de plus en plus courts imposés par le marché. La concurrence féroce entre les constructeurs automobiles exige que les nouveaux modèles automobiles soient mis sur le marché dans des délais de plus en plus courts, laissant peu de temps aux fournisseurs automobiles pour les étapes de recherche, développement et commercialisation.

L'énorme complexité de l'architecture électronique globale du véhicule due à l'intégration croissante des systèmes électroniques (gestion moteur, habitacle, etc.) est l'une des principales raisons pouvant présenter un risque pour le conducteur. Du fait des éventuels dysfonctionnements pouvant résulter de l'interaction entre ces systèmes électroniques, la difficulté de ces défauts réside dans la difficulté de les prévoir lors de la phase de développement. Certains dysfonctionnements peuvent être accidentels et la trace du problème n'est pas enregistrée dans la mémoire de l'ordinateur. C'est lorsque le dysfonctionnement n'est pas lié au système d'aide à la conduite. Dans d'autres cas, une série d'actions qui n'étaient pas nécessairement prévues par l'algorithme de gestion peut créer un conflit qui provoque une panne difficilement détectable.

Face à ces problèmes et enjeux de sécurité, chercheurs et ingénieurs développent des solutions innovantes capables d'améliorer les systèmes d'aide à la conduite afin d'éviter ces problèmes, par exemple des outils de simulation numérique (un environnement simulé qui permet de reproduire le comportement d'un membre ou l'ensemble du véhicule et son interaction avec la route et l'environnement du conducteur) est l'une des solutions innovantes. Il peut alors se jouer en simulant diverses situations de conduite pour mettre en évidence d'éventuels dysfonctionnements du système ou de son environnement.

L'intérêt croissant des programmeurs et développeurs pour la partie logicielle des systèmes d'aide à la conduite ainsi que l'intégration d'outils de développement inspirés de ceux déjà utilisés dans les industries aéronautique et ferroviaire dans l'industrie automobile contribue à valider et fiabiliser les logiciels embarqués dans ces systèmes.

1.2.6 Système d'aide à la conduite proposé

Sur la base des avancées technologiques en matière d'assistance à la conduite. Nous vous proposerons un système d'aide à la conduite qui contient trois sous-systèmes : un système de reconnaissance faciale du conducteur et un système de détection de la fatigue du conducteur ainsi qu'un système pour la classification des panneaux de signalisation

1.3 Conclusion

Nous avons présenté dans ce chapitre une introduction aux systèmes avancés d'aide à la conduite « ADAS » et leur taxonomie. Ce chapitre a décrit les étapes d'évolution des systèmes avancés d'aide à la conduite qui ont pavé le chemin vers les systèmes d'automatisation de la conduite qu'on connaît aujourd'hui.

Chapitre 2

État de l'art des systèmes d'assistance à la conduite

2.1 Introduction

L'objectif principal de ce chapitre est d'établir une vue d'ensemble du processus de chaque module. L'état de l'art fournira une bonne base de discussion tout au long de ce chapitre. Dans ce chapitre nous allons présenter l'état de l'art sur les trois modules : de classification des panneaux de signalisation, de détection de fatigue du conducteur automobile et de reconnaissance faciale du conducteur.

2.2 Reconnaissance de visages

Plusieurs méthodes de reconnaissance de visages ont été proposées ces dernières années, suivant deux grands axes : la reconnaissance à partir d'images fixes et la reconnaissance à partir de séquence d'images (vidéo).

Les approches basées sur l'IA utilisent des techniques des réseau neurones et d'apprentissage automatique, telles que SVM, CNN, etc.

H. Boughrara et al, ont présenté un algorithme d'entraînement constructif modifié pour le Perceptron multicouche (MLP) [17]. Une nouvelle procédure de formation a été proposée dans laquelle les modèles de formation ont été appris de manière incrémentielle. Cet algorithme a commencé avec un petit nombre de modèles d'apprentissage et une seule couche cachée avec un nombre limité de neurones. Au cours de l'entraînement, les neurones cachés ont augmenté lorsque le seuil d'erreur quadratique moyenne (MSE) des données d'entraînement n'a pas pu être réduit à une valeur prédéfinie. De plus, les modèles d'entrée ont été entraînés de manière incrémentielle jusqu'à ce que tous les modèles de données d'entraînement soient appris. Les expériences menées sur la base de données d'expression faciale de Cohn-Kanade ont révélé une précision de 99,23 %.

L'algorithme développé par Stuhlsatz et al était une généralisation de l'analyse discriminante linéaire (*Linear Discriminant Analysis LDA*) classique utilisant réseaux de neurones profonds [18]. La dimensionnalité intrinsèque de l'espace des caractéristiques dans le schéma proposé était limitée par le nombre de classes et la fonction discriminante optimale. Cependant, les transformations linéaires étaient insuffisantes pour extraire des caractéristiques discriminantes optimales à partir de mesures arbitrairement distribuées. Les expériences menées sur l'ensemble de données MNIST ont révélé une précision de 96,83 %.

2.3 Module de détection de fatigue du conducteur

La baisse de vigilance associée à la fatigue amène le conducteur à avoir un comportement physique observable. Ces activités comportementales associées à la conduite peuvent être considérées comme des indicateurs permettant de détecter la fatigue du conducteur.

Ces activités augmentent progressivement avec la durée de la tâche, et aussi avec la monotonie, car elles sont des mécanismes de défense du sujet qui lutte pour rester éveillé. Cependant, elles ne sont pas toujours efficaces

et, au-delà d'un certain seuil de fatigue, plus la performance se dégrade plus ces activités comportementales croissent.

Néanmoins, la perception de ces activités comportementales pour la conception d'un système embarqué est difficile à réaliser. Le problème principal est l'accès à des informations caractéristiques de ces activités comportementales.

Actuellement, la seule activité comportementale abordée dans la conception du système en temps réel est l'analyse faciale qui comprend l'analyse de l'état des yeux, les bâillements fréquents ainsi que les mouvements de la tête.

2.3.1 Analyse de d'état des yeux

Beaucoup d'efforts ont été rapportés dans la littérature sur le développement des systèmes de surveillance de la somnolence/fatigue non-intrusifs basés sur la vision.

Horng et al présentent un système qui utilise un modèle de couleur de peau dans l'espace de couleur HSI pour la détection de visage [19]. Les auteurs utilisent l'information des contours pour la localisation des yeux, et la mise en correspondance dynamique pour le suivi des yeux. L'état des yeux est défini en utilisant l'information de couleur de l'œil. Et à partir l'état des yeux Ils ont réussi à déduire l'état du conducteur. Si les yeux sont fermés pendant cinq frames consécutifs, Le conducteur est supposé être fatigué.

D'Orazio et al Ils ont imposé un algorithme de détection de l'œil qui recherche cet œil dans l'image complète [20], cette recherche est basée sur la transformé de Hough cercles pour la localisation des yeux, Puis ils ont passé vers le réseau de neurones pour faire la classification de ces yeux. Les principales limitations de cet

algorithmes sont les suivantes : il est applicable uniquement lorsque les yeux sont visibles dans l'image et il n'est pas robuste aux changements d'éclairage.

Dong et Wu ont présenté un système de détection de la fatigue du conducteur. La détection et la localisation de visage ont été effectuées en utilisant les caractéristiques de couleur de peau, ensuite les yeux sont détectés par projections. Après avoir localisé les yeux ils calculent le rapport d'aspect des yeux ¹(*Eye Aspect Ratio EAR*) en utilisant la distance de la paupière pour classer si les yeux sont ouverts ou fermés. Si les yeux sont fermés pendant cinq frames consécutifs le conducteur est considéré somnolent [21].

- **Pourcentage de Fermeture des Yeux (PERCLOS)**

L'une des mesures les plus récurrentes dans la littérature pour la détection de la somnolence est le PERCLOS qui a été proposé pour quantifier le changement apparent sur le mouvement des paupières [22] [23].

Cette mesure est basée sur le calcul du pourcentage de la fermeture de l'œil en fonction du temps et reflète les fermetures lentes des paupières plutôt que les clignements.

Le PERCLOS peut être obtenu par l'équation (1) ci-dessous :

$$f_{PERCLOS} = \frac{n_{fermée}}{N_{fermée\&ouvert}} \times 100\% \quad (2.1)$$

$N_{fermée\&ouvert}$: représente le nombre total d'ouvertures et de fermetures des yeux dans une période.

N_{fermer} : représente le nombre d'images des yeux fermés à un moment donné.

¹ Rapport d'aspect des yeux : Le rapport d'aspect est un attribut de projection d'image qui décrit la relation proportionnelle entre la largeur et la hauteur d'une image, dans le cas d'œil. Ce rapport est généralement constant lorsque l'œil est ouvert et commence à tendre vers zéro lors de la fermeture de l'œil

Une étude de Wierwille a montré qu'une valeur de PERCLOS excédant 80% pendant environ 3 minutes est révélatrice d'une lourdeur des paupières, et donc de somnolence [22].

Tous les travaux utilisant PERCLOS suivent les mêmes étapes qui consistent à délimiter la zone du visage et la région de chaque œil, puis à appliquer cette mesure pour détecter la somnolence.

Qing et al utilisent l'algorithme Adaboost pour localiser le visage et créer des modèles pour chaque œil en fonction de la position naturelle des yeux dans le visage [24]. Ensuite, Les modèles sont suivis et PERCLOS est calculé sur toutes les images vidéo. Le conducteur est alerté par une fatigue si le PERCLOS est supérieure à 40% et le temps de fermeture continue des yeux dépasse 3 secondes.

Garcia et al. [25] Présentent un système de prédiction de l'état du conducteur. Les auteurs utilisent l'algorithme de Viola et Jones dans l'étape de détection du visage et les yeux du conducteur, Puis ils estiment l'état des yeux avec la projection intégrale, et la dernière étape est prédire l'état du conducteur avec la mesure PERCLOS.

Grace a aussi utilisé le PERCLOS pour détecter la fermeture lente des paupières dans son système Copilot conçu pour surveiller l'état des chauffeurs professionnels [26]. Dans ce travail, La lumière infrarouge est utilisée pour produire l'effet d'éclaircir la pupille et de la rendre plus facile à détecter. Le PERCLOS est calculé sur des périodes de 3 minutes afin de détecter une fatigue modérée ($8\% \leq \text{PERCLOS} \leq 14\%$) ou sévère ($\text{PERCLOS} > 14\%$).

Après l'étude proposée par Ding et Grace, PERCLOS a longtemps été considérée comme une mesure standard, supérieure aux autres mesures (même l'EEG) et suffisante à l'estimation de l'hypovigilance [23].

2.3.2 Bâillements fréquents

Le bâillement est une réaction typique induite par la fatigue. Il se traduit par une ouverture prolongée et incontrôlée de la bouche bien différente des autres déformations des lèvres.

Mohanty et al modélisent cette activité par une estimation du mouvement non rigide des lèvres et considèrent qu'un bâillement est observé si un mouvement particulier des lèvres est maintenu entre cinq et dix secondes [27]. La technique proposée nécessite une série d'images de prétraitement pour fournir une première estimation de la zone à rechercher autour de la bouche en cas de bâillement. Pour tenter ensuite de distinguer le degré d'ouverture de la bouche lors de la parole et du bâillement, les auteurs ont mené une étude expérimentale de huit séquences dans lesquelles ces deux comportements sont simulés. Les résultats sont mesurés par le nombre de pixels de l'ouverture autour de la bouche. Ils ont conclu que le degré d'ouverture de la bouche lors d'un bâillement élevé est plus de trois ou quatre fois supérieur au degré de parole.

Dariusz Sawicki [28] propose un système qui peut distinguer entre les bâillements fréquents et l'ouverture de la bouche quand on parle, on chante où on sourira. Ce système est composé de trois étapes principales la première est la distinction des lèvres, ceci se fait avec l'algorithme de détection des repères faciaux et la deuxième étape est l'estimation de l'état de ces lèvres après le calcul de rapport de la hauteur de la bouche à leur largeur (Mouth Aspect Ratio MAR). Dans la dernière étape, la distinction entre la parole, le rire et le bâillement se fait à l'aide de la mesure FOM.

- **Fréquence de la bouche FOM**

La fréquence de la bouche (FOM) montre le ratio de bouche ouverte en fonction du nombre de bouche ouverte et fermée.

Le calcul de FOM est similaire au calcul de PERCLOS et ça valeur peut être calculée comme suit :

$$fFOM = \frac{n_{ouvert}}{N_{fermer\&ouvert}} \times 100\% \quad (2.2)$$

$N_{fermer\&ouvert}$: représente le nombre total d'ouvertures et de fermetures de la bouche dans une période.

n_{ouvert} : représente le nombre d'images de la bouche fermés à un moment donné.

Même si l'analyse des yeux reste le critère le plus utilisé pour déterminer l'état du conducteur puisqu'il permet de révéler la somnolence, l'utilisation du bâillement comme indicateur de la fatigue est très utile pour renforcer cette décision. En effet, l'analyse de la bouche permet de fournir de meilleurs résultats que l'analyse des yeux, puisque l'ouverture de la bouche liée au bâillement est beaucoup plus apparente dans une image.

2.4 Module de reconnaissance des panneaux de signalisation

La reconnaissance des panneaux de signalisation a reçu un intérêt croissant ces dernières années, il est même considéré comme une caractéristique très importante des véhicules intelligents. Les panneaux de signalisation portent des informations utiles substantielles qui pourraient être ignorées par les conducteurs en raison de la fatigue ou de la recherche d'une adresse, etc. Ces conducteurs sont également susceptibles de porter moins d'attention aux panneaux de signalisation lors de la conduite dans des conditions météorologiques menaçantes. Par conséquent, la détection et la reconnaissance automatique des panneaux routiers est devenu indispensables pour réduire le nombre de victimes sur la route.

Au cours des dernières années, un certain nombre de systèmes d'assistance aux conducteurs ont été proposés et mis en œuvre, et de nombreuses recherches ont été réalisées au sujet de la reconnaissance des panneaux routiers, la plupart adoptent une approche en deux phases : la localisation et la détection des panneaux et la deuxième phase est la phase de la classification qui permet de reconnaître le type de chaque panneau.

La phase de localisation et détection est très importante, car elle sert à détecter tous les panneaux et éliminer au maximum les autres objets qui peuvent apparaître dans l'image, cela permet de diminuer le taux des faux positifs. Dans le cas d'une fausse alarme dans la phase de détection, les performances seront plus faibles dans l'étape de classification, car le classifieur n'est pas toujours entraîné sur les faux positifs.

D'abord, nous allons présenter un aperçu sur les méthodes de détection et de classification récentes et efficaces.

2.4.1 Localisation et la détection des panneaux routiers

Les méthodes de détections peuvent être classées en trois classes principales : les méthodes colorimétriques qui utilisent la segmentation pour localiser les régions d'intérêts, les méthodes géométriques et les méthodes par apprentissage.

A. Modélisation colorimétrique

Une segmentation à base de couleur dominante est appliquée pour découvrir les régions d'intérêt, et ces régions sont ensuite filtrées selon un algorithme de reconnaissance ou un modèle d'apparence. Ces méthodes sont rapides, mais leur principal inconvénient est qu'elles sont sensibles aux changements d'éclairage. Les auteurs travaillent sur différents espaces colorimétriques pour pallier ce genre de problème.

L'espace HSV (Teinte - Saturation-Valeur) était le plus populaire comme étant basé sur la perception humaine. En plus, il est en grande partie insensible aux changements de luminosité, vu que sa composition est parfaitement décorrélée.

Piccioli et al ont déterminé les régions d'intérêt par groupe donc les pixels ayant des valeurs dans un domaine prédéfini par un seuil [29], Le modèle de HSV a été également employé par Paclík pour classifier les images d'essai des panneaux dans plusieurs catégories distinctives [30].

D'autres modèles d'aspect de couleur étaient utilisés mais moins populaires. Par exemple, Dans [31] les images en RGB prises par l'appareil-photo sont d'abord passées par une étape de prétraitement qui nous aide à maintenir la constance de couleur en différents états d'illumination avant de la transformée à l'espace chromatique de HSV.

Dans [32], ils ont utilisé une segmentation par seuil sur l'espace chromatique RGB après avoir fixé des seuils de changement de luminosité.

B. Modélisation géométrique

Dans le cas d'une telle approche les auteurs ne considèrent pas la segmentation couleur absolument fiables en raison de sa sensibilité à divers facteurs, tels que la distance de la cible, les conditions météorologiques, le temps de la journée, ou la réflexion des signes, la détection des panneaux se fait à partir des contours de l'image analysée par une approche structurelle ou globale.

Ces méthodes sont généralement plus robustes que celles photométriques parce qu'elles traitent le gradient de l'image et peuvent aussi traiter des images en niveaux de gris [33], mais elles sont coûteuses en temps de calcul, car la cadence du traitement dépend du nombre de contours détectés. Cependant, les méthodes géométriques peuvent traiter les images au niveau de gris, mais dans certains pays, comme le Japon, il y a des paires de signes au niveau de gris apparaissent exactement les mêmes. Pour pouvoir les distinguer, la couleur est absolument nécessaire [34]. D'autre part, certains auteurs utilisent l'une des méthodes calorimétriques pour localiser la région d'intérêt, ensuite compléter avec des méthodes géométriques afin de détecter la position des panneaux et de reconnaître sa forme géométrique.

La transformée de Hough a été utilisée par Garcia-Garrido et al afin de détecter les contours des panneaux, après ils sélectionnent que les contours fermés ce qui rend leur approche moins robuste aux bruits [35]. Après

ils utilisent un réseau de neurone pour la reconnaissance des panneaux. Leur approche est capable de détecter 97% des 435 panneaux de limitation de vitesse et 94% des 312 panneaux de danger dans un temps qui se varie de 20 ms/img jusqu'à 200 ms/img en fonction du nombre de contours traités. Même Miura et al ont appliqué la transformée de Hough pour les panneaux circulaires [36].

C. Méthodes avec apprentissage

Les deux méthodes précédentes (colorimétrique et géométrique) ne sont pas robustes aux changements d'éclairage, changements d'échelle, les occlusions, les rotations, etc. Cependant, ces problèmes pourraient également être traités en utilisant l'apprentissage automatique, mais cela nécessite une grande base de données annotées.

Dans ces méthodes un classificateur (cascade, SVM, réseaux de neurones) est entraîné sur une base d'exemples. Il est appliqué sur une fenêtre glissante qui parcourt l'image à plusieurs échelles. Ces méthodes combinent la géométrie et la photométrie mais peuvent être coûteuse en temps du calcul. Elles nécessitent la constitution d'une base d'apprentissage par type de panneaux, ce qui constitue une étape fastidieuse lorsque le nombre d'objets à détecter est grand.

Xavier Baró et al ont proposé un classificateur en cascade où chaque entrée de la cascade est la région d'intérêt détectée par le classifieur précédent [37], l'algorithme Adaboost a été utilisé pour l'apprentissage des classificateurs. Les chercheurs ont également proposé une stratégie de classification Forest-ECOC (*Error Correcting Output Codes*) pour surmonter le problème de multi-classification, l'idée étant d'intégrer plusieurs arbres dans le cadre ECOC. Les auteurs ont obtenu les résultats suivants, mais le temps de traitement n'est pas donné :

- Les panneaux d'interdiction : 70% de rappel avec 3.65% de faux positifs.

- Les panneaux d'obligation : 60% de rappel avec 0.95% de faux positifs.
- Les panneaux de danger : 65% de rappel avec 2.25% de faux positifs.
- Panneaux de droit de passage : 75% de rappel avec 2.8% de faux positifs.

L'histogramme des gradients orientés (HOG) proposé par Dalal et al [38], Pour la détection des piétons, est utilisé par Zaklouta et al [39] pour détecter les panneaux de danger en raison de sa robustesse aux changements d'échelle. Les descripteurs HOG sont utilisés comme une caractéristique pour entraîner le SVM, et pour améliorer sa précision, ils utilisent un opérateur morphologique blackhat² pour filtrer les candidats détectés.

Wu et al [40] utilisent les réseaux de neurones convolutifs CNN pour localiser et reconnaître les panneaux de signalisation, ils utilisent d'abord une machine vectorielle pour transformer l'image originale de RGB en échelle de gris pour éviter le problème de la différence de couleur due aux différentes conditions d'éclairage. Deuxièmement, ils utilisent la couche fixe dans le CNN pour localiser la région d'intérêt et les couches d'apprentissage sont utilisées pour extraire les caractéristiques discriminantes pour la classification. Avec la base GTSDDB, ils ont obtenu 99.73% dans les panneaux de danger et 97.62% dans les panneaux d'obligation. Cependant, ils sont trop loin d'une application en temps réel car pour une architecture de 180-200 couches, pour chaque région d'intérêt le nombre de paramètres est 1290326 avec un temps d'exécution 0.128 secondes (128 ms), il indique que le nombre de régions d'intérêt détectées dans une image est entre 5 et 20 régions. Alors le temps de détection pour une image de scène nécessite environ 2.5 secondes.

² La transformation de blackhat est définie comme la différence entre l'image de fermeture et l'image d'entrée, ils arrivent à réduire le nombre des faux positifs avec ce filtre.

2.4.2 Classification

A. Méthodes d'apprentissage basées sur des caractéristiques extraits manuellement

Les auteurs dans [41] ont comparé les performances de reconnaissance des panneaux routiers par l'être humain et les méthodes d'apprentissage de la machine, ils ont montré également les résultats d'un classificateur linéaire entraîné par LDA (*Linear Discriminant Analysis*), ils ont montré que les performances de LDA sont fortement liées aux caractéristiques choisies pour représenter l'image. Les auteurs ont obtenu de meilleurs résultats avec la représentation HOG2 avec une précision de 95,68%, 93,18% pour HOG1 et 92,34% pour HOG3.

L'analyse réalisée par Qu et al sur les bases de données d'apprentissage montre qu'il existe un déséquilibre dans la distribution des échantillons dans les classes des panneaux de signalisation [42]. La plus grande classe peut contenir plus de 1000 images tandis que la classe la plus petite contient seulement plusieurs images.

Pour pallier ce problème, les auteurs ont proposé une méthode de classification hiérarchique pour la reconnaissance des panneaux routiers. L'arbre de classification étant composé de deux couches, dans la première couche le classifieur Adaboost combiné avec ACF (*Aggregate Channel Features*) sont utilisés pour classer les panneaux en trois catégories selon leur forme géométrique. L'ACF est utilisé pour la représentation des caractéristiques où 10 canaux sont utilisés (trois canaux de couleur de l'espace colorimétrique RGB, les amplitudes de gradient, les six cartes de gradient orientées : horizontal, vertical, 30, 60, 120 et 150), puis ces caractéristiques sont utilisées pour entraîner le classifieur Adaboost. Dans la deuxième couche le panneau est identifié avec le Random Forest entraîné sur trois caractéristiques : HOG, LBP et HSV, ils ont atteint une précision de 95,97% sur GTSRB et de 97,94% sur STSD.

