الجمهورية الجزَائِرية الديمقرَاطية الشعبية
PEOPLE'S DEMOCRATIC REPUBLIC OF ALGERIA
وزَارة التعليم العَالي و البحث العلمي
MINISTRY OF HIGHER EDUCATION AND SCIENTIFIC RESEARCH
جَامعة عمَار ثليجي بِالأَغوَاط
UNIVERSITY OF LAGHOUAT

FACULTY OF SCIENCES
DEPARTMENT OF COMPUTER SCIENCE

Field : Mathematics and Computer Science
Option : Computer Science
Specialization : Distributed Networks, Systems, and Applications.

Memoire Submitted in Partial fulfillment of the requirements for the Master degree in Computer Science

Submitted by: Khadidja Iznasni

THEME

Driver Drowsiness Detection Using Deep Learning Approaches

Jury members:

Mr Mohamed Lahcen Bensaad	(University of Laghouat)	President
Mr Taher Bendouma	(University of Laghouat)	Examiner
Mr Younes Guellouma	(University of Laghouat)	Advisor
Ms Saida Sarra Boudouh	(University of Laghouat)	Co-Advisor

2022

الجمهورية الجزَائِرية الديمقرَاطية الشعبية
République Algérienne Démocratique et Populaire
وزَارة التعليم العَالي و البحث العلمي
Ministère de l 'Enseignement Supérieur et de la Recherche Scientifique
جَامعة عمَار ثليجي بِالأَغوَاط
UNIVERSITÉ AMAR TELIDJI LAGHOUAT

FACULTE DES SCIENCES
DEPARTEMENT D'INFORMATIQUE

Domaine : Mathématiques et Informatique
Filière : Informatique
Option : Réseaux, systèmes, et application distribuées

Mémoire de Master

Présenté par : Khadidja Iznasni

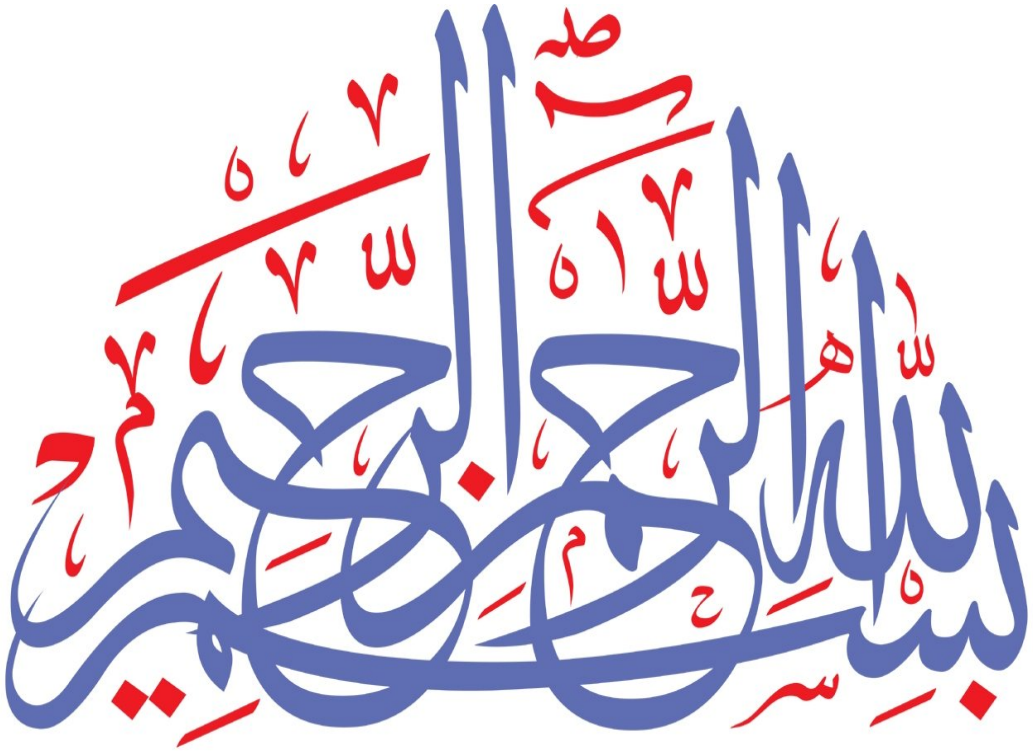
THÈME

Détection de la somnolence du conducteur à l'aide de l'apprentissage en profondeur

Soutenu publiquement devant le jury composé de :

Mr Bensaad Mohamed Lahcen	(University of Laghouat)	Président
Mr Bendouma Taher	(University of Laghouat)	Examinateur
Mr Guellouma Younes	(Université de Laghouat)	Encadreur
Ms Boudouh Saida Sarra	(Université de Laghouat)	Co-Encadreur

2022


بسم الله الرحمن الرحيم

Acknowledgments

All my gratitude and thanks to Allah Almighty, who gave me strength, courage and will to develop this work.

We would like to extend my sincere thanks and gratitude to my professor ***Prof. Nasreddine Lagraa*** *and my co-supervisor,* ***Saida Sarra Boudouh****, my husband, and my best friends* ***Sara Yassmine Brahimi*** *for their management of this work, and for providing all valuable advice to ensure its success.*

We also thank all members of the jury for giving us the honor of evaluating this modest work.

We do not forget our gratitude to all the professors of the Computer Science Department and every professor pass in our academic career, and thanks to everyone who contributed to the development of this work from near or far.

Khadidja, September 2022

Dedications

These are the feelings I have for those who have supported me And who have always been by my side, in the most difficult and hard times, my very dear husband ***Rachid*** *and my great lovely kids* ***Alaa, Abdelaali and Israa*** *may Allah bless them. I dedicate this modest work to them.*
Also I dedicate this work to my dear parents, my sisters ***Hanane, Meriem*** *and my brother* ***Mohammed Amine*** *for their patience, love, support and encouragement, for me to do better throughout my academic career, to my whole family.*

I also wish for continued health and wellness, and a life full of joy, happiness and success, for each of: My family, my teachers and with a lot of love to my dear friends specially ***Sarra Boudouh, Brahimi Sara,Fatna Mouffok and Hadjer Koribaa*** *to all those who love me, to all those I love, to all those who helped me from near or far. I present to you this humble work as a gift.*

Thanks everyone.

Khadidja Iznasni

ملخص

إن النعاس أكثر خطورة على الطريق من السائق الذي يقود بسرعة كبيرة لأنه قد يكون لديه نوبات نوم صغيرة.يسعى مهندسو السيارات إلى البحث عن حل لهذه المشكلة من خلال الاعتماد على الحلول الميكانيكية التي من شأنها منع حدوث مثل هذه الطوارئ. في هذا البحث ، اقترحنا حلاً يعتمد على نهج التعلم العميق وشبكة اتصالات السيارات. تنقسم استراتيجيتنا بشكل أساسي إلى جزأين: الأول يستخدم نهج التعلم العميق بما في ذلك نقل التعلم وتقنيات زيادة البيانات ، من أجل إنشاء نموذج دقيق يصنف صور السائق إلى النعاس أو اليقظة.في هذه الحالة ، يتم اكتشاف النعاس باستخدام كاميرا آلية مستخدمة في المركبات الذكية. بناءً على الصور التي تم الحصول عليها ، يتم توقع حالة السائق على أنها مستيقظ أو نعسان بواسطة شبكة عصبية الالتفافية مقترحة. لتحقيق ذلك ، تم تعديل أربعة نماذج من الشبكة الالتفافية مدربة مسبقًا وتكييفها مع مشكلتنا. تم استخدام كل من ResNet 50V2 و VGG16 و VGG19 و InceptionV3في العديد من التجارب ومع العديد من معلمات الضبط الدقيق لإنشاء الأنسب لحالتنا. لقد جمعنا أكثر من 6000 صورة من مجموعتي بيانات مختلفتين للنعاس والتثاؤب لتدريب نماذجنا وتقييمها. كانت النتائج التي تم الحصول عليها مرضية بشكل عام وفقًا للدراسات السابقة ، حيث وضعنا ResNet 50V2 و InceptionV3 VGG19 و VGG16 على التوالي ، متبوعة بـ VGG19 معVGG16 . وفي الوقت نفسه ، وصلVGG 16 إلى أدنى دقة لـ 50.48 ٪.
الجزء الثاني يحتوي على فرعين: بالنسبة للسيناريو الاول،نقوم ببث رسائل لتنبيه المركبات التقريبية للسيارة التي قد يشعر سائقها بالنعاس. أما بالنسبة للسيناريو الثاني ، فقد تم استخدام منبه لإيقاظ السائق باستخدام اهتزاز الموقع ، أو الأجهزة الذكية (الهاتف أو الساعة).

الكلمات المفتاحية:
شبكة السيارات ,التعلم العميق ,شبكة عصبية الالتفافية ,نعاس السائق ,VGG19 , VGG 16 ,InceptionV3 ,ResNet 50V2

Abstract

A drowsy driver appears to be far more dangerous on the road than one who is driving too fast as he might have microsleeps. Vehicle engineers are seeking to investigate this problem with a few mechanical solutions that would prevent such an emergency. In this work, we proposed a solution based on Deep Learning (Deep Learning (DL)) and Vehicular Adhoc Network (VANET) approaches. Our strategy is mainly divided into two parts: The first one employed DL approaches including Transfer Learning (TL) and data augmentation techniques, in order to create an accurate model that classify driver images into drowsy or awake. In this case, drowsiness is detected using an auto-camera used in smart vehicles. Based on the obtained images, the driver state is predicted as awake or drowsy by a proposed convolutional neural network Convolutional Neural Network (CNN). To achieve that, four pre-trained CNN models were modified and adapted to our problem. Each of Residual Network (ResNet)50V2, Visual Geometric Group (VGG)16, VGG19, and Inception V3 were employed in several trials and with various fine-tuning parameters to establish the most appropriate one for our case. We gathered more than 6000 images from two different datasets of drowsiness and yawn to train and evaluate our models. The obtained results were exceptionally satisfying according to the previous studies, putting InceptionV3 and ResNet50V2 ahead of the other models with 96.85% and 96.70% accuracy respectively, followed by VGG 19 with 92.23%. Meanwhile, the VGG 16 reached the lowest accuracy of 50.48%. The second part contains two scenario's : First, broadcasting messages to alert the approximate vehicles of the car whose driver may be drowsy. As for the second scenario, an alarm was employed to awake the driver using site vibration, or smart devices (phone or watch).

Keywords : **Drowsiness Driver ,Deep Learning ,CNN, VGG 16, VGG 19 ,ResNet50V2, InceptionV3 ,VANET.**

Résumé

Un conducteur somnolent semble être beaucoup plus dangereux sur la route qu'un conducteur trop rapide car il pourrait avoir des microsommeils. Les ingénieurs automobiles cherchent à étudier ce problème avec quelques solutions mécaniques qui empêcheraient une telle urgence. Dans ce travail, nous avons proposé une solution basée sur les approches Deep Learning (DL) et VANET. Notre stratégie est principalement divisée en deux parties : la première a utilisé des approches DL comprenant l'apprentissage par transfert TL et des techniques d'augmentation de données, afin de créer un modèle précis qui classe les images du conducteur en somnolent ou éveillé. Dans ce cas, la somnolence est détectée à l'aide d'une caméra automatique utilisée dans les véhicules intelligents. Sur la base des images obtenues, l'état du conducteur est prédit comme étant éveillé ou somnolent par un réseau neuronal convolutif proposé CNN. Pour y parvenir, quatre modèles CNN pré-entraînés ont été modifiés et adaptés à notre problème. Chacun des ResNet50V2, VGG16, VGG19 et Inception V3 a été utilisé dans plusieurs essais et avec divers paramètres de réglage fin pour établir le plus approprié pour notre cas. Nous avons rassemblé plus de 6000 images de deux ensembles de données différents de somnolence et de bâillement pour former et évaluer nos modèles. Les résultats obtenus étaient exceptionnellement satisfaisants selon les études précédentes, plaçant InceptionV3 et ResNet50V2 devant les autres modèles avec respectivement 96,85% et 96,70% de précision, suivis de VGG 19 avec 92,23%. Pendant ce temps, le VGG 16 a atteint la précision la plus faible de 50,48%. La deuxième partie contient deux scénarios : Premièrement, diffuser des messages pour alerter les véhicules approximatifs de la voiture dont le conducteur est peut-être somnolent. Quant au deuxième scénario, une alarme a été utilisée pour réveiller le conducteur à l'aide des vibrations du site ou d'appareils intelligents (téléphone ou montre).

1ex0,5 cm **Mots clés** : **Pilote de somnolence, Apprentissage en profondeur, CNN, VGG 16, VGG 19, ResNet50V2, InceptionV3, VANET.**

Contents

List of Figures

List of Tables

List of Acronyms

AI Artificial Intelligence. 2, 3, 4, 6, 11, 28

AlexNet Alex Connected Convolutional Networks. 13, 15, 16, 17, 24

ANN Artificial Neural Network. 4, 5, 13, 24, 28

AUC Area Under Curve. 10, 19, 20, 39, 40, 42

BPNN Back Propagation algorithm Neural network. 5

BSN Body Sensor Network. 25

CMPNN Complementary Neural Network. 5

CNN Convolutional Neural Network. 5, 6, 7, 9, 10, 6, 8, 12, 13, 15, 16, 18, 24, 25, 27, 29, 30, 31, 36, 41

CNNs Convolutional Neural Networks. 7, 5, 7, 8, 12, 14, 19, 21, 22, 26

CPU Central Processing Unit. 28

CUDA Compute Unified Device Architecture. 28

CUDnn Compute Unified Device Architecture Deep Neural Network. 28

DDD Driver Drowsiness Detection. 7, 9, 12, 13, 15, 18, 19, 24

DenseNet Densely Connected Convolutional Networks. 15

DL Deep Learning. 5, 6, 7, 8, 9, 10, 1, 2, 3, 4, 5, 6, 7, 8, 9, 11, 12, 13, 14, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48

DNN Deep Neural Network. 6, 7

DNNs Deep Neural Networks. 7, 12, 20, 21, 22, 30

EEG electroencephalogram. 13

FC Fully Connected Layer. 7

FD-NN Fully Designed Neural Network. 18, 19, 20

FlowImageNet Flow Image Network. 15, 16, 17

FPR False Positive Rate. 39

GANs Generative Adversarial Networks. 5

GAP Global Average Pooling. 36, 41

GMP Global Max Pooling. 36, 41

GPU Graphics Processing Unit. 1, 28, 49

HDDD Hierarchical Deep Drowsiness Detection. 20, 23

IoV Internet of Vehicles. 10

ITS Intelligent Transportation System. 9, 10, 25

KNN k-Nearest Neighbour. 4

LSTM Long Short Term Memory. 8, 20, 25

MANETs Mobile Adhoc Networks. 9

ML Machine Learning. 7, 9, 2, 3, 4, 5, 6, 8, 9, 11, 24, 29, 30

MLP Multi-Layer Perceptron. 5

NHTSA National Highway Traffic Safety Administration. 12

NLP Natural Language Processing. 7

NN Neural Network. 5, 14

NNs Neural Networks. 5, 6, 7, 11, 14, 29

OBU On-Board Unit. 10

RAM Random Access Memory. 14, 18

RBF Radial Basis Function. 5

ReLu Rectified Linear Unit. 19

ResNet Residual Network. 5, 6, 9, 1, 13, 15, 16, 17, 20, 21, 22, 24, 31, 34, 37, 38, 39, 40, 42, 47, 48

RNN Recurrent Neural Network. 6, 8, 24

RNNs Recurrent Neural Networks. 5, 7, 8

ROC Receiver Operating Characteristic. 38, 39

ROI Region Of Interest. 1, 19, 33, 37, 40, 49

RPM Revolutions Per Minute. 25

RSU Roadside Units. 10

SGD Stochastic Gradient Descent. 7

SSD Solid-State Drive. 28

SVM Support Vector Machines. 4

TL Transfer Learning. 5, 6, 18, 19, 20, 26, 30, 34

TPR True Positive Rate. 39

V2R Vehicle-to-Roadside. 10

V2V Vehicle-to-Vehicle. 10

V2X Vehicle-to-Every things. 10, 25

VANET Vehicular Adhoc Network. 5, 6, 7, 8, 9, 1, 2, 3, 9, 11, 12, 13, 24, 25, 26, 27, 43, 44, 46, 47, 48, 49

VGG Visual Geometric Group. 5, 6, 9, 1, 13, 17, 18, 19, 20, 21, 23, 24, 31, 34, 37, 39, 40, 42, 47, 48

VGG Face Visual Geometric Group Face. 15, 16

1 Introduction

Contents

1.1 Context

According to the World Health Organization, vehicle accidents kill over 1.3 million people each year [12]. These crashes are caused by a wide range of causes, nearly 90% of which are related to the "human factor", which is defined by drivers' health, values, attitudes, and behaviors, while drowsiness and fatigue are the most important factors in driver behavior that can result in catastrophic road accidents. As a result, considerable research, contributions, and realizations has been done in order to prevent or at least reduce the shocking number of these incidents. These studies and achievements have been accomplished in a variety of areas by using important techniques and methods, such as road monitoring systems, intelligent monitoring systems installed on vehicles, and communication networks between vehicles and base stations to interact and prevent in specific situations. Several studies have been published that focus on the study of the driver's state. Our current study focuses on the domain of driver behavior analysis, specifically the detection of driver drowsiness. To carry out our research, we employed two main approaches: DL for training and VANET Network for simulation. So, in order to implement the learning part, we chose to use the transfer learning solution, which is a set of pre-trained models and designed for application in the context of DL. Therefore, we selected four particular models, from Keras platform - ResNet50 V2, VGG 16, VGG 19, and Inception V3 –, that have demonstrated excellent performances in their previous implementations. To avoid going through the pre-processing stage to extract the Region Of Interest (ROI) of the images, which are the eyes and the mouth, we gathered our dataset according to the criterion of Region Of Interest ROI. So we import our dataset, "Drowsiness and Yawn," from the Kaggle platform, and then merge the two datasets to create a single one with more than 6000 images that is ready for training. Then we proceeded to the splitting and labeling phases so that we could define the awake and drowsy classes for training and test parts. Finally, we moved on to the data augmentation phase where we determined the training parameters. The four models were launched for several trials on our computer, which has a powerful Graphics Processing

Unit (GPU) ready for training, to begin the train phase with anaconda environment, Python Language, and Tensorflow environment. The first involves broadcasting an alert message to vehicles connected to the VANET network alerting them of the driver's drowsy state, and the second involves creating an alarm system that is used to awaken the driver.

