Detecting Drowsiness in Drivers using Approaches based on Machine Learning Methodologies

Ashwin Suresh Babu
ID:19151381
MSc Research Project
MSCDAD JAN22 B

January 2022

Abstract

One of the leading causes of accidents among drivers is drowsiness and fatigue. It is increasing year by year in uninformative manner. There have been many researchers conducted on this topic and one of the prominent conclusions out of those are that a 24 hour sleep deprivation can cause the same amount of effect as a person who has an blood alcohol level of 0.10% which is well above the legal limit. Some of the previous researches on the topic of driver drowsiness detection has been focused mainly on lane deviation detection. In this research we have discussed about a method where we will be using TensorFlow to train images of different facets of a human face and the actions to make the algorithm learn how to detect drowsiness, some of the algorithms used are Deep Learning model, and the machine learning model used is Logistic Regression and Support Vector machine. In this research we have divided the dataset into four main categories namely open eyes, closed eyes, yawn and no yawn. Using these 4 facets we will train the machine learning algorithms to predict the drowsiness of the driver.

Keyword – Fatigue, Drowsiness, TensorFlow, Machine Learning, Logistic Regression, Support Vector Machine, MaxPOoling2D, Conv2D

1 Introduction

1.1 Background

Driver drowsiness is a significant safety concern in the taxi industry, as it can increase the risk of accidents and other mishaps. By detecting drowsiness, taxi companies can take steps to address the problem and prevent accidents from occurring. This can help to improve the overall safety and reliability of the taxi industry, which can in turn help to build trust and confidence among passengers.

To solve this problem, a variety of methods have been created to recognize when a driver is starting to nod off. The two primary types of these methods are objective and subjective. The physiological status of a driver can be directly measured by objective means using sensors and other technologies, such as eye movement or heart rate. Conversely, subjective approaches rely on information provided by the driver, such as their level of weariness or how long they have been awake. Both of these methods have benefits and drawbacks, and they are frequently combined to give a more thorough picture of a driver's condition. Detecting driver drowsiness can also help to protect the lives of drivers, passengers, and other road users, and can potentially save lives by preventing accidents.

1.2 Aim

The main goal of this project was to build a program using machine learning algorithms to recognize if the taxi drivers are experiencing fatigue in order to improve road safety. Drivers who are drowsy during their working hours are a major risk and contribute to the majority of the taxi accidents in the world. And by using machine learning algorithms we were able to build a system that could successfully identify if a driver was feeling drowsy from using data collected of their face when their eyes were closed, open, yawning and not yawning. This data was then train and tested to accurately predict when a driver was feeling drowsy. The models where then tested to provide the utmost accuracy reliability in the aim of helping reduce accidents caused by drowsy driving.

1.3 Research Question

How can several machine learning modelling techniques(Conv2D, MaxPooling 2D, Logistic Regression and Support Vector Machine) be effectively used to conduct a classification of images of facial characteristics to accurately predict fatigue experienced by taxi drivers?

1.4 Technology and Techniques

In this project we have used python as the main programming language since it has powerful and extremely useful libraries such as TensorFlow and Scikitlearn that makes it easy to help build machine learning models. The dataset was first imported onto a python file on jupyter Lab. After importing the dataset we then used matplotlib to visualize the images of each scenarios such as the open eyes, closed eyes, yawning and no yawn.

We then move on to the pre-processing stage, since we only need the face structure for our evaluation therefore in the next step we proceed to remove the background of the face and proceed to focus on the face and put into an image array. Then convert all the different images such as open eyes, close eyes, yawn and no yawn to 0, 1, 2 and 3 respectively.

In terms of machine learning techniques and methods used, Once the pre-processing of the data is done we move to train and split the data for the modelling purposes using the train test split function from the scikit python library. We then use machine learning algorithms such as TensorFlow, Conv2D, MaxpOoling2D, logistic Regression and Support vector machine to train the models for analysis. Once that is done we use confusion matrix to get a classification report of all the models that were run.

