Dual Mode Crash Avoidance System for Manual and Autonomous Driving using Deep Learning

Ahmed Anwar[1], Ahmed Saad[1], Hind Ashraf[1], Marehan Refaat[1], Nagham Wael[1], Omar Khaled[1], Omar Mohamed[1], Omar Shaaban[1], Mohsen Rashwan[1] and Hassan Mostafa[1,2]

[1]Electronics and Electrical Communications Engineering Department, Faculty of Engineering, Cairo University, Egypt
[2]Nanotechnology Department, Zewail City of Science and Technology, Egypt

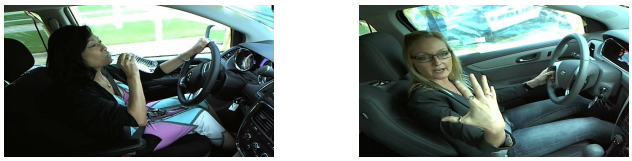
***Abstract*— In this paper, a solution for car accidents is proposed both in autonomous and manual driving vehicles. Crash avoidance in autonomous vehicles is achieved through some techniques like obstacle avoidance and lane overtaking, where the vehicle detects that there is an obstacle at a certain distance and begins to see whether there is a possibility to overtake it or not. After applying many architectures, the best results on the simulator are achieved by a CNN architecture with loss of 0.0738, model size of 127.92 MB and inference time of 1 ms. While in manual driving vehicles, a driver assistance and a real-time distraction detection system are built to decrease the probability of being distracted while driving, which in turn decreases the probability of making an accident. The distraction detection system is based on ResNet50 architecture, which achieves classification accuracy and loss of 94.3% and 0.25, respectively. A quantized version of the trained ResNet50 architecture is introduced in order to facilitate the deployment and operation on the hardware. The quantized version achieves classification accuracy and loss of 92.2% and 0.26, respectively, with 3.75x reduction in memory usage and 2x reduction in inference time, resulting in a model size of 24MB and inference time of 75ms on a CPU-based system. A speech recognition system, as a driver assistance system, is built to provide the driver some services that make him feel the luxury. Many classification networks (CNNs and RNN) are tried out by using some techniques to increase the data size for generalization. RNN is the best choice to represent the speech recognition module when implementing it, as it has less inference time of 0.69 ms, but it has 4.4M parameters. By using ensemble learning, a speech recognition system within a specific domain of commands is made with classification accuracies of 97.55%, 97.73%, and 90.238%, while the categorical cross-entropy losses are 0.1457, and 0.10314 on validation, test, and submission datasets respectively with a rate of 115 audios/sec on Tesla T4.**

I. INTRODUCTION

Advancements in science and technology have been re-shaping our lives in recent decades, where Artificial Intelligence and Machine Learning are the core for building any smart system today. A lot of car accidents occur as the vehicles can't avoid crashing especially the self-driving ones, as they may be at high velocities, like what happened at March 18, 2018, a woman in Arizona have been died due to an accident with an autonomous car operated by Uber. If an obstacle suddenly appeared, it may not have time to decrease its velocity which would lead to crashing. Also, many car accidents occur due to driver distraction, as shown in the figure, distraction has many shapes, whether this distraction is due to objects on the road, or due to the interaction of the driver with any other subject, like talking on the phone, texting, drinking, or talking with a passenger. Distraction may also occur due to the interaction with the vehicle itself, like operating the radio, turning the air conditioning on, or looking behind to grab something. These problems lead to the loss of many lives, and our project is introducing a solution to avoid it.Our problem is divided into three modules, as each module can deal with some these shapes of distraction, first module deals with the surrounding environment of the semi-self driving car,as shown in figure 1 and second and third modules deal with any distraction could happen inside the car from the driver, as shown in figure 2.

(a) Crash due to a car (b) Human in front of the car

Fig. 1: Some shapes of distraction due to the surrounding environment



(a) Drinking (b) Talking to a passenger

Fig. 2: Some shapes of distraction due to driver from State Farm dataset

A. Lane Overtaking

The lane change maneuver is one of the most thoroughly investigated automatic driving operations for autonomous vehicles after trajectory tracking. This maneuver is used as a primitive for performing more complex operations like changing lanes on a highway, leaving the road, or overtaking another vehicle. [1]

A total of 13,939 fatal crashes occurred during overtaking maneuvers in the United States from 1994 to 2005, which lead to the death of 24,565 people. Most of these accidents were caused because of failing to leave enough distance, overtaking when there was poor visibility, or by not giving way to an overtaking vehicle. These accidents forced some governments to ban overtaking like that of Netherlands. So, the objective of the autonomous vehicles is to have this feature (overtaking), but by taking the precautions and be more safe. [1]

Deep Neural Network is used for detecting the existence of an obstacle and the precautions that it will take to avoid crashing. According to the distance available between the vehicle and the obstacle and the surrounding circumstances (the availability of the neighboring lanes), the decision will be taken either by decreasing the speed of the car or by stopping or by making lane overtaking. Below, figure 3, represents the technique of the lane overtaking.

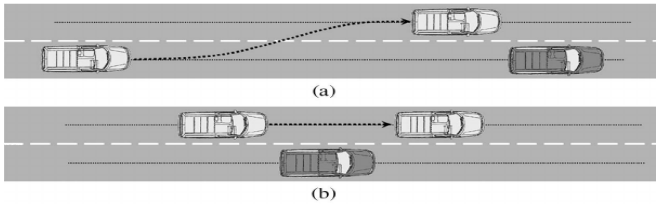


Fig. 3: Lane overtaking technique

B. Distraction Detection

Road accidents occurring due to driver distraction increase as technology and means of distraction like In-Vehicle Information Systems (IVIS) develop. The World Health Organization (WHO) stated in the 2017 Global Status Report that 1.25 million yearly deaths worldwide take place due to road traffic accidents [2]. As the Center for Disease Control and Prevention (CDC) motor vehicle safety division reported, one in five car accidents are caused due to a distracted driver. According to the National Highway Traffic Safety Administration (NHTSA), 9 people are killed and 1000 are injured every day in crashes related to distracted driving in the United States [3]. Additionally, NHTSA reported that 9% of fatal crashes in 2016 were reported as distraction-affected crashes, and that 3450 people were killed in motor vehicle crashes involving distracted drivers [3]. All the mentioned reports about distracted driving threat raise the concern towards building distraction detection systems capable of identifying the posture and state of the driver, whether the driver is in a safe-driving state or distracted, and what kind of activity causes the distraction.

The NHTSA defines distracted driving as any activity that diverts attention from driving, including talking or texting on the phone, eating or drinking, talking to a passenger and operating the radio or the navigation systems [4]. The CDC categorizes distracted driving into visual distraction (i.e taking one's eyes off the road), manual distraction (i.e taking one's hands off the driving wheel) and cognitive distraction (i.e taking one's mind off driving) [5].

A real-time distraction detection system is built to avoid the aforementioned problems of distraction. The system is based on Deep Learning, in which a CNN has been trained on different images of distracted drivers in order to classify the state of the driver whether it is a safe driving or a distracted state. The different kinds of distraction and the datasets used in this work are discussed in section III. The proposed system is able to operate in a real-time environment, with a satisfactory degree of generalization. Additionally, model optimization and hardware acceleration techniques such as pruning and quantization have been applied after training, giving the capability of deploying the model to an embedded system with less memory usage and faster inference time. The proposed compressed model in this work is based on quantization aware training technique.

C. Speech Recognition

Speech recognition is a process of converting spoken words into text. It is also known as Automatic Speech Recognition (ASR). The technology gained acceptance and shape in the early 1970s due to the research funded by the Advanced Research Project Agency in the U.S Department of Defense. It has been widely in use since the 1970s in various domains such as the automotive industry, health care, military, IT support centers, telephone directory assistance, embedded applications, automatic voice translation into foreign languages, etc.

In-car speech recognition is our point of interest, which enables the driver to interact with the car. In-car speech recognition systems have become an almost standard feature in all many new vehicles on the market today. The days of getting in our cars and driving from point A to point B without any distractions are over. Even though safe driving behaviors in many places, the law obliges to ignore the constant phone calls, emails, and text messages while behind the wheel, that kind of disconnectedness isn't the reality. A lot of voice technology was driven by the need to keep the public safe while still acknowledging the device-dependent epidemic. In-car speech recognition systems aim to remove the distraction of looking down at your mobile phone while you drive. Companies like Apple, Google, and Nuance are reshaping the way voice-activation is used in vehicles. They are the most popular companies providing this service that enables drivers to find directions, send emails, make phone calls, and play music, all by using the sound of their voice.