1.2 Organization of the Memoire

This memoire is organized as follows:

- In **Chapter 2**, we will to provide a basic overview about the domains of Artificial Intelligence (AI), ML, and DL, in addition to explaining the concepts of neural networks, their types, and their applications, we tried to explain the differences between these three areas. Furthermore, we provided an overview of the architectures of the most popular ML and DL models. Following that, we discussed VANET networks, explaining their structures, functioning, communication, applications, and challenges.
- In **Chapter 3**, we tried to focus on recent related work that were created in the same context as our study. First, we mentioned the previous studies that used DL approaches by analyzing their models, methods, and results. After that, we discussed the studies that used VANET approaches and the ones that combined between the two approaches.
- In **Chapter 4**, we explained the tools used, the models chosen, the techniques, and the obtained results after applying DL approaches. Lastly, using VANET network, we proposed our simulation scenarios.
- Finally, we finish by giving a general conclusion summarizing this work, its limitations, and the possible future perspectives.

2

Deep Learning and VANET Concept: Background

Contents

2.1 Introduction

Recently, AI has grown significantly and has been widely applied in various applications, including machine learning and deep learning. It is used in healthcare, agriculture, education, and gaming, among other things. This chapter is split into two parts. The first section covers the definition of AI, its applications, ML and its branches, DL models, and its functions. The second section introduces the concept of vehicle ad-hoc networks, including their characteristics, applications, and how vehicles communicate with one another. Furthermore, we will examine some VANET difficulties and challenges.

2.2 Artificial Intelligence

AI is the science of endowing programs with the ability to change themselves for the better because of their own experiences. The primary goal is to build an intelligent machine. The second goal is to find out about intelligence [13]. In his 2004 work, John McCarthy proposes that it is the science and engineering of making intelligent machines, especially intelligent computer programs. It is related to the similar task of using computers to understand human intelligence, but AI does not have to confine itself to methods that are biologically observable [14]. Stuart Russell and Peter Norvig define AI as Artificial intelligence means an intelligent agent. Agent means a software system that perceives its environment through sensors and acts upon that environment through actuators. Intelligence means the ability to select an action that is expected to maximize a performance measure [15] [16]. Basically AI is a simulation of natural intelligence in machines that are programmed to learn the actions of humans. Generally, AI is a technology that solves problems by combining computer science and massive datasets. It also includes the sub-types of ML and DL, which are frequently used in the context of AI. In these fields, AI algorithms are used to create expert systems that make predictions or classifications based on input data. It has also been applied in various domains.

2.3 Machine Learning

ML is the practice of programming computers to learn from data. In the following example, the program will easily be able to determine whether a given text (email) is spam or not spam. In ML, data referred to as called training sets [17]. A computer is trained how to employ the previously learned knowledge using ML without depending on human experience, it examines the available data to understand the context of these inputs and develop a potential conclusion. When a problem calls for employing several extensive lists of rules, ML is applied [18]. Machine learning methods can improve performance and make the programs simpler when there is no universally accepted solution to a complicated problem. In non-stable environments, machine learning algorithms may also adapt to new inputs [17].

2.3.1 Machine Learning Approaches:

Four key categories may be used to group machine learning algorithms:

1. Supervised Learning.
2. Unsupervised Learning.
3. Semi-supervised Learning.
4. Reinforcement Learning.

ML is a subset of AI techniques that enable computer systems to learn from previous experience (i.e. data observations) and improve their behavior for a given task. ML techniques include Support Vector Machines (SVM), Bayes learning, k-means clustering, association rule learning, regression, neural networks, and many more. The ML algorithms, which are widely in use, are Linear Classifier, Logistic Regression, glsNB, Bayesian Network, Decision Tree, Random Forest, AdaBoost, Bootstrapped Aggregation (Bagging), k-Nearest Neighbour (KNN) and Artificial

Neural Network (ANN). The Reality there are a growing number of ML algorithms and should note that there is no exact classification for ML algorithms. It suggests that several classes may be used to categorize different strategies. For example, supervised and reinforcement learning problems may be solved with Neural Networks (NNs).

2.3.2 Neural networks:

A subset of ML techniques known as ANN, often known as NNs, are largely based on biological neural networks. They are often described as artificial neurons, or a layer-organized collection of interconnected components. NNs can be used to solve supervised and reinforcement learning problems.Compared to earlier ML techniques, the Neural Network (NN) methodology offers more accuracy and efficiency, these networks are not intended to be accurate representations of the brain, but rather to simulate difficult problems, because these networks are powerful algorithms and data structures. NNs typically consist of three different kinds of layers: input, hidden (middle), and output layers, the NNs uses hidden layers to process received data and deliver it to output layers. With the right training, the network may transform inputs into higher-order feature representations, represent those representations as features at different scales or resolutions, link those representations to output variables, and eventually learn to predict the future. Back Propagation algorithm Neural network (BPNN), Radial Basis Function (RBF), Complementary Neural Network (CMPNN), and the probabilistic approach are some NNs techniques, while BPNN is the most used technique [19]. NNs may be divided into shallow networks (one hidden layer) and deep networks (multiple hidden layers). The multilayered architecture of NNs is what gives them considerable predictive power The fundamental type of NN, known as a single layer perceptron, uses a single neuron with programmable weights to categorize classes that can be separated linearly. An effective and reliable approach for modeling non-linear and complicated issues is the Multi-Layer Perceptron (MLP) [20]. The most complex task with NNs is determining the hidden layer's size. When the number of hidden layers is underestimated, the approximation will be poor, and when it is overestimated, generalization error and overfitting may follow [20]. As we come nearer today, a new era of NN called DL has evolved. The third ascent of NN has started generally in 2005 with the conjunction of several discoveries over a significant time span by late researchers Hinton, Andrew Ng, LeCun, Bengio, and other [21].

- **Type of Neural networks:** it have many types we illustrate some of them:

1. ANNs.
2. CNNs.
3. Recurrent Neural Networks (RNNs).
4. Generative Adversarial Networks (GANs).

And many others are examples of common deep learning architectures.

2.4 Deep Learning

The biggest limit of ML is the selectivity-invariance problem, which limits these systems' ability to evaluate the data in its original form. Selectivity-invariance involves ignoring features with less information and choosing those with more. Selectivity invariance, served as the primary motivation for developing the DL field. DL is now a developing and growing study area in computer vision and AI applications. In contrast to traditional algorithms, DL enables multiple layer models to learn information representations by studying the data in their original form. Unsupervised model training, which has the benefits of adaptability and transfer learning, is a requirement for DL [22]. Contrary to traditional ML, DL generally use a deep architecture of hidden layers and require a large amount of data to train a network. These methods improved the accuracy of various image processing applications, such as biology, audio, face, and object recognition. CNN, a type of Deep Neural Network (DNN), achieves excellent results when processing photos and videos. Recurrent Neural Network (RNN), another type of deep network, performs better when processing sequential data, such as text and speech [23]. The figure2.1 illustrates a Venn diagram from ML concepts to DL.

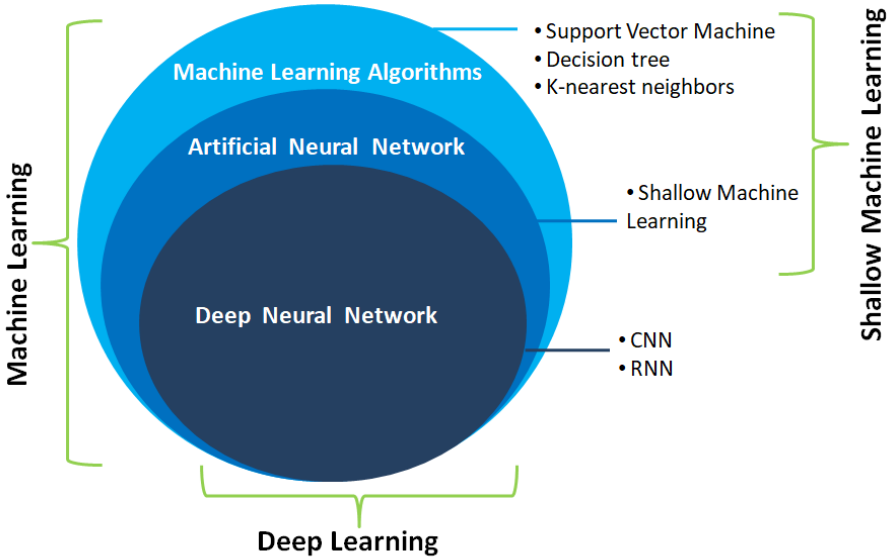


Figure 2.1: Venn diagram from ML concepts to DL [1].

Although the DL successes from the 2010s are believed to be under the confluence of three main factors:

1. New algorithmic advances that have improved application accuracy significantly and broadened applicable domains.
2. Availability of huge amount of data to train NNs.
3. Increase of computing power [24].

DL approaches have multiple abstractions of levels by using a non-linear model that transforms the original data into higher abstract levels for decision making. This simplifies finding the solution of complex and non-linear functions [25].

2.4.1 Applications of Deep Learning:

When using NNs to solve problems, the network architecture and training routine must be addressed in order to achieve high prediction accuracy [26]. DNNs, which have more hidden layers than shallow NNs, are thought to be better able to learn high-level features with greater complexity and abstraction [27].

1. Setting fine-grained specifics, such as activation functions (e.g, hyperbolic tangent, rectified linear unit (ReLU), maxout), layer types (e.g., fully connected, dropout, batch normalization, convolutional, pooling), and the architectural style of the network are all part of the definition of network architectures.

2. Setting the back - propagation learning schedules (stepwise, exponential), learning rules Stochastic Gradient Descent (SGD), SGD with momentum, root mean square propagation (RMSprop), Adam), loss functions (MSE, categorical cross entropy), regularization techniques (L1/L2 weights decay, early stopping), and hyper-parameter optimization are all necessary when defining training routines [24].

In short, DL uses a cascade of multiple layers of nonlinear processing units for feature extraction and transformation. The lower layers close to the data input learned simple features, while the higher layers learn more complex features derived from lower layer features. The architecture forms a hierarchical and powerful feature representation. This means that DL is suited for analyzing and extracting useful knowledge from both large amounts of data and data collected from different sources [28].

2.4.2 Overview of DL architectures

Veen gives a summary of the vast majority of DL designs [29]. Different DL architectures, including DNN, convolutional deep neural networks, deep belief networks, and RNNs, have been used in fields like computer vision, automatic speech recognition, natural language processing, audio recognition, and bioinformatics where they have demonstrated state-of-the-art performance on a variety of tasks. It's important to understand that several architectural styles are combined to create many applications.

- **Fully connected network:** The most popular architecture is the Fully Connected Layer (FC) network , which may be used to describe a wide range of issues using tabular data. They haven't been employed in fields like finance, physics, or biomedicine in a very long time [30].

- **Convolutional Neural Networks:** Are NNs created specifically to process images and videos. A wide range of image classifiers are now being used for a number of applications, such as classifying images of animals, X-ray scans, or galaxies. Other applications include image segmentation for crowd counting, autonomous driving, or satellite photography, among many more. When using sequential data, CNNs are looked as a good alternative to recurrent networks [25].

- **Recurrent Neural Networks:** Were created specifically to handle sequential data. Due to the internal feedback loops provided by its architecture, sequential pattern learning may simulate temporal dependencies by creating a memory, they are extensively utilized in Natural Language Processing (NLP), including Neural Machine Translation, language

Table 2.1: DL models at ILSVRC [].

Year	Model	Number of Layers
1990	LeNet	4
2012	AlexNet	8
2013	ZF Net	8
2014	VGG Net	19
2015	GoogLeNet	22
2015	ResNet	152
2016	SqueezeNet	14
2018	Inception	27

synthesis, and time series analysis, with applications in many disciplines including physics, medicine, and climatology. RNNs are used in combination with CNNs when the sequential data are images (videos) [25].

- **Long Short Term Memory network:** Since RNN architectures suffer from vanishing gradients, which results in little to no effect from early memories, Long Short Term Memory (LSTM) are the more complex architectures with improved attention mechanisms that attend to the issue of the RNN architecture.

- **Generative adversarial networks:** Interact with one another DL architectures to provide the best accurate data. Image style transfer, high resolution image synthesis, text-to-picture synthesis, image super-resolution for low dose PET reconstruction, anomaly detection, 3D object production for dental restoration, music generation, and many more applications are now in use [25].
  The CNN models LeNet, AlexNet, GoogleNet, Network in Network, and others are examples of popular CNN architectures. The following table 2.1 provides a comparison study of these CNN networks with another deep neural network.

2.4.3 Difference between DL and ML

The method used to extract the features that the classification uses is the main distinction between DL and ML. The classification performance of DL, which extracts features from several non-linear hidden layers, is significantly greater than that of ML, which depends on manually created features[2]. The figure 2.2 explains the difference between them.