B. Méthodes d'apprentissage profond (Deep Learning)

Pierre Sermanet et Yann Le Cun [43] utilisent le réseau de neurones convolutifs (ConvNets) pour apprendre de manière non supervisée les caractéristiques invariantes du panneau de signalisation en utilisant des images en entrée de taille 32x32 pixels de la base de données GTSRB, ils atteignent un rappel de 98,97%, mieux que la performance humaine (98.81 % comme est indiqué dans [41]). Ils ont augmenté la capacité et la profondeur de leur réseau en ignorant les informations de couleur, ils ont établi un rappel de 99,17%. Les auteurs ont obtenu le meilleur résultat en ignorant la couleur, ils ont donc suspecté que les canaux de couleur normalisés soient plus informatifs que la couleur brute. Les auteurs n'ont pas indiqué le temps de traitement, mais cette méthode est encore loin d'une application en temps réel vu le nombre élevé des paramètres utilisés (1437791 paramètres).

Qian, R. et al [44] utilisaient aussi CNN comme extracteur de caractéristiques et MLP (*perception multi-couche*) comme classifieur. En comparaison avec le ConvNet classique, dans la couche max pooling de MPPS (*max pooling positions*) les auteurs n'utilisent pas les valeurs max, à l'inverse, ils utilisent leurs positions. Le MPPS consiste à coder chaque position de valeur maximale en binaire de 4 bits, puis à les concaténer pour obtenir la fonctionnalité MPP. La précision obtenue en utilisant MPPS est augmentée à 98,86% sur GTSRB.

Les auteurs dans [45] proposent une nouvelle approche où le CNN fonctionne comme un extracteur de caractéristiques profondes, ce qui signifie que seules les huit premières couches sont conservées et éliminent les couches entièrement connectées. Extreme Learning Machine ELM est utilisé ensuite en tant que classifieur pour ses performances de généralisation. La méthode proposée prend 5- 6 heures en apprentissage sans GPU et atteint un taux de reconnaissance de 99.40 % sans aucune augmentation et pré-traitement de données comme dans [46], mais cette méthode n'est pas robuste au flou de mouvement.

Un nouveau record de classification des panneaux est à nouveau réalisé par Aghdam et al [47], ils proposent une nouvelle variante de leur CNN précédent proposée dans [48]. Les auteurs remplacent les images couleur par des images au niveau de gris, ils suppriment la couche de transformation linéaire. Et pour augmenter la flexibilité, une couche entièrement connectée est ajoutée au réseau et ils réduisent la taille du premier et du deuxième noyau. De plus, les images d'entrée sont redimensionnées à 44x44 pixels pour réduire le temps de traitement. La nouvelle meilleure précision obtenue est de 99,55% avec CNN simple et 99,70% avec un ensemble de 3 CNN. Le nouveau CNN est temps réel avec un temps de traitement 0,7 ms par image.

2.5 Systèmes embarqués

Les systèmes embarqués font partie intégrante de chaque composant électronique moderne. Il s'agit d'unités à faible consommation d'énergie qui sont utilisées pour exécuter des tâches spécifiques, par exemple des télécommandes, des machines à laver, des fours à micro-ondes, des étiquettes RFID, des capteurs, des actionneurs et des thermostats utilisés dans diverses applications, du matériel réseau tel que des commutateurs, des routeurs, des modems, des téléphones portables, PDA, etc.

Il est presque impossible de tourner la tête sans voir au moins une application. Ceci est sans parler des applications moins « voyantes » mais pas moins répandues : conduite assistée pour voitures, monitoring du trafic, contrôles d'accès, systèmes de sécurité, etc.

Pour atteindre un tel résultat, l'électronique a dû progressivement s'adapter pour devenir portable, puis ultra-miniaturisée : c'est la naissance des systèmes embarqués.

2.5.1 Définition

Un système embarqué peut être défini comme un système électronique et informatique autonome dédié à une tâche bien précise et répondent souvent à des contraintes temps réel [66].

Le logiciel a une fonctionnalité fixe à exécuter qui est spécifique à une application. L'utilisateur n'a pas la possibilité de modifier les programmes. Bien souvent, il n'a pas conscience d'utiliser un système à base des microprocesseurs.

2.5.2 Caractéristique

Les systèmes embarqués traitent certaines caractéristiques spécifiques et ces caractéristiques sont uniques dans chaque système embarqué. Certaines des caractéristiques importantes d'un système embarqué sont [67].

A. Spécification de domaine d'application

Un système embarqué exécute certaines fonctions prévues mais ne pouvant être employées dans aucun autre but. Des spécifications sont généralement attribuées et contribuent à la réduction du prix global du système embarqué.

B. Temps-réel

Un système temps réel est un système qui répond à des contraintes temporelles n'est pas forcément un système qui va vite. Il doit pour cela garantir un déterminisme logique et temporel. Il doit également être fiable [68].

On distingue trois types de temps réel :

- **Le temps réel dur** : C'est un système pour lequel un retard dans la délivrance du résultat rend le système inutile. C'est par exemple le cas du contrôle d'un missile.

- **Le temps réel mou** : C'est un système pour lequel un retard dans la délivrance du résultat n'est pas catastrophique. C'est par exemple le cas du système d'ouverture d'une barrière automatique.
- **Le temps réel ferme** : C'est un système pour lequel on peut tolérer quelques retards (peu souvent) dans la délivrance du résultat. C'est par exemple le cas d'un système multimédia type téléphone mobile.

C. Réactivité

Un système embarqué doit par définition surveiller et contrôler un environnement. Si ce dernier présente lui-même un comportement changeant ou de type événementiel.

D. Criticité, Fiabilité

Du fait de leur portabilité et de la mobilité des produits dans lesquels ils sont incorporés, les systèmes embarqués évoluent généralement dans de conditions environnementales non déterministes et souvent non maîtrisées. Ils sont exposés à des variations et autres contraintes environnementales susceptibles d'induire des défaillances. En même temps que s'accroît leur sophistication, les systèmes embarqués sont utilisés dans des applications de plus en plus critiques dans lesquels leur dysfonctionnement peut générer des nuisances. Ce type de systèmes doit garantir une très haute fiabilité et doit pouvoir réagir en cas de panne de l'un de ses composants [69].

E. L'autonomie

Les systèmes embarqués doivent en général être autonomes, c'est-à-dire remplir leur mission pendant de longues périodes sans intervention humaine. Cette autonomie est nécessaire lorsque L'intervention humaine est impossible, mais aussi lorsque la réaction humaine est trop lente ou insuffisamment fiable [70].

F. La consommation énergétique

La gestion de puissance est un autre facteur important qui doit être pris en considération lors de la conception des systèmes embarqués. Les processeurs utilisés dans les systèmes embarqués sont 2 à 3 décades moins puissantes qu'un processeur d'un ordinateur PC. La consommation énergétique est très faible, due à l'utilisation de batteries et/ou, de panneaux solaires voir de pile à combustible pour certains prototypes.

2.5.3 Architecture d'un système embarqué

Quelle que soit la nature et la complexité du système, on décompose un système embarqué en :

- **Système contrôlé** : environnement (procédé) équipé d'une instrumentation qui réalise l'interface avec le système de contrôle.
- **Système de contrôle** : éléments matériels (microprocesseurs...) et logiciels dont la mission est d'agir sur le procédé via les actionneurs en fonction de l'état de ce procédé indiqué par les capteurs de manière à maintenir ou conduire le procédé dans un état donné.

L'architecture d'un système embarqué se définit par le schéma ci-dessous (Figure 3.2).

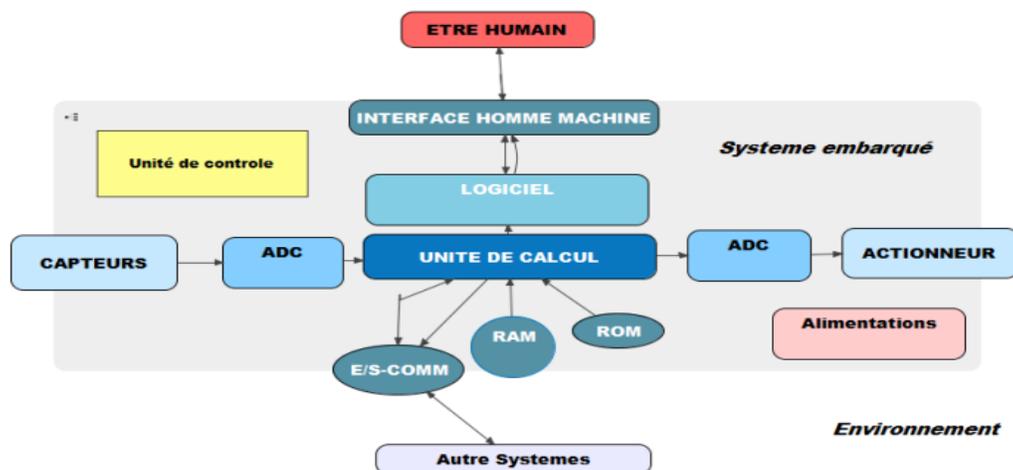


Figure 2-1 : Architecture d'un système embarqué.

2.5.4 Les différentes applications des systèmes embarqués

Les systèmes embarqués sont désormais utilisés dans des applications diverses tels que le transport (avionique, espace, automobile, ferroviaire), dans les appareils électriques et électroniques (appareils photo, jouets, postes de télévision, électroménager, systèmes audio, téléphones portables), dans la distribution d'énergie, dans l'automatisation, etc.

A. L'utilisation des systèmes embarqués dans les réseaux cellulaires (IOT)

Les systèmes embarqués seront également la pierre angulaire du déploiement de nombreuses solutions Internet des objets (IoT), en particulier dans certains secteurs verticaux et applications de l'Internet des objets industriels (IIoT).

Les principaux acteurs des développements matériels et logiciels de systèmes embarqués visent à intégrer ces transformations dans leurs produits pour tirer parti de la croissance du marché de l'IoT. Les domaines qui vont se transformer sont les systèmes d'exploitation en temps réel (RTOS) et les microprocesseurs et microcontrôleurs, suivis des empreintes mémoire et des réseaux, des communautés open source et des développeurs.

Il y a plusieurs raisons à ces mises à jour (correction, adaptation, amélioration) [71], qui impliquent l'intégration ou la suppression de données d'un état du système en impliquant un nouvel état ; Ce passage d'un état à un autre donne un aspect dynamique au système et donc une bonne réflexion pour le système réel. Aussi, chaque état du système représente pour lui une nouvelle configuration. Ainsi, la transition entre deux états sera une action de reconfiguration du système [72].

L'IoT permet à un ensemble d'objets physiques et d'applications logicielles et matérielles de communiquer partout et à tout moment en connectant le monde réel au monde numérique [73] [74]. Grâce à l'intégration de plusieurs technologies, l'IoT a permis de réduire la taille, le poids, la consommation d'énergie, le prix des appareils embarqués, avec la technologie sans fil et la capacité de communication [75].

- **Agriculture**

À l'intérieur de l'agriculture, la mise en œuvre de capteurs dans les cultures est proposée pour connaître les conditions environnementales et terrestres pour comparer les conditions météorologiques et déterminer la quantité d'eau, de fongicides, de nutriments, entre autres. De nombreuses zones agricoles peuvent améliorer leur efficacité en déterminant les mauvaises herbes nuisibles pour les plantes et les animaux avec des robots programmés qui sont responsables de leur élimination précise. Le bétail peut être surveillé pour connaître les zones où il se trouve et avertir lorsqu'il est perdu. Des accéléromètres 3D peuvent également détecter des problèmes physiques et partager ces informations entre les agriculteurs pour analyser les schémas de maladies [76].

Par exemple, le pilote EC-GSM IoT connecte des capteurs capables de mesurer les conditions environnementales, telles que la température, l'humidité, pour soutenir les solutions d'agriculture intelligente dans les marchés émergents [77].

- **Bâtiments intelligents**

Les bâtiments collectent des informations grâce à des capteurs de lumière, de chaleur, de mouvement, entre autres. En interprétant les données, Es peut économiser la consommation d'énergie électrique et la génération d'algorithmes pour apprendre les comportements des personnes dans les espaces physiques.

- **Éducation**

Certains chercheurs visent à révolutionner la pratique de l'enseignement en utilisant des systèmes embarqués en plus de l'Internet des objets. Dans ce contexte, un système a été proposé pour contrôler la présence des étudiants dans les classes universitaires, Le projet a démontré des exemples d'étude et de comparaison de certains capteurs (capteur RFID, GPS pour surveiller la position de l'étudiant à l'aide du téléphone portable, capteur biométrique, QR-Code, caméra de reconnaissance faciale, entre autres) avec un appareil Arduino Mega.

Un problème possible est qu'un étudiant peut passer la carte RFID sur un appareil embarqué installé à l'entrée de la salle de classe en se faisant passer pour un autre étudiant sans qu'il soit à l'université. Par conséquent, un « système de double contrôle » avec une combinaison de RFID, de reconnaissance biométrique et d'un clavier pour le codage des mots de passe a été initialement analysé. Ensuite, une fusion de capteurs a été testée en temps réel comme d'autres solutions, par exemple la caméra pour la reconnaissance faciale.

- **Transport**

Dans le domaine des transports, l'un des objectifs est de parvenir à l'efficacité et à la sécurité de ralentir lorsqu'un feu de circulation passe au jaune ou avertir d'une place de stationnement.

Il faut considérer que 90% des accidents sont des erreurs humaines. C'est pourquoi un environnement intelligent peut améliorer les décisions d'un conducteur en se basant sur les données du trafic ou la densité des véhicules.

Dans les aéroports, des capteurs sont installés pour connaître le flux de personnes qui passent afin de déployer du personnel supplémentaire pour faire face aux longues files d'attente peut être observé à partir d'une application et peut réorienter la route avec différentes voies pour atteindre leur destination [78].

- **Santé**

L'internet des objets permet d'aborder les domaines de la prévention et de la détection précoce, de la recherche et des soins de santé, puisque les signes vitaux peuvent être surveillés pour collecter une grande quantité de données et être en mesure de déterminer si certains modes de vie peuvent altérer leur santé, avec ces informations, les médecins peuvent effectuer une assistance à distance et permettre une action rapide. Cela nécessite une infrastructure interopérable très fiable interopérable très fiable pour l'acquisition et l'analyse des données et, surtout, le maintien de la confidentialité de l'utilisateur. Ces systèmes embarqués sont considérés comme des dispositifs portables.

B. L'émergence de l'intelligence artificiel dans les systèmes embarqués

L'intelligence artificielle est un domaine dans lequel les machines tentent d'interpréter le contenu de l'information (image ou signal) au plus près des capacités humaines [79]. C'est un domaine de recherche très vaste, et ses applications sont très diverses et variées [80].

C'est un domaine qui vise à concevoir des systèmes intelligents pouvant évoluer dans le temps en interagissant avec leurs environnements en utilisant plusieurs disciplines telles que l'informatique, les mathématiques, les neurosciences, les systèmes embarqués, etc.

L'un des algorithmes largement utilisés dans ce domaine est les réseaux de neurones artificiels, qui offrent des solutions satisfaisantes pour une grande variété d'applications du monde réel, dans divers domaines tels que l'ingénierie industrielle, la robotique et l'automobile.

Un réseau de neurones artificiels est un réseau dont les neurones sont interconnectés selon une architecture, et les liens sont configurés via un processus d'apprentissage pour une application spécifique, comme la

reconnaissance de formes ou la classification. Cependant, comme dans tous les systèmes nerveux biologiques, cet apprentissage nécessite l'ajustement des connexions (poids synaptiques) entre les neurones [81].

L'importance de ces systèmes de décision est croissante car les données sont disponibles dans plusieurs domaines d'application [82]. Les réseaux de neurones artificiels peuvent être utilisés dans diverses applications telles que les voitures intelligentes [83], la médecine [84] et la vision robotique.

Cependant, les propriétés du réseau de neurones artificiels nécessitent un circuit d'implémentation matériel puissant, d'où l'utilisation d'une plateforme capable d'implémenter cet ANN et d'utiliser le parallélisme dans le traitement et la reconfiguration. Ainsi, plusieurs chercheurs ont tenté d'implémenter ce réseau dans des plates-formes embarquées, telles que les processeurs graphiques [85] et les circuits intégrés applicatifs ASIC [86].

2.6 Conclusion

Au cours de ce chapitre, nous avons dressé un état de l'art des systèmes avancés d'aide à la conduite où un état de l'art fut élaboré pour chaque étape dans les trois modules que nous avons développés : l'identification du conducteur et la détection de son état ainsi que la classification des panneaux de signalisation.

Chapitre 3

L'apprentissage profond

3.1 Introduction

L'intelligence artificielle n'est plus un complot pour les films de fiction. La recherche dans ce domaine augmente chaque jour et fournit de nouvelles informations sur l'apprentissage automatique. Les méthodes d'apprentissage profond, sont actuellement utilisées sur de nombreux fronts tels que la reconnaissance faciale dans les réseaux sociaux, les voitures automatisées et même certains Diagnostics dans le domaine de la médecine. Cependant, cette technologie est encore mal connue. Le but de cette recherche est de clarifier le fonctionnement de l'apprentissage et de montrer ses applications actuelles. Bien sûr, avec la diffusion de ces connaissances, l'apprentissage profond pourrait avoir d'autres applications dans un avenir proche, encore plus importantes pour toute l'humanité.

Dans ce chapitre, nous allons présenter l'apprentissage profond et son évolution au cours de temps, puis nous allons parler sur les réseaux neurones artificiels. Ensuite, nous concentrons sur les réseaux de neurones convolutifs et ces différentes architectures.

3.2 Définition

L'apprentissage profond (*Deep Learning* : *DL*) est un type d'intelligence artificielle dérivé de l'apprentissage machine où la machine est capable d'effectuer des tâches précises (ex. : classification, segmentation, prédiction). Grâce aux exemples qui permettent à la machine d'apprendre par elle-même. Sans supervision, sans intervention humaine, en simulant la mécanique du cerveau humain pour exécuter avec précision des applications complexes basées sur des réseaux neuronaux artificiels [49].

3.3 Historique

L'histoire de l'apprentissage profond remonte à 1943, le neuroscientifique *Warren McCulloch* et le logicien *Walter Pitts* ont publié l'article "*A logical calculus of the ideas immanent in nervous activity*", dans lequel ils proposaient le premier neurone artificiel permettant de calculer des fonctions arithmétiques et logiques appelées logique à seuil - ou *threshold logic*- pour imiter le processus de pensée du cerveau humain. [50]

En 1958, Frank Rosenblatt a créé le perceptron [51], un algorithme de reconnaissance de formes basé sur un réseau de neurones à deux couches utilisant une simple addition et une soustraction.

Au cours des années 70, l'intelligence artificielle a fait l'objet d'un bref recul. Le manque de financement a limité la recherche en apprentissage en profondeur et en intelligence artificielle. Cependant, certains ont poursuivi la recherche sans financement pendant ces années difficiles. Kunihiko Fukushima a utilisé les premiers réseaux de neurones convolutifs conçus avec plusieurs couches de pooling et de convolution. En 1979, il a développé un réseau de neurones artificiels multi-couches, appelé Neocognitron [52], un réseau de neurones artificiels qui a appris à reconnaître les modèles visuels. Il a été utilisé pour les tâches de reconnaissance de caractères manuscrits et autres modèles de reconnaissance de formes et même le traitement du langage naturel.

En 1995, Dana Cortes et Vladimir Vapnik [53] ont mis au point la machine à vecteurs de support (SVM).

C'est un système de reconnaissance et de mappage de données similaires. Il peut être utilisé pour la catégorisation de texte, la reconnaissance manuscrite de caractères et la classification d'images en ce qui concerne l'apprentissage automatique et l'apprentissage en profondeur.

La prochaine étape importante de l'évolution du Deep Learning a eu lieu en 1999, lorsque les ordinateurs ont commencé à accélérer le traitement des données et que les unités de traitement graphique (GPU) ont été développées. Un traitement plus rapide, avec les GPU traitant les images, a multiplié par 1000 les vitesses de calcul sur une période de 10 ans. Pendant ce temps, les réseaux de neurones ont commencé à concurrencer les machines à vecteurs de support. Alors qu'un réseau de neurones pouvait être lent comparé à une machine à vecteurs de support, les réseaux de neurones offraient de meilleurs résultats en utilisant les mêmes données. Les réseaux de neurones ont également l'avantage de continuer à s'améliorer à mesure que de nouvelles données d'apprentissage sont ajoutées.

En 2009, l'atelier NIPS travaillant sur l'apprentissage profond pour la reconnaissance vocale découvre qu'avec un ensemble de données suffisamment important, les réseaux de neurones n'ont pas besoin d'un apprentissage préalable et que les taux d'erreur chutent considérablement. D'où l'apparition de la base de données ImageNet créée par Fei-Fei Li [54], professeur d'intelligence artificielle à Stanford. Il s'agit d'une base de données gratuite de plus de 14 millions d'images étiquetées.

3.4 Réseaux de neurones artificiels

Un neurone artificiel est un modèle de calcul mathématique et informatique dont la conception est inspirée du fonctionnement des neurones biologiques capable de traiter les entrées reçues d'un ensemble de connexions caractérisées par des poids synaptiques. Ces poids indiquent l'importance de la connexion [55]. L'entrée reçue

est traitée à l'intérieur d'un nœud (neurone), puis transmise au nœud suivant en tant que sortie du premier neurone.

Le nœud reçoit le signal d'entrée comme catalyseur d'autres nœuds, il calcule une fonction de seuil linéaire simple et transmet le signal de sortie à un autre nœud uniquement lorsque le signal est plus fort que le seuil limite du nœud [56], le signal résultant est étiqueté par un poids permettant de préciser dans quelle mesure le signal est amplifié ($w>0$) ou diminué ($w<0$). Cette opération entraîne la mise à jour des poids synaptiques et c'est ainsi que l'apprentissage se produit [57].

3.4.1 Modélisation d'un neurone artificiel

Un neurone formel est une représentation artificielle et schématique d'un neurone biologique.

La similitude entre le neurone biologique et le neurone artificiel est que les dendrites sont les entrées et l'axone est la sortie et qu'ils représentent tous deux les connexions avec d'autres neurones pour les neurones biologiques et artificiels respectivement. Comme montre **la figure 3.1**

Les synapses sont les poids des connexions et l'activité dans le soma ressemble aux calculs internes des nœuds.

Une autre analogie est que le seuil limite pour le passage du signal ressemble à la puissance du signal électrique nécessaire pour déclencher l'activité dans les cellules connectées.

