DROWSINESS DETECTION IN DRIVERS

Prithish Kumar Rath (2K18/CO/262)
Nilesh Nishant (2K18/CO/235)

Abstract

Driver fatigue has become one of the key reasons for road accidents in modern days. Various surveys prove that if a driver is correctly identified as fatigued, and he, or she is timely alarmed regarding the same, the cases of accidents can be remarkably reduced. Through this project, an in-depth study of various existing techniques of fatigue in a driver is studied, followed by developing a deep learning-based model to accurately identify a driver's state using a novel technique of using spatiotemporal features of the face. It can be determined that the accuracy will remarkably increase when this technique is used.

1 Introduction

Driving involves the performance of a particular sequence of actions with situational awareness, as well as, quick and accurate decision making. Situational awareness is critical in driving, as direct attention is required to process the perceived cues. Monitoring attention status, therefore, is one of the most important parameters for safe driving.

Fatigue slows down human response time, which leads to inability in safe driving. In a survey in Canada, it has been reported that 20% of fatal collisions involve fatigue. In another survey, it is reported that in Pakistan 34% of road accidents were related to fatigue. According to a US survey, 20% of fatal crashes was due to a drowsy driver. In the EU, 20% of commercial transport crashes are reported to be due to fatigue. All the statistics and numbers are alarming and seek serious research community attention to address the issue.

Due to these factors, research in the field of driver's state monitoring has been developing very rapidly, especially for things like driver workload estimation, driver activity identification, secondary task identification and driving style recognition. Many techniques have been used in the past. Some of these methods have been implemented by various multinational companies for driver assistance. Fatigue symptoms include: yawning, slow reaction time, eyelid closure, loose steering grip, etc. Humans may exhibit multiple symptoms and levels of fatigue; therefore, one symptom may not singly and accurately be employed for fatigue detection.

This project presents a brief review of existing techniques used until now in the field of fatigue detection in drivers, and also based on developing a novel architecture using deep learning methods to detect drowsiness, using spatiotemporal features of a person's face. The technique is developed such that it is accurate as well as robust under various real-life scenarios.

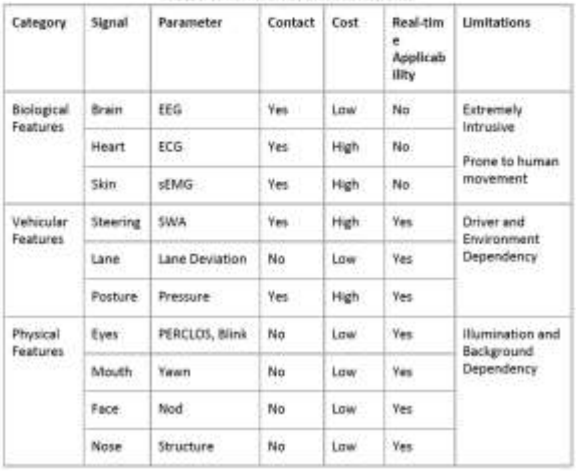
2 Literature Review

Features are used based on various requirements, and possess certain traits, which are, in brief, given in Table. On looking at various parameters associated with features to be used, we see that the biological features are intrusive in nature, due to which the driving experience and other factors of the driver will get hampered. Further, it does not have real time applicability, and the installation costs are high.

Simultaneously, vehicular features are not that accurate, because it will depend a lot on the surrounding, e.g. A turning road will have a lot of movement of the steering wheel, and it can still detect this as a drowsy driver. Simultaneously, installation cost is also high in some of the techniques, and hampers real-time applicability.

Whereas, in physical features, these disadvantages are not present. There is some dependency on the brightness of the surrounding, but still, other advantages make this way better as compared to using other features. So, in this project, physical features are used to determine the state of the driver.