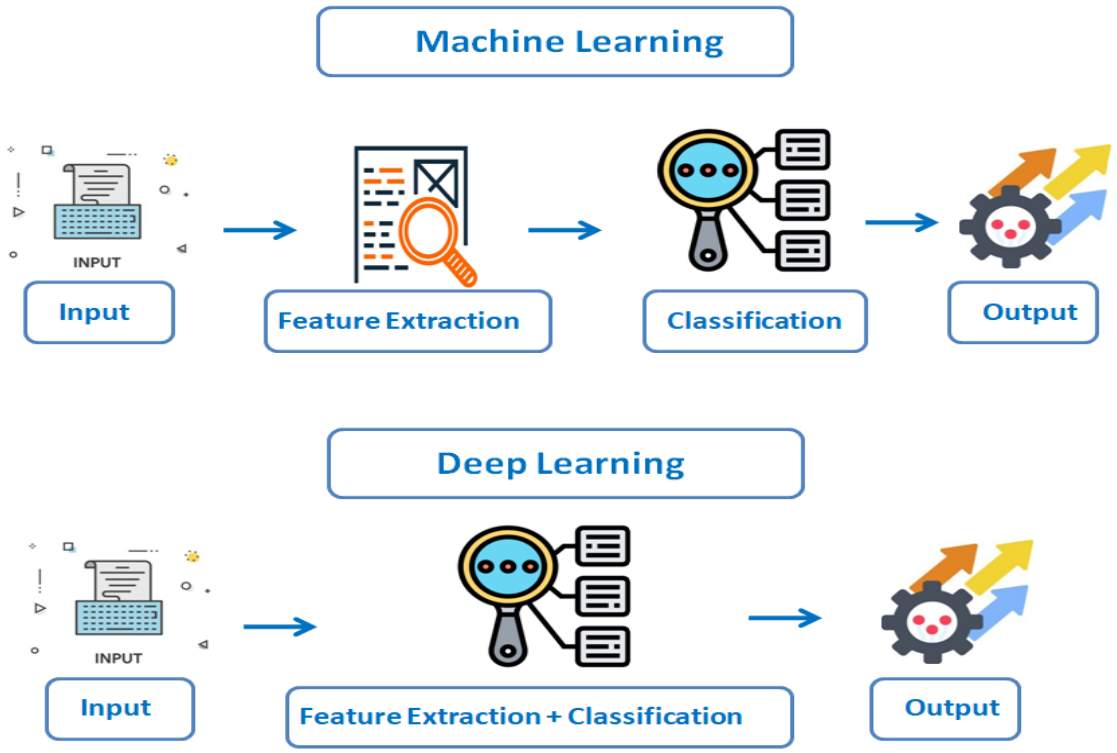


Figure 2.2: Difference between ML and DL [2].

2.5 Vehicular Ad-hoc Network

VANET is a subcategory of Mobile Adhoc Networks (MANETs), it is an important technique in Intelligent Transportation System (ITS). These networks contain There is no stable infrastructure, it can navigate with various mobile nodes, in addition, this network can offer many services like safety and security [31]. However, VANET composed a large number of vehicles characterized by a dynamic network topology due to the high mobility of vehicles, geographical positioning, the disconnection of the network during low dynamic traffic periods and no problem with the power [32]. A smart vehicle can contain many components, as shown in figure 2.3.

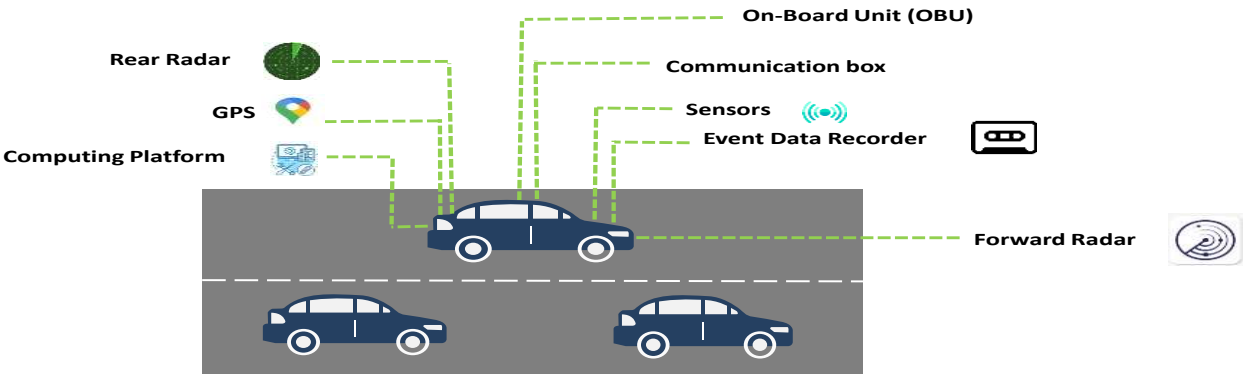


Figure 2.3: Smart and intelligent vehicles [3].

2.5.1 Communication Architectures

The critical component of ITS is how can vehicles exchange data between other, it use the concept of Internet of Vehicles (IoV) to improve driving safety via vehicle communication, that include Vehicle-to-Vehicle (V2V), Vehicle-to-Roadside (V2R) [33] and Vehicle-to-Every things (V2X) based on recent technologies.In this network each vehicle considered as a mobile node equipped with a On-Board Unit (OBU) communication. It can communicate with author vehicles, stationary access points located on the roads named as Roadside Units (RSU) through a OBU [32].

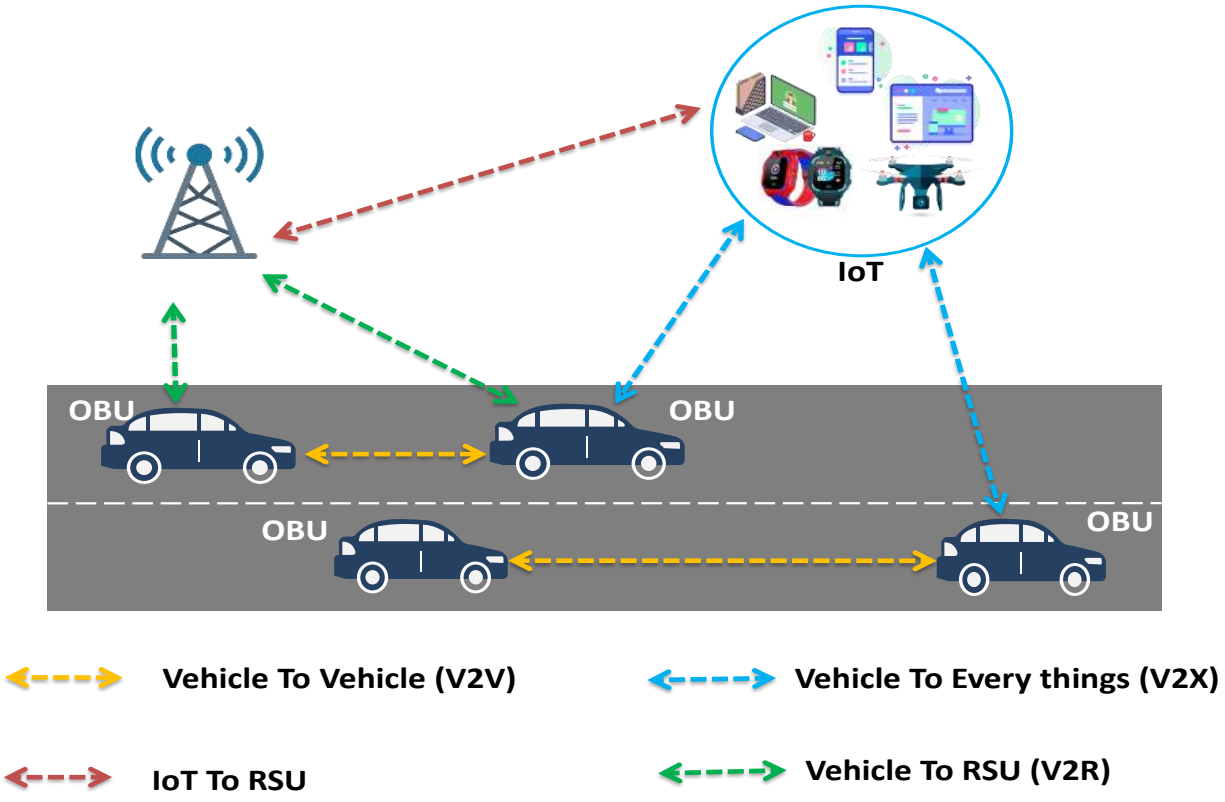


Figure 2.4: VANET System Architecture.

2.6 Conclusion

In this chapter, we present a background aboutAI, with their various domains ML and DL. Additionally, we introduced the main approaches of ML with their algorithms and mentioned the notion of NNs. Therefore, we explain the important class in AI, which is the DL with different applications and architectures. At the end of the chapter, we present the VANET, with their applications, communication architectures and challenges.

In the next chapter, we will present and analyze the existing techniques to detect drowsiness drivers based on DL and VANET approaches.

3

Driver Drowsiness Detection techniques: Related Work

Contents

3.1 Introduction

Every year, the National Highway Traffic Safety Administration (NHTSA) publishes shocking reports on how frequently accidents occur on the road. These statistics show how the risk of an accident is considerably increased when a driver is tired, drowsy, or uninformed of safe driving procedures. With the precautions, alerts, and preventative measures put in place to reduce the incidence of accidents globally, they nevertheless happen frequently and unexpectedly. Human error is mostly to blame for these accidents. The most significant human factors include intoxication, attention, drowsiness, speeding, and violating traffic and stop signals. To decrease the number of incidents affecting fatigued drivers, technology that can identify and alert drivers when they are in a poor psychophysical condition must be developed. However, creating such systems has numerous challenges. The researchers accomplished this goal by employing various techniques based on DL and VANET modeling models to alert drivers at the appropriate moment to minimize any crashes and reduce the impact of any accidents.

In this chapter, we will discuss some work and challenges that have been contributed to the field of drowsy driving in different methods using ed this goal by employing various techniques based on DL and VANET approaches .

3.2 Driver drowsiness detection technique

The main reason for many accidents occur is driver drowsiness detection, to reduce accident catastrophe they used a various techniques based on DL and VANET simulation approaches for alerting drivers at the right time to prevent any mishappening. Drowsiness in this case is captured using an auto camera located in a smart car. In the literature research of driver drowsiness detection can be classified into three main categories: behavioral, vehicular and physiological approaches. For behavioral measurement, it is based on DL with their architecture like CNN and ANN, each architecture have different models such as ResNet, Alex Connected Convolutional Networks (AlexNet) and VGG with their version. For the vehicular measurement, they used lane detection and steer wheel, for physiological measurement the DDD can be detected by various methods such as electroencephalogram (EEG) [34] as seen in figure 3.1. Also, it can detect by DL and the vehicular network approach at the same time. According to various techniques of drowsiness, we classified DDD approaches into two main categories: (i) DL and (ii) DL and VANET approaches.

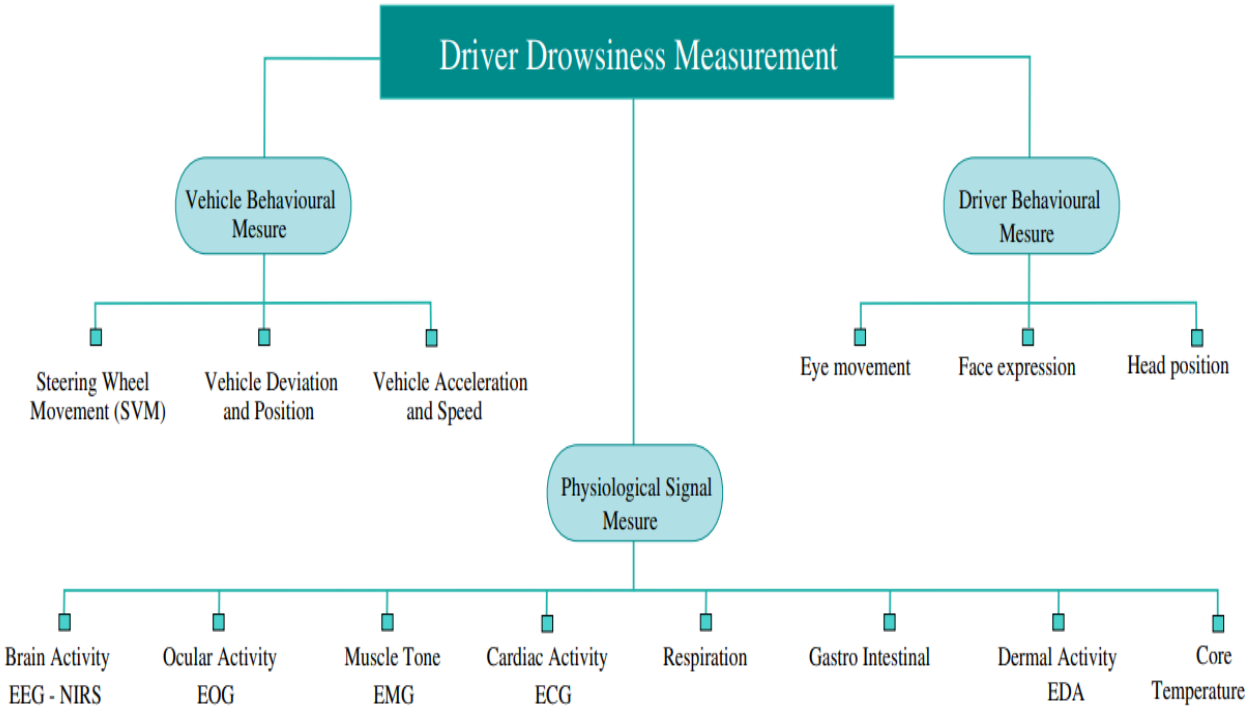


Figure 3.1: Summary of measurement approaches used DDD [4].

3.3 Based on DL Approach

3.3.1 Driver drowsiness detection using CNNs:

To correctly detect the drowsiness of the driver based on the car camera's fascial expression registrar. Zuzana Képešiová and Ján Cigánek proposed in [5], detection of driver sleepiness using the CNNs technique. To develop this application they use specific hardware with a software environment.

- **Hardware and Software characteristic:**
  - A laptop consisting of Intel® Core™ i5-7200U 2.50GHz – 2.71GHz Processor and 8GB GeForce 940MX Random Access Memory (RAM).
  - As a software environment PyCharm[1] development environment.
  - TensorFlow [2] and Keras [3] framework.

- **Dataset and Data Processing:**

  Zuzana Képešiová and Ján Cigánek collected data from 20 different kinds, each kind has eight different behavioral patterns. The behavioral patterns are divided into two main classes, aware and sleepy (table 3.1).

Table 3.1: Aware and Sleepy classes[5].

Aware(0):	**Sleepy(1):**
High Concentration.	Slow Eyelid Movement.
Looking Around.	Yawning.
Happiness.	Falling head.
Usual Face Expression.	Lethargy.

Each video is assessed into one amongst this category, with a rate of 28.16 frames per second. Each frame has mentioned by 1920 pixels wide and 1080 pixels high. The video is a recorder RGB color, it'll dig many frames, each of the five frames is considered an image. As a result, the dataset will contains 5258 frames divided into two classes, sleepy with 2619 frames and aware with 2639 frames. When a camera captures many objects in pictures (figure 3.2a) will be processed by changing from the origin dimension to 224x224 pixels and converting RGB colors to grayscale (figure 3.2b) as we shown in figure 3.2.

[1]Is one of aforementioned software applications of Jetbrains Corporation that is used to establish python scripts and applications such as NN [5].

[2]Is an open source platform for machine learning that provides an end-to-end solution. It allows you to train and deploy models using various methods, and it provides support for various data types[5].

[3]Is a high-levelNNs, written in Python and capable of running on top of TensorFlow. It was developed with a focus on enabling fast experimentation [5].

(a) Unpreprocessed video frame[5] .

(b) Preprocessed video frame [5].

Figure 3.2: Preprocessing data [5].

- **Developed and training Model:** This application is tried by Densely Connected Convolutional Networks (DenseNet) and ResNet but without high of accuracy, for that the 2 models aren't choosing. To correct the matter Visual Geometric Group Face (VGG Face)[4] was selected, VGG Face was trained by a dataset divided into batches sized of 32 during 50 epochs long process and with nadam optimizer and categorical crossentrophy loss function. As a result, this model takes **90.77%** training data accuracy and **98.07%** validation data accuracy, while the training occurred on hardware for 15 h.

3.3.2 CNN models-based ensemble approach to DDD

In the word there are many accidents occur with different kinds, 20 % of accidents happen because of drowsy condition of drivers.Mohit Dua et al. proposed [9]a system for driver drowsiness detection referred to as Deep CNN models-based ensemble approach to DDD. They divided the architecture into two parts : learning feature representation and ensemble.

- **For learning feature representation:** four models are choosing: ResNet, AlexNet, VGG Face and Flow Image Network (FlowImageNet) to extract different features by

[4]A VGG Face is a model divided into 11 blocks, were 8 blocks are convolutional blocks, where each block contains one or more convolution layers followed by a ReLu activation layer and/or max pooling layer [5].

controlling the network depth in some critical associated with drowsiness as we shown in table 3.2.

Table 3.2: Different parameters values associated with each CNN network[9].

