UNIVERSITY OF HERTFORDSHIRE

Faculty of Engineering and Information Sciences

MCOM0177 Computer Science MSc Project (Online)

Final Report
December 2021

Detecting Driver Drowsiness using Machine Learning in Python

S A Stokes

Problem Statement

According to the Global Status Report on Road Safety 2018 (World Health Organization, 2018), there are approximately 1.35 million deaths caused by road traffic accidents per year with more than half of these including pedestrians, passengers, cyclists, and motorcyclists. Road Traffic Accidents are the number one cause of death for children and young adults between the ages of 5–29 years of age.

From data received in a Road Safety Report in relation to driver fatigue (European Commission, 2018), global police reports indicate that between 1-4% of crashes are sleep related. This can be attributed to near impossible means of the police objectively diagnosing drowsiness as the cause of an accident after it happens, as opposed to alcohol for example, which can be scientifically tested for. Questionnaires and other studies however show a higher number of fatigue related accident numbers such as a study conducted in 1995 of 4600 drivers, who indicated that fatigue was a result of 9-10% of crashes (Maycock, 1995). In a study conducted in 2006, data was collected involving 15 police-reported crashes, 67 non-police reported crashes, 761 near-crashes and 8,295 incidents. Fatigue was contributed to 12% of all crashes, near-crashes, and incidents in this study (Dingus, 2006). Motorways appear to have the highest rate of fatigue related accidents as shown in a study by the British Medical Journal (Horne, 1995) which established that approximately 20% of all motorway accidents were sleep related and similarly in a study conducted in Germany that attributed fatigue to 24% of all motorway crashes (reference). These statistics are most likely an underestimate. Unless someone witnesses or survives the crash and can testify the drivers' condition, it is difficult to determine if the driver fell asleep. Because of the hazard that drowsiness presents on the road, methods need to be developed for counteracting its affects.

With self-driving cars now a reality, cars are becoming increasingly intelligent with numerous safety features built into vehicles to reduce the risk of crashes and incidents. The most recent and current example, Tesla's Model X and Model S's, are equipped with autopilot driving capabilities with a radar in place and active safety features able to predict and anticipate an accident and react accordingly before it happens (Tesla, 2019). In theory, if a driver were to fall asleep at the wheel or become fatigued whilst in an intelligent car such as a Tesla Model X or S, the chance of a serious accident occurring should be reduced with the added safety features that a regular car is not equipped with. The catch though is the average cost of a Tesla Model X or S is £100,000 (www.autotrader.co.uk, 2021.), making it an expensive solution to an everyday issue. It is not an immediately accessible/economic solution for everyone. This project aims to bridge the gap between safety and expense by offering an affordable solution.

Aim and Objectives of this project

An affordable drowsiness detection system as a method to reduce road traffic accidents has not been widely commercialized. The aim of this project is to produce an affordable and practical method of detecting drowsiness in a driver by analyzing the Eye Aspect Ratio of a driver and sounding an audible alarm to prevent falling asleep whilst in control of a motor vehicle when the

Eye Aspect Ratio decreases. Influential machine learning models and algorithms within the artificial intelligence field were investigated, tested, and analyzed to produce a product for this purpose – delivering an inexpensive and optimal solution to the problem domain.

The objectives of this project can be broken down into these parts:

List the objectives at the end as the report is completed.

Research Question(s)

1. What is the most efficient Machine Learning algorithm and model to detect the Eye Aspect Ratio of a person in real-time on a Raspberry Pi 4 to reduce the risk of falling asleep at the wheel and causing an accident?

Discussion of the methods and methodology used to explore and address the issue, selection of approach, selection of techniques, selection of evaluation approach

Drowsiness, fatigue, or sleepiness can be defined as "the natural need to fall asleep". This procedure is the consequence of the natural biological rhythm of the human body and its sleep-wake cycles. The longer the period of wakefulness, intensified pressure builds for sleep and the more challenging it becomes to withstand it. This becomes a problem where the sleep-wake cycle begins whilst a person is driving a motor vehicle and unable to satisfy the need for sleep. Driver inattention may be the result of a lack of alertness when driving due to driver drowsiness which involves no triggering event but is characterized by a progressive withdrawal of attention from the road and traffic demands. Drowsiness can be attributed to a decrease in driver performance, longer reaction times and an increased risk of crash involvement.

This decrease in driving ability can often affect a person's driving ability long before they notice that they are getting tired. Fatigue related crashes are often more serious than others because driver's reaction times are delayed, or the drivers have failed to make any maneuvers to prevent a crash. The number of hours spent driving has a significant correlation to the number of fatigue-related accidents as shown in Figure 1.