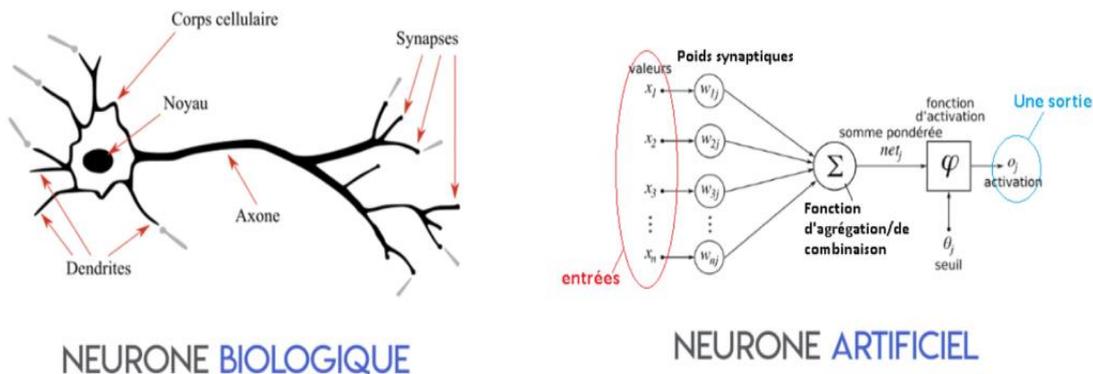


Figure 3-1 : L'Analogie Entre le Neurone Biologique et le Neurone Artificiel.

3.4.2 Principe de fonctionnement

Le neurone effectue une simple sommation mathématique des poids multipliés par la valeur d'entrée et ajoute un biais. Le produit de ces opérations passe par une fonction d'activation non linéaire et la sortie de la fonction d'activation est la sortie du neurone. Le modèle du neurone artificiel est illustré dans **la figure 3.2** :

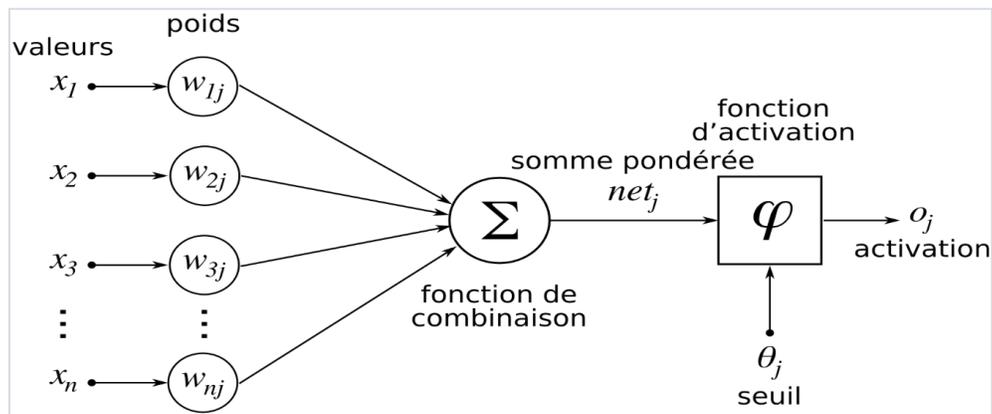


Figure 3-2 : Modèle d'un neurone artificiel.

3.4.3 Développement d'un réseau de neurones

La procédure de développement du réseau de neurones passe par trois étapes :

A. Choix d'une base de données

Afin de développer une application à base de réseaux de neurones, il est nécessaire de disposer de deux bases de données : une base pour effectuer l'apprentissage et une autre pour tester le réseau obtenu et déterminer ses performances. L'objectif de cette étape est donc de recueillir et de rassembler un nombre de données suffisant pour constituer une base représentative des données susceptibles d'intervenir en phase d'utilisation du système neuronal. Il est souvent préférable d'effectuer une analyse des données de manière à déterminer les caractéristiques discriminantes pour détecter ou différencier ces données. Cette détermination des caractéristiques a des conséquences à la fois sur les performances du réseau, et sur le temps d'apprentissage.

B. Choix d'une architecture adéquate

L'architecture adaptée est choisie en fonction de nombreux facteurs, parmi lesquels nous énumérons :

Fonctionnalité requise, nature des données à traiter, ressources matérielles/logicielles, limitations en temps réel, efforts de configuration de la base de formation/test, retards d'apprentissage.

C. Apprentissage du réseau :

L'apprentissage est une phase de développement du réseau de neurones durant laquelle le comportement de ce dernier est modifié jusqu'à l'obtention du comportement désiré. Initialisés au départ, les poids synaptiques et biais du réseau sont actualisés au cours de la phase d'apprentissage de la dernière couche vers la première afin d'essayer de minimiser la fonction de coût noté J sur une base de données d'apprentissage. J représente les erreurs faites par le modèle, en d'autres termes elle mesure l'écart entre la prédiction attendue et la prédiction effectivement donnée par le réseau.

3.4.4 Les couches des réseaux neuronaux

Une couche est la composante de base des réseaux neuronaux, composée d'un nombre précis et bien choisi de neurones uniformes ayant la même fonction d'activation. La couche agit comme un conteneur qui transforme l'entrée reçue à l'aide de ces fonctions, généralement non linéaires, et la transmet à la couche suivante [58].

La première couche d'un réseau est la couche d'entrée, la dernière est la couche de sortie et les couches intermédiaires sont appelées couches cachées.

Types de couches fréquemment utilisées

- **Dense** : Une couche qui connecte chacun de ses neurones avec tous les neurones de la couche précédente. La transformation qu'elle effectue est simple, il suffit de calculer la sortie de la fonction d'activation pour l'entrée afin de la transmettre à la couche suivante.
- **Dropout** : Une couche qui met aléatoirement la fonction d'activation de l'entrée à 0 avec une probabilité d'acceptation donnée. Utilisé pour traiter la tendance du réseau à sur-apprendre (overfitting).
- **Flatten** : Convertit l'entrée (la sortie des couches précédentes) en un vecteur 1D que nous pouvons classer, sans affecter la taille du lot.

3.4.5 Les types des réseaux de neurones

Depuis l'apparition des réseaux de neurones artificiels, plusieurs modèles neuronaux ont été proposés. Chaque modèle est caractérisé par sa règle d'apprentissage, les fonctions d'activation et l'architecture des connexions inter-neuronales. Le choix d'une architecture adéquate est un facteur tout aussi important. Certains éléments d'un réseau de neurones restent faciles à déterminer comme le nombre d'entrées, ainsi que le nombre de sorties.

A. Réseaux de neurones monocouche (perceptron simple)

Le perceptron simple est dit simple parce qu'il ne dispose que deux couches ; la couche en entrée et la couche en sortie. Le réseau est déclenché par la réception d'une information en entrée. Le traitement de la donnée dans ce réseau se fait entre la couche d'entrée et la couche de sortie qui sont toutes reliées entre elles. Le réseau intégral ne dispose ainsi que d'une matrice de poids. Le fait de disposer d'une seule matrice de poids limite le perceptron simple à un classificateur linéaire permettant de diviser l'ensemble d'informations obtenues en deux catégories distinguées. Le modèle du perceptron simple est illustré dans **la figure 3.3** :

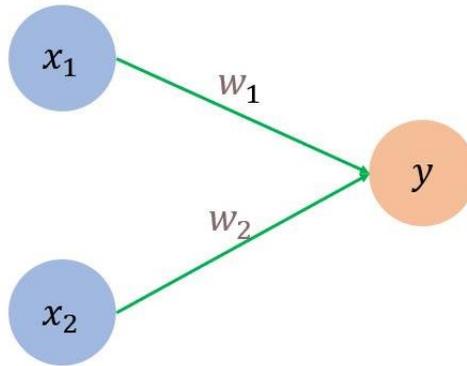


Figure 3-3 : Schéma de perceptron simple.

B. Réseaux de neurones multicouches (perceptron multicouches MLP)

Le perceptron multicouche se structure de la même façon. L'information entre par une couche d'entrée et sort par une couche de sortie. À la différence du perceptron simple, le perceptron multicouche dispose entre la couche en entrée et la couche en sortie une ou plusieurs couches dites « cachées » comme le montre la **figure**

3.4 :

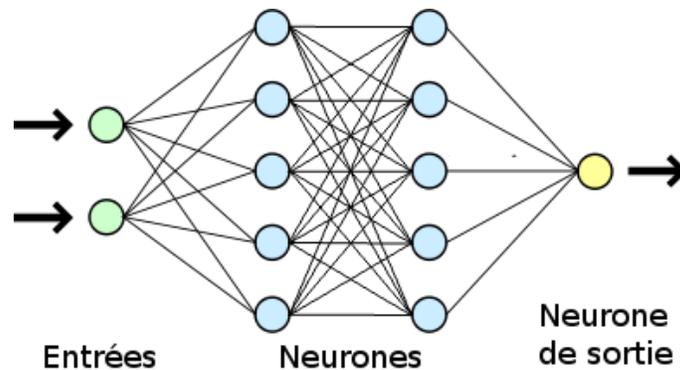


Figure 3-4 : Schéma de perceptron multicouches.

Le nombre de couches correspond aux nombres de matrices de poids disponible pour le réseau le réseau.

Un perceptron multicouche est donc mieux adapté pour traiter les types de fonctions non-linéaires.

C. Réseaux de neurones récurrents (*Recurrent Neural Network-RNN*)

Les Réseaux de Neurones récurrents traitent l'information en cycle. Ces cycles permettent au réseau de traiter l'information plusieurs fois en la renvoyant à chaque fois au sein du réseau. Comme il est illustré dans **la figure 3.5** :

La force des Réseaux de neurones récurrents réside dans leur capacité de prendre en compte des informations contextuelles suite à la récurrence du traitement de la même information. Cette dynamique auto-entretient le réseau.

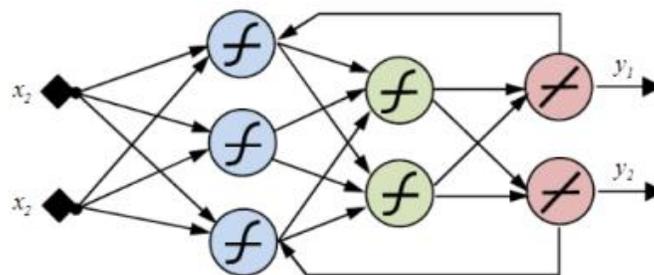


Figure 3-5 : Schéma de Réseaux de neurones récurrents.

D. Réseaux de neurones à connexions locales

Il s'agit aussi d'un réseau multicouche où chaque neurone entretient des relations avec un nombre réduit et localisé de neurones de la couche avale. Les connexions sont donc moins nombreuses que dans le cas d'un réseau MLP. (**La figure 3.6** montre le modèle du Réseaux de neurones à connexions locales)

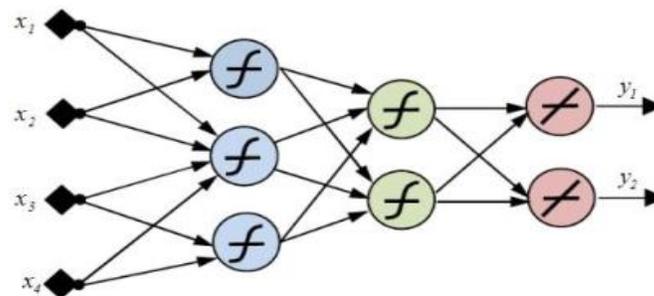


Figure 3-6 : Schéma de Réseaux de neurones à connexions locales.

3.5 Réseaux de neurones convolutifs

Un réseau neuronal convolutif (CNN) est un algorithme d'apprentissage profond qui peut prendre une image d'entrée, attribuer de l'importance (poids et biais apprenables) à divers aspects / objets de l'image et être capable de différencier l'un de l'autre. Ces dernières années, les CNN sont devenus un élément central de toutes les méthodes de vision par ordinateur et ConvNet représente l'une des plus avancées de ces méthodes.

3.5.1 Le principe de fonctionnement

Le réseau de neurone convolutif (CNN) est un sous-type de réseaux de neurones artificiels (ANN).

L'innovation des réseaux de neurones convolutifs est la possibilité d'apprendre automatiquement un grand nombre de filtres en parallèle spécifiques à un jeu de données d'apprentissage sous les contraintes d'un problème de modélisation prédictive spécifique, comme la classification d'images.

Ils sont constitués de neurones avec des poids et des biais apprenables. Chaque neurone spécifique reçoit de nombreuses entrées et prend ensuite une somme pondérée sur eux, où il le fait passer par une fonction d'activation et répond en retour avec une sortie.

Les réseaux de neurones convolutifs sont basés sur le perceptron multicouche (MLP), et ils sont inspirés du comportement du cortex visuel des vertébrés. Bien qu'efficaces pour le traitement d'images, les MLP ont beaucoup de mal à gérer des images de grande taille, ce qui est dû à la croissance exponentielle du nombre de connexions avec la taille de l'image. Par exemple, si on prend une image de taille $32 \times 32 \times 3$ (32 de large, 32 de haut, 3 canaux de couleur), un seul neurone entièrement connecté dans la première couche cachée du MLP aurait 3072 entrées ($32 \times 32 \times 3$). Une image 200×200 conduirait ainsi à traiter 120 000 entrées par neurone ce qui, multiplié par le nombre de neurones, devient énorme.

La **figure 3.7** montre la fonctionnalité de réseau de neurone convolutif

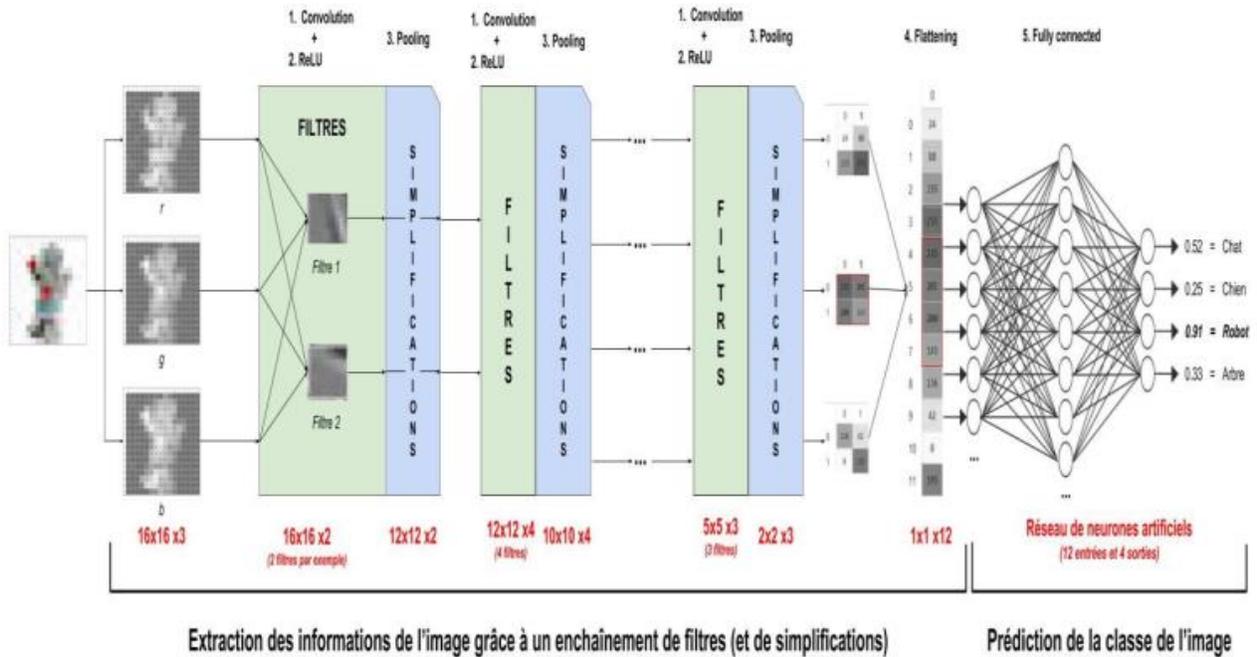


Figure 3-7 : Schéma d'un réseau de neurone convolutif.

3.5.2 Les couches du réseau de neurone convolutif

Le réseau neuronal convolutif est une séquence de couches, Chaque couche reçoit en entrée des données et les renvoie transformées. Pour cela, elle calcule une combinaison linéaire puis applique éventuellement une fonction non-linéaire, appelée fonction d'activation. Les coefficients de la combinaison linéaire définissent les paramètres (ou poids) de la couche.

Un réseau de neurones est construit en empilant les couches : la sortie d'une couche correspond à l'entrée de la suivante. Cet empilement de couches définit la sortie finale du réseau comme le résultat d'une fonction différentiable de l'entrée.

Nous utilisons trois principaux types de couches pour créer des architectures ConvNet : la couche convolutive (Convolutional layer), la couche de regroupement (Pooling layer) et la couche entièrement connectée (Fully-

connected layer) (exactement comme dans les réseaux de neurones classiques). Nous empilerons ces couches pour former une architecture ConvNet complète.

A. Couche de convolution

La couche de convolution a des noyaux (filtres) et chaque noyau a une largeur, une profondeur et une hauteur. Cette couche produit les cartes de caractéristiques à la suite du calcul du produit scalaire entre les noyaux et les régions locales de l'image.

La **figure 3.8** explique le déroulement d'une convolution avec un noyau de taille 3×3 et un pas de 2.

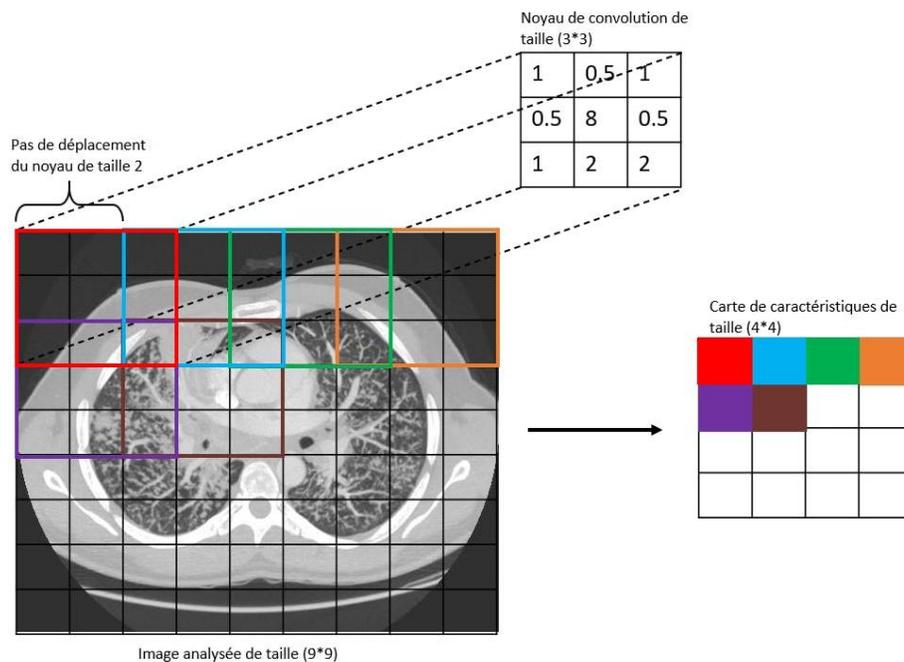


Figure 3-8 : Schéma d'une couche de convolution avec un noyau de taille 3×3 et un pas de 2.

Leur travail consiste à extraire les informations pertinentes de l'image (caractéristiques) grâce à une opération de convolution. Cette opération fait glisser un ensemble de filtres sur une image. Le poids de ces filtres est mis à jour lors de l'apprentissage et c'est grâce à eux que le réseau réussit à reconnaître les images par la suite.

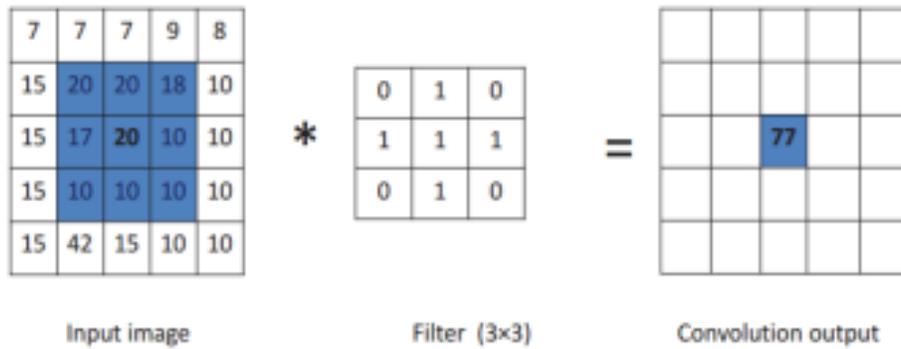


Figure 3-9 : Illustration d'une opération de convolution.

B. Couche de Pooling

L'étape de pooling est une technique de sous-échantillonnage. Généralement, une couche de pooling est insérée régulièrement entre les couches de correction et de convolution. En réduisant la taille des cartes de caractéristiques, donc le nombre de paramètres du réseau, cela accélère le temps de calcul et diminue le risque de sur-apprentissage.

L'opération de pooling la plus courante est celle du maximum : Max Pool ($2 \times 2, 2$). Elle est plus efficace que la moyenne, car elle maximise le poids des activations fortes. Elle est appliquée à la sortie de la couche précédente comme un filtre de convolution de taille (2×2) et se déplace avec un pas de 2. En sortie de la couche de pooling est obtenue une carte de caractéristique compressée par un facteur de 4.

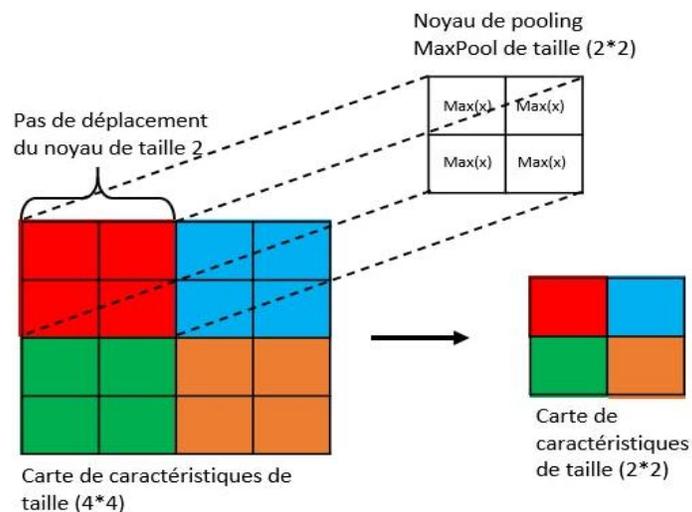


Figure 3-10 : Fonctionnement d'une opération Max pooling de taille 2×2 et d'un pas de 2.