TABLE I: COMPARISON OF VARIOUS FEATURES



Category	Signal	Parameter	Contact	Cost	Real-time Applicability	Limitations
Biological Features	Brain	EEG	Yes	Low	No	Extremely Intrusive Prone to human movement
	Heart	ECG	Yes	High	No	
	Skin	sEMG	Yes	High	No	
Vehicular Features	Steering	SWA	Yes	High	Yes	Driver and Environment Dependency
	Lane	Lane Deviation	No	Low	Yes	
	Posture	Pressure	Yes	High	Yes	
Physical Features	Eyes	PERCLOS, Blink	No	Low	Yes	Illumination and Background Dependency
	Mouth	Yawn	No	Low	Yes	
	Face	Nod	No	Low	Yes	
	Nose	Structure	No	Low	Yes	

3 Data

An academic Driver Drowsiness Detection (DDD) dataset is used, which was first introduced during the 2016 Asian Conference on Computer Vision. Videos were recorded at a 480 X640 resolution with a frame rate of 30 and 15 fps for day and night videos, respectively. For each subject, videos were recorded in a controlled setting in five conditions:

1. without glasses
2. with glasses
3. with sunglasses
4. without glasses at night
5. with glasses at night

Simulated behaviors include yawning, nodding, looking aside, talking, laughing, closing eyes and regular driving, and video segments have been labelled as drowsy or non-drowsy. The dataset consists of training (18 persons), evaluation (4 persons) and testing (14 persons) sets.

What our Data looks like (Drowsy and Non-drowsy case, respectively)

For this study, the training dataset was used for model calibration (a total of 8.5 h of video), the evaluation dataset for validation purposes (1.5 h of video), while the testing dataset was not used. First, night videos were converted from 15 to 30 fps to match the frame rate of the other videos in the dataset. Videos were then resized from 480 640 to 240 320 to reduce pre-processing time during training and disc space.

The video files were split into 100-frame sequences for training and 10-frame sequences for validation. This resulted in 9094 100-frame training records and 17,318 10-frame validation records. Note that a small fraction of 10-frame sequences from the original videos is not used for training (i.e., 10-frame sequences spanning two records), which was a trade-off for faster read performance.

3.1 Pre-processing & Data Augmentation

Since the DDD dataset contains a limited amount of training data, several pre-processing steps were implemented to increase the variety of samples supplied to the neural network during training. These pre-processing steps were tailored to the issue of drowsiness detection and increase robustness of the model when applied in a real-world setting. This not only enhances training accuracy but also lowers overfitting.

Following preprocessing/ data augmentation steps were taken -

- All the videos are converted into image frames at 30 frames per second
- All the frames are labelled according the drowsiness state (i.e. drowsy or not drowsy)
- Brightness of all the frames is normalized (by dividing from the largest value: 255).
- Some of the frames are randomly rotated by 40 degrees.
- Some of the frames are horizontally flipped.
- Images are given zoom, shear, shift (height & width) of magnitude 0.2
- The sample was rescaled to 224 x 224 x 1 . This approach achieves model invariance to translations (horizontal, vertical shifts), zooming, and face shapes.

Division in	No. of frames	No. of categories
Training Set	7,23,248	2
Test Set	1,73,299	2

Data after pre-processing

4 Models

In this project, we have adopted the approach of using the model having the following specifications:

- Method: **Deep Model**
- Type: **Convolutional Neural Network**
- Pretrained on: **ImageNet VGG16**
- Category: **Physical Feature detection**

4.1 Convolutional Neural Networks

A Convolutional Neural Network (ConvNet/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

The architecture of a ConvNet is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field.

A collection of such fields overlaps to cover the entire visual area. We have prepared two models for our purpose:

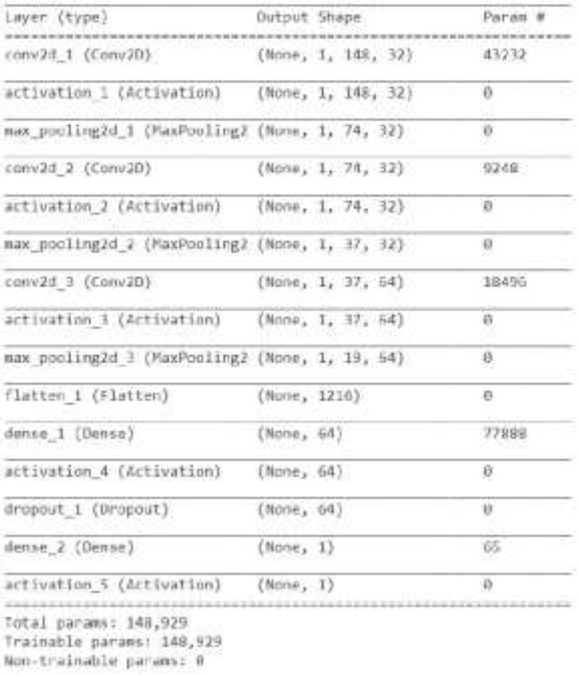
- A Baseline Model Trained from Scratch
- A Fine-tuned VGG16 Model pretrained on ImageNet dataset