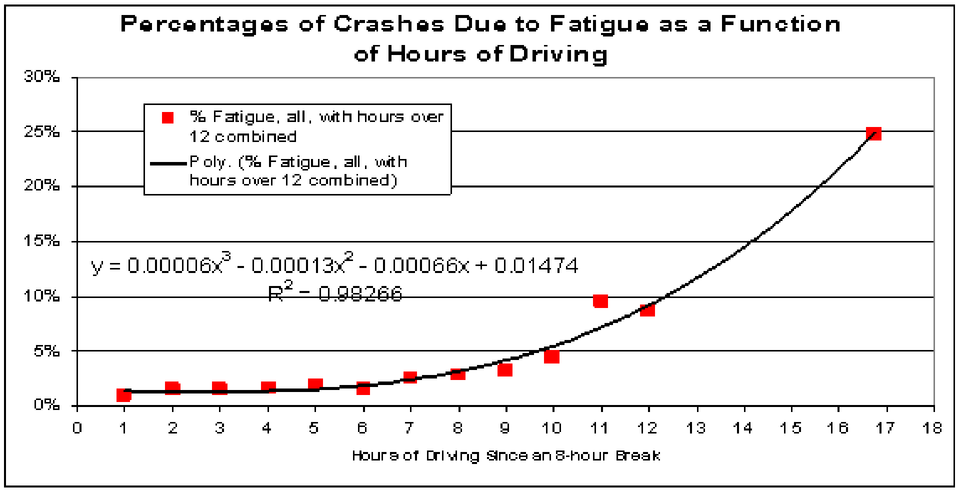


Figure 1: The figure shows the correlation between number of hours driven and the percentage of crashes related to driver fatigue (Federal Motor Carrier Safety Administration, 2008).

In Figure 1, we can see how the correlation between the number of hours driven, and the percentage of crashes associated with driver fatigue are linked (Federal Motor Carrier Safety Administration, 2008). A study led by the Adelaide Centre for Sleep Research (NHSTA) has indicated that drivers who have been awake for 24 hours have the same driving performance to a person who has a blood alcohol content of 100ml/0.1g and is seven times more likely to have a motor accident. Furthermore, NHSTA concluded that drowsy driving is just as hazardous as drunk driving (Department for Transport, Energy and Infrastructure, 2018). Thus detecting drowsiness and predicting when a person may have fallen asleep whilst driving is pivotal in preventing car accidents and potentially saving lives – contributing to the well-being of the communities we live in.

Background

In a study conducted by Dinges and Mallis (Dinges, 1998), four categories of fatigue detection technologies were identified:

1. Vehicle-based Performance Technologies
2. Ambulatory Alertness Prediction Technologies
3. Fitness for Duty Technologies
4. In-vehicle, Online, Operator Status Monitoring Technologies: Behavioral Studies using Psychological Signals and Computer Vision Systems

Vehicle-based performance technologies utilize sensors on standard vehicle components and monitors the signals sent by these sensors to identify drowsiness (Takei, 2005). The two most conventional vehicle-based performance measurements for detecting driver drowsiness are the steering wheel movement (SWM) and the standard deviation of lane positioning (SDLP).

Steering Wheel Movement (SWM) methods rely on measuring the steering wheel angle using an angle sensor mounted in the steering wheel column, which allows for detection of micro corrections. When the driver is drowsy, the number of micro corrections on the steering wheel is lower than that found in normal driving conditions (9). A potential problem with this approach is the high number of false positives. SWM based systems can function reliably only in specific environments (37) and are too dependent on the geometric characteristic of the vehicle and of the road, making them only functional on motorways (7). Standard Deviation of Lane Position (SDLP) methods are based on an externally mounted camera and associated software, which monitor the vehicles relative position to a lane. SDLP systems' limitations are mostly tied to their dependance on external factors such as road markings, weather, and lighting conditions. This means that SDLP systems can only operate reliably when all external variables are satisfied, which is not always the case. For example, not all roads have road markings, and the weather and lighting conditions can change at any time.

Both methods involve modifying or buying a car with these detection methods already included. Because of this, I have decided to go with the eyes closed method.

Ambulatory alertness prediction technologies aim to predict operator performance/alertness at varying times based on interactions of circadian rhythm, related temporal antecedents of fatigue and sleep. This technology predicts alertness using devices that seek to monitor sources of fatigue, such as how much sleep an operator has obtained via wrist activity monitors and using this data in combination with a mathematical formula that is designed to predict when future periods of increased sleepiness and fatigue will occur based on the users sleep/work history (33). The limitations of these technologies are their prediction accuracy. It is not an immediate means of accurately detecting current fatigue occurrence in a user.

Fitness for duty technologies assess the alertness or vigilance capacity of an operator before high-risk work tasks are completed such as truck driving or mining. The Truck Operator Proficiency System has achieved good results in identifying drivers who have had little to no sleep using fitness for duty tests. Failing the test means the driver is perceived to not be capable of driving safely which is a positive method of stopping a driver from commencing on long journeys with pre-existing fatigue impairment (Charlton, 1998). However, the operator's state may change during the course of duty after passing a fitness-for-duty test. For example, a transport research report (Mabbott, 1998), found that half of the drivers who reported falling asleep whilst driving truck had adequate 8 hours plus of sleep the night before. In summary, adequate sleep does not negate the development of fatigue after a significant period of vehicle operation.

Psychological signals can be evaluated to estimate fatigue using Electroencephalography (EEG) (15) (57). EEG is the recording of electrical activity along the scalp produced by the firing of neurons within the brain. In clinical contexts, EEG refers to the brain's spontaneous electrical activity as recorded from multiple electrodes placed on the scalp. There are five major brain waves distinguished by their different frequency ranges. These frequency bands from low to high frequencies respectively are called alpha, theta, beta, delta, and gamma. Alpha waves tend to occur during relaxation or keeping the eyes closed - reference (Jasper and Andrews, 1938). The theta waves represent drowsiness in adults. EEG brain waves are used as a measure to detect drowsiness (38). As the alertness level decreases EEG power of the alpha and theta bands increases, thus providing indications of drowsiness. However, using EEG as a measure of drowsiness has disadvantages, namely in terms of practicality, as it requires a person to wear an EEG cap whilst driving, which could be an invasive distraction.