We will utilize the SVM and logistic regression models to generate predictions on fresh data once they have been trained. By contrasting the predictions of the models with the labels present in the test data, we can assess how well the models performed. By doing so, we will be able to identify which model is more accurate and make the required modifications to enhance its performance.

2 Related Works

In a similar research by (Alshaqaqi et al. 2013) they have proposed a method called the PERCLOS(Percentage of Eyes Closure) where the algorithm is trained to detect the face and the eyes and calculate the percentage of the closure of the eyes. How the algorithms works is first they capture the face and eyes in initialization stage and for ever tracking they check if the outcome is good or bad and if the outcome is bad then it goes back to the initialization stage if its good then it is passed onto the eyes state identification and the drivers state identification.

From all the previous research papers examined we have come to know that there is a lot of ways to assess the situation of drowsiness of a person just from their eyes In a method proposed by (Hu & Zheng 2009) has used the method of calculating the eye blinks from the EOG data and then process it using support vector machine algorithm.

In a controlled laboratory study conducted by (Forsman et al. 2013) of two groups of people, group A consisted of men and women and group B consisted just men and they performed shifts on driving which would cause fatigue and observed that the correlation between lane variability and individual driver fatigue happens to exceed the correlation between any other variables.

In a research done by (Reddy et al. 2017) where Deep Learning Networks were used and advanced machine learning algorithms like RCNN were used for faster processing of the data They discovered that eyes and mouth made the most impact when trying to figure out the drowsiness of a driver in a real time scenario.

When we consider the fact the accidents do occur when associated with driver drowsiness it is mainly because the driver goes into an unconscious state and loses control over the steering wheel, in a research by (Eskandarian & Mortazavi 2007) they have exactly evaluated that using a truck drivers simulator where the subjects steering input is monitored and evaluated.

In a study by (Sahayadhas et al. 2012) they have evaluated a whole process using sensors to measure the correlation between three main characters which are 1. Vehicle based measures, 2. Behavioural measure and 3. Psychological measures. They came to a conclusion that the level of drowsiness totally depended on the time of the day, duration of the task and time elapsed from the last sleep.

In a more advanced study conducted by (Gao et al. 2015) they have used an eye tracking glass that are fitted with IR sensors to track the movements of the eyes and eyelids. Even though their experiment was a success they realised that the feasibility of the research was not upto the mark, since some drivers might have already glasses and having to wear an extra glass would not be convenient.

In a research by (Zhang et al. 2012) they have used Fishers linear discriminant function on 6 measure that they felt valid such as percentage of eyelid closure, duration, blink frequency, average opening levels of the eyes, opening velocity and closing velocity of the eyelids to reduce the correlations. And they have proposed to do it in a real time video format manner.

(Wang et al. 2019) Have used the latest technology such as an VR headset to track the eye movements to predict the fatigue that is being experienced by the subjects Since this was conducted in a laboratory and uses optometry tools to analyse the fatigue on the eyes and were successfully able to calculate the fatigue cause on the eyes. But this method cannot be used in an real world scenario.

(Akshay et al. 2021) have used AdaBoost based face detection to track and monitor the eye movements and eyelid tracking which they have referred to as gaze tracking. In order to evaluate the drowsiness level of the subjects in real time using IR cameras they have managed to find a correlation between their finding and PERCLOS.

In the paper by (Jose et al. 2021) they have conducted an experiment of planting an camera to monitor the motions of eyes of the driver and the yawn using OpenCV and python eye detection algorithms. Also to accurately track the movements of the face they have also used Face Mapping algorithms where the structure of the face is outlined along with numbers and then it is used to map the face. By this the face mapping becomes more accurate.

In a review by (Shi et al. 2017) they have come to conclusion that there are 4 main factors that are responsible effectively detect the fatigue on a subjects face which are Head position, gaze direction, blink frequency and yawning detection. they first collected the signals from a sensor which collected all the data and then processed all the signals by Gobar filter and Support vector machine algorithms and get the analysis results.