A command speech recognition system is built to assist the driver to interact with the surrounding environment and remove the distraction that may cause car crashes. By using the dataset provided from the Tensorflow team on Kaggle [6] as a challenge, our models are trained to recognize them well.

II. LITERATURE REVIEW

Accidents have spread significantly, either because of the dispersal of drivers doing something, or external factors resulting from the presence of cars or the appearance of people. As for the car while driving, Hence, some papers appeared attempting to solve these problems and reach solutions for how to deal when these problems are about to happen,as we found in Forward Collision Warning (FCW), the vehicle has the ability to warn the driver that there is an object in front of the vehicle. The driver has the responsibility of taking reasonable action. However, in Automatic Emergency Braking (AEB) the vehicle starts taking action through braking in case the vehicle comes close to an object in front of it. These actions are taken by the ego-vehicle based on integrating advanced sensors like Laser, Camera, and Radar. However, these vehicle's actions lead to the occurrence of many crashes, in addition to being a source of congestion because of its poorness [7].

A. Lane Overtaking

According to this paper [8],we found that they tried to solve the problems that happened due to the distraction outside the moving vehicle ,as the Crash avoidance functionality is considered as one of the most important features in self-driving Cars. Recently, it is partially integrated into the self-driving car system.The path planning problem is one of the most important problems which autonomous driving cars face, as it is a safety-critical task, and needs full knowledge of the surrounding environment. During the last ten years, path planning problem was tackled using two approaches: multi-stage pipeline, and single-stage approach.The multi-stage approach decomposes the problem into the following blocks: a) perception which is responsible for perceiving the surrounding environment, b) trajectory prediction for the surrounding objects, c) trajectory planning which compute the trajectory depends on the perception and prediction for the environment, and d) control block which is responsible for taking the good actions depending on the whole information which are gathered by the previous block.

Their data collection pipeline consists of three main blocks, The first block is the waypoints generator which is considered as the most important block in the data collection phase.Waypoints are generated based on the well-defined map using CARLA simulator[9], then a postprocessing phase is applied on these waypoints to act as the traditional ground truth for path planning functionality. The second block is a PID controller which is composed of proportional, integral and derivative components which are tuned to achieve a smooth performance for the vehicle motion following the previously generated waypoints by controlling three vehicle actions: throttle, steering angle and brake. The third block is the noise generator which is inspired by CARLA simulator[9],they collected 1500 episodes which contain 440k data frames, each data frame consists of 4 images from the forward camera, the right camera, the left camera and the backward camera with its corresponding measurements which consist of throttle, brake, steering percentage and forward speed.The data is collected covering various and different scenarios, our vehicle is moving and the other vehicles are static.

The authors used Convolution neural network as a network architecture, and they made a concatenation between the images ,as a four separated images representation depends on the camera cocoon vehicle's installation, it simply takes the four separated images which are front, right, left, and rear views as input to a DNN.
They used Bird's eye view projection,as from the four cameras, a bird's eye view image is generated ,and their results were satisfied and reached to the least loss made their model work efficiently.

B. Distraction Detection

Driver distraction is a serious problem that leads to the loss many lives. This section presents the related work in the field of distraction detection using Machine Learning and Deep Learning approaches, where various distraction detection systems have been proposed in order to limit the number of accidents that occur due to driver distraction.

[10] introduced a distracted driving dataset with with a side view of the driver. The distraction classes included in the dataset are: talking on a cellular phone, eating a cake, grasping the steering wheel and operating the shift lever. The proposed distraction detection system is based on a contourlet transform for feature extraction and random forest as a classifier with a classification accuracy of 90.5%. [11] demonstrates that a multiwavelet transform improves the accuracy of the multi-layer perceptron classifier to 90.61%, which was previously reported to be 37.06% in [10]. [12] shows that using a support vector machine (SVM) with an intersection kernel, followed by a radial basis function (RBF) kernel achieves classification accuracies of 92.81% and 94.25%, respectively, in comparison with [10] and [11]. [13] proposes a convolutional neural network (CNN) solution with a classification accuracy of 99.78% on the Southeast University Distracted Driver Dataset [10]. The introduced distraction detection system uses pre-trained sparse filters as the first convolution layer parameters, followed by fine tuning the network on the actual dataset. [14] proposes a deep learning-based solution that consists of a genetically weighted ensemble of convolutional neural networks. The proposed system achieves a classification accuracy of 90% using 5 AlexNet and 5 InceptionV3 networks. Additionally, a thinned version of the system that operates in a real-time environment on a CPU-based system is proposed using two NasNetMobile networks with a classification accuracy of 84.64%. [15] collected a driver distraction detection dataset using a developed assisted-driving test bed. The proposed distraction detection system is based on a GoogleNet model which achieves an accuracy of 89% and operates at a frequency of 11 Hz on a Jetson TX1 embedded computer board.

C. Speech Recognition

This section presents the related work in the field of speech recognition systems within a specific domain of tasks on this dataset [6] using Machine Learning and Deep Learning approaches. [16] proposes a RNN model with a classification accuracy on test set (about 7K samples) 96.5%, this model is relativly small compared to CNN models.[6] The first place at this competition on Kaggle achieved 91% on the submission set (about 150K samples).

III. METHODOLOGY AND SYSTEM SETUP

A. Lane Overtaking

1) How data is generated:

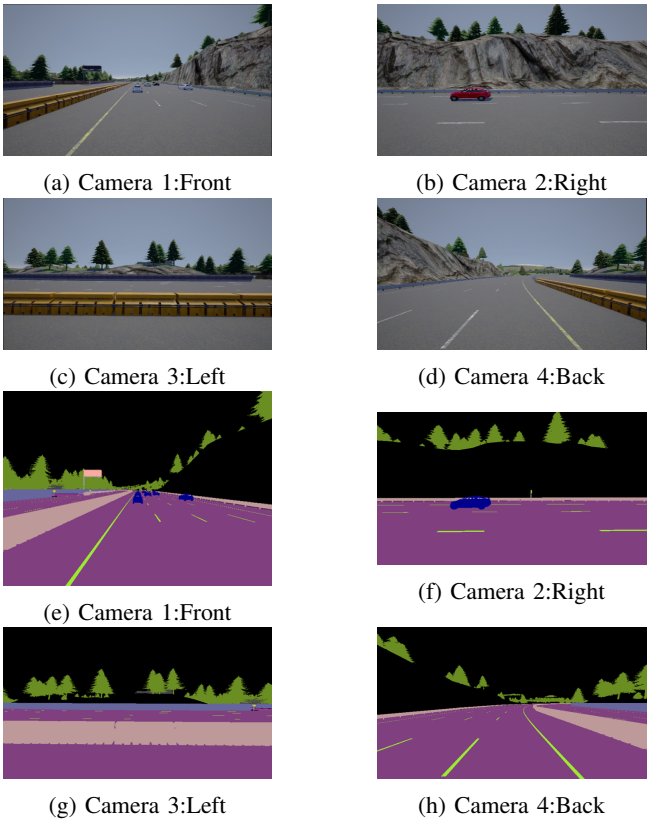
Collecting a dataset using a fleet of real vehicles on real roads can be time consuming, expensive, and very risky on other drivers and pedestrians, on the other hand, a dataset collected completely from a simulated environment is almost cost-free, risk-free, and completely customizable to include all different scenarios. Various simulators are available, such as Carla Simulator, AirSim, DeepDrive, Gazebo Simulator, or even GTA V. Here, this is evaluated in realistic simulations of urban driving on **Carla Simulator 0.9.6 version**. There are a lot of features for this version, such as: lane change extension, different towns with many lanes, ...etc.

The vehicle is moving using proportional-integral differential control (PID) and waypoints (it is given a waypoint and a speed which are translated into steer, throttle and brake). At the begining, the distances between the vehicle and the other vehicles that are in the same lane and in front are calculated. At a specific limit (6m), it begins to see whether there is a possibility for lane change or not (according to the street restrictions). If there is a possibility, it would be right or left or both. If it is less than 5m, it will stop. If it is both, the default will be left, but if it is full, the right will be checked. First, the distances between the waypoint of the same location of the vehicle in the available lane and the other vehicles that are in the same lane and in front or behind are calculated, the minimum distance and its location are gotten. Then, if the lane is empty, it will turn to this lane. If not, as long as the difference between the vehicle and the nearest vehicle that is behind is < 3m, it will stop otherwise the minimum distance will be calculated, if 0 <= the minimum distance < 9m, it will stop otherwise it will turn to this lane.

2) Dataset:

It is a group of images with 27 labels that have been collected under a some specific assumptions,as the data had been collected from highway using many types of cars like (chevorlet,citroen,mini.cooper,dodge,nissan,volks and mercedes),town is divided into sections according to x and y values, as the town is divided into 8 sections and the other cars are static unlike the car that had been used in collecting data.