Computer vision has been a prominent technology in monitoring the human behavior. The advantage of computer vision techniques is that they are non-invasive, and thus are more amenable to use by the general public. Machine learning provides us tools to build computer vision systems that can detect and recognize the facial motion and appearance changes occurring during drowsiness (30)(58). A majority of the published research on computer vision approaches to detect fatigue has focused on the analysis of blinks (53). Percent closure (PERCLOS), which is the percentage of eyelid closure over the pupil over time and reflects slow eyelid closures or droops rather than blinks, is analyzed in many studies (16)(28). Although useful, this does not incorporate the scenario of an eye becoming shut, nor do these studies have any means of waking the driver when this event occurs.

For the work described in this project, a behavioral method for driver drowsiness will be proposed and explored using computer vision. The method is based on the detection of behavioral cues, specifically the closing of the eyes for an extended period to indicate a driver has begun the natural sleep-wake cycle whilst in control of a motor vehicle. This will be completed by using a video camera or image acquisition and rely on a combination of computer vision and machine learning methodologies to detect events of interest - specifically the Eye Aspect Ratio, measure them, and make a prediction on whether the driver has become drowsy. If the sequence of captured live images and measured parameters such as the closed eye state suggest that the driver is drowsy, an action in the form of an audible alarm will be produced to alert the driver and stop them from falling into a state of sleep whilst driving.

The methods, approach, and machine learning techniques for the detection model for use on the Raspberry Pi 4 were evaluated with the following considerations:

1. It must be algorithmically simple and easy to implement.
2. It must be computationally inexpensive and fast as real-time performance will be required
3. It must be accurate.
4. It must be robust and adaptable to different situations such as variations in lighting, angles of the face, camera motion such jumping up and down on rough terrain, and changes of visual appearance such as adding or removing glasses.

Background on Machine Learning Techniques

Here we will give an introduction to the machine learning concepts that have been used for the content of this project.

Viola-Jones Object Detection Framework

Object Detection using Haar feature-based cascade classifiers is a machine learning approach where a cascade function is trained using the Viola-Jones algorithm from a dataset containing positive and negative images (Viola and Jones, 2001). The algorithm is given images of faces for positive detections and images without faces for negative detections.

Haar features are a sequence of rescaled square shape functions proposed by Hungarian mathematician Alfred Haar in the 1909 proposal "square-shaped functions forming a wavelet family" (find a reference). The features sought by the Viola-Jones detection framework universally involve the sums of image pixels within rectangular areas. They share some similarities to Haar basis functions, which have been used historically in the field of image-based object detection (Papageorgiou, 1998). Figure 2 illustrates the five different types of features used in the Viola-Jones framework.

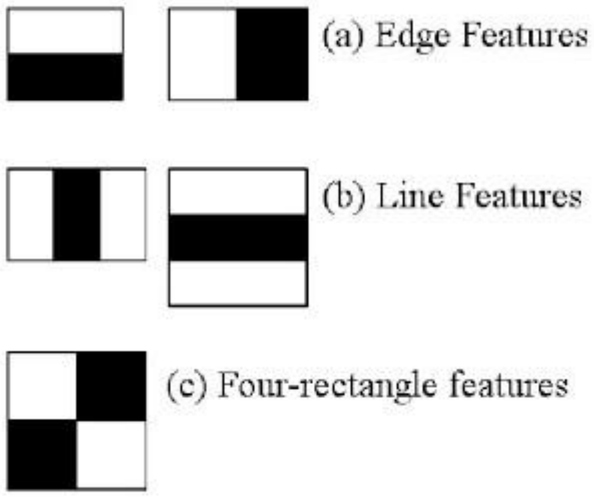


Figure 2: Example Rectangle Features of the Viola-Jones Algorithm
(docs.opencv.org, n.d.).

Depending on the feature each one is looking for, these are broadly classified into three categories as seen in Figure 2. The two rectangle features (a) are responsible for finding the edges in a horizontal or vertical direction. The three rectangle features (b) are responsible for finding if there are lighter regions surrounded by darker regions on either side. The four rectangle features (c) are responsible for finding the change of pixel intensities across diagonals.

All human faces share similar characteristics. When analysing faces in an image, there are some generalizations we can assume, such as the eyebrow is darker than what is above and below it, the eye region is darker than the upper cheeks and the nose bridge region is brighter than the eyes.

Using the rectangle features of the Viola Jones Algorithm, we can begin to make detections with these assumptions.

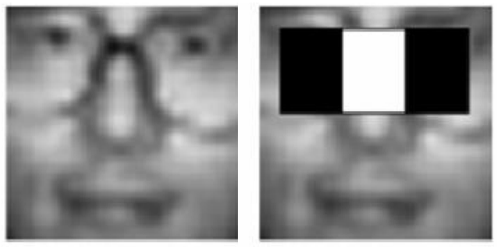


Figure 3: Example Haar feature that shows similarities of the brighter bridge of the nose region in contrast to the darker eye regions (Ngo, 2009).

The objective is to therefore calculate the sum of all the image pixels lying in the darker areas of the haar feature and the sum of all the image pixels in the lighter area of the haar feature. The haar features traverse the image pixel by pixel calculating a single value to make calculations. This process requires a large number of computational mathematical calculations which Viola and Jones reduced in their algorithm by using the Integral Image approach.