C. Couche entièrement connectée

Cette couche est à la fin du réseau. Elle permet la classification de l'image à partir des caractéristiques extraites par la succession de bloc de traitement. Elle est entièrement connectée, car toutes les entrées de la couche sont connectées aux neurones de sorties de celle-ci. Ils ont accès à la totalité des informations d'entrée. Chaque neurone attribue à l'image une valeur de probabilité d'appartenance à la classe i parmi les C classes possibles. Chaque probabilité est calculée à l'aide de la fonction « softmax » dans le cas où les classes sont exclusivement mutuelles.

3.5.3 Les fonctions d'activation

La fonction d'activation est une fonction mathématique appliquée à un signal en sortie d'un neurone artificiel, Elle va permettre le passage d'information ou non de l'information si le seuil de stimulation est atteint. Concrètement, elle va avoir pour rôle de décider si on active ou non une réponse du neurone.

A. ReLU (Rectified Linear Units) :

La fonction d'activation la plus simple et la plus utilisée. Elle donne x si x est supérieur à 0, 0 sinon. Autrement dit, c'est le maximum entre x et 0 :

$$f_{ReLU} = \max(x, 0) \quad (3.1)$$

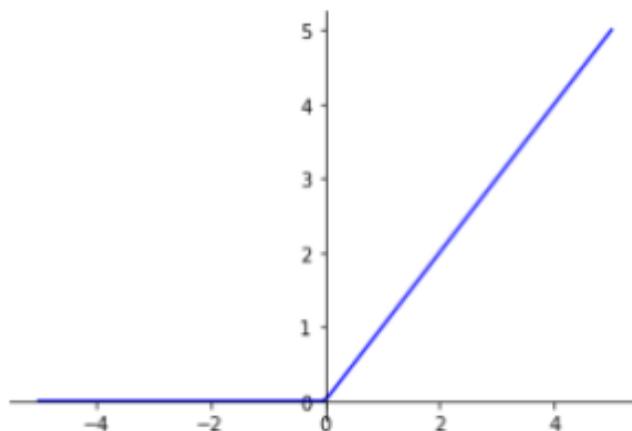


Figure 3-11 : La fonction d'activation ReLu

Cette fonction permet d'effectuer un filtre sur nos données. Elle laisse passer les valeurs positives ($x > 0$) dans les couches suivantes du réseau de neurones. Elle est utilisée presque partout mais surtout pas dans la couche finale, elle est utilisée dans les couches intermédiaires

B. Sigmoid :

La fonction Sigmoid donne une probabilité (une valeur entre 0 et 1). Elle est donc très utilisée pour les classifications binaires, lorsqu'un modèle doit déterminer seulement deux labels.

$$f_{sigmoid} = \frac{1}{1 + e^{-x}} \quad (3.2)$$

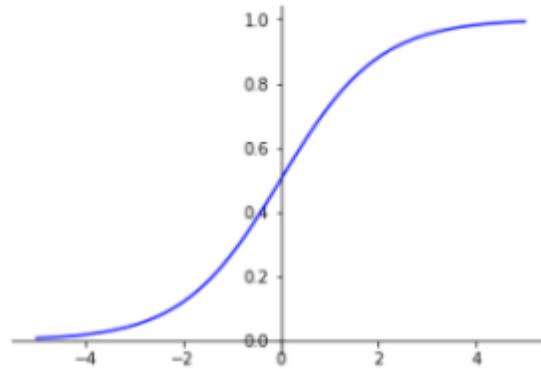


Figure 3-12 : La fonction d'activation Sigmoid

C. SoftMax :

La fonction Softmax permet de transformer un vecteur réel en vecteur de probabilité.

On l'utilise souvent dans la couche finale d'un modèle de classification, notamment pour les problèmes multiclasse.

Dans la fonction Softmax, chaque vecteur est traité indépendamment. L'argument axis définit l'axe d'entrée sur lequel la fonction est appliquée.

$$f_{softmax} = \frac{e^x}{\sum x_i} \quad (3.3)$$

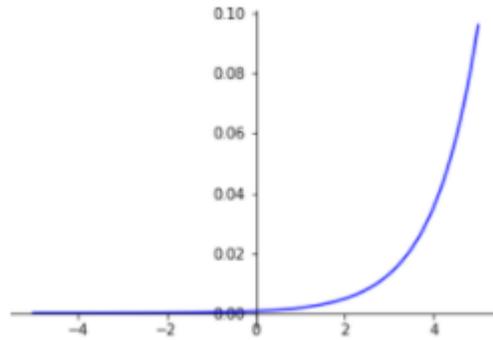


Figure 3-13 : La fonction d'activation SoftMax

3.5.4 Les architectures du réseau de neurones les plus courants

De nos jours, les CNN sont considérés comme les algorithmes les plus largement utilisés parmi les inspirés des techniques d'Intelligence Artificielle (IA). L'histoire de CNN commence par expériences menées par les neurobiologistes Hubel et Wiesel (1959, 1962) [59]. Leur travail a fourni une plate-forme pour de nombreux modèles cognitifs, et CNN a remplacé presque tous ceux-ci. Au fil des décennies, différents efforts ont été menés pour améliorer les performances des CNN.

Il existe de nombreuses architectures CNN réputées. Les architectures CNN les plus populaires sont :

A. VGG

L'utilisation réussie des CNN dans les tâches de reconnaissance d'image a accéléré la recherche en conception architecturale. À cet égard, Simonyan et al ont proposé un principe de conception simple et efficace pour les architectures CNN. Leur architecture, nommé comme le Groupe visuel Géométrie (VGG) de l'université d'Oxford [60], puis Simonyan et Zisserman développée VGG architecture de 16 couches (VGG-16) et 19 couches (VGG-19) [60], Sa principale réalisation a été pour compenser un des réseaux plus profonds par remplacé les filtres 11x11 et 5x5 avec une pile de 3x3 couche de filtres. L'utilisation de filtres de petite taille offre un avantage supplémentaire de faible complexité de calcul en réduisant le nombre de paramètres.

D. ResNet

Kaiming He et al. [61] ont développé un réseau résiduel (ResNet). Cette architecture CNN présente des connexions de saut uniques et une utilisation essentielle de la normalisation par lots (Batch Normalization). Encore une fois, l'architecture n'a pas de couches entièrement connectées à la fin du réseau. Le principal inconvénient de ce réseau est qu'il est très coûteux à évaluer en raison de la vaste gamme de paramètres. Cependant, jusqu'à présent, ResNet est considéré comme un modèle de réseau neuronal convolutif à la pointe de la technologie et constitue l'option par défaut pour l'utilisation des ConvNets dans la pratique. Il avait été le gagnant de l'ILSVRC 2015.

E. Inception- V1

Inception-V1 a été le gagnant du concours 2014-ILSVRC, Il est également connu comme GoogleNet, Il a été développé par une équipe de Google (Christian Szegedy et al). [62]

C'est une architecture utilise des modules Inception, ce bloc encapsule des filtres de différentes tailles (1x1, 3x3 et 5x5) pour capturer des informations spatiales à différentes échelles, suivi du filtre Concat qui permet de concaténer les résultats des filtres. En outre, la densité de la connexion a été réduite en utilisant la mise en commun moyenne globale à la dernière couche, au lieu d'utiliser une couche entièrement connectée. Ces réglages de paramètres ont provoqué une diminution significative du nombre de paramètres de 60 millions à 4 millions de paramètres.

F. Xception

Xception l'architecture est une architecture de création extrême proposée par François Chollet de Google [63], entièrement sur la base de la profondeur des couches de convolution séparables qui ont été présentés par

Laurent Siphre en 2014 [64], est une convolution de la profondeur (convolution de canal sage) suivi d'une convolution ponctuelle (convolution 1×1).

G. MobileNet-v1

MobileNet est une architecture légère développée par Howard et al. [65] de Google, le modèle MobileNet est conçu pour être utilisé dans des applications mobiles, et c'est le premier modèle de vision par ordinateur mobile de TensorFlow, cette architecture utilise des convolutions séparables en profondeur. Il réduit considérablement le nombre de paramètres par rapport au réseau avec des convolutions régulières avec la même profondeur dans les filets. Il en résulte des réseaux de neurones profonds légers. Howard et al introduisent deux hyperparamètres globaux simples qui font un compromis efficace entre latence et précision. Ces hyperparamètres permettent au constructeur de modèles de choisir le modèle de la bonne taille pour leur application sur les contraintes du problème.

3.6 Conclusion

Dans ce chapitre, nous discutons d'abord de l'apprentissage en profondeur, y compris son bref historique, puis nous mettons en lumière les réseaux de neurones et ces différents types. Ensuite, nous avons concentré notre attention sur les CNN, leurs différentes architectures et couches. Enfin, nous avons donné quelques exemples.

Chapitre 4

Résultats Expérimentaux et Réalisation

4.1 Introduction

Ce chapitre est le cœur de notre travail, dans lequel nous expliquerons tout le processus par lequel nous sommes passés pour développer notre système et réaliser le projet.

Tout d'abord, nous allons représenter l'ensemble de définition et d'architecture des systèmes embarqués et jeter un coup d'œil au matériel, en donnant les spécifications de chaque pièce et ce à quoi elle sert dans le projet.

De même pour les logiciels, un aperçu général, comment ils sont installés (avec les commandes utilisés) et leur rôle dans le projet bien sûr.

Ensuite, nous aborderons les différentes applications que nous avons implémentées dans le système, les matériaux utilisés dans chacune d'elles et comment elles fonctionnent.

4.2 Outils utilisés

De nombreux outils ont été utilisés pour réaliser ce travail, qu'il s'agisse de matériel ou de logiciels. Nous allons donc examiner la liste des outils et passer en revue chaque élément.

4.2.1 Hardware

A. Raspberry Pi 3 Model B+

➤ **Aperçu général :**

Le Raspberry Pi est un ordinateur mono carte pouvant se connecter à un téléviseur, à un clavier et disposant d'une connectivité Wi-Fi et Bluetooth. Il se boote depuis la carte micro-SD et fonctionne sous OS Linux (Raspberry Pi OS basé sur Debian) ou une version spéciale modifiée de Windows (Windows 10 IoT Core Dashboard). Il est fourni sans boîtier, alimentation, clavier, écran et souris dans le but de diminuer le coût et de favoriser l'utilisation de matériel de récupération.

Le Raspberry Pi 3 modèle B+ (Pi3B+) est le dernier de la génération d'ordinateurs Raspberry Pi 3 prenant en charge plus de RAM et offrant des performances de CPU, de GPU et d'E/S considérablement améliorées

Le Pi3B+ est disponible avec 1 giga-octets de SDRAM LPDDR2.

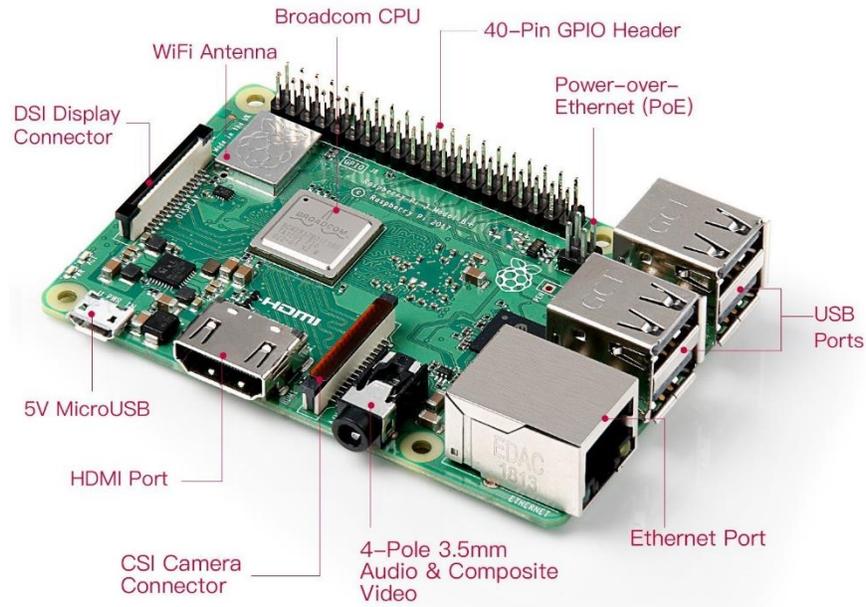


Figure 4-1 : Les ports de Raspberry Pi 3 model B+.

Le Raspberry Pi peut effectuer des tâches d'un PC de bureau (feuilles de calcul, traitement de texte, jeux). Il peut également diffuser des vidéos en haute définition grâce à son GPU Broadcom VideoCore IV (permet le décodage des flux Blu-ray full HD).

Cette carte nécessite une carte SD munie d'un OS, une alimentation, un clavier USB, une souris USB, un boîtier et des câbles. Pour préparer une carte SD bootable, il faut disposer d'un PC avec lecteur de carte.

➤ **Caractéristiques :**

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- 1 GO DE SDRAM LPDDR2
- LAN sans fil 2,4 GHz et 5 GHz IEEE 802.11.b/g/n/ac, Bluetooth 4.2, BLE
- Gigabit Ethernet sur USB 2.0 (débit maximal de 300 Mbps)
- Connecteur GPIO 40 broches étendu
- HDMI pleine grandeur
- 4 ports USB 2.0
- Port caméra CSI pour la connexion d'une caméra Raspberry Pi
- Port d'affichage DSI pour la connexion d'un écran tactile Raspberry Pi
- Sortie stéréo 4 pôles et port vidéo composite

- Port Micro SD pour charger votre système d'exploitation et stocker des données
- Entrée d'alimentation CC 5V/2,5A
- Prise en charge de la technologie PoE (Power-over-Ethernet) (nécessite un adaptateur PoE séparé).

➤ **Les ports E/S du Raspberry Pi :**

Dans le Raspberry Pi, les ports GPIO (General Purpose input/output) sont des entrées /sorties pour usage général, ils permettent de se communiquer avec d'autres interfaces et les broches GPIO peuvent être commutées (multiplexées) dans divers autres modes soutenus par des blocs périphériques dédiés tels que I2C, UART et SPI.

En plus des options de périphériques standard que l'on trouve sur les anciens Pis, des périphériques I2C, UART et SPI supplémentaires ont été ajoutés à la puce BCM2837B0 et sont disponibles comme options de mux supplémentaires sur le Pi3.

Cela donne aux utilisateurs beaucoup plus de flexibilité lors de la fixation de matériel supplémentaire par rapport aux anciens modèles.

Pour utiliser une broche dans un Raspberry Pi, il y a deux modes à programmer, soit le mode BOARD ou le mode BCM.

- **BOARD :** (accessible par la ligne de code `GPIO.setmode(GPIO.BOARD)`) spécifie que vous vous référez aux broches par le numéro de la broche dans la fiche - c'est-à-dire les numéros imprimés sur la carte (par exemple P1) et au milieu des diagrammes ci-dessous.
- **BCM :** (accessible par la ligne de code `GPIO.setmode(GPIO.BCM)`) signifie que vous vous référez aux broches par le numéro de "Broadcom SOC channel", ce sont les numéros après "GPIO" dans les rectangles verts autour de l'extérieur des diagrammes ci-dessous :

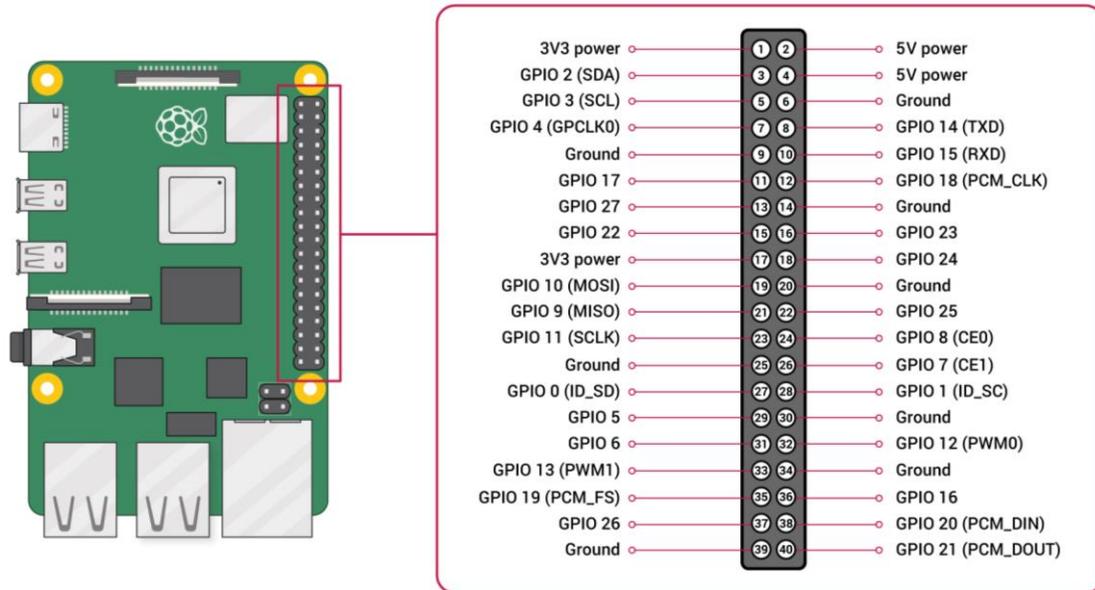


Figure 4-2 : Les différents pins de Raspberry Pi et ses nombres.

B. Pi Camera V2

➤ Aperçu général :

Le module caméra Raspberry Pi v2, avec son capteur Sony IMX219 de 8 mégapixels, peut être utilisé pour prendre des vidéos haute définition. Il supporte les modes vidéo 1080p30, 720p60 et VGA90, ainsi que prendre des photos. Il se fixe via un câble ruban de 15 cm sur le port CSI du Raspberry Pi.

La caméra fonctionne avec tous les modèles de Raspberry Pi (le nôtre inclus). On peut y accéder via les API MMAL et V4L, y compris la bibliothèque Python Picamera qui est celle que nous utiliserons.

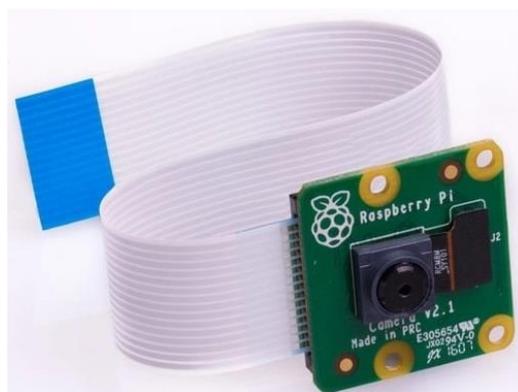


Figure 4-3 : Raspberry Pi Camera Module v2.

➤ **Caractéristiques :**

- Capteur 8 Mégapixels
- Dimension du capteur 3280 x 2464 pixels
- Résolutions vidéo 1080p30, 720p60 et 640x480p90
- Dimension du pixel 1,4 μm X 1,4 μm avec technologie OmniBSI
- Dimension de la partie optique 1/4"

Puisque toutes les applications de notre système sont basées sur la vision par ordinateur, ce sont les deux principaux composants, un Raspberry Pi comme ordinateur ou unité de traitement principale et une caméra pour la visualisation et la capture des entrées.

C. Afficheur LCD I2C 20*4

➤ **Aperçu général :**

Il s'agit d'un module d'affichage LCD 20x4 avec interface I2C haute vitesse conçue pour les microcontrôleurs.

Il est capable d'afficher 20 caractères sur quatre lignes, des caractères blancs sur un fond bleu.

Il utilise l'interface de communication I2C. Avec cette interface, seulement 2 broches sont nécessaires (le 3ème et le 5ème pin dans le raspberry pour les communications SDA et SCL de l'I2C respectivement) pour afficher les informations sur n'importe quel projet basé sur RPi. Cela permettra d'économiser au moins 4 broches pour d'autres utilisations. Tous les connecteurs sont standard XH2.54 (type Breadboard). Vous pouvez le connecter directement avec un fil de liaison.

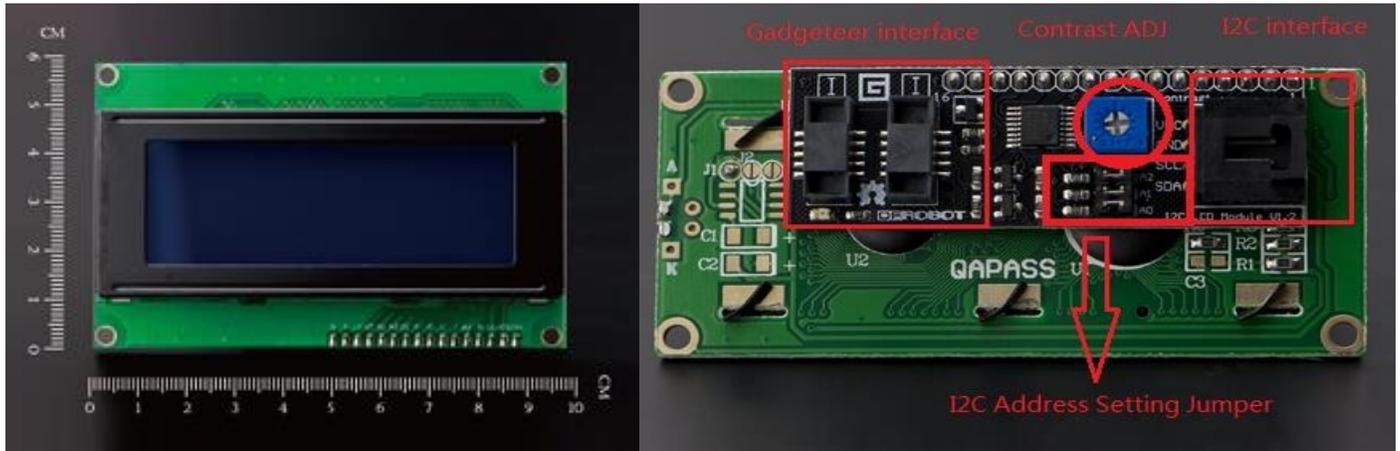


Figure 4-4 : Spécifications et taille du module d'affichage LCD I2C 20x4 avec sac à dos.