Model	**No.of convolution layer**	**No.of FC layer**	**No. of parameters**
AlexNet	5	3	58,299,140
VGG Face	13	3	138,357,544
FlowImageNet	3	5	80,283,396
ResNet	48	2	23,587,712

After training the model from image input and extraction features, each network can be classified into multiple classes by using **Soft Max** function in each network.

- **For ensemble strategy:**

  Is used as if any of the model is shows a positive result, for instance showing drowsiness, which implies there are some frames within the video that are show the drowsy behavior of the driving force. This Algorithm 1 1, describes a video stream given as the input and frames are extracted from this input stream. These extracted frames are fed to the said deep learning models for training and prediction. of these four models work independently and provide independent outputs.

  The models classify the videos into four classes as:

  – non drowsiness.
  – drowsiness with yawning.
  – nodding.
  – eye blinking.

  These model outputs are ensemble and final output will be shown as: drowsiness or non-drowsiness.

Algorithm 1: : Driver Drowsiness Detection System [9]

Input : Video;

code

Driver Drowsiness Detection(Input Video)

{

// Categorize the video according to Behavioral features

1.FlowImageNet Output=model FlowImageNet(Input Video)

//Categorize the video according to Environmental features

2.AlexNet Output=model AlexNet (Input Video)

// Categorize the video according to Facial feature representation

3.VGG-16 Output=model VGG-16 (Input Video)

// Categorize the video according to hand gestures

4.ResNet Output=model ResNet (Input Video)

// Ensemble the outputs

5. output=ensemble(AlexNet Output,VGG-16 Output,FlowImageNet Output, ResNet Output)

if $(output > thresholdvalue)$ **then**

| 6. print ("Drowsy State")

end

else

| 7. print ("Not Drowsy State")

end

}

- **Hardware and Software characteristic:**
  - A laptop consisting of Intel® Core™ i5 2.4GHz Processor and 8GB RAM.
  - As a software environment Python 3.6 on Jupyter Notebook platform and Windows 10 development environment.
  - Videos of 640 x 480 pixels, 30 frames per second AVI format without audio.

The final accuracy after training, testing and ensemble came out to be **85%**.

3.3.3 Real-Time DDD System Based on CNN

To address the issue of driver safety on the road, Maryam Hashemi et al. proposed in [10] a system that uses CNN, to identify the driver's condition of sleepiness. One Fully Designed Neural Network (FD-NN) and two other networks using TL in VGG-16 and VGG-19 with extra designed layers TL-VGG are shown as viable networks for classifying eye state. The authors first discuss the eye recognition and preprocessing method, then they go into detail about the data collection method for detecting driver fatigue and analyze a different dataset in this area. Regression tree machine learning is employed for this.

- **Hardware and Software characteristic:**
  - A laptop consisting of Intel® Core™ i7-6700K 4.00GHz Processor and 16GB RAM NVIDIA GeForce GTX 1070Ti GPU.
  - Python has been chosen as the platform's language of programming.
- **Regression tree machine learning method:** The Viola and Jones algorithm is used for head detection and headbox estimation in order to identify the eye region and feed it into the network. The Kazemi and Sullivan work is implemented for the facial landmark approach.
- **Dataset and Data Processing:** The study examined two datasets. The first is the **ZJU**[5] dataset, while the second is a combination of the **ZJU** dataset and the authors' own built database.

  Additionally, The extended dataset is combined with **ZJU** to a comprehensive database to train the model, it includes 2383 closed eyes and 2458 open eyes, taken from 80 video clips. These clips were taken of 20 different people, with four clips being recorded for each person:
  - One for the frontal view without glasses.
  - One for the frontal view while wearing thin rim glasses.
  - One for the frontal view while wearing black frame glasses.
  - The final clip for the upward view without glasses of the left and right eyes are collected independently using a symmetry-based technique for Treatment.

[5]The **ZJU Eyeblink** Database is a collection of 4157 images2100 with open and 2057 closed eyes from 4 distinct people [10].

To train a network with sub-sample images, all of these images have been geometrically normalized to 24x24 pixel. The data set was separated into two groups after being collected from various rotations and distances. The first category accounts for the fact that the driver's head is straight and their eyes are spinning. The second group applies the data when the driver turns his head, but it does so inside the permitted range.

- **Networks Architectures:** This study proposes three distinct neural networks, and compares the result of each network for both datasets:
  - **Fully Designed Neural Network:** The network used a 2D convolutional layer with a 3x3 filter size and assigned Rectified Linear Unit (ReLu) as the activation function. Reduce the number of features by using maxpooling with a 2x2 size. at learning state **25%** of connections are deactivated to avoid overfitting. Implement data flatten to vectorize it for the next layers. They employed the **Sigmoid** function for binary classification output as the final activation layer.
  - **Transfert learning TL-VGG16 and TL-VGG19:** It is suggested that the VGG-16 and VGG-19 be employed as pre-trained CNNs for features extraction. Using a smaller dataset and shorter training times is another advantage of transfer learning, which is used to build deeper networks with improved accuracy.
- **Training model and Results:**

Images were collected at a frame rate of six during the final detection process. The network will detect danger and send the driver a verbal alert if it determines that the probability of an eye being close is more than **50%** for more than 12 successive images. Tiredness can be indicated by prolonged periods of closed eyes (greater than one second).

The ROIof the training process classified each eye into one of two groups of open or closed eyes, and the author used this information as input to the DDD system using the TL-VGG16, TL-VGG19, and FD-NN networks.

The authors try performing a study on accuracy and AUC by comparing the results of the training and test processes using the three networks TL-VGG16, TL-VGG16, and FD-NN on the two datasets of **ZJU** and extended zju dataset separately. The following tables 3.3 and 3.4 display the obtained results.

Table 3.3: Accuracy and AUC of Three Proposed Network on **ZJU** Dataset[10].

Network	Accuracy	AUC	Epoch
TLVGG16	94.96%	99.0%	50
TLVGG19	95.45%	99.0%	100
FD-NN	98.15%	99.8%	50

Table 3.4: Accuracy and AUC of 3 Proposed Networks on Extended Dataset[10].

Network	Accuracy of validation	AUC of validation	Accuracy of test	AUC of test	Epoch
TLVGG19	96.42%	99.4%	96.09%	99.3%	100
FD-NN	97.01%	99.4%	96.79%	99.3%	50
TLVGG16	98.53%	99.8%	97.54%	99.5%	100

According to the authors, the advantages of the suggested methodologies, include excellent accuracy and minimal processing complexity. To choose the most reliable network, accuracy and other factors were assessed for the three prospective networks. According to on the experiment results, the FD-NN network has a **98.15%** accuracy rate and a **99.8%** AUC. The required time for eye state classification in the FD-NN network is **1.4 ms**, making it reliable for real-time tasks as well.

3.3.4 Hierarchical DNNs to detect driver drowsiness

In their paper, Samaneh Jamshidi et al. proposed in [11] an Hierarchical Deep Drowsiness Detection (HDDD) network, which consists of deep networks with split spatial and temporal phases. The suggested technique employs ResNet to determine the driver's face, the lighting condition, and whether or not the driver is wearing glasses. This results in a significantly higher percentage of eyes and mouth detection in the following step. The temporal information between the frames is then used by the LSTM network.

The four main phases of the proposed framework for drowsiness detection are face location, situation recognition, eye and mouth state detection, and drowsiness detection. Only one frame is considered at in the first three phases, and these phases' decisions are unaffected by the temporal dimension. The fourth phase of the proposed method is the temporal component, which employs a series of frames.

- This approach uses a hierarchical framework to identify pertinent information from the current frame, such as lighting conditions and the presence of eyeglasses.
- After face extraction, the result is transferred to a network for situation detection, and the output is supplied to a network pre-trained for that condition in accordance with the obtained information.

The driver executes a series of eight operations in five distinct scenarios, including Day-BareFace, Day-Glasses, Sunglasses, Night-BareFace, and Night-Glasses, using the **NTHU-DDD** dataset that the authors used. All experiments and results were obtained in accordance with the two parts for training and evaluation. The following four stages were used to complete this work:

- **Face location:** All the frames are initially labeled with the **MTCNN** method order to get face coordinates. The frames that in this method are unsuccessful are manually labeled. The images are first resized to 112x112 and fed into the ResNet network with four outputs. The Adam algorithm was employed as an optimizer,

with a learning rate of 0.0001. This network has been trained for 20 epochs. The weights that had the highest accuracy in the evaluation data were retained. The accuracy of the evaluation data is **97.53%**, compared to **96.06%** accuracy of the training data.

- **Situation detection:** The status of the face images obtained in the previous step is categorized into the following five states: Sunglasses, Day-BareFace, Day-Glasses, Night-BareFace, and Night-Glasses. The cropped face images that have been resized to 64x64 are then input into a ResNet18 network to its fully-connected layer , and a new fully-connected layer is created with the same number of neurons as the outputs. For measuring error, Cross-Entropy is employed, and, similar to the previous phase, the Adam optimizer is applied. The accuracy of the training data is **98.977%**, while the accuracy on the assessment data is **98.03 %**.
- **State detection:** After labeling the states of the mouth and eyes in the **NTHU-DDD** [6]dataset, it was determined that the mouth may be in any of the three states listed above the following: normal, speaking-laughing or sleepy. The eyes might be in either a normal or sleepy condition. The labels on the frames that stated State detection: After labeling the states of the mouth and eyes in the **NTHU-DDD** dataset, it was determined that the mouth may be in any of the three states listed above the following: normal, speaking-laughing, or sleepy. The eyes might be in either a normal or sleepy condition. The labels on the frames that stated "sleepy and speaking" or " sleepy and laughing" were removed.sleepy and speaking" or " sleepy and laughing" were removed.

  Five different DNNs were trained, one for each moth and eye condition. The dropout approach is used to address the issue of overfeeding that the authors faced during learning because of the large number of parameters. Then, they try their best to resolve this issue by simplifying the network and reducing the parameters. However, it was discovered that by doing so, the labels were affected by previous frames and did not correspond to just one frame.

  For this reason, each frame must have correct labeling. is a result of the CNNs extracting and classifying each frame's spatial data. The authors have decided to employ the VGG16 network as a result. This network's fully connected layer was initially linked to 1052 neurons, then five, and it was then trained for each state separately. The validation accuracy of the proposed state detection model using evaluation data in the **NTHU-DDD** dataset for each situation is mentioned in the following table 3.6.

[6]The entire dataset (including training, evaluation, testing dataset) contains 36 subjects of different ethnicities recorded with and without wearing glasses/sunglasses under a variety of simulated driving scenarios, including normal driving, yawning, slow blink rate, falling asleep, burst out laughing, etc., under day and night illumination conditions [35].

Table 3.5: Validation accuracy of the proposed state detection model using evaluation data in the **NTHU-DDD** dataset for each situation[11].

Situation	**Accuracy(%)**
Day-BareFace	92.67
Day-Glasses	82.81
Sunglasses	80.76
Night-BareFace	88.34
Night-Glasses	79.62

- **Face location:** All the frames are initially labeled with the **MTCNN** method order to get face coordinates. The frames in this method that are unsuccessful are manually labeled. The images are first resized to 112x112 and fed into the ResNet network with four outputs. The Adam algorithm was employed as an optimizer, with a learning rate of 0.0001. This network has been trained for 20 epochs. The weights that had the highest accuracy in the evaluation data were retained. The accuracy of the evaluation data is **97.53%**, compared to **96.06%** accuracy of the training data.
- **Situation detection:** The status of the face images obtained in the previous step is categorized into the following five states: Sunglasses, Day-BareFace, Day-Glasses, Night-BareFace, and Night-Glasses. The cropped face images that have been resized to 64x64 are then input into a ResNet18 network to its fully-connected layer was modified, and a new fully-connected layer is created with the same number of neurons as the outputs. For measuring error, Cross-Entropy is employed, and, similar to the previous phase, the Adam optimizer is applied. The accuracy of the training data is **98.977%**, while the accuracy on the assessment data is **98.03 %**.
- **State detection:** After labeling the states of the mouth and eyes in the **NTHU-DDD** dataset, it was determined that the mouth may be in any of the three states listed above the following: normal, speaking-laughing or sleepy. The eyes might be in either a normal or sleepy condition. The labels on the frames that stated State detection: After labeling the states of the mouth and eyes in the **NTHU-DDD** dataset, it was determined that the mouth may be in any of the three states listed above the following: normal, speaking-laughing, or sleepy. The eyes might be in either a normal or sleepy condition. The labels on the frames that stated "sleepy and speaking" or "sleepy and laughing" were removed.sleepy and speaking" or " sleepy and laughing" were removed.

  Five different DNNs were trained, one for each moth and eye condition. The dropout approach is used to address the issue of overfeeding that the authors faced during learning because of the large number of parameters. Then, they try their best to resolve this issue by simplifying the network and reducing the parameters. However, it was discovered that by doing so, the labels were affected by previous frames and did not correspond to just one frame.

  For this reason, each frame must have correct labeling. is a result of the CNNs

extracting and classifying each frame's spatial data. The authors have decided to employ the VGG16 network as a result. This network's fully connected layer was initially linked to 1052 neurons, then five, and it was then trained for each state separately. The validation accuracy of the proposed state detection model using evaluation data in the **NTHU-DDD** dataset for each situation is mentioned in the following table 3.6.

Table 3.6: The accuracy of the proposed method in the NTHU-DDD data set[11].

methods	HDDD (%)
Day-Barefaced	94.21
Day-Glasses	85.95
Sunglasses	82.09
Night-Barefaced	91.25
Night-Glasses	82.43
Average	87.19

The average accuracy of the drowsiness detection system increased to **87.19%** percent in general.

3.4 Comparison and summary

In this section, a comparative study of DL approaches is presented. Table 3.7 summarizes the main characteristics of the discussed categories, where DL approaches are classified according to their model requirements, and other criteria such as dataSat and accuracy obtained.
By studying these methods and as presented before, they classified into tow modes by transfer learning and without using transfer learning. We can say that the VGG and all versions are poorly studied in the Application of deep learning to driver drowsiness detection.

Table 3.7: DL approaches for driver drowsiness detection.

	Models							Other Criteria	
DL approach-based	VGG 16	VGG 19	VGG Face	AlexNet	ResNet	FlowImageNet	Inception V3	DataSet	Accuracy
RT-DDD [36]	✓	✓	✗	✗	✗	✗	✗	**ZJU**	**90%**
DDD-CNN [5]	✗	✗	✓	✗	✗	✗	✗	**2000**	**90.77%**
HDNN-DDD [37]	✗	✗	✗	✗	✓	✗	✗	**NTHU**	**87.19%**
DCNN-DDD [9]	✗	✗	✓	✓	✓	✓	✗	**NTHU**	**85%**

3.5 Based on DL and VANET Simulation Approach

DDD is the main cause of many accidents, thus to reduce the degree of accidents, there are a plethora of studies on the detection of driver fatigue and drowsiness using various types of measurement technologies, methods using deep learning being one of them, as well as technologies based on driving behavior (vehicle-based), driving behavior (video-based), and driver physiological signals measure.

Driving behavioral measurement: this involves keeping an eye on the car and it surrounds while evaluating how you drive. The degree of drivers' attention is estimated using various signs supplied by sensors built into the vehicle. **Measurement of Driving Physiological Markers:** This method is based on various physiological signals that are visually correlated with driver weariness and sleepiness. **Driver Behavioural Measure:** Rather of concentrating on the driver's driving activities, this technique focuses on the driver's behavior by watching his movements and facial expressions. Some of these motions might be symptoms of depletion and fatigue. Our methodology, which is based on DL and VANET, is used with this technology.

As seen in the accompanying diagram 3.1, each of these technologies is composed of various methodologies.

The majority of these techniques and methods are combined with ML and DL techniques, such as ANN, CNN, RNN, and other architectures that have expanded their capabilities with several additional models, the most common of which are as follows: ResNet, VGG, and AlexNet. Another approach is studies with VANET simulation to identify driver sleepiness.

In this context we can classify the DDD approaches into three mains categories: (i) VANET (ii) DL and (iii) DL with VANET approaches as shown in figure 3.1.

The detection of drowsiness is one of the behavior of the driver that the researchers

have focused on in their past studies on VANET networks to identify. This proved their performance in the field. One of the most recent contributions is that of Genaro Rebolledo-Mendez and associates, whose work is mentioned in "Developing a Body Sensor Network to Detect Emotions During Driving." They profited from the trend of Body Sensor Network (BSN) applications, a recent development in automobile safety. But combining diverse body sensors with VANET was a challenge. The authors suggest a detector of human emotions by providing an exploratory study proving the possibility of detecting one emotional state in real time using a BSN. Based on these results, they proposed a middleware architecture that could connect a vehicle's onboard unit to city emergency services, VANET, and roadside devices [38].