Dataset has been collected using a 4 cameras to get the RGB images and 4 sensors to get the semantic segmentation images and the FPS is 20 as shown in the figure 4



(a) Camera 1:Front (b) Camera 2:Right

(c) Camera 3:Left (d) Camera 4:Back

(e) Camera 1:Front (f) Camera 2:Right

(g) Camera 3:Left (h) Camera 4:Back

Fig. 4: The RGB and Semantic images getting from the four cameras

Dataset has been divided into training data [14,313 for RGB and 14,313 for Semantic] ,validation data [2,362 for RGB and 2,362 for Semantic] and test data [2,031 for RGB and 2,031 for Semantic]. During taking the images that express the course of the car, its actions and the surrounding environment,labels have been collected for the four images were taken by the four cameras in the same instant which include the location of the car in that instant, its throttle, gear, steer, brake, velocity, rotation, lane invasion, sensor of collision, weather of the town and some other labels that describe the car and the surrounding environment.

3) Preprocessing:

RGB and semantic images have been collected from four cameras (forward, backward ,left and right),there is a little overlapping between the four cameras ,as FOV for all cameras is “110 degree”.

- Concatenation: The purpose of concatenation is to concatenate between every out coming four images from the four images for each instant to be one image with all information,as The front image has been attached to the image on the right, with the image on the left,with the image at the backward and this was applied for RGB ones and the semantic ones as shown in figure 5 and figure 6 .

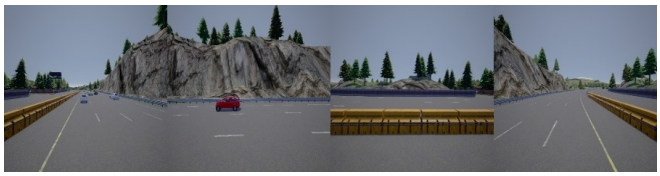


Fig. 5: Concatenation of four RGB images in one image

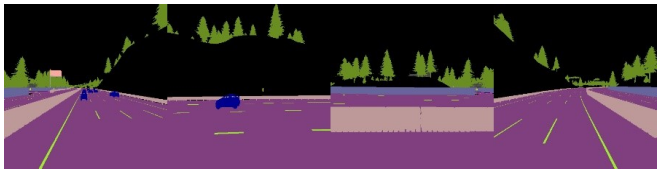


Fig. 6: Concatenation of four Semantic images in one image

- Augmentation: It aims to increase data set,seeing different shapes of images that could be happened at the real life during taking the pictures to collect the data set as a result human errors such as the images may be shifted or as a result of climate changes and improving the performance ,as the machine will deal with a variation of data shapes. In order to augment the training set, the following transformations are applied on every group of images:
  - Rotation and Shear
  - Motion Blur
  - Rain
  - Clouds
  - Additive Gaussian Noise
  - Addition
  - Multiplication

4) Regression Model:

The problem is a regression problem in which that it has an input image and the output is that the prediction of car control signals: throttle which is a continuous value between 0 and 1, steer which is a continuous value between -1 and 1, and brake which is discrete value 0 or 1. We aim to reduce the loss function given by this equation1

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y - \hat{y})^2 \quad (1)$$

According to that there is more than one approach to minimize loss function to be able to make the car take the right action. If there is an obstacle in front of it, it will check both; right lane and left lane in case that our car in the middle lane, if it is in the most right lane, it will check the left only and if it is in the most left lane; it will check the right only. According to prediction, it will decide to go left, right or it will stop. Moreover, if there is no obstacle in front of it, it will go straight. There are mainly two approaches: CNN and LSTM.For CNN,different models have been tried such as:
CNN with RGB based on one image,four images,one image with augmentation and four images with augmentation.
CNN with Semantic Segmentation based on one image,four images,one image with augmentation and four images with augmentation.CNN with Gray Scale based on one image.
For LSTM there is LSTM with Gray Scale based on one image.

- First Approch: CNN
  In this approach, CNN takes the input frame RGB image, Semantic Segmentation image, or Gray Scale image and it gives the output of control signals: throttle, steer, and brake which makes the vehicle enables to move and takes different action. The main layers in the architecture are the convolution layer, dropout layer, batch normalization layer, pooling layer, flatten layer, fully connected layer and Adam optimizer with learning rate = 0.001 as shown in figure 7 and the CNN architecture flowchart shows that the branching of three different activation function : Sigmoid for throttle, Tanh for steer and Sigmoid for brake as shown in figure 9.

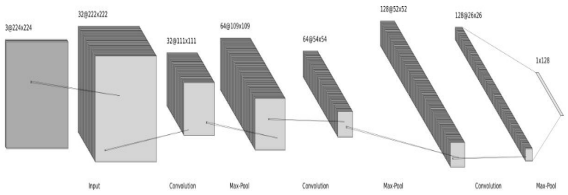


Fig. 7: CNN Architecture

For Gray Scale,the input to CNN is (224,224,4) as the idea is to convert the RGB image to Gray Scale and take the current frame and last three frames in the past so the total number of channels becomes four which help the model to predict the output control signals depend on the current and past frames as shown in figure 8.

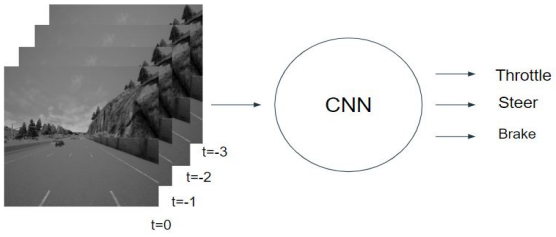


Fig. 8: Temporal information in CNN input (Past)

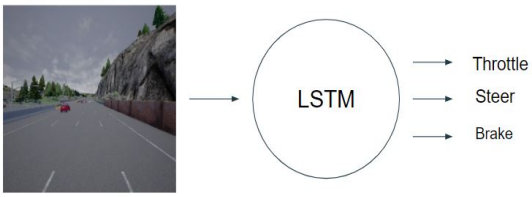


Fig. 10: Sequence prediction using LSTM

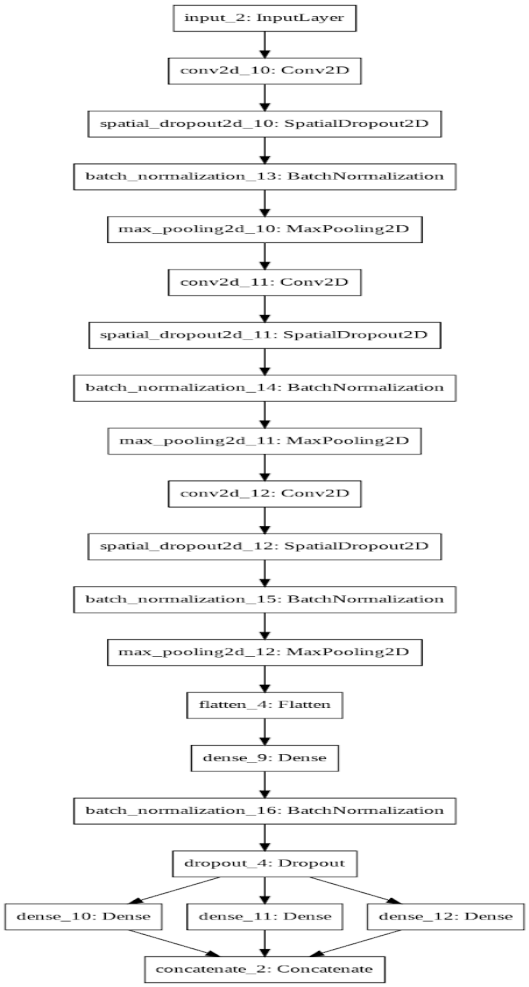


Fig. 9: CNN architecture flowchart

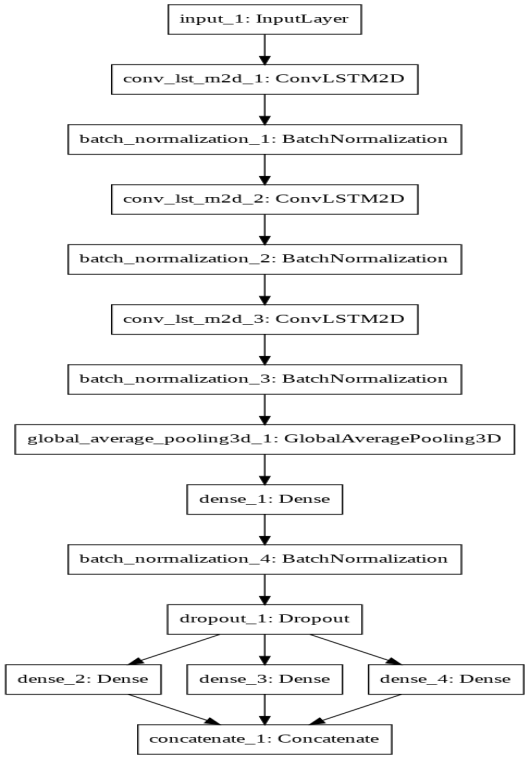


Fig. 11: LSTM architecture flowchart

- Second Approach: LSTM
  The main idea of LSTM that it has feedback connections. Here the input shape is (4,224,224,1), it depends on the current frame and the last three past frames so the total number of frames becomes four ; all of them are Gray Scale and the one refers to the time dimension as shown in figure 10 The architecture is the same in the case of CNN, the difference is there is no Drop-out layer and Pooling layer. We use here the global average pooling layer instead of the flatten layer as shown in figure11.