The issue with novel calculations of haar features is that the average of a given region must be computed multiple times with the time complexity of these computations being O(n*n). The integral image approach proposed by Viola and Jones was utilized to achieve a O(1) run time for an increase in speed and performance. This is achieved by each features rectangular area always being adjacent to at least one other rectangle. Edge features require 6 array references, line features require 8 and diagonal features require 9. By calculating the sum of each of the pixels in the original image above and to the left of the current pixel, a recursive formula is utilized, reducing the number of operations required.

The simplification of the calculation of the sum of pixels does not change the fact that most of the features calculated will be irrelevant because the only features of importance are those of what we are trying to detect. For example, in a standard 24x24 pixel sub-frame, there are a total of M = 162,336 possible features (Stack Overflow, 2009), making it computationally expensive to use every single sub-window over the entire image to evaluate every pixel when testing an image. Viola and Jones introduced a variation of the Adaboost learning algorithm to improve this. The adaboost learning algorithm chooses the best features out of the 162,336 choices by categorizing them into positive and negative features. Adaboost applied all of the features to each positive and negative image to create weak learners. It uses a combination of weak classifiers to create a strong classifier from the mean prediction of all weak learners that the algorithm can learn from to make detections. Essentially, each iteration improves accuracy by learning from non-detections and detections. This boosting technique reduced the number of features down to 6000.

The subset of all 6000 features will again run on the training images to detect if there’s a facial feature present or not in a sub-window size of 24x24 over the entire image. Despite a drastic improvement on computational time, it could be improved further by using Cascading Classifiers. The idea behind this is that not every feature stage needs to run on every window. By grouping the features into different stages, we can apply them at different times of the detection process. The first stages contain simpler features in comparison to the features in the later stages designed to find specific details of the face. A stage only progresses when all detections are made for that stage. For example, if a feature fails on a specific window in the first stage, then we can say that there are no facial features present, and the other feature stages will not run in that window. We then discard this window and not test it again and move on to the next window where a facial feature may be present. If a facial feature is present in the next window, then all feature stages will be run and the next stage will initiate. When all stages have passed, a facial feature has successfully been detected.

HOG and SVM

Maybe add an image showing the image process?

Histogram of Orientated Gradients (HOG) is a feature descriptor used in image processing for the purpose of object detection and extracting features (or histograms) proposed by Dalal and Triggs who implemented the feature descriptor to make object detections in static images and videos. The theory behind this feature descriptor is that local object shape and appearance in an image can be defined by the calculations of magnitude (gradients) and orientation of the edges that appear in the image. The image is divided into small connected regions called cells, and for pixels within each cell, a histogram of gradient directions is compiled. The descriptor is the concatenation of these histograms. The local histograms can be contrast-normalized by calculating a measure of the intensity across a larger region of the image called a block. This normalization results in better invariance changed in illumination and shadowing. Because the HOG descriptor operates on local cells, it is invariant to geometric and photometric transformations, except for object orientation. Such changes would only appear in larger spatial regions. This is beneficial, as it means other areas of movement in a region can be ignored so long as they stay at a regular constant. For example, extracting the eyes of a person's face whilst the head is moving.

Histograms are created using the HOG feature descriptor by calculating the gradients and orientation of an image. The histograms created in the HOG feature descriptor are not generated for the whole image. Instead, the image is divided into smaller cells, and the histogram of oriented gradients is computed for each cell. This means the histograms for the smaller sub-windows represent the whole image. By dividing the image into smaller cells and generating the histograms, a matrix is composed for each cell. To calculate the gradients of an image, every pixel in the image is considered. Gradients are the slight changes in the x and y coordinate directions that define the shape of an object. A sub-window is analyzed retrieving a pixel matrix for every pixel in the sub-window. The middle pixel of the matrix is selected. To determine the gradient of the x-direction, the pixel value on the left of the middle pixel needs to be subtracted from the pixel on the right of the middle pixel. Likewise, for the y-direction, the pixel value below the middle will be subtracted from the pixel value above of the middle pixel. This will give two resultant gradients for the x and y directions for this pixel forming two new matrices – one storing gradients in the x direction and one storing gradient in the y direction. This process is repeated for all pixels in the image. Using these calculated gradients, the orientation for each pixel value can be calculated. Using Pythagoras theorem, the total gradient magnitude for a pixel can be calculated using the mathematical equation $\sqrt{[(G_x)^2+(G_y)^2]}$ where G_x is the x-direction gradient value and G_y is the y-direction gradient value. The calculation of the orientation (direction) for the same pixel is resulted by using the atan formula $\Phi = \operatorname{atan}(G_y / G_x)$ which produces an orientation value. Using the gradient and orientation calculations, every pixel value will be evaluated within the sub-windows, generating the features used for detection. These features can then be utilized for object detection by providing them as features to machine learning algorithms such as the Support Vector Machine (SVM) algorithm.