(Deng & Wu 2019) have proposed a method called DriCare which is no contact method that tracks the eye direction and movements, yawning, blinking and eyes closure from video images that they collect using a camera mounted in the vehicle. They were successfully able to make a prediction of the fatigue with an accuracy level of 92%.

(Moujahid et al. 2021) have come up with a method where they have 5 situation to detect the face better and get higher accuracy in predicting the drowsiness of the subject in the situation. The situation include Bareface, glasses, sunglasses, night bareface and night glasses. After the data were collected from the five different scenarios they were analysed using SVM algorithms and decision fusion method.

In a similar fashion to what we have attempted here (Valsan et al. 2021) has also used OpenCV python to track the image of face and other important details from the page to analyse the situation. They created two tests one with 100 images and another with 1000 images and carried out to find the true and false rate of the prediction.

3 Research Methodology

In this section we will discuss about how the experiment was setup, the environment used, process, the data, and the transformation into our desired results. To detect and evaluate the different facial characteristics.

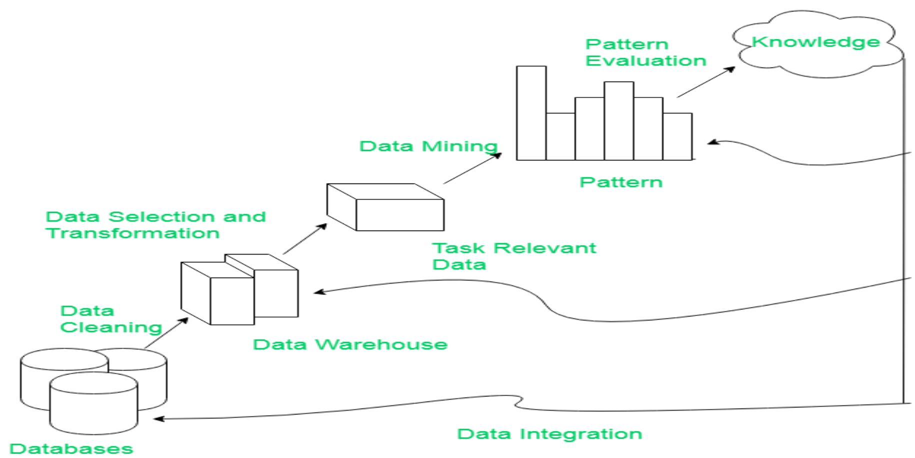

Figure 1: The KDD Methodology

The whole experiment was conducted using the KDD Methodology. The KDD technique (knowledge discovery in databases) is a systematic strategy for extracting insightful information from large, complicated datasets. It often has a number of phases, which includes data preparation and cleaning, feature engineering and selection, model training and assessment, and result interpretation. KDD approach frequently combines machine learning, statistics, and data mining methods with domain expertise, as well as domain-specific data pre-treatment and visualization.

The dataset (*driver drowsiness using keras — kaggle.com* n.d.) that we have used in this research is acquired from the open-source platform Kaggle. At first, we import and install all the necessary libraries for this project in python, then we import the dataset to our local directory and store it. Then we use python to access that directory and the contents inside that which is our dataset. We first list out the labels in the particular directory to understand the data better. our dataset included images of different facial characteristics of a human face such as open eyes, closed eyes, yawning and no yawn, so in order to better understand we first visually plot the data using matplotlib function from each category.

Figure 2: Visual representation of all the four facets of the data

3.1 Data Preparation

Since there is background in the photos of Yawning and No yawn and since our aim was to create a more accurate prediction we wanted to make the data a little easier to train therefore we write a function that will enable us to detect faces in images and returns a list of those faces, along with their corresponding labels. Now that we have pre-processed the data we then use the append function to convert the data into a NumPy array for further evaluation. Since our dataset consists of images, images have features, and labels, for example, a feature includes the pixel count of the raw image. So in the next step we create two variables x and y, and append the feature and label data to these data respectively for further machine learning modelling.