In the most recent research, the researchers introduced the use of deep learning and VANET simulation, proving their efficiency in predicting driver behavior, among which driving drowsiness is one. We may cite some of the research conducted in this area. Such as the work of Seong Kyung Kwon and colleagues, who developed the "settings Open Access Article Driving Behavior Classification and Sharing System Using CNN-LSTM Approaches and V2X Communication" project [39].

Their concept is based on a 4-layer CNN-2 stack LSTM-based sharing and V2X system that classifies driving behavior and uses time-series data as an input to reflect temporal changes. Using only the 3-axis acceleration of the driving vehicle and sharing it with the surrounding vehicles via a VANET network, the proposed system categorizes driving behavior into defensive, normal, and aggressive driving.

In the same context, Mohammad Shahverdya and his coworkers have developed their project, "Driver behavior detection and classification using deep convolutional neural networks," by investigating an ITS to address the issue of privacy violation and the potential for spoofing in the VANET network using cutting-edge yet effective DL method for analyzing the driver behavior that is comprised of a 2D CNN on images constructed from driving signals based on recurrence plot technique. Additionally, the use of driving signals such as acceleration, gravity, throttle, speed, and Revolutions Per Minute (RPM) to distinguish between five different driving behaviors, such as normal, aggressive, distracted, sleepy, and intoxicated driving [40].

3.6 Conclusion

We have attempted to focus on related studies that have developed their projects in the context of driver drowsiness detection in this chapter. Using various techniques, including Deep Learning and VANET or a combination of the two DL and VANET approaches, in the final section of the chapter, we present the works discussed in this chapter in a comparative table.

In the next chapter, We will outline our procedure and the results we got when we describe our approach for detecting driver drowsiness.

4

Driver Drowsiness Detection using DL And VANET Approaches: Our contribution

Contents

4.1 Introduction

Deep learning techniques have been very successful in all aspects of daily life. Through the application of intelligent detection or prevention systems, such as driver drowsiness detection, they can be used in the field of road safety to help many people save their lives against road accidents. Reducing the significant number of road accidents caused by this phenomenon has been a challenge for many years. Our study is a study that has contributed to the field of driver drowsiness detection, through the application of several DL techniques, mentioning CNNs, TL, and data augmentation techniques that allow the machine to learn faster. Also, we illustrate another approach using VANET network to prevent accidents.
In this chapter, we'll go into depth about the work we did and describe the hardware and software tools that we used.

4.2 Our Proposed method:

Several studies has been implemented to fix the problem of driver drowsiness. However, none of the proposed solutions combined a drivers drowsiness detection CNN model, as a first step to simulation scenarios, in order to alert the network of the drowsy driver as well as to take an action to awake the driver. To do so, in our proposed method (Figure 4.1) we started with a first phase that involves DL approaches including deep transfer learning and data augmentation techniques. To create an accurate CNN model that classifies images into drowsy or awake driver, thus detect driver drowsiness. As for the second part, it is based on the outcome of the first one. In which, after the prediction of a drowsy driver, the simulation is carried out in two scenarios. The first case involves broadcasting the information received by the training model, in the VANET network to warn vehicles in the range of the vehicle whose driver may be drowsy. As a result, the vehicles must react to this warning message by slowing down, stopping their cars, or even going faster. In order to avoid a road accident with the car that presents a risk to others. The second scenario is based on the employment of an alarm to awaken the driver who is asleep, either through a device (such as smartphone or smartwatch), or a vibrator that moves the driver's seat to wake him up.

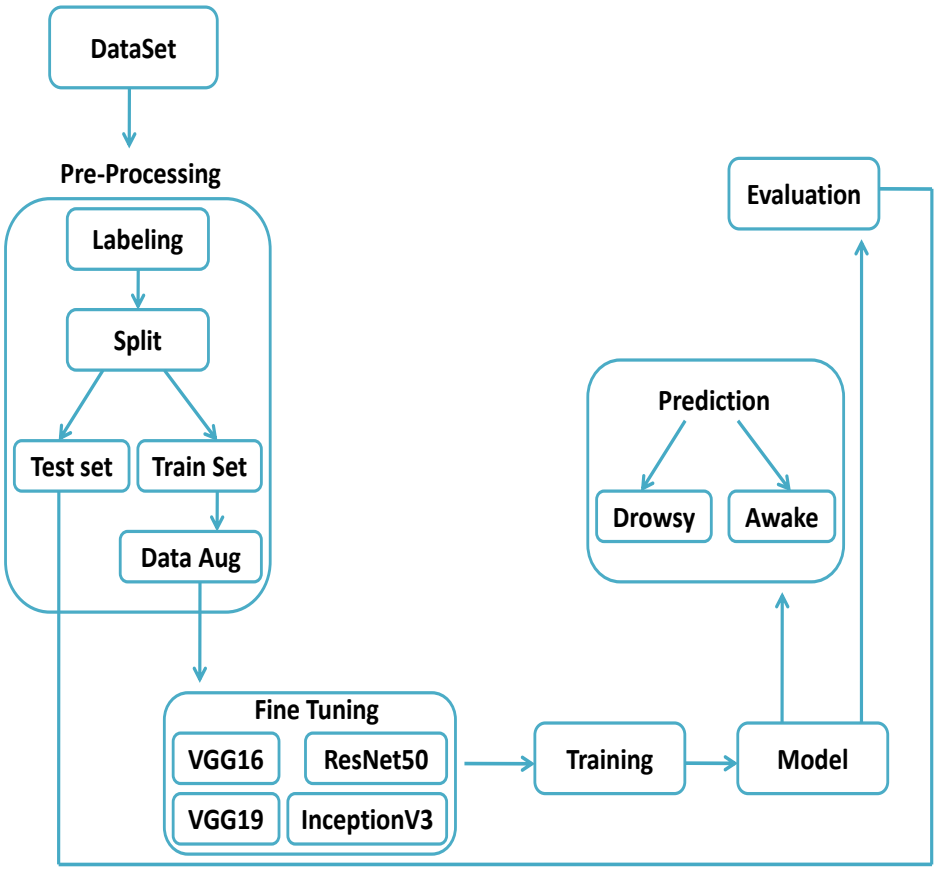


Figure 4.1: Stage 01: Proposed DL Approach.

4.2.1 Development Environment

It is not sufficient to have large data sets for training in various branches of artificial intelligence, you also must have powerful hardware and deploy software designed specifically for AI. This is possible because of the platform's powerful Central Processing Unit (CPU) and GPU, performance-enhancing tools, and other characteristics.

* **Hardware Characteristics and Software Environments:** In our situation, we used a personal laptop with powerful characteristics and important software that may aid us in creating efficient training results.
* Our computer is a **DELL G15 5510** with an Intel® Core™ i7-10870H 2.20GHz, 16 GB of RAM and a 512 GBSolid-State Drive (SSD) hard drive. It is powered by GPU .
  - Nvidia RTX 3060 GPU with a performance capacity of 16 GB.

We need to employ a variety of platforms, tools, and libraries for the software part. Our Nvidia RTX 3060 graphics card needs the Compute Unified Device Architecture (CUDA) toolkit to increase the training performance of our dataset, thus installing CUDA and Compute Unified Device Architecture Deep Neural Network (CUDnn) is necessary to have a successful learning experience.

We did CUDA 11.6 andCUDnn 8.1.1 installation first.

* **CUDA:** is a parallel computing platform and programming model developed by NVIDIA for general computing on graphical processing units GPU. With CUDA, developers are able to dramatically speed up computing applications by harnessing the power of GPU. The CUDA Toolkits from NVIDIA provides everything you need to develop GPU-accelerated applications. The CUDAToolkit includes GPU-accelerated libraries, a compiler, development tools and the CUDA runtime [41].
* **CUDnn:** Is a GPU-accelerated library of primitives for Deep Neural Network. CUDnn provides highly tuned implementations for standard routines such as forward and backward convolution, pooling, normalization, and activation layers. Deep learning researchers and framework developers worldwide rely on CUDnnfor high-performance GPU acceleration. It allows them to focus on training neural networks and developing software applications rather than spending time on low-level GPU performance tuning [42].

In order to employ Python as a learning tool that demonstrated its superior capacity and opportunity in the world of DL and AI in general, we also needed to use the Anaconda platform.

* **Anaconda:** In order to employ Python as a learning tool that demonstrated its superior capacity and opportunity in the world of DL and AI in general, we also needed to use the Anaconda platform who carries the logo "Data science technology for groundbreaking search, a competitive edge and a better world"[43]. In order to do the training, we first installed Python 3.9 on the Anaconda platform, under which we imported the keras library, tensorflow, matplotlib, numpy, panda, etc.
* **Keras:** The open-source software program known as Keras offers a Python interface for ANN. The high-level Keras API makes it simple to build and train

models while using TensorFlow and ML by offering the TensorFlow library interface [44].

* **Tensorflow:** TensorFlow is an end-to-end open source platform for machine learning, that enables building and implementing ML models simple for both beginner and expert users[45].
* **Visual Studio Code:** is another tool we use to edit Python code.
* **The proposed architecture Neural Network CNN:** We chose to use CNN networks because, as part of our contribution to the detection of driver drowsiness using deep learning techniques, we must classify images.CNN are a subsection of NNs, which are currently one of the models for image classification that is generally regarded as the most efficient, A deep CNN model has a restricted number of processing layers that can learn different incoming features.CNN has different types of layers as shown in figure4.2 .

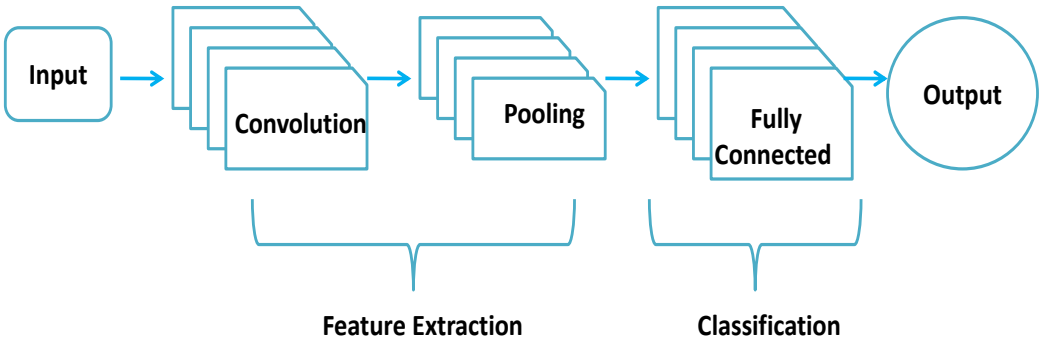


Figure 4.2: CNN Architecture [6].

- **Convolutional Layer:**
  is a set of learnable filters (kernels). The filter sizes may be 3x3 to 5x5 to 7x7. The number of channels in the input is represented by the filter's third dimension [6] . A filter is a collection of multipliers that enables us to extract features that are essential for classifying an image. There are several filters that may be randomly started and trained to do tasks like identifying an object in an image.

- **Pooling Layer:** After filtering, the amount of the input typically increases, hence CNN frequently uses the pooling layer, also known as sub sampling or down sampling, to reduce the size of the data. The filter size and strides are the two layer parameters with the most commonly applied values. Max pooling and Average Pooling are two types of pooling layers where the maximum and average value are taken, respectively.
- **Fully Connected Layer:**is the final layer in The CNN. the dropout regularization approach can be used to reduce over fitting. The output layer, the last fully connected layer, has the same number of neurons as the classes [6].

* **Deep Transfer learning:**
TL is the process of employing a model that has already been trained to address a different problem. It's now highly popular in DL since it can train DNNs with relatively limited data. Using the TL, which is ML technique, we can apply the learning from a model that was previously used for a similar task. This is very beneficial for data scientists because the majority of real-world problems usually lack the millions of labeled data points required to train such complex models.

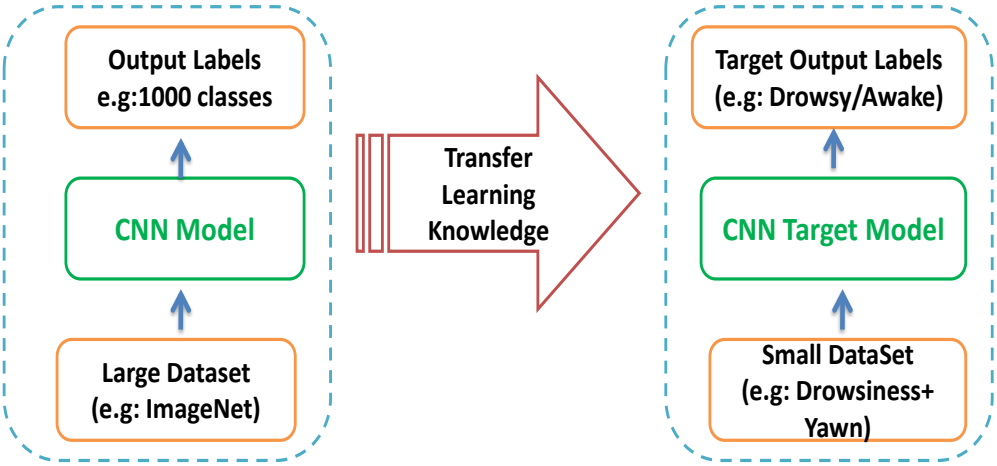


Figure 4.3: Concept of Transfer Learning [7].

* **Transfer Learning Models of training:**
According to the Keras API, there are various models available for excellent training, some of them are pre-trained, so we are under the Deep transfer learn-

ing tab, while others are not, so the programmer must train the model from scratch. In our scenario we opted to the Deep transfer learning, and preferred to employ the VGG16, VGG19, Inception V3 andResNet50 V2 models.

- **RESNET 50 V2:** ResNet 50-V2 is famous for its depth (152 levels) and the addition of residual blocks. By introducing identity skip connection so that the layers can transfer their input to the following layer, the residual solved the problem of training an complicated deep architecture [12]. ResNet 50-V2 architecture is seen in figure 4.4.

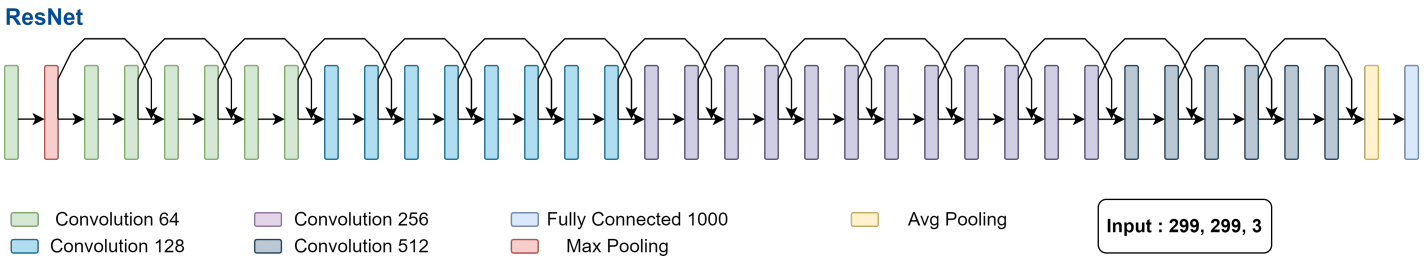


Figure 4.4: Architecture of ResNet[8].

- **VGG:** VGG16 is a CNN trained on the ImageNet database. VGG19 was proposed by Oxford University's Visual Geometry Group in 2014. It performed well on the ImageNet dataset. Figure 4.6. illustrates more details about VGG 16 and VGG 19 architecture.