SVM is a supervised machine learning algorithm which can be used for classification challenges. Given the HOG features of a sub-window in an image, the SVM allocates a score, which governs how confident the algorithm is that the object which is being searched for is present (Burges, 1998). This score is calculated using binary classification. To classify a window with a feature vector x, the SVM computes the following function $h(x) = wTx + b$ where w is the weight vector and b is the bias. These two parameters yield a hyperplane in feature space, which separates windows containing an object from the background windows in a sliding-window progression (Dutta, 2015). Support Vector Machines support linear and non-linear variants. Non-linear variants use kernel functions to learn hyperplanes in a higher dimensional space. These methods can separate feature vectors of windows that do not appear to be separable in the original feature space. However, compared to the linear SVM, training and classification is computationally expensive for such methods (Cristianani and Shawe-Taylor, 2005), hence for the purpose of this project, linear SVMs will be considered. The prime objective of an SVM is to learn a hyperplane, which can be interpreted as an object model, to capitalize on the margin between two different classes.

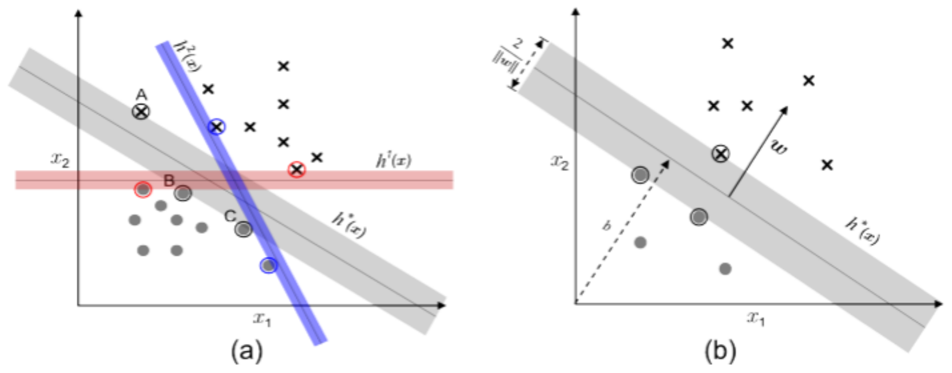


Figure 4: Example of the positive class marked as 'x' and negative class marked as 'o' in a 2D feature space. (Dutta, 2015)

Figure 4 shows two explanatory hyperplanes in blue and red with their respective margins in subfigure (a). The hyperplane that has the maximum margin is drawn in grey. The training examples that lie on the margin are called the Support Vectors. We denote this specific hyperplane as the primal hyperplane h*. The aim of the SVM is to learn the weights and bias of this hyperplane when given a training dataset. The theory of the SVM states that the one that maximizes the margin between two classes also performs best on data that it has not been exposed to during training. Subfigure (b) parameters w, b denote the normal vector of the hyperplane and the distance between the hyperplane and the origin in feature space (Christianini and Shawe-Taylor, 2000). Based on this, the SVM algorithm then learns using positive and negative images whether the object is present or not. This process concatenates the HOG features over the detection window. This produces a model which can be utilized to detect the object that the SVM has been trained for. For the purpose of this project, this will be the eyes of a face.

Deep Learning DNN and CNN (For OpenCV DNN)

Artificial Neural Networks (ANN) can be either shallow or deep. When an ANN has more than one hidden layer in its architecture, they are called Deep Neural Networks (DNN). These networks process complex data with the help of mathematical modelling. These Networks need a huge amount of data to train. Deep Neural Networks have an input layer, an output layer and at least more than one hidden layer between them. DNNs are an instrument of machine learning built to simulate the activity of the human brain – specifically, pattern recognition and the passage of input through various layers of simulated neural connections. The neurons pass the signal to other neurons based on the input received. If the signal value is greater than the threshold value, the output will be passed, else will be ignored. The data is passed to the input layer which yields output to the next layer and so on until it reaches the output layer, where it provides the prediction.

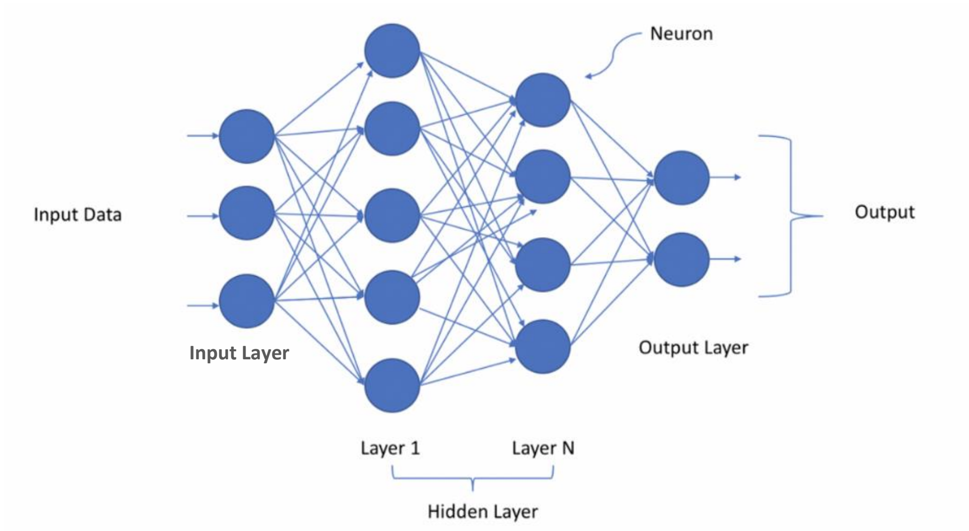


Figure 5: A simplified representation of a DNN. Input data is passed from a dataset, which is then passed to the input layer, then onto n number of hidden layers and nodes which make associations and grade importance of the input to determine the output (Jojo John Moolayil, 2019).