3.2 Data Processing

The dataset that we are using has a total of 4800 images for us to train the models with. The format of the images is JPG(JPEG) format, the distribution of images along all four categories is as follows; eyes open - 725, Eyes closed - 725, Yawn - 726, No Yawn - 2624.

Once the data was pre-processed it was time for further evaluation. We then created a NumPy array and then reshaped it into a 4-dimensional array. We then used the LabelBinarizer function to binarize the labels of the feature into 0s and 1s. And then we split the data into train and test with the test size of 0.30, which means the training dataset gets 70% and 30% of the data is allocated to the test split to build our model and run machine learning algorithms. Both the LabelBinarizer and TrainTestSplit functions are imported from the python library sklearn. We then used the ImageDataGenerator function the from Keras library to augment the data and make random transformations of the data in order to create new artificial training examples. We then used keras to train the data.

4 Design specification

Since the data that we have used are images, therefore we have used appropriate modelling techniques to run the data. Images may be modelled using a wide range of approaches, including both conventional machine learning techniques and deep learning techniques. Typical methods used for modelling photos include Logistic Regression, Support Vector Machine, and Keras.

4.1 Data Modeling techniques

4.1.1 Keras Model

For the construction and training of neural network models in Python, Keras is a high-level deep learning framework. For image classification, it offers a user-friendly API that makes it simple to design, train, and assess complicated neural networks. With a dataset made up of photos, a convolutional neural network (CNN) may be built and trained using Keras to sort the images into several categories. A CNN is a particular kind of neural network created with image categorization tasks in mind. It has several convolutional, pooling, and fully-connected layers, and it can automatically learn sophisticated characteristics from the photos.

4.1.2 Logistic Regression

A straightforward and well-liked method for binary classification is logistic regression. The input data must be in an appropriate format in order to perform logistic regression for picture classification. In order to do this, the picture data is generally flattened into a 1D array, and the target labels are checked to make sure they are in a 1D array of integer class labels. A logistic function is used in logistic regression to represent the connection between the input characteristics and the target labels. The probability that an input falls into the positive class is represented by the probability value between 0 and 1 that this function outputs.

4.1.3 Support Vector Machine

Support vector machines (SVMs) are a well-liked classification method. They may be applied to binary and multi-class classification and work best when there is a distinct line of demarcation between the classes in the data. The input data must be in the proper format for an SVM to be used for picture classification. Flattening the picture data into a 1D array and making sure the target labels are in a 1D array of integer class labels are the usual steps involved in doing this. A hyperplane in the feature space that maximum separates the various classes in the data is found by an SVM to perform its function. The nearest points to each class, referred to as the support vectors, are selected as the hyperplane such that they are equally far from it.

5 Implementation

The implementation of this research was done using Python Programming Language, and Jupyter Notebook as the IDE. In order to run the code for the project we have used python libraries such as NumPy, Pandas, Scikit Learn. The dataset[1] was extracted from a public source Kaggle.

The dataset consists of 4 different facial characteristics that will be used to train the models used in this research. the facial characters play an important role in this implementation as they are the deciding factor for the successful prediction

[1] Dataset: https://www.kaggle.com/code/kanyadharaakash/driver-drowsiness-using/data

of identifying fatigue on the subject. The four facial characters are images of subjects with their eyes open, closed, yawning and no yawn.

We have used ImageAugmentation function to augment the images in different manner so that we can train the model more accurately. Then we split the data into test and train data for the deep learning model. The deep learning model used Conv2D and MaxPooling2D to classify the images into four categories. Once we categorized the dataset into four categories, then we train that model. The first deep learning model has an accuracy of 83.04%.

Now for the Machine learning models, we again append the label and feature values into a list do the process all over again. Once that was done we then again we split the data into train and test for the execution of machine learning models. We then ran two highly used machine learning models like the Logistic Regression model and the support Vector machine, after all the execution SVM and Logistic regression had an accuracy of 92.38 and 92.04 respectively.

6 Evaluation

In this section we will discuss about the results that we have achieved from the experiment by running from two machine learning models and one deep learning model.