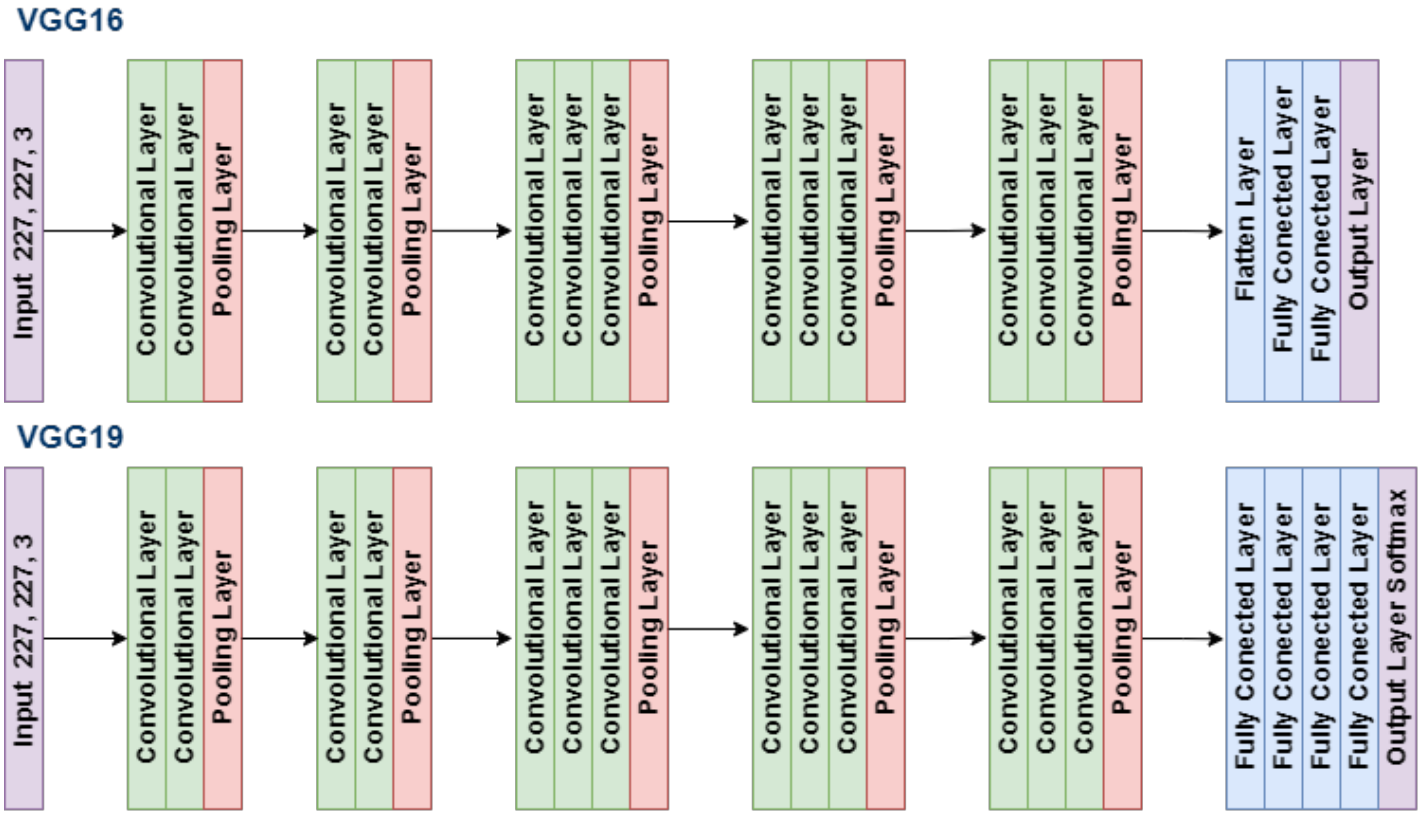


Figure 4.5: VGG 16 and VGG 19 Architecture [8].

· **Inception V3:** The Inception-v3 significantly achieves all others in object recognition. The Inception-v3 model is comprised of three parts: the basic convolutional block, the modified Inception module, and the classifier. The basic convolutional block, which alternates between convolutional and max-pooling layers, is used for feature extraction. The improved Inception module is built on the Network-In-Network concept (Lin et al., 2014) [46] [47]
Inception V3's proposed architecture is based on factorizing the classic 7x7 convolution into three 3x3 convolutions[48].

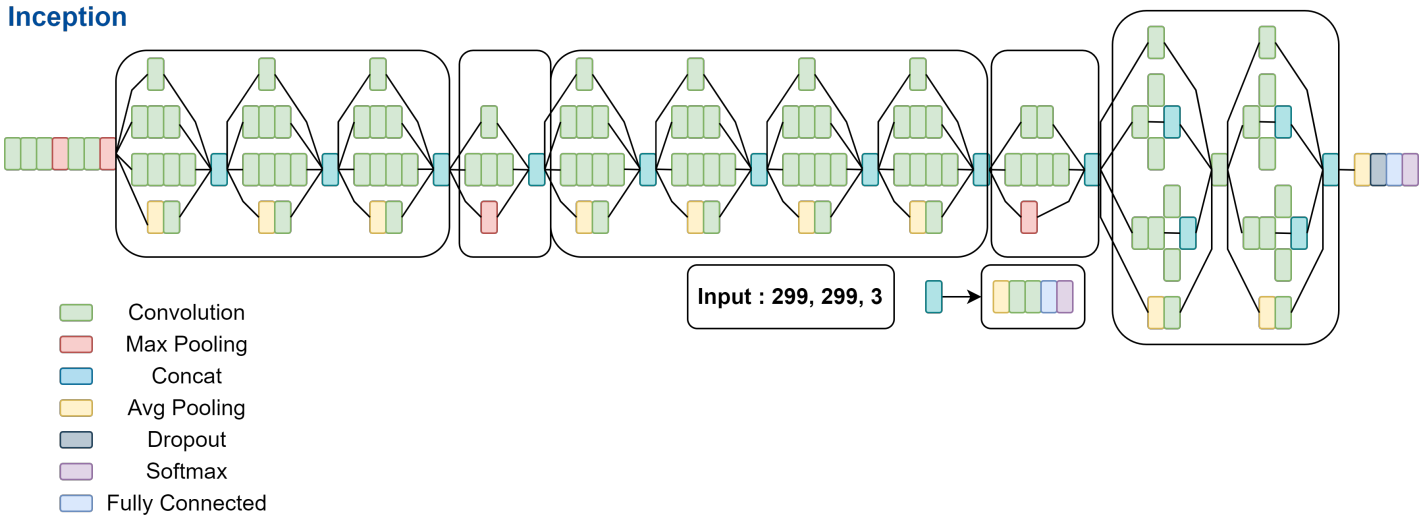


Figure 4.6: Inception V3 Architecture [8].

∗ **Dataset:** We employed two datasets for this project, Yawn Dataset and Drowsiness dataset, which are available from Kaggle [49], a website that provides an endless amount of datasets among all study domains. The 5119 images in the Yawn Dataset have already been given labels, and the Drowsiness dataset, which has 2900 images, We then divided the images into the two classes, train and test, using a split of 70% for the train phase and 30% for the test phase as shown in table 4.4.

Figure 4.7: Drowsiness and Yawn DataSet.

Table 4.1: Description of Used Dataset

Class	Number of Images		
Database	Drossiness Dataset	Yawn Dataset	Total
Drowsy(Closed Eyes)	726	0	726
Drowsy(Yawn)	0	2591	2591
Awake(Open Eyes)	726	0	726
Awake(Not Yawn)	0	2528	2528
Total	1452	5119	6571

4.3 Deep Learning Approach

4.3.1 Pre-processing section:

In first time, we attempted to gather our datasets. The first dataset contains a set of images of open and closed eyes, with images of drivers yawning or not yawning, which prompted us to load another dataset that contains images of open and closed mouths to avoid going through the phase of extracting Region Of Interest (ROI), which is a step in the pre-processing phase that include the extraction of the most important part of the image [6]. The Keras platform allows us to apply the Image Data Generator phase in every models that we have used in our implementation, which is used to assign labels to images in binary mode in the dataset, our classes are: sleepy or awake, and are divided into 70% for the train and 30% for the test (Table 4.4). After that the training set was augmented.

4.3.2 Data Augmentation:

The data augmentation process, which provides a range of options that may be applied to images to improve training results, such as rotation, zoom, translation, and so on as we can see in table 4.2.
Then, using TL, we tried to adapt the pre-trained models we selected to all of our images in order to identify which one is most appropriate for our study.ResNet50V2, VGG 16,VGG 19, and InceptionV3 are the models used. It is worth mentioning that all of these procedures are carried out using the four training models. After that, we

Table 4.2: Data Augmentation.

Random Rotation	0.2
Random Width	0.15
Random Height	0.15
Random Zoom	0.01
Random Translation	width factor =0.01, height factor=0.01
Random Flip	Horizontal

added the pre-processing function of each selected pre-trained model, which contain also some data augmentation parameters.
Additionally, we created our model in two aspects:

* **Functional API:** which enables us to design models in more flexible manner by identifying connections between layers.
* **Sequential API:** which allows us to build models layer by layer but not design them.

The following figure 4.8, demonstrates the use of Sequential and Functional API with their results obtained:

The following table 4.3 details the learning rate, the number of epoches utilized in the training and test phases, the learning time, and all other hyper-parameters:

```
[16]  model.summary()

...  Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_4 (InputLayer)        [(None, 224, 224, 3)]     0

 sequential_1 (Sequential)   (None, 224, 224, 3)       0

 inception_v3 (Functional)   (None, 5, 5, 2048)        21802784

 global_average_pooling2d_1   (None, 2048)             0
 (GlobalAveragePooling2D)

 dropout_1 (Dropout)         (None, 2048)              0

 dense_1 (Dense)             (None, 1)                 2049

=================================================================
Total params: 21,804,833
Trainable params: 21,770,401
Non-trainable params: 34,432
_________________________________________________________________
```

Figure 4.8: Inception V3 Architecture.

Table 4.3: Fine Tuning Parameters.

Global Pooling Layers(2D)	Global Max Poling /Global Average Pooling
Dropout Layer	0.3 / 0.2
Optimizer	Keras SGD
Learning Rate	0.0001 / 0.001
Momentum	0.9
Nesterov	True
Loss Function	Binary Crossentrop
From Logits	True
Metrics	Binary Accuracy
Number of Epochs	60
Steps Per Epoch	10

4.3.3 Fine-tuning process:

The models were imported with its own weights that are parameterized on the ImageNet dataset at the start, with an input shape of (224, 224, 3), and without including top, which means that we did not include the fully-connected layer at the top of the networks. As well After that, the output of each model was adjusted to

Table 4.4: DataSet Splitting.

Class	Drowsy	Awake	Total
Number of Images	3254	3317	6571
Training Set (70%)	2278	2322	4600
Testing Set (30%)	976	995	1971

suit our problem's output using the **Sigmoid** activation function, since our case is a binary classification.
We also employed Global Average Pooling (GAP) or Global Max Pooling (GMP) techniques, which is a pooling process that substitutes fully-connected layers in CNNs by creating one feature map for each corresponding class of the categorization tasks. The output, which is the Max or Average depending on whether GMP or GAP is used, and will take it for each feature map and feed the resulting vector directly into the **Sigmoid** layer [50]. Supposing we take the final convolutional layer output $(5, 5, 2048)$ feature map from the pre-trained model Inception V3 (Figure 4.8). The purpose of GAP (or GMP) is to compute the average (or maximum) for each of the 2048 feature maps. As a result, the feature maps will generate 2048 feature points, which we can then concatenate into a 1×2048 feature vector and feed into our sigmoid output for image classification.
Table 4.5 shows the feature vector of each model after the global pooling. We tested the pre-trained models including a global average pooling, then a global max pooling separately in order to determine which one of them is the most suitable for each model in our case.

Table 4.5: Models Feature Maps

Model	Last Convolutional Layer	After Global Pooling
ResNet50V2	$(7, 7, 2048)$	$(None, 2048)$
InceptionV3	$(5, 5, 2048)$	$(None, 2048)$
VGG16	$(7, 7, 512)$	$(None, 512)$
VGG19	$(7, 7, 512)$	$(None, 512)$

4.3.4 Discussion and results:

* **Training phase:** During the training phase, we ran several tests on the four models that we opted for, in order to determine the best parameters that may be used to increase training results and accuracy. The training phase took the same time for all trials of each model which is about 16 min for 60 epoch. As a consequence of examining the results in the table 4.6, we may make the following observations:
  - The use of the drowsiness Dataset, which contains 2900 images and was

divided into two key classes: drowsy and awake during the split and labeling phases; the ResNet50 and VGG 16 models, provided us with a low accuracy of 60%, which is due to the images of the drivers' faces, which determine the driver's yawn or not yawn state without ROI that is in this case the mouth.

- On the other hand, we got a very excellent accuracy of 100% for the two models ResNet50 and VGG16 by removing the images that do not include ROI to evaluate the driver's level of drowsiness.

In order to optimize the training, we have to go through the pre-processing step to extract the ROI of the images that can indicate whether the drivers are yawning or not.

To avoid going through the pre-processing phase to extract ROI, we preferred to look for a dataset that contains images of open and closed mouths that determine the yawn or not yawn state of the driver rather than going through the pre-processing phase to extract ROI. To do so, we loaded the Yawn Dataset from Kaggle platform, which contains 5119 images of open and closed mouths.
The training results indicated that the VGG19 reached the highest accuracy of 100%, followed by ResNet50V2 and Inception V3 with 94.99% accuracy, meanwhile VGG 16 achieved the lowest accuracy of (55%) as shown in table 4.6.

Table 4.6: Training.

DataSet	**Class**	**Model**	**Accuracy**	**Loss**
Drowsiness	Awake/Drowsy	ResNet50V2	60%	0.676
Drowsiness	Awake/Drowsy	VGG16	69.99%	0.693
Drowsiness	ClosedEye/OpenEye	ResNet50V2	100%	0.0027
Drowsiness	ClosedEye/OpenEye	VGG16	100%	0.00107
Drowsiness+Ywn	Awake/Drowsy	ResNet50V2	94.99%	0.0230
Drowsiness+Ywn	Awake/Drowsy	VGG16	55%	0.6900
Drowsiness+Ywn	Awake/Drowsy	VGG19	100%	0.0099
Drowsiness+Ywn	Awake/Drowsy	InceptionV3	94.%99	0.6900

We obtained results from different trials for each pre-trained model. The training metrics were accuracy and loss; as shown in figure 4.9 of a trial for ResNet50 V2.

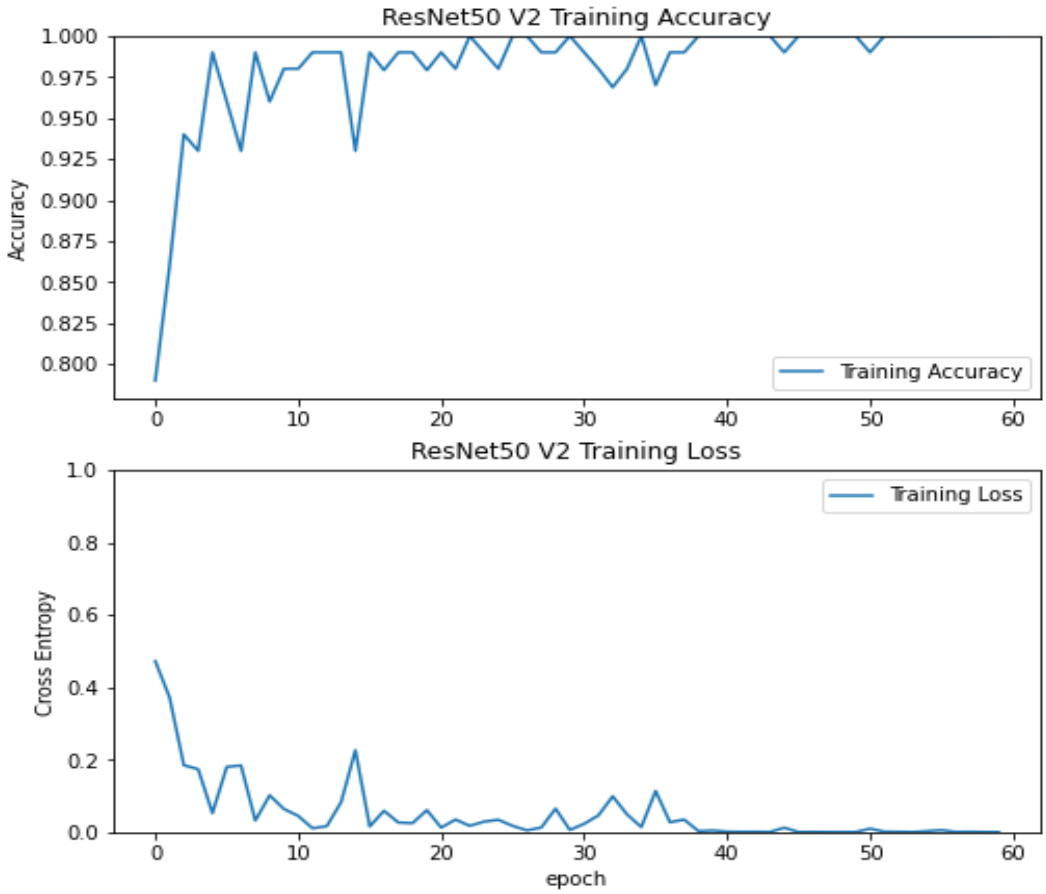


Figure 4.9: ResNet50V2 Training Accuracy And Loss .