➤ **Caractéristiques :**

- Adresse I2C : 0x20-0x27 (0x20 par défaut)
- Nombre de caractères : 20 caractères x 4 lignes
- Rétro-éclairage LED bleu avec couleur blanche des caractères
- Contraste réglable
- Tension d'alimentation : 5V (via Pin) 3.3V (via IDC10)
- Interface : I2C/TWI x1, IDC10 x2
- Direction de vue : Grand angle de vue
- Taille des points : 0,55 x 0,55 mm (0,02x0,02")
- Pas des points : 0,60 x 0,60 mm (0,023x0,023")
- Taille des caractères : 2,96 x 4,75 mm (0,11x0,19")
- Pas des caractères : 3,55 x 5,35 mm (0,14x0,21")
- Size : 98x60x24mm (3.86x2.36x0.94")

D. Autres Composants

D'autres composants électroniques ordinaires ont été utilisés dans le circuit du système, tels que des boutons poussoirs pour alterner entre les applications, des lampes LED pour indiquer quelle application est en cours, un buzzer pour servir d'alarme, un haut-parleur pour les guides vocaux, un câble Ethernet RJ45 pour les connexions RPi-PC, des résistances, des jumpers, des batteries, etc.

E. Schéma de câblage

Le schéma de câblage du système réalisé est représenté dans la figure 3.6 ci-dessous :

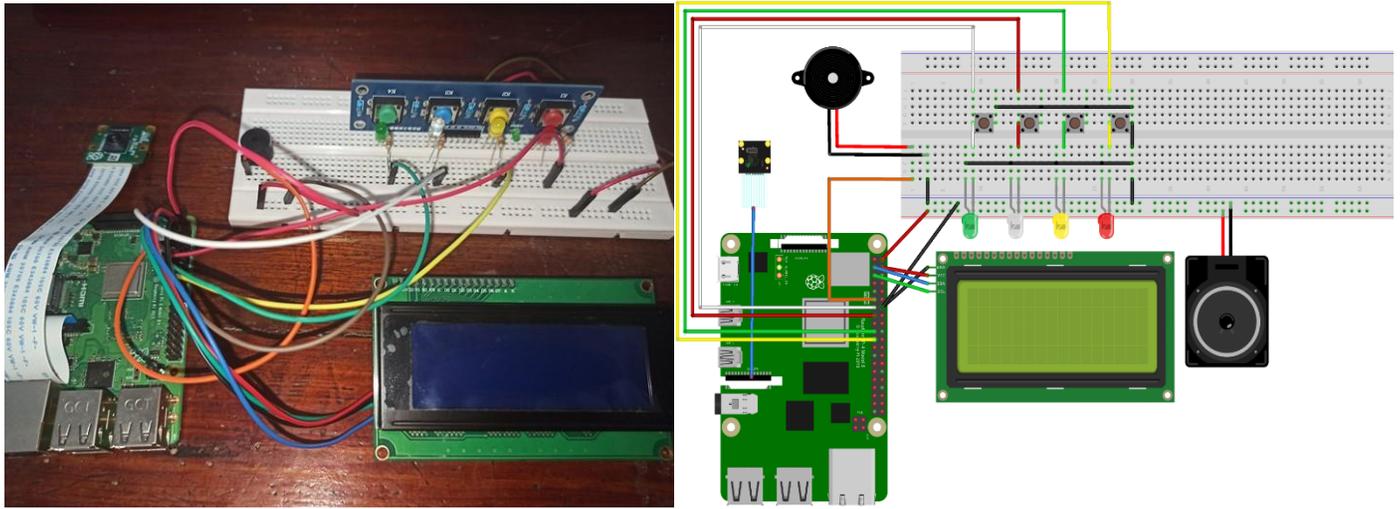


Figure 4-5 : Schéma de câblage du système réel et sous le logiciel Fritzing.

4.2.2 Software

Raspberry Pi OS (anciennement Raspbian) : le système d'exploitation officiel supporté est le système d'exploitation que nous avons installé pour exécuter nos applications. Il s'agit d'un système d'exploitation basé sur Debian Linux pour Raspberry Pi. Ce système d'exploitation est hautement optimisé pour la gamme RPi d'ordinateurs compacts à carte unique équipés de CPU ARM. Il fonctionne sur presque tous les Raspberry Pi.

Visual Studio Code : Visual Studio Code (VSCode) est un éditeur de code source léger et puissant développé par Microsoft. Il est doté d'une interface interactive très agréable et possède plusieurs fonctions d'aide et d'extensions. Il supporte un large éventail de langages de programmation, y compris celui que nous allons utiliser, Python.

Python : Python est un langage de programmation de haut niveau, interprété, à objets orientés, utilisé pour un très large ensemble d'applications et de domaines tels que le développement de jeux, la création d'applications

de bureau ainsi que le développement d'applications. Il est très convivial pour les débutants et facile à apprendre et à utiliser. Nous avons utilisé la version 3.9.5.

OpenCV : Une bibliothèque de vision par ordinateur à code source ouvert qui a été créé pour regrouper les applications de vision par ordinateur. Elle dispose de tonnes d'algorithmes optimisés (plus de 2500) à cet égard. Elle supporte une variété de systèmes d'exploitation, de langages de programmation, de matériels et de logiciels. Installé avec la commande : `pip install opencv-contrib-python`

Dlib : une boîte à outils moderne en C++ (qui a un très bon support pour python, le langage que nous avons utilisé) contenant des algorithmes d'apprentissage automatique et des outils pour plusieurs applications allant des dispositifs embarqués jusqu'aux environnements de calcul à haute performance. Le processus d'installation est un peu difficile et nécessite de nombreuses étapes, mais la commande exacte de la bibliothèque est : `pip install dlib`.

TensorFlow : est une bibliothèque logicielle de bout en bout, gratuite et à code source ouvert, utilisée pour construire et déployer des modèles d'apprentissage machine, aussi bien pour les débutants que pour les professionnels, développé par Google. Installé avec la commande : `pip install --upgrade tensorflow`

Keras : est une API d'apprentissage profond de haut niveau, simple, flexible et puissante, dont le backend est TensorFlow. Elle vise à faciliter et à minimiser les actions requises pour résoudre les problèmes, ce qui en fait le Framework de Deep Learning le plus utilisé. Installé avec la commande : `pip install keras`

L'entraînement des modèles d'apprentissage profond avec des millions de neurones sur un ordinateur ordinaire prendrait un temps énorme, sans compter les essais et les échecs, il fallait alors trouver un moyen de le faire. « Google Colaboratory » était ce que nous recherchions.

Google Colaboratory : ou « Colab » est un service cloud qui donne un accès à des ressources computationnelles, notamment des GPU, ce qui le rend très adapté aux applications d'apprentissage profond comme les nôtres, permettant à quiconque d'écrire et d'exécuter du code python directement sur le navigateur. Il est fourni gratuitement par Google.



Figure 4-6 : Logos des outils utilisés dans cette étude.

4.3 Les applications du système

L'objectif de ce projet de fin d'études est de développer des options automatiques pour les voitures intelligentes en utilisant des techniques d'IA. Dans les chapitres précédents, nous avons parlé de certaines de ces applications et de l'état de l'art des méthodes utilisées pour le développement.

Notre système contient trois applications principales, toutes basées sur la vision par ordinateur, la première est un démarreur de reconnaissance de visage, la deuxième application est la détection de la fatigue et de la somnolence du conducteur et la dernière est la détection des panneaux de signalisation. Nous allons examiner chacune d'entre elles individuellement et expliquer le principe de fonctionnement, les résultats obtenus et les limitations de l'application.

4.3.1 Démarreur avec reconnaissance faciale

L'idée de cette application est d'utiliser le système embarqué pour reconnaître le visage du conducteur afin de donner un signal de sortie pour initier le démarreur et mettre la voiture en marche. La reconnaissance faciale est aussi utilisée pour ajuster certaines caractéristiques de la voiture selon les préférences de chaque personne (visage).

L'état de l'art de la reconnaissance des visages est le réseau neuronal convolutif de l'apprentissage profond que nous avons expliqué en détail dans le chapitre 3, nous allons donc l'utiliser maintenant et expliquer le processus étape par étape comme montre dans la figure 4.8 :

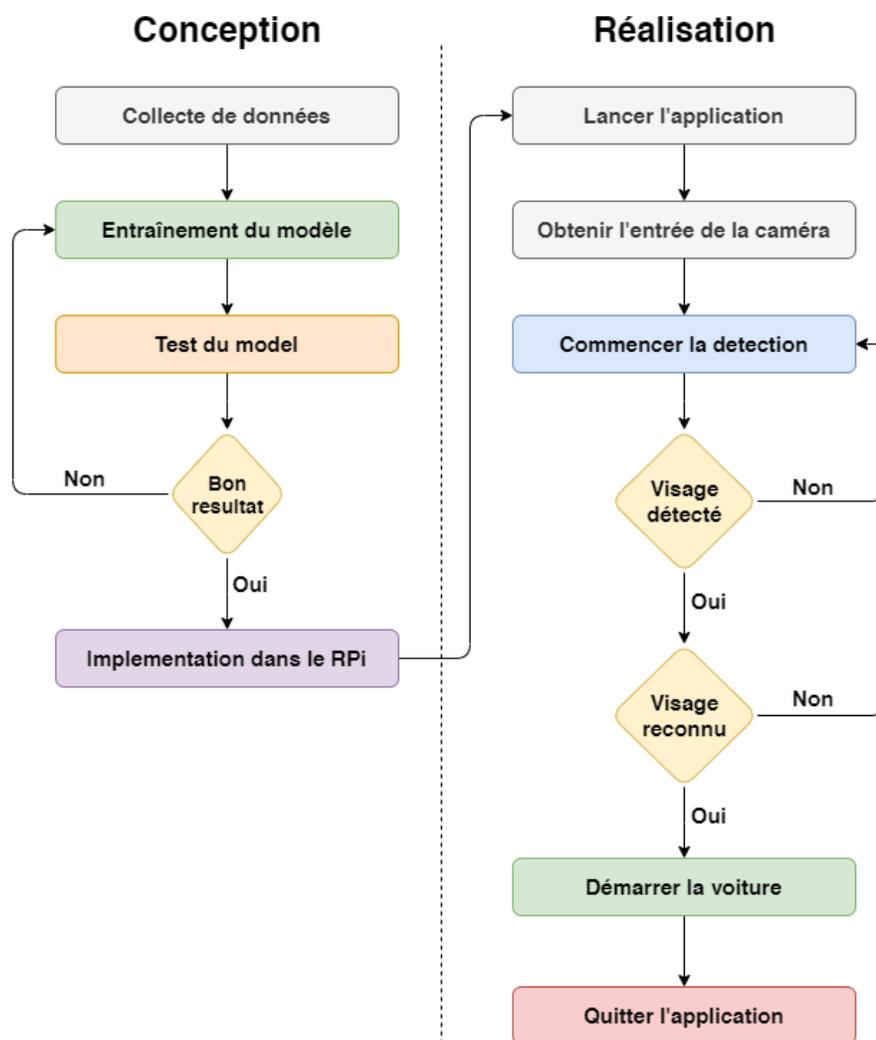


Figure 4-7 : Organigramme de processus de démarreur avec reconnaissance faciale.

A. Préparation des données

➤ **Récupération de la base de données**

Tout d'abord, nous devons générer notre propre base de données avec nos visages. Pour ce faire, nous avons écrit un programme qui utilise le détecteur de visage frontal en cascade de Haar³ pour détecter les visages dans l'image. Si le visage est détecté, le programme prend un nombre prédéfini de photos et les stocke dans des dossiers d'entraînement et de test. Si le visage n'est pas détecté, un message "aucun visage détecté" est affiché et le programme arrête de prendre des photos jusqu'à ce qu'un visage soit à nouveau dans le cadre.

Nous avons pris 400 photos pour le jeu d'entraînement et 100 pour le test (les jeux de validation et de test sont les mêmes) de taille 224*224 pour une raison que nous verrons plus tard.

➤ **Prétraitement des images**

Les images ont été prises dans de bonnes conditions d'éclairage, de sorte que le seul prétraitement requis est la normalisation des valeurs des pixels, qui est effectuée directement par les fonctions intégrées de prétraitement de l'architecture utilisée dans Keras.

➤ **Augmentation de nombre des images**

L'augmentation est faite en appliquant certaines fonctions sur les images qui les changent un peu, mais c'est toujours la même image.

³ Haar cascade est un algorithme d'apprentissage machine qui utilise les caractéristiques de détection des bords ou des lignes proposées par Viola et Jones dans leur article de recherche "Rapid Object Detection using a Boosted Cascade of Simple Features" publié en 2001 pour détecter des objets, dans notre cas des visages.

Les fonctions d'augmentation sont fournies par l'API d'apprentissage profond Keras et sont simples à appliquer, les fonctions utilisées sont illustrées dans la figure suivante :



Figure 4-8 : Exemple des photos d'entraînement augmentée avec les fonctions utilisées sur la base de données.

L'augmentation a permis de faire passer le nombre d'images de la base de données de 400 à 1200, ce qui est suffisant pour entraîner le CNN.

B. Entraînement du modèle

Pour la reconnaissance des visages, nous avons utilisé une architecture CNN pré-entraînée parmi celles expliquées dans le chapitre 3, section 5.4.

Au début, nous avons utilisé l'architecture VGG16 et elle a parfaitement fonctionné pour la reconnaissance des visages, mais nous avons eu quelques problèmes lors de l'implémentation dans le RPI. Nous avons donc changé pour MobileNetV2, car il est conçu pour les mobiles, comme son nom l'indique, il s'agit d'une architecture légère pour les systèmes embarqués.

Une autre raison est que les dispositifs mobiles et les systèmes embarqués utilisent tous l'architecture ARM dans leurs CPU.

L'architecture MobileNetV2 a aussi donnée un résultat excellent durant l'entraînement du modèle pour 2 époques seulement.

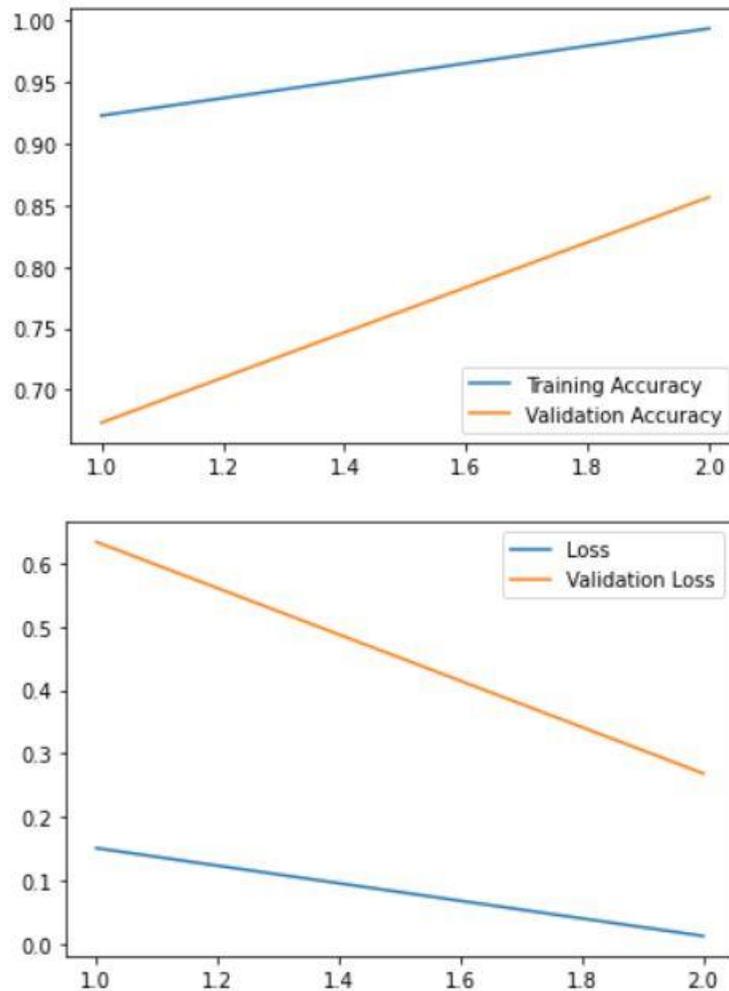


Figure 4-9 : Les courbes de précision et d'erreur du MobileNetV2 pendant 2 époques d'apprentissage.

Pour les valeurs exactes, le modèle a obtenu loss: 0.0134 - acc: 0.9937 - val_loss: 0.2690 - val_acc: 0.8567

C. Test et implémentation

➤ Tests

Une fois l'entraînement terminé, nous sauvegardons notre modèle au format h5 à l'aide de la fonction Keras `model.save()`, puis pour le tester, il suffit de charger le modèle avec `model.load()` dans un programme séparé, d'utiliser la caméra comme entrée (les frames de la vidéo doivent avoir même taille et même

prétraitement que l'ensemble d'entraînement) et ensuite le modèle donnera la prédiction basée sur l'indice de la liste des classes. Nous avons deux classes, `class[0] = "hicham"` et `class[1] = "mahmoud"`.

Nous avons fixé un seuil de prédiction à 0.85, donc si le pourcentage de la classe prédite est supérieur à 85% le nom de la classe (personne, visage) apparaît dans le cadre ou sinon il écrit "inconnu".



Figure 4-10 : Reconnaissance faciale avec l'architecture de CNN MobileNetV2 avec une personne non-reconnue.

➤ **Implémentation**

Puisque tout a bien fonctionné, deux dernières étapes sont nécessaires. La première est d'implémenter le modèle dans le RPi, ce qui se fait facilement en copiant les fichiers (le code de test python et le fichier du modèle h5) dans la RPi et en l'exécutant. Facile ! Nous avons ajouté quelques lignes de code pour faire apparaître le nom sur l'écran LCD avec un bip du buzzer et un message de bienvenue du haut-parleur.

La deuxième étape consiste à configurer le schéma de circuit du démarreur de la voiture pour qu'il soit excité par notre système embarqué si le visage est reconnu. Ce que nous avons fait dans la figure ci-dessous :

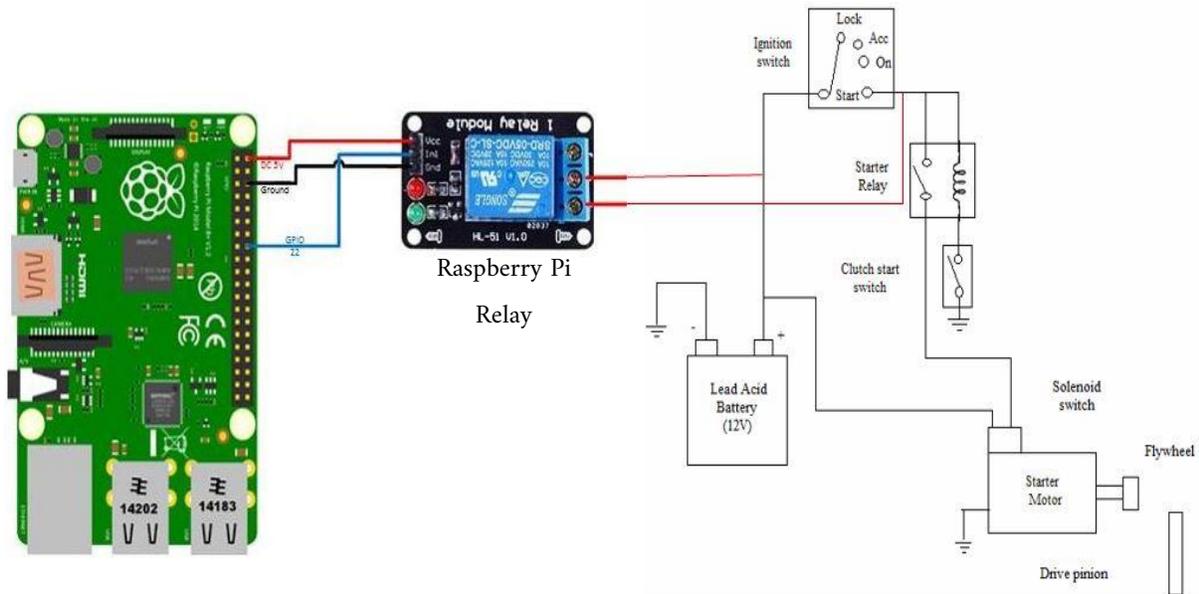


Figure 4-11 : Circuit de démarreur de la voiture commandé par le Raspberry Pi.

Lorsqu'un visage est reconnu, nous réglons la valeur GPIO22 sur HIGH pour commander le relais (le premier relais, celui que nous avons installé) afin que le courant passe par lui plutôt que par le commutateur d'allumage. Lorsque le relais est fermé, le courant passe de la batterie au second relais (celui qui est supposé de se fermer à partir du commutateur d'allumage), puis au solénoïde du démarreur qui, à son tour, ferme le circuit direct entre la batterie de la voiture et le démarreur. Et alors la voiture démarre.

Après 5 secondes de reconnaissance constante, la valeur de GPIO22 est mise à LOW à nouveau afin de couper l'alimentation au circuit de démarrage, juste comme vous tournez la clé pendant 4-6 secondes puis la lâchez.

4.3.2 Système de détection de la fatigue

A. Préparation des données

➤ Récupération de la base de données

La base de données utilisée dans cette expérimentation est la base de détection de bâillement (*Yawning Detection Dataset: YawDD*). Elle se compose d'une séquence des vidéos acquise à partir d'une caméra Canon avec une résolution de 640*480 pixels à 30 images par seconde (30 FPS) au format AVI sans audio.



Figure 4-12 : Exemples de données dans YawDD.

L'étape la plus importante dans l'apprentissage des réseaux de neurone profond est de bien préparer l'ensemble d'apprentissage et de validation.

Nous avons utilisé uniquement les vidéos où la caméra est placée sur le tableau de bord et à partir de celles-ci, nous avons utilisé que les vidéos des personnes qui ne portent pas de lunettes car le détecteur de visage utilisé (celui de la bibliothèque Dlib qui est basé sur l'algorithme de l'apprentissage machine HOG ne fonctionne pas bien si le visage n'est directement en face de la caméra). En Total 14 vidéos de YawDD ont été utilisées.

Les images sont classées dans des dossiers « état ouvert ou fermé » en fonction de l'état des yeux et de la bouche. Notre base de données est donc composée de deux dossiers principaux, Yeux et Bouche, à l'intérieur

de chaque dossier il y a deux dossiers, un pour l'entraînement et l'autre pour la validation du modèle (pour les données de test, nous avons utilisé celles de la validation). 9 vidéos ont été choisies pour le dossier d'entraînement et les 5 autres pour la validation.

Avant de séparer chaque image de la vidéo, le prétraitement habituel doit être effectué avec une fonction supplémentaire : le redimensionnement des images de 640*480 à 320*240, car elles sont trop grandes et prendraient trop de puissance de traitement. En plus de la mise à l'échelle des gris et de la normalisation, bien sûr.