6.1 Deep Learning Model

The aim of the experiment was to execute machine leaning techniques to achieve an reliable accuracy to detect drowsiness of drivers. After the deep learning model we can see that it has shown an accuracy of 83.04%. Even though we have to convert the image dataset into a 4D array and then train the model. Even with so much data to work with the deep learning network under performs from the other linear regression models.

6.2 Machine Learning Model

In order to execute the Logistic regression and SVM models we had to append the image dataset into lists again and then convert it into a N-Dimensional NumPy Array. Because of the fact that both these algorithms does not support multi-dimensional arrays. In order to use logistic regression for multi-class classification, you would need to transform the data into a binary format that the algorithm can understand. Even though both logistic regression and support vector machines

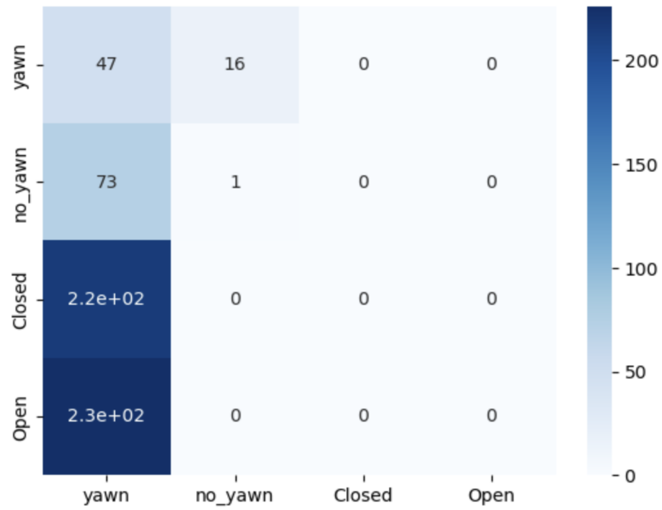

Figure 3: Heat map of the Deep Learning Model

(SVMs) are classification algorithms, but they are designed to solve different types of problems and have different strengths and weaknesses.

Table 1. Accuracy of Machine Learning Models

Machine Learning Models	Accuracy
Logistic Regression	92.04
Support Vector Machine	92.38

7 Discussion

From the experiment we have found some interesting points that could have caused the most suitable modelling technique for images, the deep learning model to be the under performer. There may not be sufficient data in the dataset to train the deep learning model successfully. If the dataset is limited, a deep learning model may not have learnt enough to produce reliable predictions since deep learning models need huge quantities of data to function properly. It's possible that the deep learning model won't be effective for the task. The strengths and disadvantages of various machine learning models vary, and some are better suited to particular tasks than others. The logistic regression models could be more effective for the given job, but the deep learning model may have trouble producing precise predictions.

The deep learning model's hyperparameters may not be properly calibrated. A machine learning model's behaviour is determined by its hyperparameters, and if they are not set properly, the model may not work successfully. It's likely that the logistic regression models' superior accuracy was due to the fact that they were more appropriately calibrated for the job. We may be unaware of other elements that are in play. Without knowing more about the dataset, the models, and the job, it is challenging to pinpoint exactly why the deep learning model did worse because there are several factors that can affect how well a machine learning model performs.

8 Conclusions and Future Work

The shear importance of data when collected properly and when used meticulously and appropriately can make a lot of difference to the society. With help of machine learning and using face recognition along with prediction algorithm we are able to identify the physical fatigue or drowsiness that is being experienced by a taxi driver. Having the information of face at different points such as closed eyes, open eyes, yawn and no yawn we are able to train an algorithm to detect the same.

From the results we can conclude that, the SVM model has performed the best due to its suitability for the job at hand, the SVM model outperformed the other two models. The machine learning method known as SVM (support vector machine) is frequently used for classification tasks like the one in the question. SVM is highly good at creating precise predictions because it finds the hyperplane that maximum separates the various classes in the dataset. It's likely that the SVM model in this instance performed better than the other models in identifying patterns in the data and making predictions.