A

layer consists of many neurons which form a gateway of passing the signal to the next connected neuron. The weight has the influence of the input to the output of the next neuron and finally, the last output layer. The weights initially assigned are random, but as the network gets trained iteratively, the weights are optimized to make sure that the network makes a correct prediction.

Data could come from ImageNet or some source similar.

Input data is provided to the to the network and the output prediction gives a correct or incorrect answer. Based on the output, the feedback is fed back into the network, and the network learns by adjusting its weights between the layers based on this feedback. The more layers a model utilizes, higher levels of patterns can be captured. Each layer performs specific types of sorting and ordering to make a feature hierarchy. Deep learning-based object detection have some primary detection methods:

1. Faster R-CNNs ***https://blog.paperspace.com/faster-r-cnn-explained-object-detection/***
2. Single Shot Detectors (SSDs)
3. Then talk about Resnet used with SSDs for OpenCV DNN part.
4. Max Margin Object Detectors

Researchers at UC Berkely developed a deep convolutional network called region-based convolutional neural networks (R-CNN) that can detect 80 different types of objects in an image (Gupta et al., 2014). Compared to feature extractions seen in the HOG image descriptors which then assign them to an SVM model, R-CNN simply extracts the features based on a convolutional neural network (CNN). The R-CNN generates 2000 features using the Selective Search algorithm. After being resized to a fixed pre-defined size, the second module extracts a feature vector from each region. The third module uses a pre-trained SVM algorithm to classify the region proposal to either the background or one of the object classes. The drawbacks of this method is that it relies on the Selective Search algorithm for generating features, which is computationally expensive and requires hundreds of gigabytes of disk space. Fast R-CNN speeds up the R-CNN network. Fast R-CNN proposed a new layer called ROI pooling that extracts equal-length feature vectors from all regions of interest in the same image. Compared to R-CNN which has multiple stages (region proposal generation, feature extraction and classification using SVM), Faster R-CNN builds a network that only has a single stage as opposed to three previously. Fast R-CNN shares computations across all features rather than doing the calculations for each proposal independently. This is achieved by utilizing the introduced ROI Pooling layer. Fast R-CNN does not cache the extracted features and thus does not need a large amount of disk storage compared to R-CNN (He et al., 2018). However, it still utilizes the time-consuming Selective Sort algorithms to generate region proposals.

Faster R-CNN is an extension of Fast R-CNN which utilizes a region proposal network (RPN) for more efficiency. RPN is a fully convolutional network that generates proposals with various scales and aspect ratios. The RPN implements the terminology of neural network with attention to tell the object detector (Fast R-CNN) where to look. Rather than using pyramids of images which use multiple instances of an image but at difference scales or pyramid of filters with multiple filters with different sizes, Faster R-CNN introduces the use of anchor boxes. An anchor box is a reference box of a specific scale and aspect ratio. With multiple reference anchor boxes, multiple scales and aspect ratios can exist for a single region, which can be thought of as a pyramid of reference anchor boxes. Each region is then mapped to each reference anchor box, and thus detecting objects at different scales and aspect ratios. The convolutional computations are shared across the RPN and the Fast R-CNN which therefore reduces computational time. One drawback of the Faster R-CNN however is that the RPN is trained where all anchors are extracted from a single image and all samples from a single image may be correlated where their features are similar, resulting in the network taking a long time to reach convergence. Because of this, it has been ruled out.

(Ren et al., 2017). Faster R-CNN reference

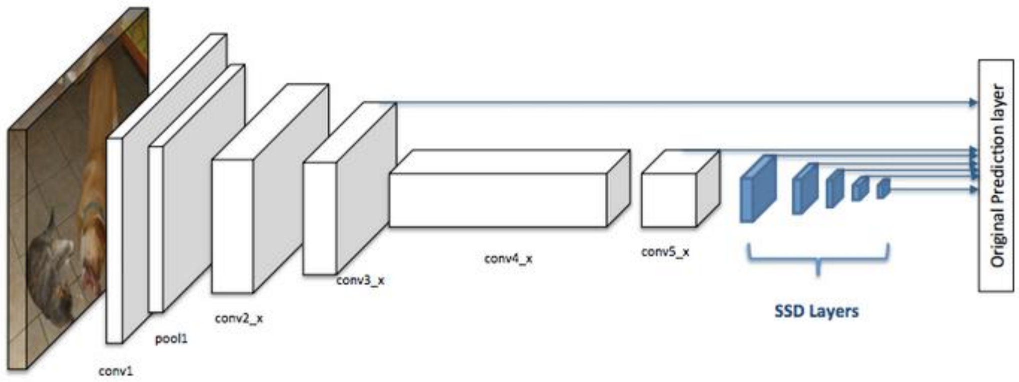
Single Shot Detectors (SSD) with ResNet (for opencv dnn)
https://developers.arcgis.com/python/guide/how-ssd-works/

Single Shot Detectors only need to take one single shot to detect multiple objects within an image, while RPN based approaches seen in the R-CNN series need two shots, one for generating region proposals and one for detecting the object of each proposal. Thus, SSD is much faster compared with two-shot RPN-based approaches.