B. Distraction Detection

1) Dataset:

Two datasets are used in this work. The first dataset is the AUC Distracted Driver Dataset [17] [18], while the second dataset is the State Farm Distracted Driver Detection Dataset [19]. Both datasets contain images for distracted drivers with 10 different classes. Figure 12 and 13 show the 10 different classes from the AUC and State Farm datasets, respectively. Table I and II demonstrate the dataset distribution for both AUC and State Farm datasets, respectively.

Fig. 12: AUC Distracted Driver Dataset. (a) Safe driving (b) Texting - right (c) Talking on the phone - right (d) Texting - left (e) Talking on the phone - left (f) Operating the radio (g) Drinking (h) Reaching behind (i) Hair and makeup (j) Talking to a passenger.

Fig. 13: State Farm Distracted Driver Dataset. (a) Safe driving (b) Texting - right (c) Talking on the phone - right (d) Texting - left (e) Talking on the phone - left (f) Operating the radio (g) Drinking (h) Reaching behind (i) Hair and makeup (j) Talking to a passenger.

TABLE I: AUC dataset distribution

Dataset	Number of samples
Train	10555
Validation	1123
Total	11678

TABLE II: State Farm dataset distribution

Dataset	Number of samples
Train	18732
Validation	3692
Test	79726
Total	102150

2) Preprocessing:

Preprocessing is divided manly into dividing the dataset into training and validation sets based on unique drivers, and augmentation.

The problem of unique drivers is crucial to inspect in order to achieve a degree of satisfactory model generalization. When splitting the State Farm dataset into training and validation sets, the splitting is carried out so that no driver in the training set appears in the validation set. This ensures that the validation results reflect the performance of the model on new unseen drivers. If some drivers in the validation set appear in the training set, the validation set results will not be a proper evaluation metric to assess model generalization, as the training and validation sets will be highly correlated. The AUC dataset is split into training and validation sets based on unique drivers, so this preprocessing step is performed only on State Farm dataset, by selecting some drivers for the validation set and excluding them from the training set.

The purpose of augmentation is to increase the number of samples in the dataset and introduce artificial alterations in the original images that the trained model may encounter in the future after training, which in turn increases the possibility of generalization. In order to augment the training set, the following transformations are applied on every image:

- Rotation and Shear
- Motion Blur
- Cropping
- Additive Gaussian Noise
- Addition
- Multiplication

3) End-to-End Classification System:

Different experiments have been carried out in order to determine the best architecture and approach for classification. The first experiments are performed on the State Farm dataset only, while in the second experiment, the AUC and State Farm datasets have been merged into a larger dataset. Adding the two datasets together shows a better classification performance and generalization, due to the variance in the lighting conditions, recording cameras and drivers. The experiments are performed by utilizing Google Colaboratory resources, which are Nvidia Tesla P100-PCIE-16GB GPU, Intel ® Xeon ® CPU @ 2.20 GHz, and 25GB RAM.

The experiments including only the State Farm dataset are carried out using two approaches. In the first approach, a custom CNN has been built and trained from scratch. Different architectures have been tested for their generalization after training. Figure 14 shows the best performing CNN architecture. The architecture consists of four convolutional blocks, each block consists of three layers: convolutional layer, spatial dropout layer and a max-pooling layer. The number of filters of the convolutional layers increases as the network depth increases, it starts by 64 filters in the first layer and ends by 256 filters in the the third and fourth layers. A flattening layer lies after the last convolutional block, followed by a fully connected layer of 1024 neurons. A last fully connected layer of 10 neurons is added for classification. Dropout layers are introduced between the flattening layer and the first fully connected layer, and between the two fully connected layers to avoid overfitting. The total number of model parameters is 2.1M.

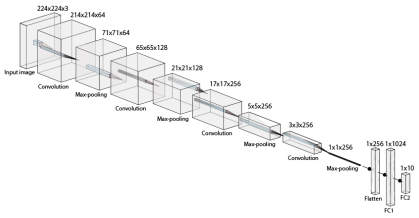


Fig. 14: Custom CNN architecture

The second approach is based on Transfer Learning. Various pre-trained CNNs on the ImageNet dataset have been trained on the State Farm dataset. The CNNs used in the experiment are VGG16, ResNet50, InceptionV3, Xception, MobileNet and DenseNet. The last fully connected layers in the CNNs have been removed and replaced by new fully connected layers with randomly initialized weights. The training process is performed using ImageNet pre-trained weights as an initialization, then the whole network is trained without freezing the weights of any convolutional layer, which shows better results than keeping certain weights unchanged. The results of the two approaches are discussed in section IV. The Transfer Learning based experiment shows that ResNet50 outperforms other pre-trained CNNs used in the experiment. An ensemble between all the trained CNNs is performed by calculating the average of the predicted probabilities by all the models for each class on every sample of the dataset. The advantage of the ensemble approach is that it improves the classification accuracy on the validation and test sets, on the other hand, it is computationally expensive due to the existence of six models contributing to the final prediction, leading to more memory usage and greater inference time.

C. Speech Recognition

1) Dataset:

The dataset is a mandatory block of a successful deep learning system. This section will introduce speech datasets in two different languages. The speech recognition system is trained on the English dataset although the Arabic dataset is mentioned because it has been collected by our team. The Arabic dataset is small which prevents our team to build a reliable Arabic speech recognition system however it is an open-source dataset to increase it and build a reliable system soon.

English dataset:

This dataset is publicly available from Tensorflow team on kaggle [6] as a challenge to build an algorithm that understands simple speech commands in English. The predicted 12 labels are "yes", "no", "up", "down", "left", "right", "on", "off", "stop", "go" and everything else should be considered either "unknown" or "silence". These are the core classes that the challenge compute the accuracy score on them however there are additional 20 classes that are available to use which are "zero", "one", "two", "three", "four", "five", "six", "seven", "eight", "nine", "bed", "bird", "cat", "dog", "happy", "house", "Marvin", "sheila", "tree", and "wow". This dataset does not cover the modeling of the "unknown" or the "noise" classes. In section Preprocessing the creation of these two classes are covered.

Arabic dataset:

Arabic speech dataset is rare to be found as a reliable open source, from this point a website [20] for collecting an open source Arabic speech dataset is deployed to have a reliable dataset structure as an Arabic version from the dataset created by Tensorflow team on kaggle [6]. Building the website will be discussed briefly below. The targeted classes are shown in figure 15. Although the number of users who recorded these words is still small, it is a serious initiative to construct a reliable speech Arabic dataset for the Arabic AI community.

'صفر', 'واحد', 'اثنان', 'ثلاثة', 'اربعة', 'خمسة', 'ستة',
'سبعة', 'ثمانية', 'تسعة', 'تشغيل', 'ايقاف', 'قف', 'انطلق',
'نوافذ', 'فوق', 'تحت', 'يسار', 'يمين', 'نعم', 'لا', 'تكييف', 'راديو',

Fig. 15: Targeted Arabic labels

2) Preprocessing:

Preprocessing is divided into three parts Modeling of the missing classes, feature levels and augmentation.

Dataset distribution and Modeling of the missing classes:

Modeling of classes "unknown" and "silence" was a mandatory task for this competition to achieve a high score. The dataset contains a folder for different kinds of background noises. Background clips had been cut into 1-sec slices and mixed for the "silence" class modeling. The "unknown" class is conducted from two resources. The first one is from the other unused classes that exist in the dataset like "marvin", "bird", etc. The second resource is from random people talking video on youtube[21]. The whole dataset is split into training, validation, testing, and unlabeled submission set. The submission set is the dataset that is submitted to kaggle to calculate the model score on it and it is about 152000 samples.