4.1.1 Baseline Model

For the purpose of implementing a baseline model so as to obtain a model which has been trained from scratch (i.e., without any pretraining), we have prepared a CNN model with 3 Convolutional Layers and 'Relu' activation function. These layers are followed by Max-pooling layers after each Conv layer. At the top, two dense fully connected layers are attached which finally classifies a frame as drowsy or non-drowsy using a 'Sigmoid' activation function.

The flow of data in this model, the following are some highlighting points:

- Input to the model: (3 x 224 x 224)
- Output of ConvLayers: (1 x 56 x 64)
- Input to Fully Connected Layers: (1 x 1792)
- Final Output: 0 or 1 (Not Drowsy or Drowsy)



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_1 (Conv2D)            (None, 1, 148, 32)        43232
_________________________________________________________________
activation_1 (Activation)    (None, 1, 148, 32)        0
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 1, 74, 32)         0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 1, 74, 32)         9248
_________________________________________________________________
activation_2 (Activation)    (None, 1, 74, 32)         0
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 1, 37, 32)         0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 1, 37, 64)         18496
_________________________________________________________________
activation_3 (Activation)    (None, 1, 37, 64)         0
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 1, 19, 64)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 1216)              0
_________________________________________________________________
dense_1 (Dense)              (None, 64)                77888
_________________________________________________________________
activation_4 (Activation)    (None, 64)                0
_________________________________________________________________
dropout_1 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 65
_________________________________________________________________
activation_5 (Activation)    (None, 1)                 0
=================================================================
Total params: 148,929
Trainable params: 148,929
Non-trainable params: 0
```

Baseline Model Architecture

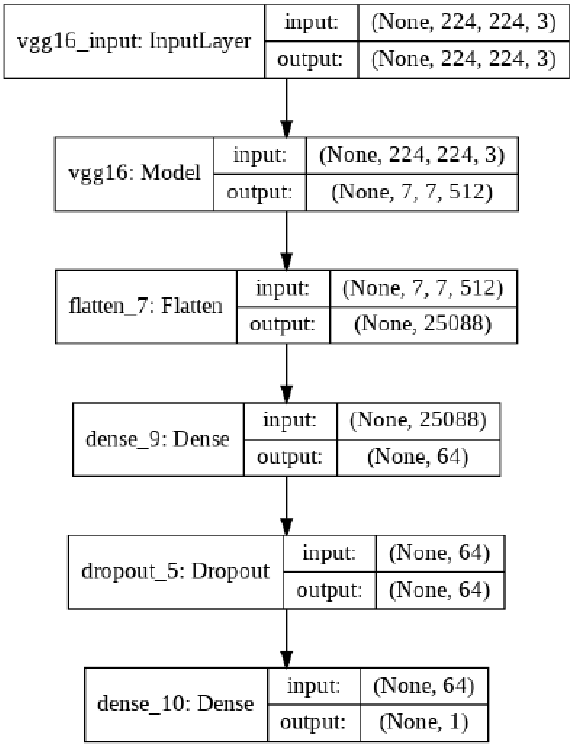
We have trained our Baseline Model in batches of 16 which means the total number of batches that are trained per epoch (iteration) = 45,203 batches

Our major concern while training such networks is Overfitting.

To avoid this, we have predominantly used heavy Data Augmentation. But to really, tackle the issue, we also employ a Regularization technique called L2 Regularization. Which consists in forcing model weights to take smaller values thus mitigating the risk of overfitting up to a great extent.