SSD model has two main parts: Backbone model and SSD head

The Backbone model of the SSD is a typical pre-trained image classification network that works as the feature map extractor. This is typically a network like ResNet from which the final image classification layers of the model are removed to give us only the extracted feature maps. We are thus left with a deep neural network that is able to extract semantic meaning from the input image while preserving the spatial structure of the image.

The SSD head is made up of a one or more convolutional layers stacked together and it is added to the top of the backbone model. This gives us the output as the bounding boxes over the objects in the spatial location of the final layer activations. These convolutional layers detect the various objects in the image.



In the above figure, the first few layers (white boxes) are the backbone, the last few layers (blue boxes) represent the SSD head.

Previously in Haar Cascades (double check, could be HOG) we have seen an image classification model being used on top of an object detection model and the image classifier detects objects by sliding a window across the image and classify whether the image in that window (sub-window cropped out region of the image) is of the desired type. Instead of using sliding windows, SSD divides the image using a grid and has each grid cell be responsible for detecting objects in that region of the image. Detection objects simply means predicting the class and location of an object within that region. If no object is present, we consider it as the background class and the location is ignored. Each grid cell is able to output the position and shape of the object it contains.

SSD deals with multiple objects in one grid cell or multiple objects of different shapes by using anchor boxes. Each grid cell in SSD can be assigned with multiple anchor boxes. These anchor boxes are pre-defined and each one is responsible for a size and shape within a grid cell. SSD encompasses a matching phase whilst training which matches the most appropriate anchor box with the bouding boxes of each object detected in an image. Essentially, the anchor box with the highest degree of overlap with an object is responsible for predicting that objects class and location. Thhis property is used for training the network and for predicting detected objects at their locations once the network has been trained. In practice, each anchor box is specified by an aspect ratio because not all objects are square in shape. The SSD architecture allows pre-defined aspect ratios of the anchor boxes to account for this. The ratios parameter can be used to specify different aspect ratios of the anchor boxes associates with each grid cell at different scale levels. It is not necessary for the anchor boxes to have the same size as the grid cell. The scale parameter is used to specify how much the anchor boxes need to be scaled up or down with respect to each grid cell.

Receptive field is defined as the region in the input space that a particular CNNs feature is looking at. Because of the convolution operation, features at different layers represent different sizes of region in the input image. As it goes deeper, the size represented by a feature gets larger. These feature maps refer to a set of features created by applying the same feature extractor at different locations of the input map and allows for object detection at different scales.

Max-Margin Object Detector (MMOD) is a CNN face detector for dlib

Evaluation of Methods

The You Only Look Once (YOLO) model was considered as a competitor for using a fully convolutional approach. Compared to the YOLO object detection model which also detects objects with a single forward pass, SSD is simple and fast. Although YOLO achieves higher detection rates, SSD is superior at running on limited resource systems and is beneficial for landmark detections (found here: https://algoscale.com/blog/yolo-vs-ssd-which-one-is-a-superior-algorithm/)

At the end, give a brief paragraph about certain comparisons you can make. Use this to help: https://www.baeldung.com/cs/svm-vs-neural-network

Based on the background literature review, talk about the choices you made for the project and why. Pros and cons, sum up.

Face Detection in Images

These object detections are then combined with DNN modules.

HOG features are fed to the SVM which determines whether there is a positive detection or negative, this is how it is trained. A model is then composed, which can be used to make detections.

OpenCV's deep learning face detector is based on the Single Shot Detector (SSD) framework with a ResNet base network (unlike others such as MobileNet).

Explain and find references for how good HOG and SVM work and justify choice. Some can be found here: https://www.ijert.org/research/image-classification-using-hog-and-lbp-feature-descriptors-with-svm-and-cnn-IJERTCONV8IS04021.pdf

Dlib offers HOG and MMOD

OpenCV Offers Haar Cascades and DNN

Details om how SSD works with resnet https://debuggercafe.com/object-detection-using-ssd300-resnet50-and-pytorch/

How does your background research link to face detection? It has solely been focused on object detection for now…

Remember to use words like 'in a study shown by… this happened' to support points or not

Application of the methods and evaluation of the results

There will be 4 main applications of the methods:

Detection stage

Tracking stage

Warning stage

Alert stage

HOG and SVM process: Data prep – change size. HOG feature extraction. SVM Training, get SVM model, give test images – positive negative, make predictions.

Resizing of the image when using the algorithm to train images to compute haar features, which is done in pycharm.

HOG Feature stage: Preprocess the Data (64 x 128)

https://www.analyticsvidhya.com/blog/2019/09/feature-engineering-images-introduction-hog-feature-descriptor/

1. Maycock, G. (1995) *Driver Sleepiness as a Factor in Car and HGV Accidents*, Transport Research Laboratory (TRL), Crowthorne, Berkshire, UK
2. https://ec.europa.eu/transport/road_safety/specialist/knowledge/fatique/fatigue_and_road_crashes/frequency_of_fatigue_related_crashes_en (European COmmision)
3. file:///C:/Users/sam_s/Downloads/9789241565684-eng.pdf (WHO, 2018)
4. Dingus, T. A., Klauer, S. G., Neale, V. L., Petersen, A., Lee, S. E., Sudweeks, J., Perez, M. A., Hankey, J., Ramsey, D., Gupta, S., Bucher, C., Doerzaph, Z. R., Jermeland, J. & Knipling, R.R. (2006) *The 100-Car Naturalistic Driving Study:* Phase II - Results of the 100-Car Field Experiment. DOT HS 810 593. National Highway Traffic Safety Administration, Washington, DC Accessed 4 February 2008: https://www.nhtsa.gov/sites/nhtsa.gov/files/100carmain.pdf
5. Horne, J.A. & Reyner, L.A. (1995) *Sleep related vehicle accidents*. British Medical Journal, 310, 565-567
6. Tesla. (2019). *Autopilot and Full Self-Driving Capability*. [online] Tesla.com. Available at: https://www.tesla.com/support/autopilot.

www.autotrader.co.uk. (n.d.). *New & used Tesla Model X cars for sale | AutoTrader*. [online] Available at: https://www.autotrader.co.uk/cars/tesla/model-x [Accessed 4 Nov. 2021].