The training, validation, and testing datasets contain unique speakers that the speaker who contributes to the training set doesn't contribute to the testing set nor the validation set and vice versa. This guarantees well-generalized models that don't bias to the speaker's identity. The dataset distribution is shown in table III.

TABLE III: Dataset distribution

Class	Train	Validation	Test
Down	1842	264	253
Go	1861	260	251
Left	1839	247	267
No	1853	270	252
Off	1839	256	262
On	1864	257	246
Right	1852	256	259
Silence	1600	199	201
Stop	1885	246	249
Unknown	34154	4421	4470
UP	1843	260	272
Yes	1860	261	256
Total	54292	7197	7238

Features level:

The large amount of data does not guarantee getting the best results for training the model. It is mandatory to ensure that all of our data are well organized and unique and the duration of each sound segment is equal and has same rate to ensure access to the same features when processing on sound clips. MFCC is considered one of the most important features and have a great impact on models' performance. The idea of MFCC is to convert audio in time domain shown in figure 16 into frequency domain shown in figure 17 then transform it to MFCC. Our ear has cochlea which basically has more filters at low frequency and very few filters at higher frequency. This can be mimicked using Mel filters. So, the idea of MFCC is to convert time domain signals into frequency domain signal by mimicking cochlea function using Mel filters shown in figure 18, this is to extract the best features found in the sound by using suitable number of filters.

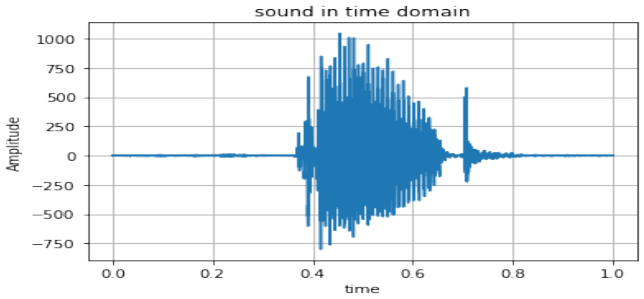


Fig. 16: word (bed) in time domain

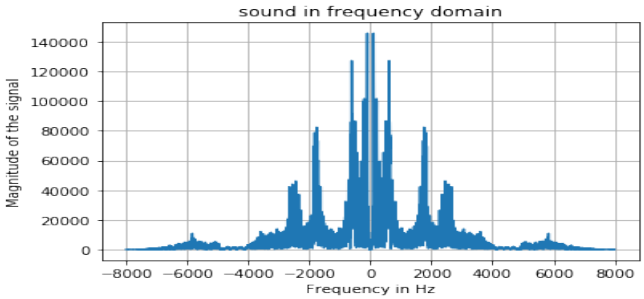


Fig. 17: word (bed) in frequency domain

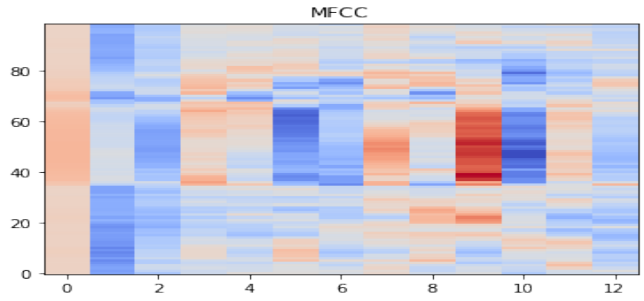


Fig. 18: word (bed) in MFCC

Augmentation:

Data augmentation is a technique to artificially get more training data from the existing training data. It is a type of data pre-processing that makes different transformations to the training data, and gives the training data the property of diversity. This technique makes the classifier learns features that are irrelevant to the time, frequency or power (signal strength). Many functions are used to increase data which are:

1) Changing speed
2) Volume control
3) Mask
4) Time shifting
5) Cropping audio
6) Adding random noise
7) Fourier transform
8) Pitch change

Generative Adversarial Networks (GANs):

Generative Adversarial Networks (GANs) have had a huge success since they were introduced in 2014 by Ian J. Goodfellow and co-authors in this paper [22], they've become one of the hottest sub fields in deep learning, going from generating fuzzy images of digits to photo realistic images of faces. GANs learn a probability distribution of a dataset by pitting two neural networks against each other. The generator is able to take random noise (latent space vector of a certain dimension) and map it to image which will be the input to the discriminator. The competition between these two models is what improves their knowledge, until the generator succeeds in creating realistic data. As discussed in Feature Level, GANs will be used to generate MFCC waveforms, then they will be converted to time-domian waveforms. WGAN is a neural network that is used to generate new data [23]. This is done by using new concept in calculating loss function, that tells how similarity the generated samples are to the actual data by calculating the distance between the actual data distribution and the generated data distribution. After training the generator and critic, WGAN has been tuned with weights that enables it to generate augmented versions of the actual data. In our case, more than 100K samples in total of classes 'yes, no, up, down, right, left, go, stop, on, off' are generated to begin our third training phase for some models. But before that, GAN labelling of the generated samples is checked by passing them as inputs to the best ensemble version of models to predict generated samples labels.

3) Techniques used to increase performance:

There are some approaches that make a big breakthrough in machine learning and deep learning fields, which are:

- **Semi-supervised learning technique:**
  It is an intermediate between supervised and unsupervised learning. Due to the expensiveness and the difficultly of collecting new data and annotating it, semi-supervised technique is applied to increase the amount of data and train our models well. As discussed in the section of dataset that the competition provides a submission dataset that kaggle evaluates the score on it, but it is unlabelled. After training our models with annotated dataset using supervised learning, the models have been tuned with a set of weights that can predict answers for similar untagged data, and the submission data has been used for this purpose.
- **Ensemble learning:**
  Ensemble learning is a powerful technique used when many models are trained either on same or different features of the dataset. Its effect on increasing performance appears in submission on kaggle.

4) Speech recognition classification networks:

Convolutional neural networks (CNNs): The success of convolutional neural networks made data pre-processing an important step and widened the view to think of speech signals as images. This is a good point that CNN made a great success in many fields, so its implementation on another fields such as Speech recognition may reflect its success and make a breakthrough in that field. Our first attempt to solve the problem is to try CNN models and convert speech dataset into MFCC waveform images. Many different models are trained for solving the same problem. In this section, the used CNN models are covered from their architectures point of view. Our second attempt is to train a sequence model that its input is raw data (time domain waveform) with some extensions added to it. The variation in the architectures of the models used to solve the problem and the features is a good point that will help when ensemble learning is performed. Many CNN models are trained to solve the problem, which are DenseNet-BC (densenet with bottleneck extension and compression ratio) [24], WideResNet which is a neural network that solves the problems faced ResNet [25], ResNext which provides a new factor to update ResNet [26], and DPN-92 [27]. All these neural networks are randomly initialized, this helps in breaking symmetry and every neuron is no longer performing the same computation, and pretrained models such as: ResNet, NasNet, xception, VGG-16, VGG-19 and MobileNet.

Sequence Models: A recurrent neural network (RNN) is a class of artificial neural networks where connections between nodes form a directed graph along a temporal sequence. This allows it to exhibit temporal dynamic behavior. RNN is very useful in speech recognition systems. In some cases it outperforms CNNs with the same number of parameters however, it takes a long time in training. In this section we have implemented the RNN proposed in the paper with some modifications on the input size to be less than the proposed [16]. Two approaches are taken to outperform the results of the paper [16]. The first one is using the semi supervised technique while the second approach is the two stages classifiers which will be explained in this section.

"silence" and "unknown" classes are too large due to their good modeling as explained. This forces us to do down sampling to these classes to get a balanced distribution. Two stages model solves this problem. The first stage is 2 or 3 classes classifier model that is able to take all the data size while the second stage is 11 or 10 classes classifier. This technique allows to use all the available data of the "unknown" and "silence" samples.

Figure 19 shows an example of 2 stages. The first model is a "silence"/"voiced" classifier while the second model classifies what the user have said if the first model passes the voiced word to it.