4.1.2 Final Model (with VGG16 Image Net)

We prepare our final model after fine tuning weights from a VGG16 model pre-trained on ImageNet dataset. The architecture of this model is as follows:



Final Model Structure

4.1.2.1 Pre-training/Transfer Learning

Transfer learning (TL) is a research problem in machine learning (ML) that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem. Usually in pre-trained models, core convolutional layers are trained on the larger dataset and their weights are fixed. After this, the final layers of the network (dense fully connected layers) are fine-tuned using our target dataset (smaller)

Due to scarcity of readily available data for our project, we are also using the application of Transfer Learning by pre-training our model on ImageNet Dataset

ImageNet Dataset : ImageNet is an image database organized according to the WordNet hierarchy (currently only the nouns), in which each node of the hierarchy is depicted by hundreds and thousands of images.

The total number of images in the dataset are over 14 million distributed over more than 20,000 categories (or labels).

4.1.2.2 VGG16 Network

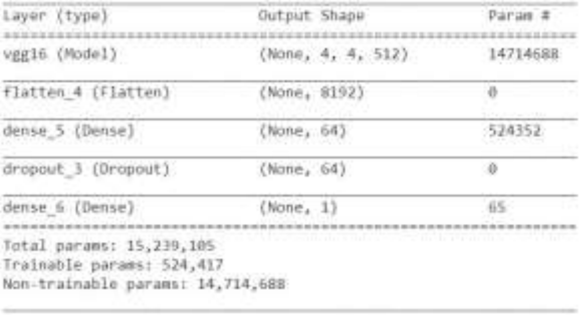
VGG16 is a convolution neural net (CNN) architecture which was used to win ILSVR (ImageNet) competition in 2014. Most unique characteristic about VGG16 is that instead of having a large number of hyper-parameters it focuses on having convolution layers of 3x3 filter with a stride 1 and always used the same padding and maxpool layer of 2x2 filter of stride 2. It follows this arrangement of convolution and max pool layers consistently throughout the whole architecture. In the end it has 2 FC (fully connected layers) followed by a softmax for output. The 16 in VGG16 refers to it has 16 layers that have weights. This network is a pretty large network and it has about 138 million (approx.) parameters.

The final layers of the network are kept dynamic so that the model can take advantage of transfer learning while at the same time get fine-tuned for the actual target (drowsiness detection) by training the fully connected dense layers.

- Input to the model: (3 x 224 x 224)
- Output of ConvLayers: (4 x 4 x 512)
- Input to Fully Connected Layers: (1 x 8192)
- Final Output: 0 or 1 (Not Drowsy or Drowsy)

We have trained our Baseline Model in batches of 16 which means the total number of batches that are trained per epoch (iteration) = 45,203 batches

To avoid the problem of overfitting, we have predominantly used heavy Data Augmentation. But to really, tackle the issue, we also employ a Regularization technique called L2 Regularization. which consists in forcing model weights to take smaller values thus mitigating the risk of overfitting up to a great extent.