Dinges, D. F., Mallis, M. M. (1998), *Managing fatigue by drowsiness detection: Can technological promises be realised?* Fremantle, Western Austrlia. Elsevier Science Ltd [online] Available at: https://m3alertness.com/wp-content/uploads/2014/01/Dinges-and-Mallis-1998-Technology-chapter-in-Hartley.pdf

Federal Motor Carrier Safety Administration. (2008). *Regulatory Impact Analysis and Small Business Analysis for Hours of Service Options.* [online] Available at: https://web.archive.org/web/20080126005807/http://www.fmcsa.dot.gov/rules-regulations/topics/hos/regulatory-impact-analysis.htm

Department for Transport, Energy and Infrastructure. (2018). *Fatigue-related Crashes in South Australia,* Government of South Australia [online] Available at: https://web.archive.org/web/20180219173252/https://www.dpti.sa.gov.au/__data/assets/pdf_file/0010/51022/Fatigue_Fact_Sheet_2010.pdf

Takei, Y. and Furukawa, Y. (2005). *Estimate of driver's fatigue through steering motion*. [online] IEEE Xplore. Available at: https://ieeexplore.ieee.org/document/1571404 [Accessed 11 Nov. 2021]. – USE THIS REFERENCE AS A TEMPLATE FOR OTHERS

7.

Mabbott, N.A., Lydon, M., Hartley, L. & Arnold, P. (1999). Procedures and Devices to monitor operator alertness whilst operating machinery in open-cut coal mines. Stage 1: State-of –the-art review. [online] Available at: https://www.acarp.com.au/abstracts.aspx?repId=C8035

Charlton, S.G. & Ashton, M.E. (1998). Review of Fatigue Management Strategies in the Transport Industry. Land Transport Safety Authority. Wellington, New Zealand.

Viola, P. and Jones, M. (2001). Rapid Object Detection using a Boosted Cascade of Simple Features. *ACCEPTED CONFERENCE ON COMPUTER VISION AND PATTERN RECOGNITION*. [online] Available at: https://www.cs.cmu.edu/~efros/courses/LBMV07/Papers/viola-cvpr-01.pdf.

Papageorgiou, M., Oren, M., Poggio, T., (1998). General framework for object detection. *International Conference on Computer Vision.* [online] Available at: https://www.researchgate.net/publication/3766402_General_framework_for_object_detection.

docs.opencv.org. (n.d.). *OpenCV: Face Detection using Haar Cascades*. [online] Available at: https://docs.opencv.org/3.4/d2/d99/tutorial_js_face_detection.html.

Ngo, H. and Rakvic, R. and Broussard, R. and Ives, R. (2009). An FPGA-based design of a modular approach for integral images in a real-time face detection system. [online] Available at: https://www.researchgate.net/publication/268348020_An_FPGA-based_design_of_a_modular_approach_for_integral_images_in_a_real-time_face_detection_system

Stack Overflow. (2009). *algorithm - Viola-Jones' face detection claims 180k features*. [online] Available at: https://stackoverflow.com/questions/1707620/viola-jones-face-detection-claims-180k-features [Accessed 16 Nov. 2021].

Burges, C.J.C. (1998). *Data Mining and Knowledge Discovery*, [online] 2(2), pp.121–167. Available at: https://link.springer.com/article/10.1023/A%3A1009715923555 [Accessed 15 May 2019].

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2005.

N. Cristianini and J. Shawe-Taylor. An introduction to support vector machines : and other kernel-based learning methods. Cambridge University Press, 2000

Dutta, S. (2015). Pedestrian Detection using HOG and SVM in Automotives. [online] Available at: http://surajitdutta.com/downloads/research_report_part2.pdf [Accessed 18 Nov. 2021].

Gupta, S., Arbeláez, P., Girshick, R. and Malik, J. (2014). Indoor Scene Understanding with RGB-D Images: Bottom-up Segmentation, Object Detection and Semantic Segmentation. *International Journal of Computer Vision*, [online] 112(2), pp.133–149. Available at: https://www.cv-foundation.org/openaccess/content_cvpr_2014/papers/Girshick_Rich_Feature_Hierarchies_2014_CVPR_paper.pdf.

Ren, S., He, K., Girshick, R. and Sun, J. (2017). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(6), pp.1137–1149.

He, K., Gkioxari, G., Dollar, P. and Girshick, R. (2018). Mask R-CNN. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp.1–1.

Jojo John Moolayil (2019). *A Layman's Guide to Deep Neural Networks*. [online] Medium. Available at: https://towardsdatascience.com/a-laymans-guide-to-deep-neural-networks-ddcea24847fb.

8.
9.