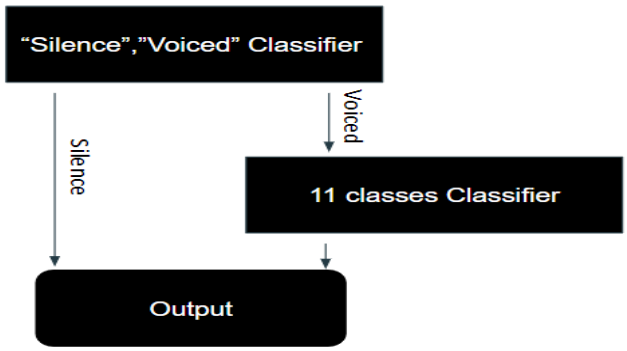


Fig. 19: 2 stages block diagram for silence/voiced classifier

IV. RESULTS AND DISCUSSION

A. Lane Overtaking

In this section, the results of each approach are discussed.There are two approaches CNN and LSTM.The problem here is a regression problem which depends on loss function so the loss in control signals:throttle,steer and brake have been calculated for training, validation and test.After that the average of each loss has been taken to be easily compare among different architectures.The following architectures have been tried in case of one image from the front camera and case of four images from the front, back, right and left side cameras.Moreover, each of this architecture have been tried with and without augmentation.

1) First Approach CNN:

- RGB: The following results have been done for RGB one image without augmentation,one image with augmentation, four images without augmentation and four images with augmentation as shown in Table IV. The total number of parameters is 11.2 M, the model size is 127.92 MB, the inference time is 1 ms on Tesla T4 GPU.

TABLE IV: Training,validation and test loss in case of RGB images

Model	Training Loss	Validation Loss	Test Loss
One image	0.043	0.118	0.072
One image augmented	0.039	0.110	0.064
Four images	0.051	0.099	0.070
Four images augmented	0.034	0.078	0.068

- Semantic Segmentation:The following results have been done for Semantic Segmentation one image without augmentation,one image with augmentation, four images without augmentation and four images with augmentation as shown in Table V. The total number of parameters is 11.2 M, the model size is 127.92 MB, the inference time is 1 ms on Tesla T4 GPU.

TABLE V: Training,validation and test loss in case of Semantic Segmentation images

Model	Training Loss	Validation Loss	Test Loss
One image	0.039	0.114	0.071
One image augmented	0.015	0.101	0.073
Four images	0.035	0.097	0.064
Four images augmented	0.035	0.082	0.059

- Gray Scale:The following results have been done for Gray Scale one image without augmentation as shown in Table VI.The total number of parameters is 11.2 M, the model size is 127.92 MB, the inference time is 0.0002 s on Tesla P100-PCIE-16GB GPU.

TABLE VI: Training,validation and test loss in case of Gray Scale images

Training Loss	Validation Loss	Test Loss
0.036	0.091	0.038

2) Second Approach LSTM:

- Gray Scale:The following results have been done for Gray Scale one image without augmentation as shown in Table VII.The total number of parameters is 296,387, the model size is 3.47 MB, the inference time is 0.028 s on Tesla P100-PCIE-16GB GPU.

TABLE VII: Training,validation and test loss in case of Gray Scale images

Training Loss	Validation Loss	Test Loss
0.228	0.217	0.216

As shown in the previous tables the different results, after that the models have been tried on Carla simulator. For RGB the results have not been efficient enough to enable the vehicle to take the right action.Semantic Segmentation with augmentation based on one image has been given the best results which make the vehicle to behave right while the four images have not given the best results as expected because the way of concatenation has not been the best way to concatenate four images as the size of each image after concatenation becomes 224*896 and it has been resized to become 224*224 to be ready as input for CNN so the image became unclear which makes the model has not enabled to make a good prediction.For CNN with Gray Scale,the results have been better than the Semantic Segmentation but it has not worked well in the Carla simulator.LSTM has been very slow in training and in the simulator.At the end,the best choice is CNN Semantic Segmentation based on one image with augmentation.

B. Distraction Detection

In this section, the results of each classification experiment are discussed. The first experiments are carried out using the State Farm dataset, while the second experiments are performed using both the AUC and State Farm datasets. Model optimization and hardware acceleration techniques are demonstrated. Moreover, saliency maps of the model predictions are illustrated in order to clarify the degree of generalization.

1) State Farm Dataset:

The experiments based on the State Farm dataset are performed using both the custom CNN and pre-trained models discussed in section III. Table VIII shows the training and validation accuracy and loss using the custom CNN, which are the best results achieved by training a CNN from scratch. It has been observed that transfer learning facilitates the training process and improves the ability of the CNNs to generalize on the problem of distraction detection. Table IX demonstrates the validation accuracy and loss for different pre-trained CNNs after training on State Farm dataset, as well as the results of the ensemble between all the CNNs. ResNet50 outperforms other CNNs with a validation accuracy and loss of 88.4% and 0.38, respectively. The ensemble improves the validation accuracy and loss to 90.2% and 0.31, respectively, however, this approach leads to high memory usage, and it is more difficult to be deployed in a real-time environment.

TABLE VIII: Training and validation accuracy and loss on State Farm dataset for the custom CNN

Metric	Train	Validation
Accuracy	86%	78%
Loss	0.4	0.8

TABLE IX: Validation accuracy and loss on State Farm dataset for different CNNs using Transfer Learning

Model	Validation Accuracy	Validation Loss
ResNet50	88.4%	0.38
VGG16	84.8%	0.48
InceptionV3	85%	0.5
Xception	85.8%	0.45
MobileNet	85.8%	0.54
DenseNet	85.8%	0.47
Ensemble of classifiers	90.2%	0.31

2) AUC and State Farm Datasets:

After testing the ResNet50 model trained on the State Farm dataset on the AUC dataset, classification accuracy has been calculated to be 33%. This result illustrates the inability to generalize to new cases, as the AUC dataset includes different recording cameras, lighting conditions and drivers. Hence, the two datasets are combined to form a larger dataset, with more variance in lighting conditions and drivers than any of the two datasets separately. After training the ResNet50 model on the new formed dataset without augmentation, the classification accuracy and loss have been observed to be 90.4% and 0.34 respectively, which is close to the results obtained by the ensemble approach on the State Farm dataset only. After applying augmentation on the new dataset, the classification accuracy and loss have been improved to 92.5% and 0.26, respectively. By fine tuning the learning rate and dropout percentage, the classification accuracy and loss have been further improved to 94.3% and 0.25, respectively. Table X demonstrates the proposed real-time distraction detection system specifications. Figure 20 shows the normalized confusion matrix for the validation set. Most of the samples in the validation set are classified correctly, with some confusion between hair and makeup and drinking, and between talking to a passenger and safe driving classes. The model predicted the hair and makeup class samples accurately with an accuracy of 81%, with 9% misclassification from the drinking class. Regarding the talking to a passenger class, 78% of the samples are classified correctly, with 14% misclassification from the safe driving class.

TABLE X: Proposed System Specifications

Model	ResNet50
Validation Accuracy	94.3%
Validation Loss	0.25%
Test Loss	0.247
Number of Parameters	23.5 M
Model Size	90 MB
Inference Time (GPU)	8 ms
Inference Time (CPU)	160 ms

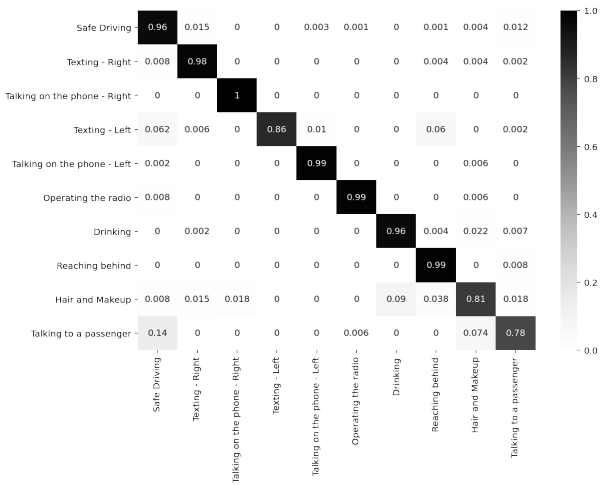


Fig. 20: Normalized confusion matrix for the validation set

3) Model Optimization and Hardware Acceleration:

Model optimization is an essential step in order to compress the trained model for an efficient deployment on hardware. The optimization process results in accelerating the forward propagation when inferring an input, which leads to less inference time, and more classified frames per second (FPS). The proposed optimization technique is quantization aware training.

Quantization is the process of converting the representation of the model parameters from float32 to int8, thus reducing the model size and the inference time, which in turn accelerates the model performance on hardware. Quantization aware training is used in this work, in which the model parameters are quantized based on 2 introduced new parameters according to PyTorch Quantization documentation [28], which are the scale and zero point. Equation 2 demonstrates how each model parameter is quantized. Each parameter is divided by the scale value, then the result is added to the zero point value. After scaling the model parameter and adding the zero point value, the result is rounded to the nearest integer. The two introduced parameters (scale and zero point) are computed using an optimization technique which minimizes the error between the actual label and the prediction of the quantized model, thus the ResNet50 architecture has been trained on a subset of the dataset in order to find the optimum scale and zero point parameters. Table XI shows the effect of quantization on the validation accuracy, validation loss, test loss, model size and inference time on CPU.