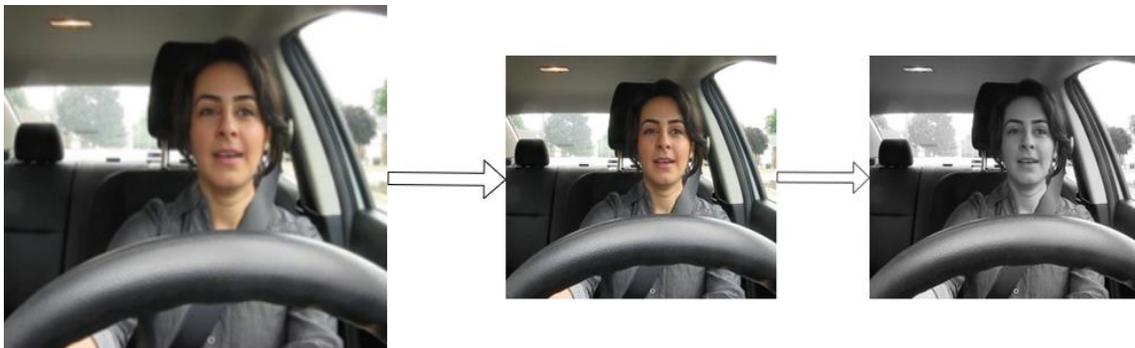


Figure 4-13 : Prétraitement des données, redimensionnement et mise à l'échelle des gris.

La classification des images dans cette étape est faite en calculant le rapport d'aspect des yeux (EAR) et de bouche (MAR).



Figure 4-14 : La détection de l'état des yeux et de la bouche avec EAR et MAR.

B. Entraînement du modèle

D'après notre étude du Master (*Etude comparative entre les différentes architectures des réseaux de neurones convolutifs (CNNs) pour la détection de la fatigue du conducteur*), nous avons conclu que l'architecture CNN la mieux adaptée à la détection de la fatigue et aux systèmes embarqués est l'architecture MobileNetV2.

Le modèle a été entraîné sur l'ensemble de données construit, nous avons ajouté quelques couches supérieures spécifiées pour notre problème qui étaient GlobalAveragePooling2D(2*2), 1024 neurone Dense layer, Dropout(0.4), Dense(224), Dropout(0.6) et une dernière couche FC (Dense) avec le nombre de nos classes qui est de 2 pour chaque modèle.

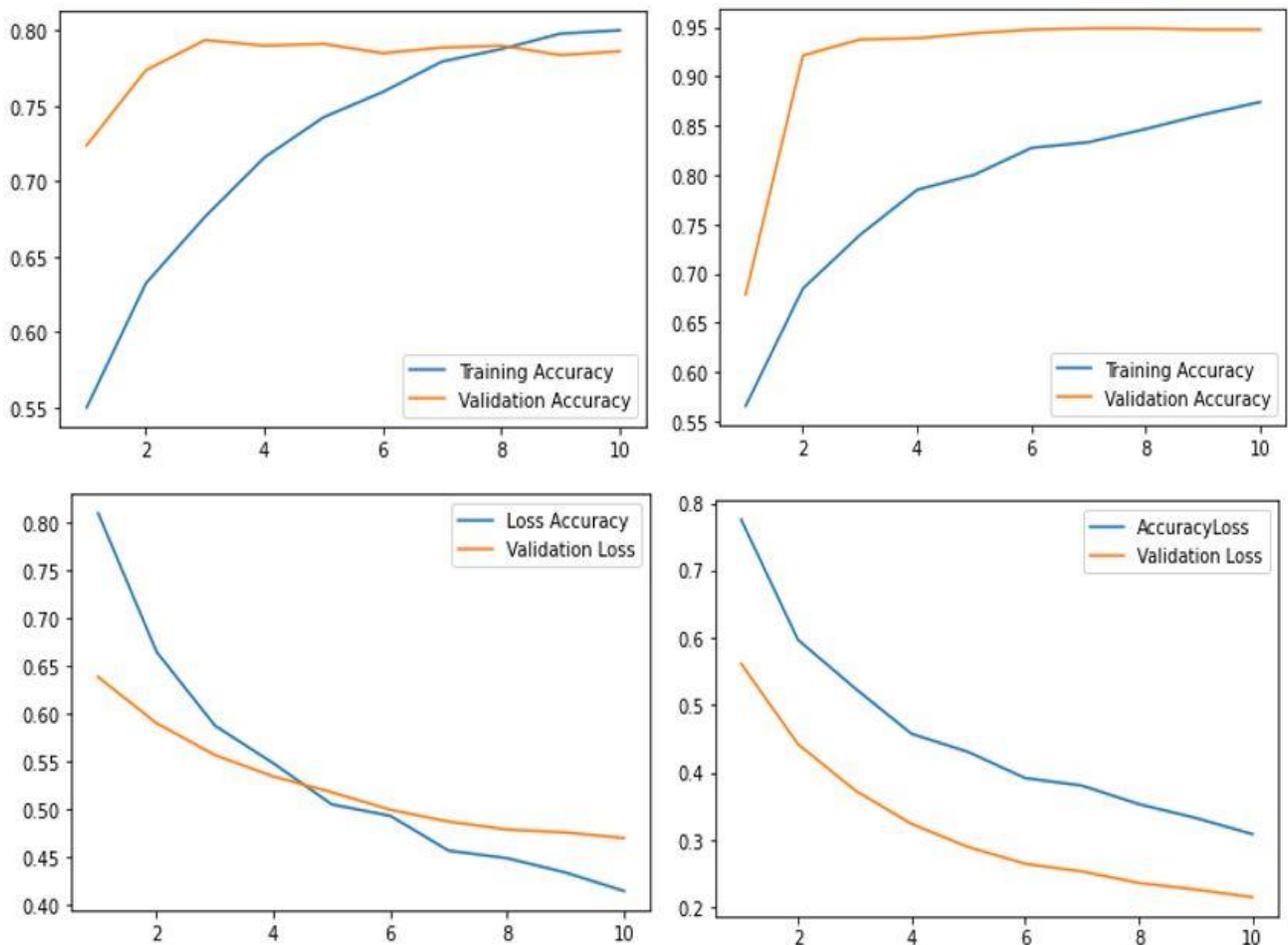


Figure 4-15 : Les courbes de précision et l'erreur du modèle MobileNetV2 pendant 10 époques d'apprentissage pour yeux et de bouche à gauche et à droite respectivement.

Pour les valeurs exactes,

- Le modèle des yeux a obtenu loss: 0.4148 - accuracy: 0.8002 - val_loss: 0.4703 - val_accuracy: 0.7862.
- Le modèle de la bouche a obtenu loss: 0.3088 - accuracy: 0.8740 - val_loss: 0.2153 - val_accuracy: 0.9475.

C. Test et implémentation

➤ Tests

Cette architecture a obtenu un très bon score dans tous les critères pour les yeux et la bouche, une grande précision d'apprentissage et de validation. De plus, il s'agit d'une architecture légère qui convient parfaitement au système intégré Raspberry Pi dans notre projet.

Nous devons noter que, comme pour les données d'entraînement, le modèle ne sera pas capable de classier des images normales mais seulement des images masquées. Ainsi, la même technique de prétraitement, détection du ROI et masquage de toutes les informations inutiles sera appliquée au flux vidéo de la PiCamera avant que le modèle puisse classer les images.



Figure 4-16 : Un exemple des images classifiées correctement avec le model MobileNetV2.

➤ Détection de la fatigue

D'après notre étude de Master la détection de fatigue est faite selon l'organigramme ci-dessous :

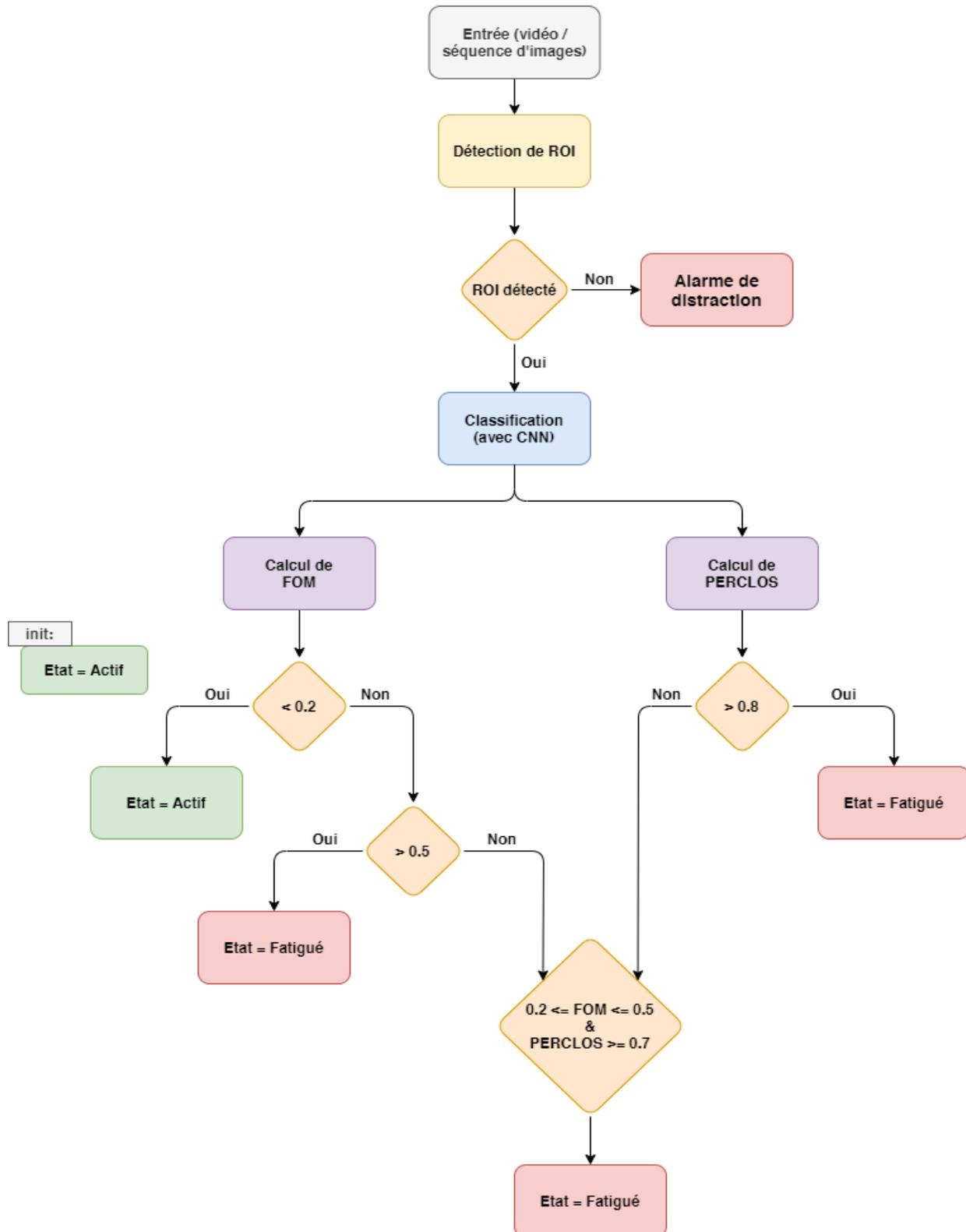


Figure 4-17 : L'organigramme de décision de la détection de la fatigue.

➤ **Implémentation**

Pour mettre en œuvre cette application dans le système embarqué, les mêmes conditions doivent être recrées, c'est-à-dire que la PiCamera doit être placée dans le tableau de bord et avoir des conditions d'éclairage acceptables.

Ensuite, le même code est lancé pendant la conduite, le PERCLOS et le FOM sont calculés pendant une unité de temps et le programme se comportera selon les conditions définies dans l'organigramme de la **figure 4.18** :

Lorsque le programme démarre, la LED verte est allumée, si le PERCLOS ou le FOM atteignent une valeur proche de l'état de fatigue, elle commence à clignoter et un message "soyez prudent, reposez-vous bientôt". Au moment où le PERCLOS ou le FOM atteint les valeurs seuils, la LED rouge s'allume, le buzzer se met à biper, l'écran LCD affiche " Veuillez-vous reposer MAINTENANT ! " avec le même message du haut-parleur.

D. Système de détection des distractions

Pour améliorer la sécurité et l'expérience de conduite, ainsi que pour ajouter des fonctionnalités plus utiles à notre système. Nous avons ajouté un système de détection des distractions pour empêcher le conducteur d'être distrait de la route.

Le système de détection de la fatigue a besoin de détecter le visage du conducteur pour calculer les paramètres PERCLOS et FOM mentionnés ci-dessus. Pour cela nous avons utilisé le *dlib.frontal_face_detector* mais dans ce cas particulier il a deux inconvénients :

- Il donne de fausses informations (reste bloqué sur l'ancien) lorsque le visage est légèrement tourné.
- Il est lourd à traiter pour le Raspberry Pi.

Donc, pour la détection des visages, nous avons utilisé le détecteur de visage frontal en cascade de Haar à la place, qui est plus rapide que celui de dlib mais plus sensible aux visages non frontaux et ne peut pas les détecter. Ce qui est précisément ce qu'il nous faut pour cette application.

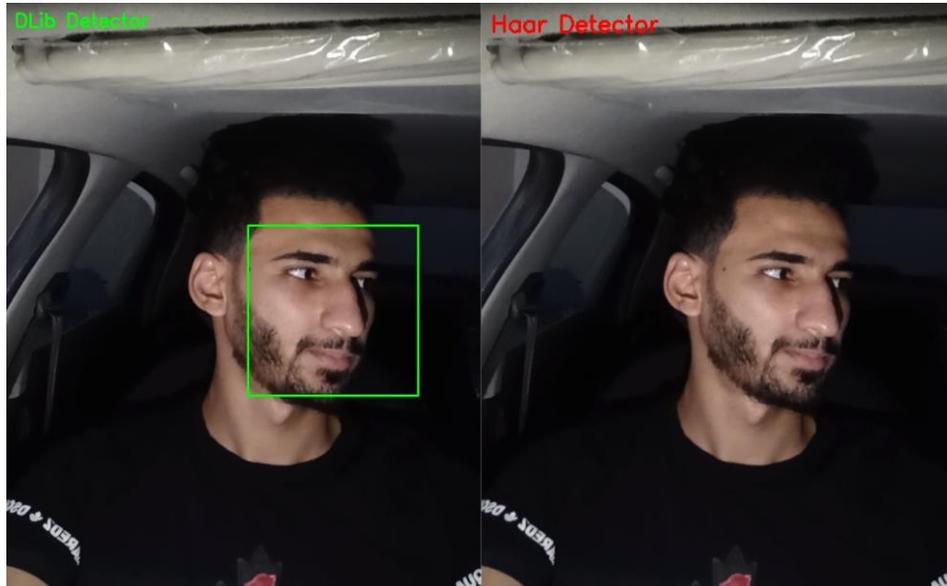


Figure 4-18 : Comparaison entre le détecteur de visage frontal de dlib et Haar.

Avant d'implémenter l'application au système, nous devons noter que le détecteur en cascade de Haar présente un problème de précision des fausses détections (Détections fausses positives). Mais comme l'échelle du visage et sa distance par rapport à la caméra dans la voiture sont presque toujours les mêmes, il est facile d'ajuster les paramètres par essai et erreur.

- **scaleFactor**: Combien la taille de l'image est réduite à chaque échelle d'image. Utilisé pour détecter les visages plus petits (plus loin) dans l'image, mais nous n'avons qu'un seul visage (celui du conducteur) donc il a été fixé à 1.05.
- **minNeighbors**: Ce paramètre contrôle combien de rectangles (voisins) doivent être détectés pour que la fenêtre soit étiquetée comme un visage. Nous avons essayé plusieurs valeurs pour aboutir à 8.
- **minSize**: Un tuple de largeur et de hauteur (en pixels) indiquant la taille minimale de la fenêtre. Les boîtes englobantes plus petites que cette taille sont ignorées. Nous avons trouvé que la meilleure valeur est (30,30).

Après avoir ajusté ces paramètres et trouvé les meilleurs pour détecter un visage dans la voiture, si aucun visage n'est détecté, cela signifie que le conducteur a détourné son visage de la route. Nous avons donc ajouté un code

dans le système de détection de la fatigue : si aucun visage n'est détecté pendant 3 secondes, le buzzer se déclenche pour alerter le conducteur.

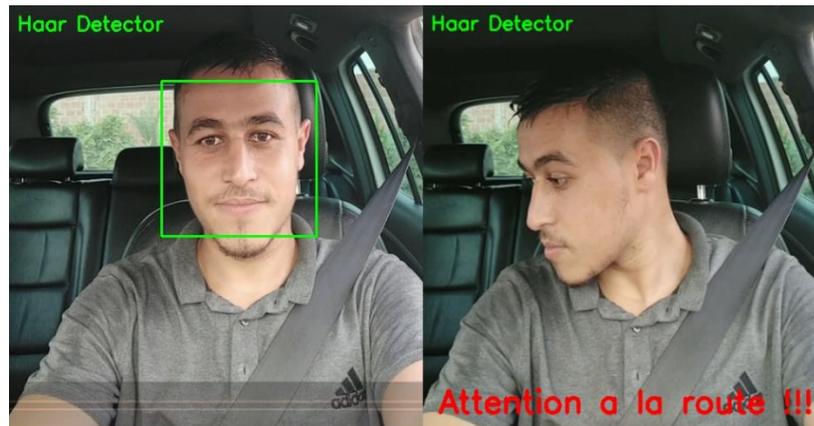


Figure 4-19 : Distraction du conducteur détectée avec le message d'alerte à l'écran.

4.3.3 Détecteur de panneaux de signalisation

Le processus de détection des panneaux de signalisation se divise en deux problèmes principaux. Le premier et le plus difficile est la détection de la région d'intérêt dans l'image capturée (détecter le panneau dans la vidéo de la route) et le second problème est la prédiction de la classe (nom) du panneau. Dans le temps limité et compte tenu de la difficulté de la première tâche, nous avons réussi à accomplir le problème de prédiction avec une grande efficacité et de bons résultats.

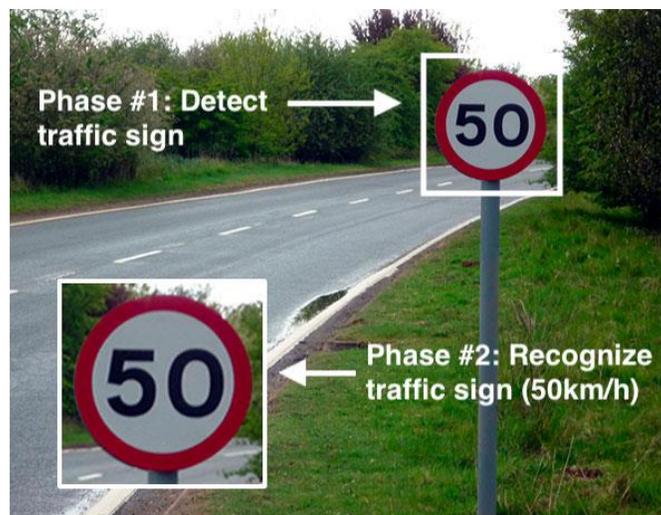


Figure 4-20 : Les deux étapes de la détection des panneaux de signalisation.

Le processus de détection est le suivant :

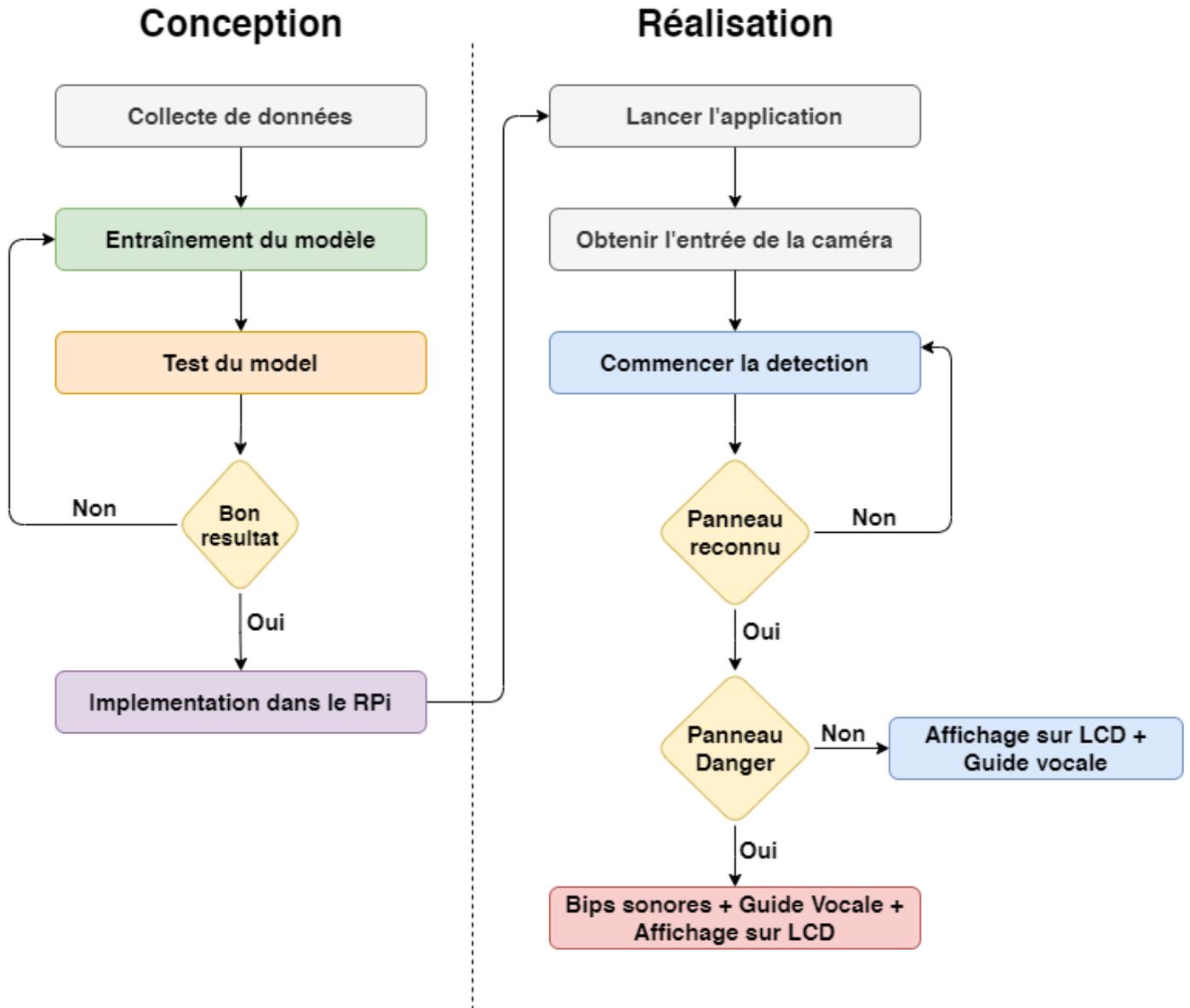


Figure 4-21 : Le processus de l'application de classification des panneaux de signalisation.