There is certainly an exorbitant amount of future work that needs to be put into this topic to more understand the severity of this issue. Some of the improvement that can be made is using the advanced technology if we could come up with a system that could detect the drowsiness of the taxi drivers in real time that could absolutely make a huge difference in the way the taxi drivers operate and can help prevent the amount of accidents that are caused due to overworked taxi drivers.

References

Akshay, S., Abhishek, M., Sudhanshu, D. & Anuvaishnav, C. (2021), Drowsy driver detection using eye-tracking through machine learning, *in* '2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC)', IEEE, pp. 1916–1923.

Alshaqaqi, B., Baquhaizel, A. S., Ouis, M. E. A., Boumehed, M., Ouamri, A. & Keche, M. (2013), Driver drowsiness detection system, *in* '2013 8th International Workshop on Systems, Signal Processing and their Applications (WoSSPA)', IEEE, pp. 151–155.

Deng, W. & Wu, R. (2019), 'Real-time driver-drowsiness detection system using facial features', *Ieee Access* **7**, 118727–118738.

driver drowsiness using keras — kaggle.com (n.d.), https://www.kaggle.com/code/saurabhprajapat/driver-drowsiness-using-keras/data. [Accessed 18-Jul-2022].

Eskandarian, A. & Mortazavi, A. (2007), Evaluation of a smart algorithm for commercial vehicle driver drowsiness detection, *in* '2007 IEEE intelligent vehicles symposium', IEEE, pp. 553–559.

Forsman, P. M., Vila, B. J., Short, R. A., Mott, C. G. & Van Dongen, H. P. (2013), 'Efficient driver drowsiness detection at moderate levels of drowsiness', *Accident Analysis & Prevention* **50**, 341–350.

Gao, X.-Y., Zhang, Y.-F., Zheng, W.-L. & Lu, B.-L. (2015), Evaluating driving fatigue detection algorithms using eye tracking glasses, *in* '2015 7th International IEEE/EMBS Conference on Neural Engineering (NER)', IEEE, pp. 767–770.

Hu, S. & Zheng, G. (2009), 'Driver drowsiness detection with eyelid related parameters by support vector machine', *Expert Systems with Applications* **36**(4), 7651–7658.

Jose, J., Vimali, J., Ajitha, P., Gowri, S., Sivasangari, A. & Jinila, B. (2021), Drowsiness detection system for drivers using image processing technique, *in* '2021 5th International Conference on Trends in Electronics and Informatics (ICOEI)', IEEE, pp. 1527–1530.

Moujahid, A., Dornaika, F., Arganda-Carreras, I. & Reta, J. (2021), 'Efficient and compact face descriptor for driver drowsiness detection', *Expert Systems with Applications* **168**, 114334.

Reddy, B., Kim, Y.-H., Yun, S., Seo, C. & Jang, J. (2017), Real-time driver drowsiness detection for embedded system using model compression of deep neural networks, *in* 'Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops', pp. 121–128.

Sahayadhas, A., Sundaraj, K. & Murugappan, M. (2012), 'Detecting driver drowsiness based on sensors: a review', *Sensors* **12**(12), 16937–16953.

Shi, S.-Y., Tang, W.-Z. & Wang, Y.-Y. (2017), A review on fatigue driving detection, *in* 'ITM Web of Conferences', Vol. 12, EDP Sciences, p. 01019.

Valsan, V., Mathai, P. P. & Babu, I. (2021), Monitoring driver's drowsiness status at night based on computer vision, *in* '2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)', IEEE, pp. 989–993.

Wang, Y., Zhai, G., Chen, S., Min, X., Gao, Z. & Song, X. (2019), 'Assessment of eye fatigue caused by head-mounted displays using eye-tracking', *Biomedical engineering online* **18**(1), 1–19.

Zhang, W., Cheng, B. & Lin, Y. (2012), 'Driver drowsiness recognition based on computer vision technology', *Tsinghua Science and Technology* **17**(3), 354–362.