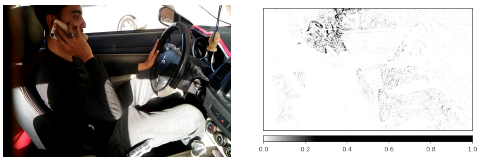
$$Q(x, scale, zero\_point) = round(\frac{x}{scale} + zero\_point) \tag{2}$$

TABLE XI: Effect of quantization on the validation accuracy, validation loss, test loss, model size and inference time

Metric	Before Quantization	After Quantization
Validation Accuracy	94.3%	92.2%
Validation Loss	0.25%	0.262
Test Loss	0.247	0.28
Model Size	90 MB	24 MB
Inference Time (CPU)	160 ms	75 ms

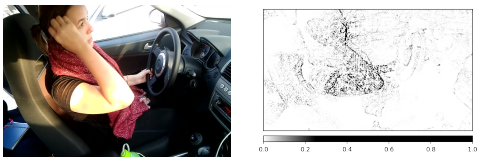
4) Model Interpretability:

Interpretability and visualization play a key role in evaluating the Neural Networks performance. In order to assess model generalization, it is crucial to identify the most relevant parts of the image that the trained model considers when predicting a certain class for an input image. Figure 21 and 22 demonstrate the saliency map based on the ResNet50 model prediction for the talking on the phone - right and hair and makeup classes from the AUC dataset, while figure 23 and 24 demonstrate the saliency map for the drinking and texting - right classes from the State Farm dataset.



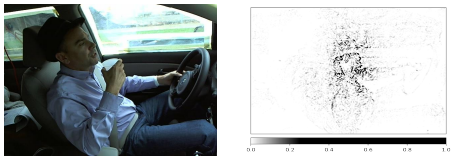
(a) (b)

Fig. 21: Visualization of the important parts of the image that contribute in the prediction. (a) Original talking on the phone - right sample from the AUC dataset (b) Saliency map



(a) (b)

Fig. 22: Visualization of the important parts of the image that contribute in the prediction. (a) Original hair and makeup sample from the AUC dataset (b) Saliency map



(a) (b)

Fig. 23: Visualization of the important parts of the image that contribute in the prediction. (a) Original drinking sample from the State Farm dataset (b) Saliency map

(a) (b)

Fig. 24: Visualization of the important parts of the image that contribute in the prediction. (a) Original texting - right sample from the State Farm dataset (b) Saliency map

C. Speech Recognition

The results of CNN models are shown in table XII while RNN models are shown in table XIII.

TABLE XII: Results of all the proposed CNN models

Model	Val. acc.	Test. acc.	Val. loss	Test loss	Parameters	msec/sample
Xception	0.906	0.908	0.629	0.702	24.046 M	0.41
Mobilenet	0.949	0.954	0.2	0.173	4.034 M	0.138
Resnet50	0.956	0.959	0.167	0.152	26.229 M	0.414
Densenet	0.959	0.964	0.145	0.137	13.565 M	0.69
Nasnet	0.939	0.946	0.258	0.219	87.262 M	1.5
VGG16	0.965	0.963	0.137	0.138	15.258 M	0.27
VGG19	0.957	0.962	0.157	0.139	21.092 M	0.27
Densenet (semi)	0.959	0.962	0.153	0.139	13.776 M	0.69
Mobilenet (semi)	0.872	0.874	0.435	0.430	4.034 M	0.138
Nasnet (semi)	0.943	0.945	0.22	0.201	87.262 M	1.5
Resnet50 (semi)	0.954	0.954	0.173	0.171	26.229 M	0.414
VGG16 (semi)	0.957	0.958	0.17	0.163	15.258 M	0.27
VGG19 (semi)	0.957	0.96	0.166	0.155	20.193 M	0.27
Densenet (random initialized)	0.952	0.958	0.202	0.189	0.769 M	3.3
Densenet (semi) (random initialized)	0.948	0.95	0.193	0.172	0.769 M	3.3
DPN92	0.951	0.954	0.201	0.19	34.240 M	4.6
DPN92 (semi)	0.94	0.94	0.19	0.172	34.240 M	4.6
Resnext	0.935	0.937	0.244	0.229	34.427 M	4.8
Resnext (semi)	0.942	0.947	0.231	0.219	34.427 M	4.8
WideResnet	0.963	0.97	0.18	0.165	75.205 M	5.48
WideResnet (semi)	0.959	0.961	0.191	0.166	75.205 M	5.48

TABLE XIII: Results of all the proposed RNN models

Model	Val. acc.	Test. acc.	Val. loss	Test loss	Parameters	Input Size	msec/sample
RNN	0.927	0.933	0.246	0.249	1.292 M	(80*125)	2.48
RNN (2 stages silence)	0.987	0.989	0.218	0.211	4.96 M	(80*125)	6
RNN (2 stages unknown)	0.928	0.928	0.423	0.5	4.96 M	(80*125)	6
RNN (2 stages 3 classes)	0.926	0.93	0.472	0.461	4.96 M	(80*125)	6.2
RNN	0.932	0.936	0.237	0.222	4.407 M	(32*32)	0.69
RNN (2 stages silence)	0.987	0.989	0.211	0.217	8.814 M	(32*32)	3
RNN (2 stages unknown)	0.956	0.959	0.167	0.152	8.814 M	(32*32)	3
RNN (2 stages 3 classes)	0.93	0.932	0.44	0.487	8.814 M	(32*32)	3.3
RNN (semi)	0.961	0.963	0.141	0.131	4.407 M	(32*32)	0.69
RNN (semi) (2 stages silence)	0.995	0.995	0.106	0.109	8.814 M	(32*32)	3
RNN (semi) (2 stages unknown)	0.964	0.959	0.265	0.307	8.814 M	(32*32)	3
RNN (semi) (2 stages 3 classes)	0.963	0.962	0.284	0.276	8.814 M	(32*32)	3.3

Different combinations of ensemble learning between models have been carried out targeting to get the best accuracy on submission dataset (150k samples). The best combination of models is Densenet (randomly initialized) (semi supervised), Wideresnet, Resnext (semi supervised) ,DPN92 and RNN (32*32) (semi supervised). This combination has achieved accuracy of 0.9755, 0.973 and 90.238 on the validation, test and submission set respectively. This achievement placed our team on the 25^{th} out of 1314 competitors on the competition. The Confusion matrix of the validation set is shown in figure 25 while that of the test set is shown in figure 26.

The Classification report of the validation set is shown in figure 27 while that of the test set is shown in figure 28.

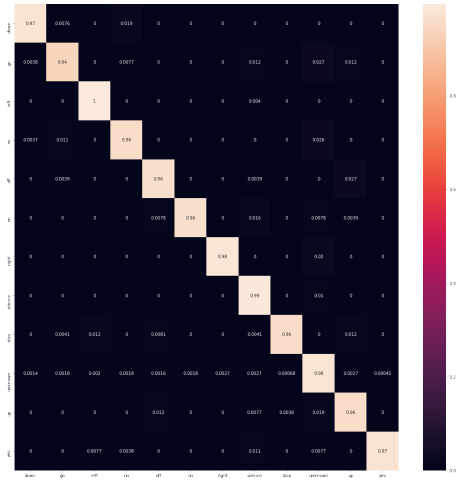


Fig. 25: Confusion matrix of the validation set

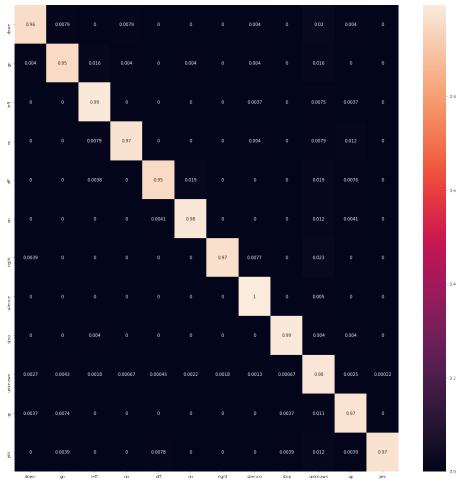


Fig. 26: Confusion matrix of the test set

	precision	recall	f1-score	support
down	0.97	0.97	0.97	264
go	0.94	0.94	0.94	260
left	0.95	1.00	0.97	247
no	0.94	0.96	0.95	270
off	0.95	0.96	0.96	256
on	0.97	0.96	0.97	257
right	0.95	0.98	0.97	256
silence	0.88	0.99	0.93	199
stop	0.98	0.96	0.97	246
unknown	0.99	0.98	0.99	4421
up	0.91	0.96	0.93	260
yes	0.99	0.97	0.98	261
accuracy			0.98	7197
macro avg	0.95	0.97	0.96	7197
weighted avg	0.98	0.98	0.98	7197