```
Layer (type)                 Output Shape              Param #
=================================================================
vgg16 (Model)                (None, 4, 4, 512)         14714688
_________________________________________________________________
flatten_4 (Flatten)          (None, 8192)              0
_________________________________________________________________
dense_5 (Dense)              (None, 64)                524352
_________________________________________________________________
dropout_3 (Dropout)          (None, 64)                0
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 65
=================================================================
Total params: 15,239,105
Trainable params: 524,417
Non-trainable params: 14,714,688
_________________________________________________________________
```

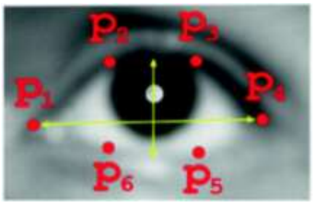
Final Model Architecture

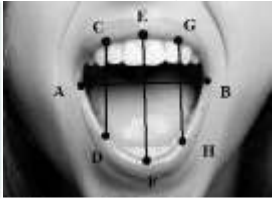
4.2 Real time detection

We first detect a face using **dlib's frontal face detector**. Once the face is detected , we try to detect the facial landmarks in the face using the dlib's landmark predictor. The landmark predictor returns 68 (x, y) coordinates representing different regions of the face, namely - mouth, left eyebrow, right eyebrow, right eye, left eye, nose and jaw. We don't need all the landmarks, here we need to extract only the eye and the mouth region.

We calculate the Eye Aspect Ratio (EAR) it is the ratio of the length of the eyes to the with of the eyes which detects the eyes are closed or open.

Then we calculate the Mouth aspect ratio (MAR) it measures the ratio of the length of the mouth to the width of the mouth. It detects if the person is yawning or not. All these ratio values are taken from the training model.





5 Results

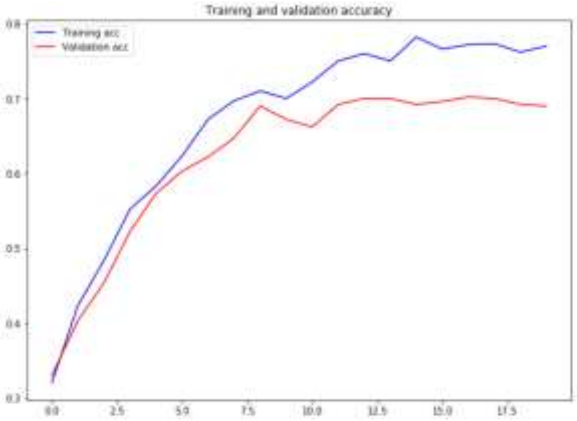
Performance comparison of the various architectures and approaches Vs our Final model

Scenario	**Baseline Model**	**Final Model**
No Glasses	69.33%	73.25%
Glasses	70.85%	75.64%
Sunglasses	71.35%	76.22%
Night No glasses	67.67%	73.50%
Night Glasses	62.42%	65.63%
ALL	**68.56%**	**73.20%**

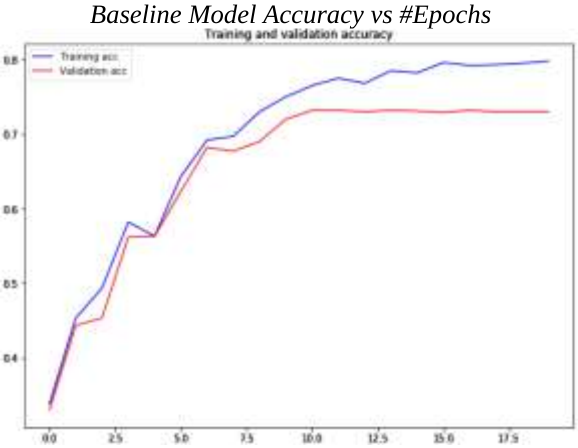