We can notice that the model started reaching its highest accuracy and stabilized starting from the 20th epoch. As for the loss values, the stabilization started from the 40th epoch.

4.3.5 Testing phase:

Several evaluating metrics are used in this phase, such Accuracy, Sensitivity, Specificity, the Receiver Operating Characteristic (ROC), and the confusion matrix. As mentioned before, 30% of our gathered dataset was used for evaluation set.
Each of the accuracy, sensitivity, and specificity were calculated based on the values obtained as True Positive, True Negative, False Positive, False Negative.

* **TP:** True Positive is the total number of truly identified drowsy images.
* **TN:** True Negative is the total number of truly identified awake images.
* **FP:** False Positive is the total number of awake driver images predicted by the model as drowsy driver.
* **FN:** False Negative is the total number of drowsy driver images predicted by the model as awake driver.

Table 4.7: Confusion Matrix Example.

	Predicted Class Labels	
True Class Labels	Drowsy	Awake
Drowsy	**TP**	**FN**
Awake	**FP**	**TN**

The accuracy of the model's performance is determined by measuring the correct predictions made.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP}\%[6] \tag{4.1}$$

The ROC curve is a graph of operational points that shows the True Positive Rate (TPR) as a function of the False Positive Rate (FPR), also known as sensitivity or recall it is defines in Equation 4.2, whereas equation 4.3 defines theFPR, also known as specificity 4.11.

$$Sensitivity = \frac{TP}{TP + FN}\%[6] \tag{4.2}$$

$$Specificity = \frac{TN}{TN + FP}\%[6] \tag{4.3}$$

Looking at the rates of sensitivity and specificity we notice that the VGG 16 had a TPR (Sensitivity) of 0% compared to its FPR (specificity) of 100% which shows its failure in the detection of Drowsy images, on the other hand the ResNet has the highest TPR between the four models. Furthermore, The ROC curve is closer to the top left corner, and the AUC (Area Under Curve) was 0.97 for each of ResNet50 V2 and Inception V3, as shown in figure 4.11, who could detect the majority of the images when compared to VGG16.

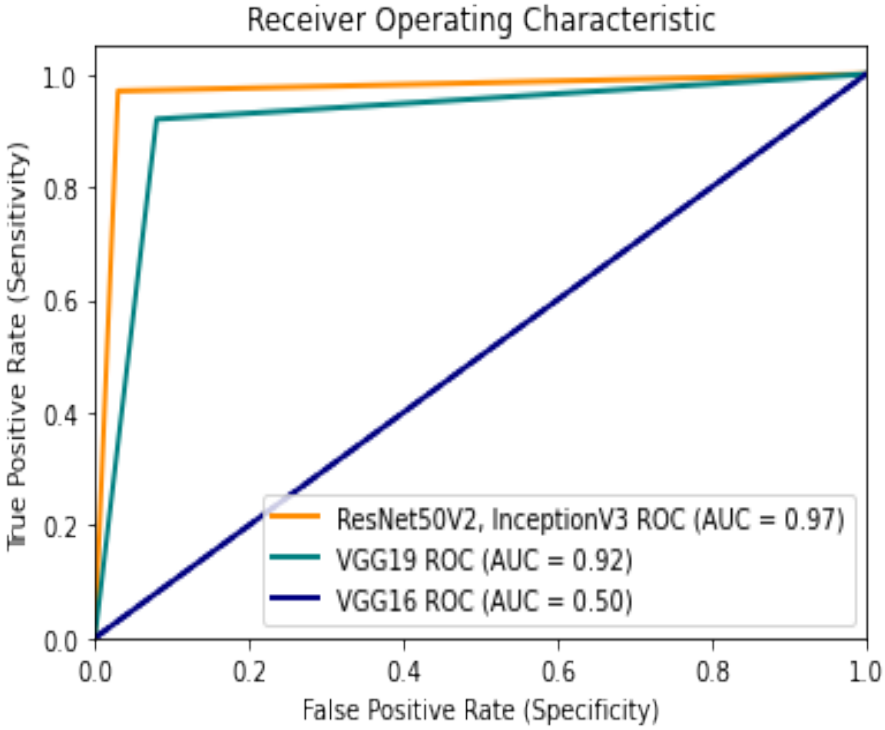


Figure 4.10: Receiver Operating Characteristic Graph.

In the testing table 4.8, the best results out of each trial of each model were considered. We were able to determine the accuracy, sensitivity, specificity, and AUC (Area under the ROC Curve) from each confusion matrix of each model. Furthermore, during the testing phase, we ran several trials on our models, including the ResNet50 and VGG 16 models on the drowsiness dataset, which gave accuracy of 66.70% and 51.09%, respectively, and this low accuracy is due to the dataset that contains images with no ROI, as we described in the train phase. However, when we merged the yawn and Drowsiness dataset to test our models, we notice that the highest accuracy was obtained by Inception V3 with 96.85%, followed by ResNet50V2 with 96.70%. Meanwhile, VGG 19 reached a good accuracy as well with 92.23%. On the other hand, the lowest accuracy was achieved by VGG16 with 50.48%. The reason behind that is mainly due to the model's architecture (figure4.6. architecture VGG16) that is simpler as compared to the ResNet and Inception V3 architectures.

4.3.6 Comparison with Others Approaches

Based on the obtained results, we can conclude that our technique generates superior results when compared to the previous research presented in the second chapter for detecting driver drowsiness. So, by comparing the methods used in earlier studies, we can refer the good results to several reasons, including the combination of the two drowsiness and yawn datasets that contain pre-processed images with their ROI. We can also observe that, although some of the previous studies used the same pre-trained models, such as VGG 16, VGG 19, our results are still the best; we can attribute this to the parameter that we changed in our models, which allowed us

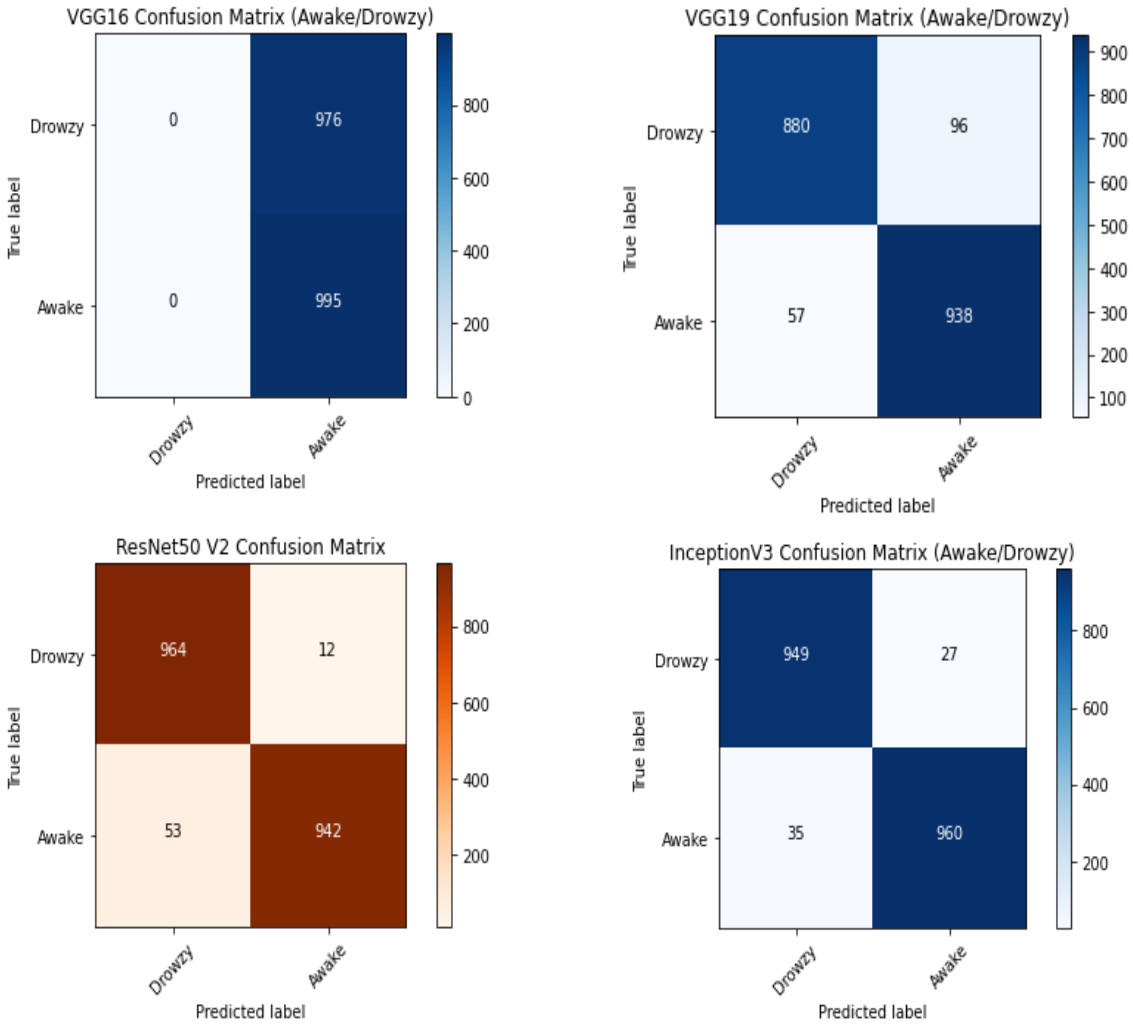


Figure 4.11: Confusion Matrix of Models.

to achieve the best results. Among the reasons that also allowed us to obtain these results is the use of the GAP and GMP parameter, which are two crucial parameters in the Data Augmentation phase, and which is a pooling process employed in traditional CNN to replace fully connected layers.

Table 4.8: Testing.

DataSet	Class	Model	L.rate	Global Pooling	Dropout	AUC	Accuracy	Sensitivity	Specificity
Drowsiness	Awake/Drowsy	ResNet50V2	0.001	GAP	0.2	0.67	66.70%	49.19 %	84.36 %
Drowsiness	Awake/Drowsy	VGG16	0.001	GMP	0.3	0.51	51.09	97.01%	05.27%
Drowsiness	closE/OpenE	ResNet50V2	0.001	GAP	0.2	1	99.77	99.54%	100 %
Drowsiness	closE/OpenE	ResNet50V2	0.001	GMP	0.3	0.99	98.85	100%	97.70%
Drowsiness	closdE/OpenE	VGG16	0.001	GMP	0.3	1	99.54	100%	99.54%
Drowsiness	closE/OpenE	VGG16	0.001	GAP	0.2	0.5	50 %	100%	0%
Drow+Yawn	Awake/Drowsy	ResNet50V2	0.001	GAP	0.2	0.97	96.70%	98.77%	94.67%
Drow+Yawn	Awake/Drowsy	VGG16	0.001	GAP	0.2	0.51	50.48%	0%	100%
Drow+Yawn	Awake/Drowsy	VGG19	0.001	GAP	0.2	0.92	92.23%	88.44%	94.27%
Drow+Yawn	Awake/Drowsy	InceptionV3	0.001	GAP	0.2	0.97	96.85%	97.23%	96.48%

4.4 Simulation with VANET Approach

Drowsiness, defined as the state of sleepiness of drivers , could be a major cause of street accidents and has critical suggestions for street security. A few dangerous mishaps can be anticipated in the event that the drowsy drivers are warned in time. This state can settled by numerous of methods using deep learning with different strategies talked about within the final segment. in this section we outline another approach in our contribution which is employing a VANET simulation with a scenarios explained in the figure 4.12.

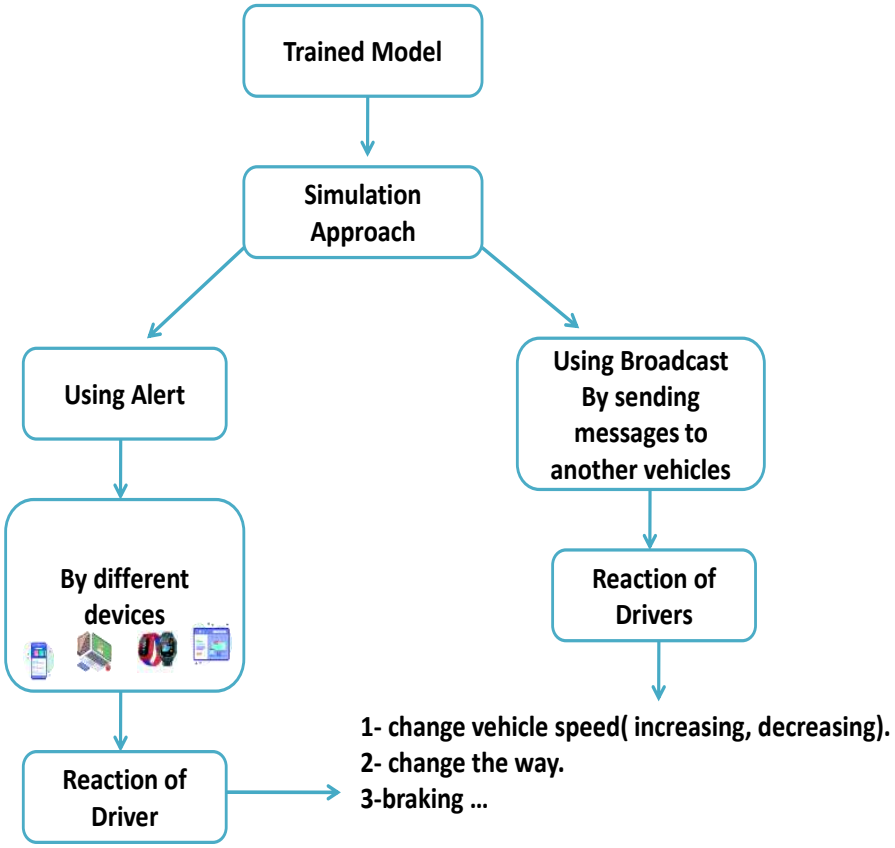


Figure 4.12: Stage 02: Proposed VANET Approach.

4.4.1 VANET Approach

* **Using Alert:**
  A assortment of drowsiness location strategies exist that screen the drivers' drowsiness state whereas driving . in the event that they are not concentrating on driving. In our situation all vehicles are combined with a camera to taking pictures from time to time , When camera capture photographs the deep learning approach will be executed and classify the picture to drowsy or awake . In the event that the result of deep learning taken a tired picture the driver ought to be educated by sending alarm to his devices such as phone, portable workstation

or his hour to require a choice around changing vehicle speed, changing the way or braking the car.

* **Using Broadcast Messages:** Another strategy is proposed within the moment case utilizing recreation with VANET approach which is broadcast messages within the network of VANET . in this situation in case the result gotten by the deep learning strategy is drowsy this vehicle will be broadcast messages to another vehicles that found within the same extend to prevent accidents. When the drivers gotten this messages they will be alter the way or alter the speed of the car. This case are divided into protocols in each situation as we illustrate in the figure 4.13:
  - When the drowsy driver in the vehicle located in the center of a set of vehicles in the road.
  - When the drowsy driver in the vehicle located in the end of set of vehicles in the road.
  - When the drowsy driver in the vehicle located in the first of set of vehicles in the road.