A. Préparation des données

Comme nous utilisons la méthode d'apprentissage profond pour la vision par ordinateur, connue pour sa soif de données, nous avons dû trouver un ensemble de données suffisamment grand pour entraîner le réseau de neurones convolutifs.

➤ Récupération de la base de données

Ensemble de données de reconnaissance de panneaux de signalisation allemands (*German Traffic Sign Recognition Dataset: GTSRB*) est un ensemble de données de classification d'images. Les images sont des photos de panneaux de signalisation qui sont classées en 43 classes et ont des conditions de lumière variables et des arrière-plans riches.



Figure 4-22 : Un exemple d'image de chacune des 43 classes de l'ensemble de données.

L'ensemble d'entraînement contient 39209 images étiquetées et l'ensemble de tests contient 12630 images non étiquetées. La distribution n'est pas uniforme, les plus grandes classes ont 10 fois plus d'images que les plus petites classes comme on peut voir dans la figure 4-9. C'est prévisible étant donné que dans la vie réelle, certains panneaux apparaissent plus fréquemment que d'autres, mais cela peut poser des problèmes lors de l'entraînement du réseau comme le surapprentissage que nous verrons plus tard comment le surpasser.

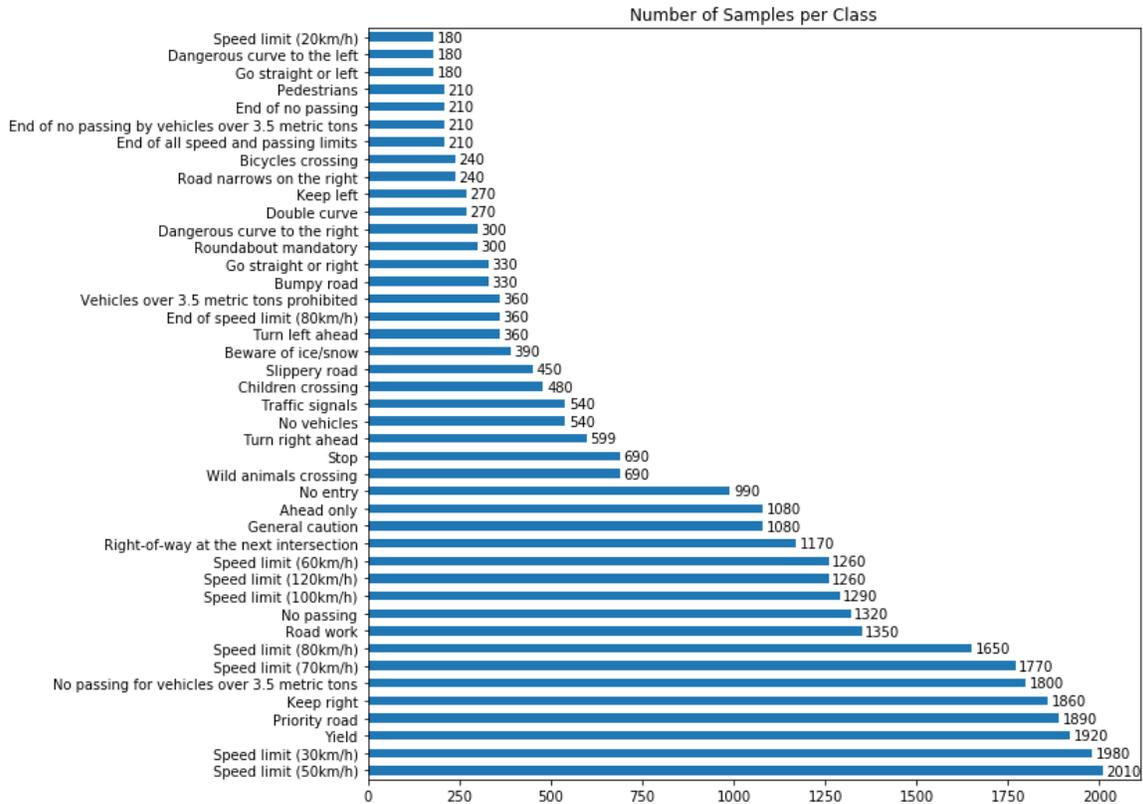


Figure 4-23 : La étiquettes et le nombre de chaque classe dans la base de données.

➤ Prétraitement des images

Avant d'alimenter le réseau avec les images, un prétraitement doit être effectué pour obtenir un résultat optimal et clair de chaque image.

- **Niveaux de gris** : La première étape consiste à transformer l'image des 3 canaux RVB (Rouge, Vert, Bleu) en un canal en niveaux de gris.
- **Égalisation** : L'image est ensuite égalisée pour obscurcir les zones sombres et blanchir les zones claires, ce qui améliore le contraste et la clarté de l'image.
- **Normalisation** : Nous normalisons ensuite les images en divisant les valeurs des pixels par 255 pour qu'elles soient comprises entre 0 et 1 afin de les injecter dans le réseau de neurones.



Figure 4-24 : Exemple d'une image prétraitée.

➤ **Augmentation de nombre des images**

Nous avons parlé précédemment de l'absence d'uniformité de l'ensemble de données, ce qui peut entraîner un surajustement du modèle. Une solution consiste à augmenter le nombre de données dans les petites classes pour obtenir un nombre égal d'images.

L'augmentation est faite en appliquant certaines fonctions sur les images qui les changent un peu, mais c'est toujours la même image.

Les fonctions d'augmentation sont fournies par l'API d'apprentissage profond Keras et sont simples à appliquer, les fonctions utilisées sont illustrées dans la figure suivante :



Figure 4-25 : Exemple d'une photo augmentée avec les fonctions utilisées sur la base de données.

Les étapes sont faites par ordre, donc l'augmentation a été faite après le prétraitement, pas comme montré dans la figure ci-dessus, mais pour une meilleure visualisation de ce que chaque fonction fait, nous avons utilisé celle-là.

➤ Répartition des données

Maintenant que les données sont prêtes, nous avons besoin d'une dernière étape pour entraîner le modèle : diviser les données en données d'entraînement, de validation et de test.

Au lieu de créer plusieurs dossiers, nous avons laissé toutes les images ensemble et utilisé une fonction de la bibliothèque **sklearn** appelée `train_test_split()`. Nous avons fixé le ratio de test et de validation à 0,2, ce qui signifie que si nous avons 1000 images, la division en fera 200 pour le test et les 800 restantes 160 pour la validation.

C'est la deuxième méthode utilisée pour éviter le sur-apprentissage.

B. Entraînement du modèle

Pour cette tâche, nous n'avons pas utilisé un réseau neuronal pré-entraîné, mais nous avons créé un modèle simple de réseau neuronal convolutif qui consiste en trois ensembles linéaires de couches qui sont présentées dans la **figure 4-12** ci-dessous.

La première a 2 Conv2D consécutifs avec un filtre et un kernel de 5*5 puis une couche de Maxpooling 2*2.

La 2ème est la même avec une couche de Dropout de 0.5 (qui est aussi utilisée pour éviter l'overfitting d'ailleurs)

Dans la troisième couche, nous aplatissons les formes des données pour qu'elles soient en une seule dimension avec la couche Flatten, puis nous ajoutons une couche entièrement connectée (Dense) de 500 neurones, une couche Dropout de 0,5 et une dernière couche dense avec un neurone pour chaque classe, soit 43 neurones.

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 28, 28, 60)	1560
conv2d_1 (Conv2D)	(None, 24, 24, 60)	90060
max_pooling2d (MaxPooling2D)	(None, 12, 12, 60)	0
conv2d_2 (Conv2D)	(None, 10, 10, 30)	16230
conv2d_3 (Conv2D)	(None, 8, 8, 30)	8130
max_pooling2d_1 (MaxPooling2D)	(None, 4, 4, 30)	0
dropout (Dropout)	(None, 4, 4, 30)	0
flatten (Flatten)	(None, 480)	0
dense (Dense)	(None, 500)	240500
dropout_1 (Dropout)	(None, 500)	0
dense_1 (Dense)	(None, 43)	21543

=====
Total params: 378,023
Trainable params: 378,023
Non-trainable params: 0

Figure 4-26 : Résumé des couches du réseau de neurones.

Ce modèle n'a que 378.023 paramètres (poids et biais), ce qui n'est pas beaucoup, et l'apprentissage se fait facilement sur un PC normal.

Nous avons fait l'entraînement sur un PC avec 8GB de RAM, un CPU intel core i7 4700MQ 2.40GHz et un GPU GTX 765M pour 50 époques, le modèle a cessé de s'améliorer à partir de la 20ème époque mais a donné de très bons résultats.

Ensuite, nous sommes tombés sur un problème de compatibilité, car le PI3B+ est capable d'exécuter python 3.5 seulement, Tensorflow 1.14.0 et Keras 2.2.5 et le PC a python 3.9.5, Tensorflow 2.5.0 et Keras 2.5.0 et depuis TF 1.15.0 les versions ne sont pas compatibles et ne peuvent pas se rencontrer. La mise à jour des

versions sur le Raspberry n'était pas possible et essayer de rétrograder les versions sur le PC a également posé de gros problèmes. La solution optimale a donc été d'utiliser Colab à nouveau puisque la mise à jour des versions est facile à réaliser.

L'entraînement a pris trop de temps dans Colab pour une raison inconnue (nous pensons que c'est parce que la version inférieure n'utilise pas le GPU ou quelque chose comme ça) donc nous avons entraîné le modèle pour 5 époques seulement et il a quand même donné d'excellents résultats comme le montre la figure suivante :

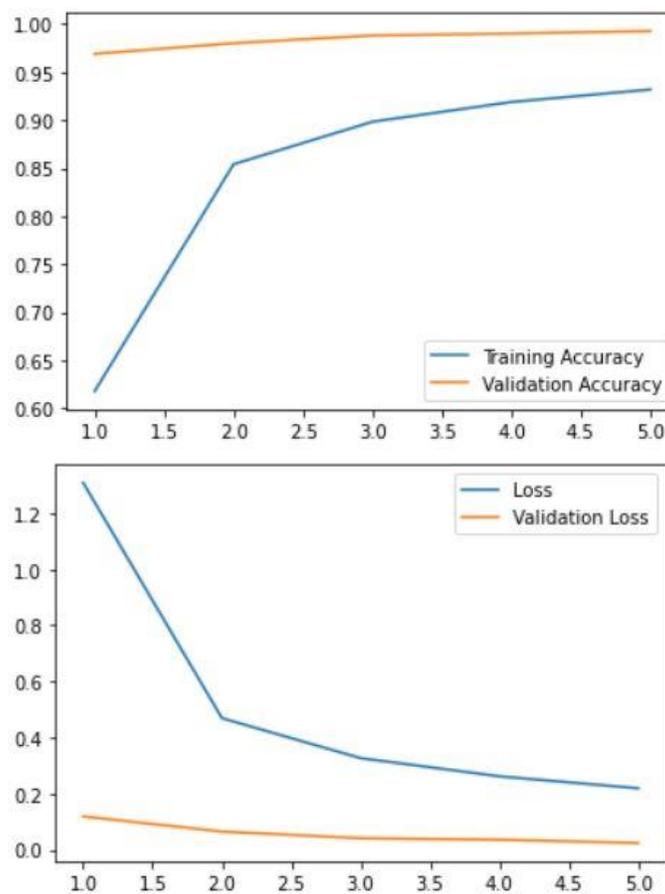


Figure 4-27 : Les courbes de précision et d'erreur du modèle pendant 5 époques d'apprentissage.

Pour les valeurs exactes, le modèle a obtenu loss : 0.2196, acc : 0.9319, val_loss : 0.0243, val_acc : 0.9928

C. Test et implémentation

➤ Test :

Après l'entraînement du modèle, il est maintenant temps de tester ses performances sur la classification des panneaux de signalisation.

L'entrée doit être la même que celle de l'entraînement, elle doit passer par les étapes de prétraitement avant d'être envoyée au modèle pour la classer, donc elle doit être de 32×32 , mise à l'échelle de gris, égalisée et normalisée.

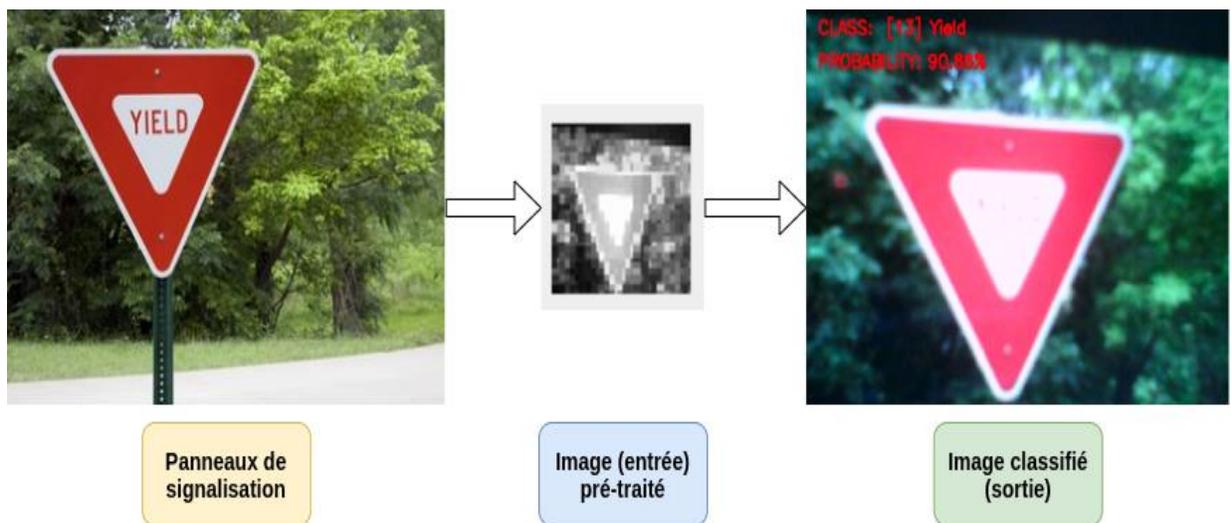


Figure 4-28 : Le processus d'alimentation du modèle en images à classer.



Figure 4-29 : Exemples de différents panneaux de signalisation classés.

Le modèle a bien prédit les bonnes classes et nous pouvons maintenant l'utiliser dans le RPI.

➤ **Implémentation :**

Nous avons utilisé le même code de test avec quelques améliorations pour le système embarqué afin d'en exploiter le meilleur.

Dans la voiture, nous ne disposons pas d'un PC pour visualiser la sortie du modèle, nous avons donc utilisé l'écran LCD pour afficher le nom de la classe prédite, voir **figure 4.30**.

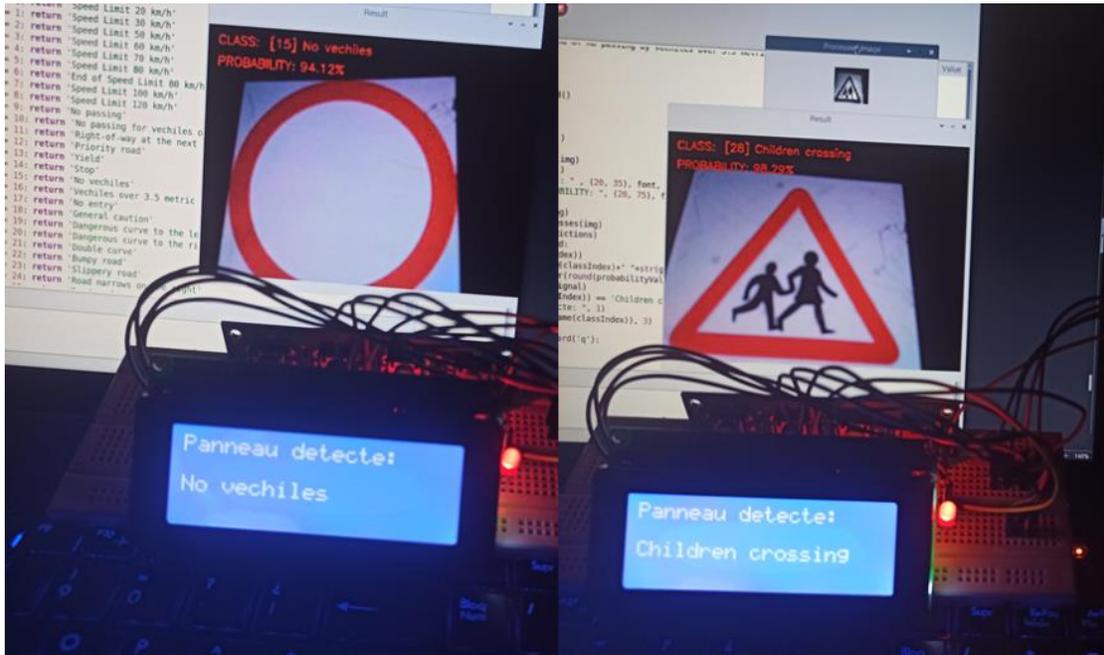


Figure 4-30 : L'étiquette du panneau de signalisation prédite affichée à l'écran.

Nous savons que sur les routes, certains panneaux sont plus importants que d'autres, ils doivent tous être respectés mais certains sont dangereux que des autres, il faut les compléter avec un guide vocal du haut-parleur attaché si le panneau est détecté.

Pour simuler l'expérience de conduite et tester les guides que nous avons mis sur les panneaux de vitesse, nous avons harnaché des boutons poussoirs à utiliser comme entrée au système, vert 20km/h, bleu 60km/h, rouge 80km/h et jaune 120km/h puis nous présentons un panneau devant la PiCamera afin qu'il soit classifié et que le guide correspondant soit activé, si la vitesse d'entrée est supérieure à ce qui est indiqué dans le panneau, le buzzer se déclenchera et le haut-parleur vous dira de ralentir à la vitesse limitée, si le contraire le haut-parleur vous alertera que vous êtes autorisé à accélérer un peu.

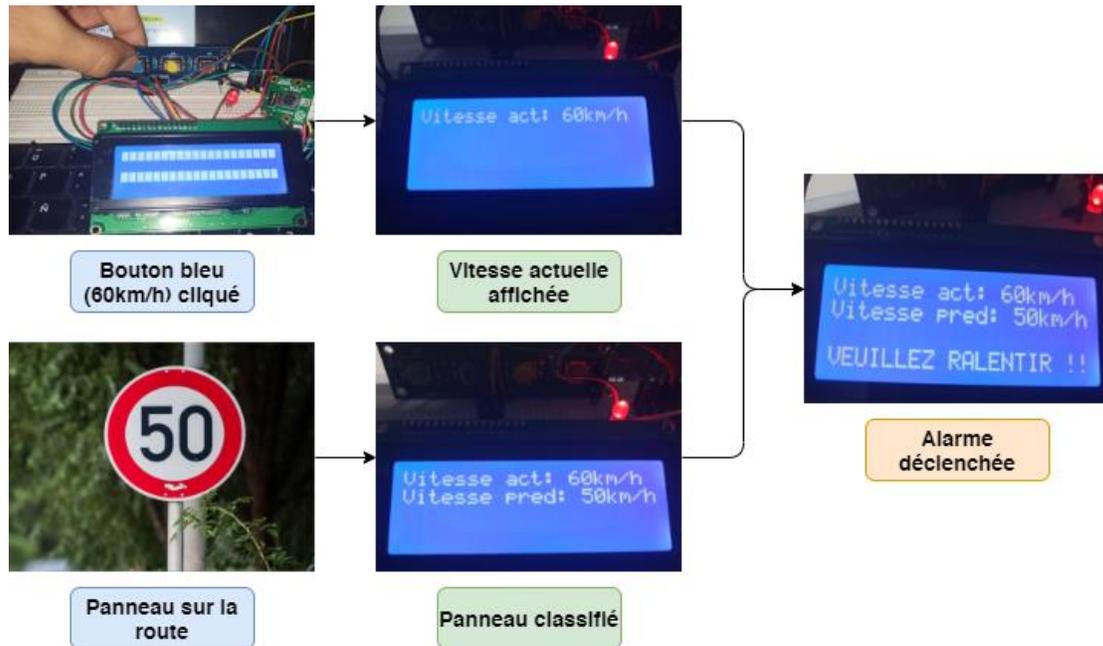


Figure 4-31 : Un exemple de la simulation faite.

4.4 Conclusion

L'utilisation de l'apprentissage profond pour construire un tel système est très difficile, en particulier avec les systèmes embarqués, car les CNN nécessitent une grande puissance de traitement, ce qui limite le travail.

Dans ce chapitre, nous avons utilisé les réseaux neuronaux convolutifs pour faire reconnaissance faciale, la détection de la fatigue du conducteur et la classification des panneaux de signalisation en expliquant le processus, étape par étape, du tout début de la mise en place de l'environnement de programmation à la construction d'une base de données appropriée pour chaque application, puis à l'entraînement du modèle de CNNs et jusqu'à l'explication des composants électroniques et l'intégration dans les systèmes embarqués avec les défis qui l'accompagnent.

Et si nous avons appris une chose de cette expérience en apprentissage profond, c'est que les données sont plus importantes que le modèle lui-même.

Conclusion Générale

Les applications d'intelligence artificielle se répandent largement dans tous les domaines aujourd'hui. Parmi ces domaines, on trouve l'industrie automobile, où il est possible de construire des voitures intelligentes avec toutes sortes d'applications qui aident le conducteur à obtenir la meilleure expérience de conduite.

Notre objectif pour ce projet était de développer un tel système et de le réaliser en utilisant un Raspberry Pi comme un système embarqué qui peut être utilisé dans des voitures réelles.

L'utilisation de l'IA pour améliorer l'expérience de conduite a beaucoup de potentiel, en particulier son sous-ensemble en constante augmentation, l'apprentissage profond. C'est pourquoi nous l'avons choisi parmi d'autres méthodes pour développer notre système capable de démarrer une voiture par reconnaissance faciale, de surveiller l'état de fatigue du conducteur, de l'alerter en cas de risque et de l'empêcher d'être distrait de la route, ainsi que de lire les panneaux de signalisation.

Pour ce faire, nous avons utilisé les réseaux neuronaux convolutifs qui non seulement nous ont permis de réduire la quantité d'outils nécessaires pour construire le système (ce qui est crucial pour ne pas perturber le conducteur), mais qui ont également été un excellent choix pour obtenir les meilleurs résultats. Sans parler de l'énorme avantage et du défi que cela a représenté pour nous, des ingénieurs en Automatique, d'apprendre une technique informatique aussi brute.