Comparison to other popular works VS ours

Model	**Accuracy**
InceptionV1, Szegedy et al	69.6%
MobileNetV2_1.4, Sandler et al.	72.8%
Ours (Final Model)	**73.20%**

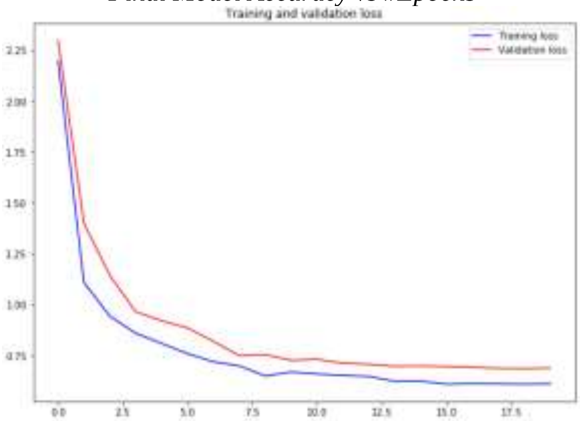
Moreover, we also visualize the training and testing accuracy as well as loss after each iteration. The concatenated graphs of the same have been shown below:



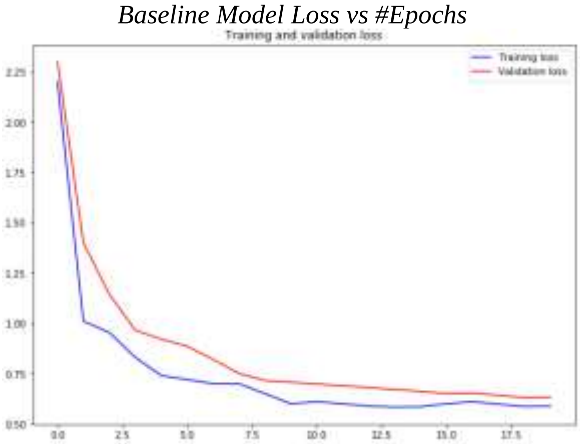
Baseline Model Accuracy vs #Epochs



Final Model Accuracy vs #Epochs

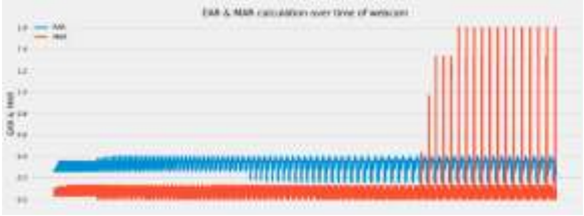


Baseline Model Loss vs #Epochs

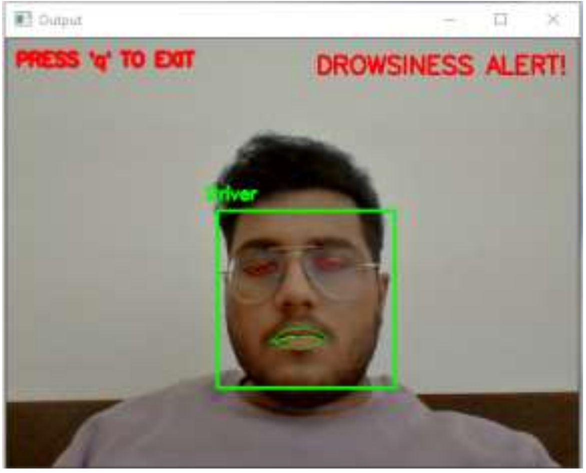


Final Model Loss vs #Epochs

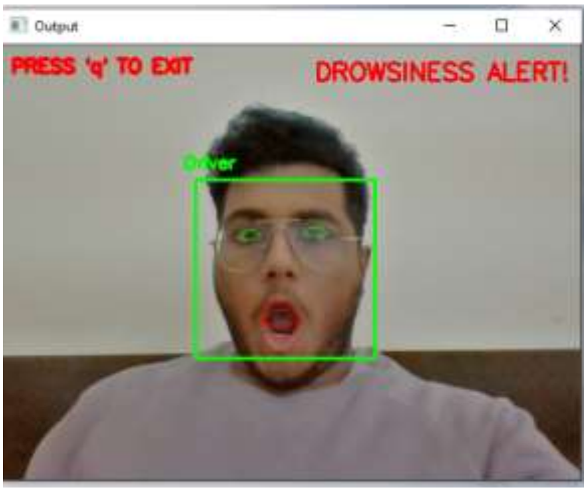
5.1 Real time detection



EAR &MAR calculation over time of webcam



Drowsy eyes ALERT



Yawn ALERT

6 Conclusion

- The process of drivers' drowsiness detection is very important for both individual and community safety
- The growing use of Artificial Intelligence can be immensely useful in predicting fatigue of a driver
- Our model automatically predicts a driver as drowsy or not by recognizing and key features from its face and predicting the output in real-time with an accuracy of 73.2%
- It has various advantages as the installation is simple - of a camera, and it does not hinder the driver's experience and comfort like in other techniques such as biological features or vehicle-based features
- Not prone to external disturbances, considers only the driver's face.

7 References

[1] M. R. Endsley, "Toward a theory of situation awareness in dynamic systems," Hum. Factors, J. Hum. Factors Ergonom. Soc., vol. 37, no. 1, pp. 32–64, 1995.

[2] Sandler M, Howard A, Zhu M, Zhmoginov A, Chen LC (2018) MobileNetV2: inverted residuals and linear bottlenecks. 2018 IEEE conference on computer vision and pattern recognition (CVPR). IEEE, Salt Lake City, UT, pp 4510–4520

[3] Szegedy C, Liu W, Jia Y, Sermanet P, Reed S, Anguelov D, Erhan D, Vanhoucke V, Rabinovich A (2015) Going deeper with convolutions. In: 2015 IEEE conference on computer vision and pattern recognition (CVPR), IEEE, Boston, MA, pp 1–9. https://doi.org/10.1109/CVPR.2015.7298594

[4] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556