Fig. 27: Classification report of the validation set

	precision	recall	f1-score	support
down	0.94	0.96	0.95	253
go	0.91	0.95	0.93	251
left	0.94	0.99	0.96	267
no	0.98	0.97	0.97	252
off	0.98	0.95	0.97	262
on	0.94	0.98	0.96	246
right	0.97	0.97	0.97	259
silence	0.94	1.00	0.97	201
stop	0.98	0.99	0.98	249
unknown	0.99	0.98	0.99	4470
up	0.93	0.97	0.95	272
yes	1.00	0.97	0.98	256
accuracy			0.98	7238
macro avg	0.96	0.97	0.96	7238
weighted avg	0.98	0.98	0.98	7238

Fig. 28: Classification report of the test set

V. CONCLUSION

All in all, three modules have been introduced in order to minimize the car crashes. First module, lane overtaking, which helps in detecting the obstacles and take suitable decisions according to the surrounding conditions whether by stopping or by performing lane change. Second module, distraction detection, which detects whether the driver is internally distracted or not, and classifies the kind of distraction (drinking, talking on the phone, operating the radio, texting,.... etc.). Third module, speech recognition, which assists the driver to interact with the surrounding environment and prevents the distraction that may cause car crashes.

As shown in figure 29, there may be an integration of the three modules into one system, where first it detects whether the driver is distracted or not. If yes, there will be an alarm then it will check the input command, if not it will check the input command. If the input command is to turn on the lane overtaking module, the system will check whether it can be done or not according to the surrounding situations and take the suitable actions. If the input command is not to turn on the lane overtaking module, it will do the event according to the input command then return to see whether the driver is distracted or not and so on.

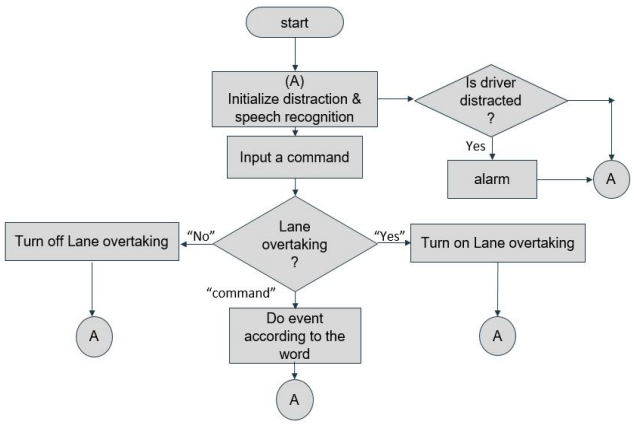


Fig. 29: Block diagram of the proposed system

References

[1] J. E. Naranjo, C. Gonzalez, R. Garcia, and T. De Pedro, "Lane-change fuzzy control in autonomous vehicles for the overtaking maneuver," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 3, pp. 438–450, 2008.

[2] W. H. Organization, "World health statistics 2017: Monitoring health for the sdgs, sustainable development goals," 2017.

[3] National Highway Traffic Safety Administration, *Traffic safety facts research notes 2016: Distracted driving. department of transportation, washington, dc: Nhtsa; 2015*. [Online]. Available: https://crashstats.nhtsa.dot.gov/Api/Public/ViewPublication/812517.

[4] T. M. Pickrell, H. R. Li, and S. KC, *Traffic safety facts, 2016*. [Online]. Available: https://www.nhtsa.gov/risky-driving/distracted-driving.

[5] U. D. o. H. &. H. Services, *Distracted driving, 2016*. [Online]. Available: https://www.cdc.gov/motorvehiclesafety/distracted_driving/.

[6] P. Warden, "Speech commands: A public dataset for single-word speech recognition.," *Dataset available from https://www.kaggle.com/c/tensorflow-speech-recognition-challenge/data*, 2017.

[7] M. K. Pal, N. Debabhuti, P. Sadhukhan, and P. Sharma, "A novel real-time collision avoidance system for on-road vehicles," in *2018 Fourth International Conference on Research in Computational Intelligence and Communication Networks (ICRCICN)*, IEEE, 2018, pp. 141–146.

[8] E. Mohammed, M. Abdou, S. A. Engineer, and O. A. Nasr, "End-to-end deep path planning and automatic emergency braking camera cocoon-based solution," in *Machine Learning for Autonomous Driving, NeurIPS 2019 Workshop*, 2019.

[9] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "Carla: An open urban driving simulator," *arXiv preprint arXiv:1711.03938*, 2017.

[10] C. H. Zhao, B. L. Zhang, J. He, and J. Lian, "Recognition of driving postures by contourlet transform and random forests," *IET Intelligent Transport Systems, 6(2):161-168*, 2011.

[11] C. Zhao, Y. Gao, J. He, and J. Lian, "Recognition of driving postures by multiwavelet transform and multilayer perceptron classifier," *Engineering Applications of Artificial Intelligence, 25(8):1677â"1686*, 2012.

[12] C. Zhao, B. Zhang, J. Lian, J. He, T. Lin, and X. Zhang, "Classification of driving postures by support vector machines," *In Image and Graphics (ICIG), 2011 Sixth International Conference on, pages 926â"930. IEEE*, 2011.

[13] C. Yan, F. Coenen, and B. Zhang, "Driving posture recognition by convolutional neural networks," *IET Computer Vision, 10(2):103â"114*, 2016.

[14] H. Eraqi, Y. Abouelnaga, M. Saad, M. Moustafa, "Driver distraction identification with an ensemble of convolutional neural networks," *Journal of Advanced Transportation, Machine Learning in Transportation*, 2019.

[15] Duy Tran, Ha Manh Do, Weihua Sheng, He Bai, Girish Chowdhary, "Real-time detection of distracted driving based on deep learning," *IET Intelligent Transport Systems*, July 2018.

[16] D. C. de Andrade, S. Leo, M. L. D. S. Viana, and C. Bernkopf, *A neural attention model for speech command recognition*, 2018. eprint: arXiv:1808.08929.

[17] Y. Abouelnaga, H. Eraqi, and M. Moustafa, *"Real-time Distracted Driver Posture Classification"*. Neural Information Processing Systems (NIPS 2018), Workshop on Machine Learning for Intelligent Transportation Systems, Dec. 2018.

[18] H. Eraqi, Y. Abouelnaga, M. Saad, M. Moustafa, *"Driver Distraction Identification with an Ensemble of Convolutional Neural Networks",* Journal of Advanced Transportation, Machine Learning in Transportation (MLT) Issue, 2019.

[19] State Farm, *State Farm Distracted Driver Detection. Can computer vision spot distracted drivers? 2016*. [Online]. Available: https://www.kaggle.com/c/state-farm-distracted-driver-detection/data/.

[20] Saad, Nwishy, and Shabaan, *Ankh: A website for collecting an open source arabic speech dataset*, https://ahmedsaad.pythonanywhere.com/, 2020.

[21] Your Questions Answered channel. (Oct. 2016). 10 hours of people talking, [Online]. Available: https://www.youtube.com/watch?v=PHBJNN-M_Mo.

[22] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, *Generative adversarial networks*. [Online]. Available: https://arxiv.org/pdf/1406.2661.

[23] Ishaan Gulrajani, Faruk Ahmed, Martin Arjovsky, Vincent Dumoulin, Aaron Courville, *Improved training of wasserstein gans*. [Online]. Available: https://arxiv.org/abs/1704.00028.

[24] Gao Huang, Zhuang Liu, Laurens van der Maaten, Kilian Q. Weinberger, *Densely connected convolutional networks*. [Online]. Available: https://arxiv.org/abs/1608.06993.

[25] Sergey Zagoruyko, Nikos Komodakis, *Wide residual networks*. [Online]. Available: https://arxiv.org/abs/1605.07146.

[26] Saining Xie, Ross Girshick, Piotr DollÃ¡r, Zhuowen Tu, Kaiming He, *Aggregated residual transformations for deep neural networks*. [Online]. Available: https://arxiv.org/abs/1611.05431.

[27] Yunpeng Chen, Jianan Li, Huaxin Xiao, Xiaojie Jin, Shuicheng Yan, Jiashi Feng, *Dual path networks*. [Online]. Available: https://arxiv.org/abs/1707.01629.

[28] PyTorch, *Quantization*. [Online]. Available: https://pytorch.org/docs/stable/quantization.html.