La méthode que nous avons suivie pour construire chaque application était

- Acquérir, préparer et prétraiter une base de données appropriée pour chaque problème.

- Entraîner le modèle d'apprentissage profond (CNN) pour obtenir les meilleurs résultats.
- Tester le modèle sur des exemples réels sur PC.
- Optimiser le modèle pour qu'il fonctionne correctement sur les systèmes embarqués (RaspberryPi).

Les résultats obtenus dans chaque application ont été très motivants car nous avons réussi à obtenir des modèles de haute précision qui sont fiables pour la tâche requise avec efficacité. Cependant, chaque travail a ses limites et peut être amélioré encore plus, sans oublier que nous avons utilisé des unités de faible puissance de traitement pour effectuer des applications aussi exigeantes du Deep Learning.

Notre perspective pour les projets futurs peut être citée en différents points :

- Acquérir des ensembles de données plus larges, plus riches et plus diversifiés, car c'est la chose la plus importante pour les réseaux neuronaux.
- Entraîner les modèles pour plus d'époques et utiliser des techniques de réglage fin pour atteindre des scores et des précisions plus élevés.
- Faire évoluer le système embarqué de cette version prototype vers un état esthétique plus complet.
- Utiliser des systèmes embarqués plus puissants, comme le Jetson Nano de Nvidia, qui intègre un GPU suffisamment performant pour exécuter des applications de Deep Learning.

Références

- [1] «Rapport de situation sur la sécurité routière 2015,» organisation mondiale de la santé, 2015.
- [2] Y.Page, T.Hermitte et S.Cuny, «La sécurité routière en france, bilan de l'année 2011,» *In Annals of Advances in Automotive Medicine/Annual Scientific Conference*, p. 101, 2011.
- [3] M.G. MENDOZA, «Surveillance temps-réel des systèmes Homme-Machine,» *Institut National des Sciences Appliquées de Toulouse*, 2004.
- [4] J. Song, a. Kim et K. Boo, «A study on an anti-lock braking system controller and rear-wheel controller to enhance vehicle lateral stability,» *Proceedings of the Institution of Mechanical Engineers, Part D : Journal of Automobile Engineering*, vol. 221, n° 17, p. 777-787, 2007.
- [5] A. T. VanZanten, «Evolution of electronic control systems for improving the vehicle dynamic behavior,» *in Proceedings of the 6th International Symposium on Advanced Vehicle Control*, vol. 2, p. 9, 2002.
- [6] R. Sferco, Y. Page, J.-Y. L. Coz et P. Fay, «Potential effectiveness of electronic stability programs (esp) – what european field studies tell us,» *in Proceeding of 17 th ESV Conference. Amsterdam, The Netherlands*, 2001.
- [7] M. Aga et A. Okada, «Analysis of vehicle stability control (vsc)'s effectiveness from,» *in Proceedings of the 18 th ESV Conference, paper*, 2003.
- [8] K. Bengler, K. Dietmayer, B. Farber, M. Maurer, C. Stiller et H. Winner, «Three decades of driver assistance systems : Review and future perspectives,» *IEEE Intelligent Transportation Systems Magazine*, vol. 6, n° 14, pp. 6-22, 2014.
- [9] NADINE FILION, «guideautoweb,» 07 01 2018. [En ligne]. Available: <https://www.guideautoweb.com/articles/11403/le-regulateur-de-vitesse-a-utiliser-avec-moderation/>. [Accès le 29 06 2021].
- [10] U. Lundgren, J. Peters, M. Smulders, S. Johnston et M. Lörtzer, «Independent research by dutch research institute tno shows that satellite navigation systems have a positive influence on road safety,» *tech. rep*, 2007.
- [11] H. Tan, F. Zhao, H. Hao, Z. Liu, A. A. Amer et H. Babiker, «Automatic Emergency Braking (AEB) System Impact on Fatality and Injury Reduction in China,» *International Journal of Environmental Research and Public Health*, vol. 17, n° 13, p. 917, 2020.

- [12] «Volvo City Safety loss experience – a long-term update,» *Highway Loss Data Institute*, vol. 32, n° 11, pp. 1-24, 2015.
- [13] M. DOYLE, A. EDWARDS et M. AVERY, «AEB real world validation using UK motor insurance claims data,» *In Proceedings of the 24th ESV Conference*, pp. 13-58, 2015.
- [14] M. Audrey, «Commande assistée au conducteur basée sur la conduite en formation de type "banc de poissons,» *Physique mathématique [math-ph]*. Université de Bordeaux, 2014.
- [15] SAE, «Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems,» *SAE Standard J3016*, p. 01–16, 2014.
- [16] «parlonssciences,» 21 01 2021. [En ligne]. Available: <https://parlonssciences.ca/ressources-pedagogiques/documents-dinformation/intelligence-artificielle-et-automobiles>. [Accès le 29 06 2021].
- [17] H. BOUGHRARA, M. CHTOUROU, C. BEN AMAR et al, «Face recognition based on perceived facial images and multilayer perceptron neural network using constructive training algorithm,» *IET computer vision*, vol. 8, n° 16, pp. 729-739, 2014.
- [18] A. STUHLSTATZ, J. LIPPEL et T. ZIELKE, «Feature extraction with deep neural networks by a generalized discriminant analysis,» *IEEE transactions on neural networks and learning systems*, vol. 23, n° 14, pp. 596-608, 2012.
- [19] W.-B. HORNG, C.-Y. CHEN, Y. CHANG et al, «Driver fatigue detection based on eye tracking and dynamic template matching,» *In : IEEE International Conference on Networking, Sensing and Control, 2004. IEEE*, pp. 7-12, 2004.
- [20] T. D'ORAZIO, M. LEO et A. DISTANTE, «Eye detection in face images for a driver vigilance system,» *In : IEEE Intelligent Vehicles Symposium, 2004. IEEE*, pp. 95-98, 2004.
- [21] W. DONG et X. WU, «Fatigue detection based on the distance of eyelid,» *In : Proceedings of 2005 IEEE International Workshop on VLSI Design and Video Technology, 2005. IEEE,,* pp. 365-368, 2005.
- [22] W. WIERWILLE, «Overview of research on driver drowsiness definition and driver drowsiness detection,» *In Technical International Conference on Enhanced Safety of Vehicles (ESV)*, p. 23–26, 1994.
- [23] D.Dinges et R.Grace, «Perclos : a valid psychophysiological measure of alertness as assessed by psychomotor vigilance,» *In US Department of Transportation Federal Highway Administration*, 1998.
- [24] W. QING, S. BINGXI, X. BIN et al, «A perclos-based driver fatigue recognition application for smart vehicle space,» *In : 2010 Third International Symposium on Information Processing. IEEE,,* pp. 437-441, 2010.

- [25] I. García, S. Bronte, L. M. Bergasa, J. Almazán et J. Yebes, «Vision-based drowsiness detector for real driving conditions,» *IEEE Intelligent Vehicles Symposium*, pp. 618-623, 2012.
- [26] R. Grace, Richard et S. STEWARD, «Drowsy driver monitor and warning system,» *In International Driving Symposium on Human Factors in Driver Assessment, Training and Vehicle Design.*, 2001.
- [27] M. MOHANTY, A. MISHRA et A. ROUTRAY, «A non-rigid motion estimation algorithm for yawn detection in human drivers,» *International Journal of Computational Vision and Robotics*, vol. 1, n° %11, pp. 89-109, 2009.
- [28] M. Ochocki et D. Sawicki, «Yawning Recognition based on Dynamic Analysis and Simple Measure,» *in Proceedings of the International Conference on Computer-Human Interaction Research and Applications*, pp. 111-117, 2017.
- [29] G. PICCIOLI, E. DE MICHELI, P. PARODI et al, «Robust method for road sign detection and recognition,» *Image and Vision Computing*, vol. 14, n° %13, pp. 209-223, 1996.
- [30] P. PACLIK, J. NOVOVIČOVÁ, P. PUDIL et al, «Road sign classification using Laplace kernel classifier,» *Pattern Recognition Letters*, vol. 21, n° %113-14, pp. 1165-1173, 2000.
- [31] U. ZAKIR, I. ZAFAR et E. A. EDIRISINGHE, «Road sign detection and recognition by using local energy based shape histogram (LESH),» *International Journal of Image Processing*, vol. 4, n° %16, pp. 566-582, 2011.
- [32] BENALLAL, Mohamed et MEUNIER, «Real-time color segmentation of road signs,» *In IEEE CCECE 2003-Canadian Conference on Electrical and Computer Engineering*, pp. 1823-1826, 2003.
- [33] R. BELAROUSSI et J. P. TAREL, «Détection des panneaux de signalisation routière par accumulation bivariée,» *Traitement du signal*, vol. 27, n° %13, pp. 265-296, 2010.
- [34] A. RUTA, Y. LI et X. LIU, «Real-time traffic sign recognition from video by class-specific discriminative features,» *Pattern Recognition*, vol. 43, n° %11, pp. 416-430, 2010.
- [35] GARCIA-GARRIDO, M. Angel, SOTELO, M. Angel, MARTIN-GOROSTIZA et Ernesto, «Fast traffic sign detection and recognition under changing lighting conditions,» *In : 2006 IEEE Intelligent Transportation Systems Conference. IEEE*, pp. 811-816, 2006.
- [36] J. MIURA, T. KANDA, S. NAKATANI et al, «An active vision system for on-line traffic sign recognition,» *IEICE TRANSACTIONS on Information and Systems*, vol. 85, n° %111, pp. 1784-1792, 2002.
- [37] X. BARÓ, S. ESCALERA, J. VITRIA et al, «Traffic sign recognition using evolutionary adaboost detection and forest-ECOC classification,» *IEEE Transactions on Intelligent Transportation Systems.*, vol. 10, n° %11, pp. 113-126, 2009.

- [38] N. DALAL et B. TRIGGS, «Histograms of oriented gradients for human detection,» *In : 2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05). IEEE*, pp. 886-893, 2005.
- [39] F. ZAKLOUTA et B. STANCIULESCU, «Warning traffic sign recognition using a HOG-based Kd tree,» *In : 2011 IEEE intelligent vehicles symposium (IV). IEEE*, pp. 1019-1024, 2011.
- [40] Y. WU, Y. LIU, J. LI et al, «Traffic sign detection based on convolutional neural networks,» *In : The 2013 international joint conference on neural networks (IJCNN). IEEE*, pp. 1-7, 2013.
- [41] J. STALLKAMP, M. SCHLIPSING, J. SALMEN et al, «Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition,» *Neural networks*, vol. 32, pp. 323-332, 2012.
- [42] Y. QU, S. YANG, W. WU et al, «Hierarchical traffic sign recognition,» *In : Pacific Rim Conference on Multimedia. Springer, Cham*, pp. 200-209, 2016.
- [43] P. SERMANET et Y. LECUN, «Traffic sign recognition with multi-scale convolutional networks,» *In : The 2011 International Joint Conference on Neural Networks. IEEE*, pp. 2809-2813, 2011.
- [44] R. QIAN, Y. YUE, F. COENEN et al, «Traffic sign recognition with convolutional neural network based on max pooling positions,» *In : 2016 12th International conference on natural computation, fuzzy systems and knowledge discovery (ICNC-FSKD).IEEE*, pp. 578-582, 2016.
- [45] ZENG, Yujun, XU, Xin, FANG, Yuqiang et al, «Traffic sign recognition using extreme learning classifier with deep convolutional features.,» *In : The 2015 international conference on intelligence science and big data engineering (IScIDE 2015), Suzhou, China., pp. 272-280, 2015.*
- [46] D. CIREAAN, U. MEIER, J. MASCI et al, «Multi-column deep neural network for traffic sign classification,» *Neural networks*, vol. 32, pp. 333-338, 2012.
- [47] H. H. AGHDAM, E. J. HERAVI et D. PUIG, «A practical approach for detection and classification of traffic signs using convolutional neural networks,» *Robotics and autonomous systems*, vol. 84, pp. 97-112, 2016.
- [48] S. EICKELER, M. VALDENEGRO, T. WERNER et al, «Future computer vision algorithms for traffic sign recognition systems,» *In : Advanced Microsystems for Automotive Applications 2015. Springer, Cham*, pp. 69-77, 2016.
- [49] M. HARGRAVE, «Deep Learning Definition,» 06 April 2021. [En ligne]. Available: <https://www.investopedia.com/terms/d/deep-learning.asp>. [Accès le 23 April 2021].
- [50] W. S. McCulloch et W. Pitts, «A logical calculus of the ideas immanent in nervous activity,» *The Bulletin of Mathematical Biophysics*, vol. 5, p. 115–133, 12 1943.

- [51] F. ROSENBLATT, «The perceptron, a perceiving and recognizing automaton Project Para,» *Cornell Aeronautical Laboratory*, 1997.
- [52] K.FUKUSHIMA, «Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition,» *Biological Cybernetics*, vol. 36, n° 14, p. pp. 193–202., 1980.
- [53] COVER, Thomas, Peter et HART., «Nearest neighbor pattern classification,» *IEEE transactions on information theory*, vol. 13, n° 11, pp. 21-27, 1967.
- [54] DENG, Jia, DONG, Wei, SOCHER, Richard et al., «Imagenet: A large-scale hierarchical image database,» *In : 2009 IEEE conference on computer vision and pattern recognition. IEEE*, pp. 248-255, 2009.
- [55] L. Tucci, «What is artificial neuron? - Definition from WhatIs.com,» May 2018. [En ligne]. Available: <https://searchcio.techtarget.com/definition/artificial-neuron>. [Accès le 21 April 2021].
- [56] M. Robins, «The Difference Between Artificial Intelligence, Machine Learning and Deep Learning,» Intel, 27 May 2020. [En ligne]. Available: <https://www.intel.com/content/www/us/en/artificial-intelligence/posts/difference-between-ai-machine-learning-deep-learning.html>. [Accès le 22 April 2021].
- [57] I. Nedjar, «Medical images indexation and annotation,» Thèse de Doctorat, Université Tlemcen, Tlemcen, Algeria, 2015.
- [58] T. Dettmers, «Deep Learning in a Nutshell: Core Concepts | NVIDIA Developer Blog,» NVIDIA, 03 November 2015. [En ligne]. Available: <https://developer.nvidia.com/blog/deep-learning-nutshell-core-concepts/>. [Accès le 22 April 2021].
- [59] H. Wiesel, «Receptive fields of single neurones in the cat's striate cortex,» *The Journal of Physiology*, vol. 148, n° 13, p. 574–591, 1959.
- [60] A. K. Simonyan, «Very deep convolutional networks for large-scale image recognition,» *arXiv preprint arXiv :1409.1556*, 2014.
- [61] K. He et e. al, «Deep residual learning for image recognition,» *in Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770-778, 2016.
- [62] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke et A. Rabinovich, «Going deeper with convolutions,» *in Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1-9, 2015.
- [63] F.Chollet, «Xception: Deep learning with depthwise separable convolutions,» *in Proceedings of the IEEE conference on computer vision and pattern recognition*, p. 1251–1258, 2017.

- [64] L. Sifre et S. Mallat, «Rigid-motion scattering for image classification,» *Ph. D. thesis*, 2014.
- [65] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto et H. Adam, «Mobilenets: Efficient convolutional neural networks for mobile vision applications,» *arXiv preprint arXiv:1704.04861*, 2017.
- [66] F. KRIEF, «Les systèmes embarqués communicants : mobilité, sécurité , autonomie».
- [67] S. KV, *Introduction to Embedded System*, 1er Edition, McGraw-Hill, 2009.
- [68] S.RAMLA, «Diagnosticabilité modulaire appliquées au Diagnostic en ligne Des Systèmes Embarqués,» Thèse de Doctorat, L'ENSI- Ecole Centrale de Lille, Lille, France, 2013.
- [69] W.Theurer, «Une méthodologie de modélisation multi-modèles distribuée par Métier pour les systèmes embarqués,» Thèse de Doctorat, ENSAE de Toulouse, Toulouse, France, 2006.
- [70] I.STURDNIA, «Détection d'intrusion pour des réseaux embarqués automobiles : Une approche orientée langage,» thèse de doctorat, l'Institut National des Sciences Appliquées de Toulouse, 2015.
- [71] ABABSA, Tarek, DJEDL, Nouredine, DUTHEN et Yves, «Genetic programming-based self-reconfiguration planning for metamorphic robot,» *International Journal of Automation and Computing*, vol. 15, n° 14, pp. 431-442, 2018.
- [72] G. ABDELLAOUI, H. MEGNAFI et F. T. BENDIMERAD, «A novel model using Reo for IoT self-configuration systems,» *In : 2020 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP). IEEE*, pp. 1-5, 2020.
- [73] ATZORI, Luigi, IERA, Antonio et Giacomo.MORABITO, «The internet of things: A survey. Computer networks,» vol. 54, n° 112, pp. 2787-2805, 2010.
- [74] MIORANDI, Daniele, SICARI, Sabrina, D. PELLEGRINI, Francesco et al, «Internet of things: Vision, applications and research challenges,» *Ad hoc networks*, vol. 10, n° 17, pp. 1497-1516, 2012.
- [75] WEINSTEIN et Ron, «RFID: a technical overview and its application to the enterprise,» *IT professional*, vol. 7, n° 13, pp. 27-33, 2005.
- [76] JABEEN, Qamar, KHAN, Fazlullah, HAYAT, M. Nouman et al, «A survey: Embedded systems supporting by different operating systems,» *arXiv preprint arXiv:1610.07899*, 2016.
- [77] H. MEGNAFI, «Frequency plan optimization based on genetic algorithms for cellular networks,» *Journal of Communications Software and Systems*, vol. 16, n° 13, pp. 217-223, 2020.

- [78] D.Uckelmann, M.Harrison et F.Michahelles., «Architecting the Internet of Things,» *Springer Science & Business Media*, 2011.
- [79] A. MELLIT, H. MEKKI, A. MESSAI et al, «FPGA-based implementation of intelligent predictor for global solar irradiation, Part I: Theory and simulation.,» *Expert Systems with Applications*, vol. 38, n° 13, pp. 2668-2685, 2011.
- [80] N. SHAH, P. CHAUDHARI et K. VARGHESE, «Runtime programmable and memory bandwidth optimized FPGA-based coprocessor for deep convolutional neural network,» *IEEE Transactions on Neural Networks and Learning Systems*, vol. 29, n° 112, 2018.
- [81] Y. SUN et A. CHENG, «Machine learning on-a-chip: A high-performance low-power reusable neuron architecture for artificial neural networks in ECG classifications,» *Computers in biology and medicine*, vol. 42, n° 17, pp. 751-757, 2012.
- [82] CHOWDHURY, S. Roy et H. SAHA, «Development of a FPGA based fuzzy neural network system for early diagnosis of critical health condition of a patient.,» *Computers in biology and medicine*, vol. 40, n° 12, pp. 190-200, 2010.
- [83] A. MOHAMED, A. ISSAM, B. MOHAMED et al, «Real-time detection of vehicles using the haar-like features and artificial neuron networks,» *Procedia Computer Science*, vol. 73, pp. 24-31, 2015.
- [84] B. MOHAMED, A. ISSAM, A. MOHAMED et al, «ECG image classification in real time based on the haar-like features and artificial neural networks,» *Procedia Computer Science*, vol. 73, pp. 32-39, 2015.
- [85] O. YADAN, K. ADAMS, Y. TAIGMAN et al., «Multi-gpu training of convnets,» *arXiv preprint arXiv:1312.5853*, 2013.
- [86] Y.-H. CHEN, T. KRISHNA, J. S. EMER et al, «Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks,» *IEEE journal of solid-state circuits*, vol. 52, n° 11, pp. 127-138, 2016.

Résumé

L'objectif de cette étude est d'utiliser les techniques d'intelligence artificielle afin de développer des options automatiques pour les voitures intelligentes. Nous avons développé trois applications différentes telles que le système de détection de la fatigue et de la distraction, le démarreur de voiture à reconnaissance faciale et un système de reconnaissance des panneaux de signalisation. Nous avons utilisé la vision par ordinateur dans le processus de développement pour sa facilité et sa technique non exigeante en termes d'instruments, comme l'apprentissage profond (Deep Learning) qui est la technique la plus précise et en constante évolution pour la vision par ordinateur. Nous avons ensuite implémenté les modèles CNN formés dans un Raspberry Pi pour que le système embarqué puisse être placé dans de vraies voitures et tester les applications en temps réel.

Nous avons expliqué le processus de création de chaque application étape par étape, depuis les informations théoriques et les réalisations de l'état de l'art jusqu'à nos applications, la formation, les tests et la mise en œuvre.

Mots clés : Intelligence Artificielle, Vision par Ordinateur, Apprentissage Profond, CNN, Détection de la Fatigue, Reconnaissance Faciale.

Abstract:

The use of AI in embedded systems for the development of automatic options for smart cars

The objective of this study is to use Artificial Intelligence technics in order to develop smart options for intelligent cars. We developed three different applications such as driver's fatigue and distraction detection system, face recognition car starter and a traffic signs recognition system. We use computer vision in the development process for its easiness and non-instrument demanding technique so as Deep Learning being the ever growing and most accurate technique for computer vision. We then implemented the trained CNN models into a Raspberry Pi for the embedded system to be placed in real cars and test the applications in real-time.

We explain the process for creating each applications step by step from theoretic information and state of the art realizations to our applications, training testing and implementation.

Keywords: Artificial Intelligence, Computer Vision, Deep Learning, CNNs, Fatigue detection, Face Recognition.

ملخص:

هذه الدراسة تهدف إلى استخدام تكنولوجيا الذكاء الاصطناعي لتطوير تطبيقات ذكية لسيارات ذكية. طورنا ثلاث تطبيقات مختلفة مثل نظام رصد التعب وتشتت انتباه السائق، نظام تشغيل السيارة بالتعرف على الوجوه ونظام التعرف على إشارات المرور. استخدمنا الرؤية الحاسوبية في عملية التطوير لسهولة استخدامها التي لا تتطلب استخدام أجهزة مكلفة ومشتتة. كما أن تقنية التعلم العميق هي تقنية دائمة التطور والأكثر دقة لرؤية الحاسوب. بعد ذلك قمنا بتطبيق نماذج سي إن إن المدربة في راسبري باي للنظام المضمن ليتم وضعه في السيارات الحقيقية واختبار التطبيقات في الوقت الحقيقي.

شرحنا عملية إنشاء كل تطبيق خطوة بخطوة بدءاً من المعلومات النظرية ومروراً بالجوانب التقنية، وصولاً إلى تطبيقاتنا واختبارات التدريب وتطبيقاتها.

الكلمات المفتاحية: الذكاء الاصطناعي، رؤية الحاسوب، التعلم العميق، شبكة الخلايا العصبية التلافيفية، رصد التعب، التعرف على الوجه.