  It'll be way better to utilize **Omnetpy** to interfaces python code that contains the deep learning approach. The important component within the network it the choosing of routing protocol it'll be superior to choose a proactive protocols since they have a low idleness to get messages by drivers.
  Based on the evaluation results, The Inception V3 is the most suitable model for our study. Hence, in our proposed algorithm (Figure 4.14) we explained our scenarios. First, we defined the function of production starting with pre-processing phase. After the prediction step, If the state of the driver is drowsy, the Alert and broadcast part comes along. The First case is based on sending the alert to the sensor's speed. After that, using the sensors, an alert will be sent to the driver's seat to wake him up, or to a smart device (watch or phone). As for the broadcasting case, is based on the usage of packet messages, which contain the position of the drowsy driver's car. Each vehicle that is in the same range and received a packet will take an action based on its position.

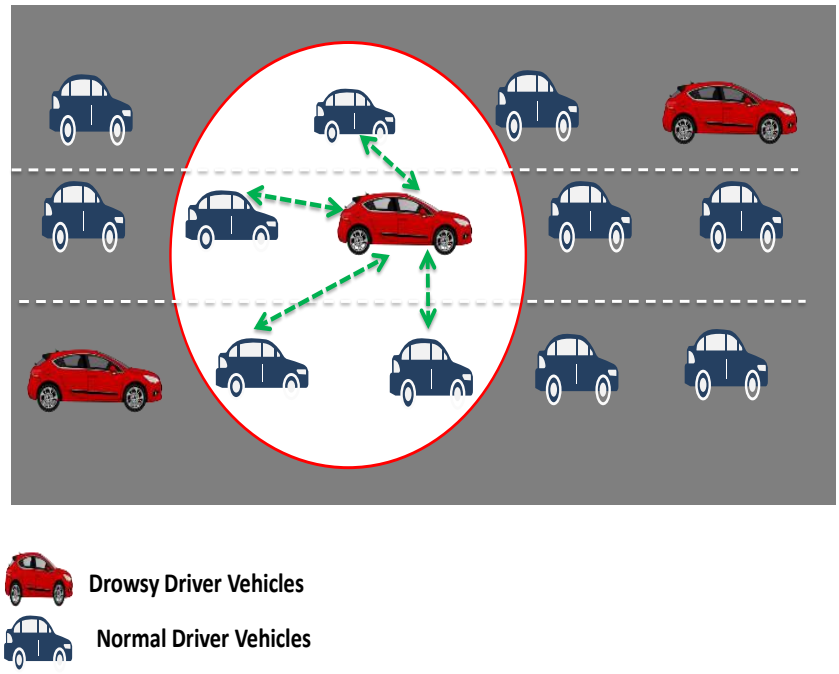


Figure 4.13: Using Broadcast Messages.

Algorithm :

```
Def Broadcast (Packet) :
    Position= ["Behind", " Front"]
    Packet = Position
    If Packet == "Behind" :
        Reduce The Spead
    Elif Packet == "Front" :
        Accelerate
    Else :
        Go To The Secure Cross


Def Predict (Image_Path) :
    Class=["Drowsy","Awake"]
    model = tf.Keras .models.load_model("Inception V3.h5")
    Image=cv2.imread(Image_Path)
    Image=cv2.resize(Image,(224,224))
    Image=Image.reshape(-1,224,224,3)
    Image=Image/255.0
    Prediction=model.predict(Image)
    If Predection[0][0] < 0.5 :
        percentage= Predection[0][0] * 100
        return Class[1], percentage
    Else :
        percentage = (1 - Prediction [0][0]) * 100
        return Class[0], percentage
        Send Alert To Speed Sensors
        Send Alert To Driver Seat To Awake Him OR Send
    Alert to Smart Devices
        Broadcast
```

Figure 4.14: Proposed Algorithm Of VANET Approach.

4.5 Conclusion

In our study, which is the detection of driver Drowsiness by the use of the two approaches of DL and VANET, we opted for using the Deep Transfer Learning method, which proposes the pre-trained models, which include the ResNet50V2, VGG 16, VGG 19, and Inception V3 used in our study. These models are very good models relative to the results we obtained after training and testing on the combination of our two Datasets: Drowsiness and Yawn. Then, after several trials on the selected model and with changing of several hyper-parameters, we found that the Inception V3 model provided the greatest results, with an accuracy of **96.85%**, which is the highest in comparison to earlier studies.
In the second part of our study, the simulation phase supposed to be implemented using scenario of broadcasting and system alarm based on the learning results. However, due to a lack of time and documentations about tools for connecting training results with theVANET network using the OmnetPy simulator, we were incapable to implement our proposed scenario.

5

Conclusion and future perspectives

Contents

5.1 Summary of our Work

Artificial intelligence and computer vision have taken up a great place in our daily lives and have demonstrated a great performance. AI and its features allow us to invest in areas that have been difficult to invest in for a short period. Among these sectors are those of road safety, which is used to organize the road and even save lives because road accidents have claimed the lives of many people. Drowsy driving is a behavior of drivers that could result in major accidents in the area of traffic safety. To address this issue, researchers have worked to detect driver drowsiness using Machine Learning and Deep Learning approaches, which have shown to be excellent solutions to difficult challenges in this area. Our study is divided into two parts: detecting driver drowsiness using DL approaches and using the results comes from the first part as input into the simulation using the VANET approach. We combined two datasets, drowsiness and yawn, including more than 6000 images. After organizing our gathered dataset, we passed into the train and test phases. The training and testing results were great compared to previous studies. During the training phase, we obtained 100% accuracy for the ResNet50V2 and VGG 19 models using only the Close/Open eye images in the Drowsiness dataset, as well as for the VGG19 using the new dataset (drowsiness+yawn). For the test phase, the Inception V3 model provided the highest accuracy of 96.85% compared with previous studies, ResNet50V2 provided 96.70% accuracy, VGG19 provided 92.23% accuracy, and VGG16 provided the lowest accuracy of 50.48%, which may be due to several limitations that we will discuss in more detail.

5.2 Aventages and Limitations

During our study, we had several advantages in addition to several limitations. Among the advantages, we could train and learn on a personal PC equipped with a very powerful GPU, we also managed to achieve excellent accuracy compared to several previous studies, and we could test several pre-trained models, allowing us to compare and see which parameters are more suitable for driver drowsiness detection. The usage of an unbalanced dataset, which was a combination of two datasets (Drowsiness and Yawn), where we can observe that the number of eyes open/closed (1452 images) was much lower from Open/Close mouth (5119 images), led the test of some models to achieve a poor accuracy. The size of the dataset can be an issue during the training and test stages. as we could test more than four models; As we can see, the loss value for the Inception V3 model was a little high, although it provided greater accuracy than the other models, and this is also related to the size of the dataset. Among the limitations that can be mentioned is the lack of a pre-processing step for extracting the ROI on the Drowsiness dataset, which could prevent us from removing more than 1448 images that do not have the ROI on the driver's mouth, causing us to decrease the size of the new dataset (Drowsiness + Yawn). The biggest limitation we had was that we couldn't implement the simulation part due to lack of time, as well as the implementation language chosen Omnet, which we couldn't connect with the training implementation language **Python 3.9** so we needed more time to move forward with the implementation phase.

5.3 Future Perspectives

To build on the work accomplishments, several future perspectives for designing a new driver drowsiness detection technique should be considered. One of the next work directions is to implement our second technique in Omnetpy or another simulator using simulation with a VANET network approach. Additionally, we intend to improve our technique by using a balanced dataset and a larger number of images. Finally, future studies will primarily focus on the use of various classification models based on Deep Learning and Transfer Learning Approaches with different hyper-parameters and machine characteristics.

Bibliography

[1] Christian Janiesch, Patrick Zschech, and Kai Heinrich. Machine learning and deep learning. *Electronic Markets*, 31(3):685–695, 2021.

[2] Protima Khan, Md Fazlul Kader, S. M. Riazul Islam, Aisha B Rahman, Md Kamal, Masbah Toha, and Kyung Kwak. Machine learning and deep learning approaches for brain disease diagnosis: Principles and recent advances. *IEEE Access*, 9:37622 – 37655, 02 2021.

[3] Biswa Ranjan Senapati and Pabitra Mohan Khilar. Automatic parking service through vanet: A convenience application. In Himansu Das, Prasant Kumar Pattnaik, Siddharth Swarup Rautaray, and Kuan-Ching Li, editors, *Progress in Computing, Analytics and Networking*, pages 151–159, Singapore, 2020. Springer Singapore.

[4] Messaoud Doudou, Abdelmadjid Bouabdallah, and Véronique Berge-Cherfaoui. Driver drowsiness measurement technologies: Current research, market solutions, and challenges. *International Journal of Intelligent Transportation Systems Research*, 18(2):297–319, 2020.

[5] Zuzana Képešiová, Ján Cigánek, and Štefan Kozák. Driver drowsiness detection using convolutional neural networks. In *2020 Cybernetics Informatics (K I)*, pages 1–6, 2020.

[6] Saida Sarra Boudouh and Mustapha Bouakkaz. Breast tumor detection in mammogram images using convolutional neural networks. In *2022 IEEE 9th International Conference on Sciences of Electronics, Technologies of Information and Telecommunications (SETIT)*, pages 330–336, 2022.

[7] Saida Sarra Boudouh and Mustapha Bouakkaz. Breast cancer: Breast tumor detection using deep transfer learning techniques in mammogram images. In *2022 International Conference on Computer Science and Software Engineering (CSASE)*, pages 289–294, 2022.

[8] Saida Sarra Boudouh and Mustapha Bouakkaz. Breast cancer: Toward an accurate breast tumor detection model in mammography using transfer learning techniques. 2022.

[9] Mohit Dua, Shakshi, Ritu Singla, Saumya Raj, and Arti Jangra. Deep cnn models-based ensemble approach to driver drowsiness detection. *Neural Computing and Applications*, pages 1 – 14, 2020.

[10] Maryam Hashemi, Alireza Mirrashid, and Aliasghar Beheshti Shirazi. Driver safety development: Real-time driver drowsiness detection system based on convolutional neural network. *SN Computer Science*, 1(5):1–10, 2020.

[11] Samaneh Jamshidi, Reza Azmi, Mehran Sharghi, and Mohsen Soryani. Hierarchical deep neural networks to detect driver drowsiness. *Multimedia Tools and Applications*, 80:1–14, 04 2021.

[12] Road traffic injuries. https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries#:~:text=Approximately%201.3%20million%20people%20die,pedestrians%2C%20cyclists%2C%20and%20motorcyclists., 2022.

[13] Roger C. Schank. What is ai, anyway? *AI Magazine*, 8(4):59, Dec. 1987.

[14] John McCarthy. What is artificial intelligence? page 14, 11 2004.

[15] Jonas Schuett. A legal definition of ai, 08 2019.

[16] Nils J. Nilsson. Stuart russell and peter norvig, artificial intelligence: A modern approach. *Artificial Intelligence*, 82:369–380, 1996.

[17] John McCarthy. Machine learning: Step-by-step guide to implement machine learning algorithms with python. page 106, 05 2018.

[18] Neha Sharma, Reecha Sharma, and Neeru Jindal. Machine learning and deep learning applications-a vision. *Global Transitions Proceedings*, 2(1):24–28, 2021. 1st International Conference on Advances in Information, Computing and Trends in Data Engineering (AICDE - 2020).

[19] Nicholas Carlini and David A. Wagner. Towards evaluating the robustness of neural networks. *2017 IEEE Symposium on Security and Privacy (SP)*, pages 39–57, 2017.

[20] Gabriel Goh, Nick Cammarata, Chelsea Voss, Shan Carter, Michael Petrov, Ludwig Schubert, Alec Radford, and Christopher Olah. Multimodal neurons in artificial neural networks. 2021.

[21] Pramila P. Shinde and Seema Shah. A review of machine learning and deep learning applications. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pages 1–6, 2018.

[22] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural networks : the official journal of the International Neural Network Society*, 61:85–117, 2015.

[23] Xing Hao, Guigang Zhang, and Shang Ma. Deep learning. *Int. J. Semantic Comput.*, 10:417–, 2016.

[24] Giang Nguyen, Stefan Dlugolinsky, Martin Bobak, Viet Tran, Álvaro López García, Ignacio Heredia, Peter Malik, and Ladislav Hluchy. Machine learning and deep learning frameworks and libraries for large-scale data mining: A survey, 2019.

[25] Nitin Kumar Chauhan and Krishna Singh. A review on conventional machine learning vs deep learning. In *2018 International Conference on Computing, Power and Communication Technologies (GUCON)*, pages 347–352, 2018.

[26] Sampo Kuutti, R. Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 22:712–733, 2021.

[27] Sampo Kuutti, R. Bowden, Yaochu Jin, Phil Barber, and Saber Fallah. A survey of deep learning applications to autonomous vehicle control. *IEEE Transactions on Intelligent Transportation Systems*, 22:712–733, 2021.

[28] Pramila P. Shinde and Seema Shah. A review of machine learning and deep learning applications. In *2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)*, pages 1–6, 2018.

[29] asimovinstitute. The neural network zoo.

[30] Jiahao Li, Bin Li, Jizheng Xu, Ruiqin Xiong, and Wen Gao. Fully connected network-based intra prediction for image coding. *IEEE Transactions on Image Processing*, 27:3236–3247, 2018.

[31] Farooque Azam, Sunil Kumar, K.P. Yadav, Neeraj Priyadarshi, and Sanjeevikumar Padmanaban. An outline of the security challenges in vanet. In *2020 IEEE 7th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, pages 1–6, 2020.

[32] Mustafa Maad Hamdi, Lukman Audah, Sami Abduljabbar Rashid, Alaa Hamid Mohammed, Sameer Alani, and Ahmed Shamil Mustafa. A review of applications, characteristics and challenges in vehicular ad hoc networks (vanets). In *2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA)*, pages 1–7, 2020.

[33] Prabhat Sharma and Swapnil Jain. Review of vanet challenges and protocol for architecture design and intelligent traffic system. In *2nd International Conference on Data, Engineering and Applications (IDEA)*, pages 1–4, 2020.

[34] Muhammad Ramzan, Hikmat Ullah Khan, Shahid Mahmood Awan, Amina Ismail, Mahwish Ilyas, and Ahsan Mahmood. A survey on state-of-the-art drowsiness detection techniques. *IEEE Access*, 7:61904–61919, 2019.

[35] Driver drowsiness detection dataset. http://cv.cs.nthu.edu.tw/php/callforpaper/datasets/DDD/, 2022.

[36] Qaisar Abbas. Hybridfatigue: A real-time driver drowsiness detection using hybrid features and transfer learning. *International Journal of Advanced Computer Science and Applications*, 11, 01 2020.

[37] Samaneh Jamshidi, Reza Azmi, Mehran Sharghi, and Mohsen Soryani. Hierarchical deep neural networks to detect driver drowsiness. *Multimedia Tools and Applications*, 80(10):16045–16058, 2021.

[38] Genaro Rebolledo-Mendez, Angélica Reyes, Sebastian Paszkowicz, Mari Carmen Domingo, and Lee Skrypchuk. Developing a body sensor network to detect emotions during driving. *IEEE Transactions on Intelligent Transportation Systems*, 15(4):1850–1854, 2014.

[39] Seong Kyung Kwon, Ji Hwan Seo, Jun Young Yun, and Kyoung-Dae Kim. Driving behavior classification and sharing system using cnn-lstm approaches and v2x communication. *Applied Sciences*, 11(21), 2021.

[40] Mohammad Shahverdy, Mahmood Fathy, Reza Berangi, and Mohammad Sabokrou. Driver behavior detection and classification using deep convolutional neural networks. *Expert Systems with Applications*, 149:113240, 2020.

[41] Cuda [online]. https://developer.nvidia.com/cuda-toolkit, 2022.

[42] Cudnn [online]. https://developer.nvidia.com/cudnn, 2022.

[43] Anaconda [online]. https://www.anaconda.com, 2022.

[44] Keras[online]. https://keras.io, 2022.

[45] Create production-grade machine learning models with tensorflow[online]. https://www.tensorflow.org, 2022.

[46] Chunmian Lin, Lin Li, Wenting Luo, Kelvin C. P. Wang, and Jiangang Guo. Transfer learning based traffic sign recognition using inception-v3 model. *Periodica Polytechnica Transportation Engineering*, 47(3):242–250, 2019.

[47] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network, 2013. cite arxiv:1312.4400Comment: 10 pages, 4 figures, for iclr2014.

[48] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[49] kaggle [online]. https://www.kaggle.com/datasets/, 2022.

[50] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *CoRR*, abs/1312.4400, 2014.