Drowsiness Detection System(webapp)

<a href = "https://youtu.be/YyIMsBBEukw">Click Here to see the demo</a> <br>

<img src="https://img.shields.io/github/repo-size/fear-the-lord/Drowsiness-Detection"> <img src="https://img.shields.io/github/license/fear-the-lord/Drowsiness-Detection"> <img alt="GitHub last commit" src="https://img.shields.io/github/last-commit/fear-the-lord/Drowsiness-Detection"> <img alt="Libraries.io dependency status for GitHub repo" src="https://img.shields.io/librariesio/github/fear-the-lord/Drowsiness-Detection"> <img src = "https://hitcounter.pythonanywhere.com/count/tag.svg?url=https://github.com/fear-the-lord/Drowsiness-Detection"> <img src = "https://img.shields.io/youtube/views/YyIMsBBEukw?style=social">

## Motivation:

According to the National Highway Traffic Safety Administration, every year about 100,000 police-reported crashes involve drowsy driving. These crashes result in more than 1,550 fatalities and 71,000 injuries. The real number may be much higher, however, as it is difficult to determine whether a driver was drowsy at the time of a crash. So, we tried to build a system, that detects whether a person is drowsy and alert him.

## Installing and Configuring dlib:

We need to create an enivronment in order to install dlib, as it cannot be directly installed using pip. So, follow this commands in order to install dlib into your system if you haven't installed it previously. Make sure you have Anaconda installed, as we will be doing everyting in Anaconda Prompt.

### Step 1: Update conda

```bash

conda update conda

```

### Step 2: Update anaconda

```bash

conda update anaconda

```

### Step 3: Create a virtual environment

```bash

conda create -n env\_dlib

```

### Step 4: Activate the virtual environment

```bash

conda activate env\_dlib

conda install -c conda-forge dlib

```

If all these steps are completed successfully, then dlib will be installed in the virtual environment <b>env\_dlib</b>. Make sure to use this environment to run the entire project.

### Step to deactivate the virtual environment

```bash

conda deactivate

```

## Running the system:

### Step 1:

Clone the repository into your system by:

```bash

git clone https://github.com/fear-the-lord/Drowsiness-Detection.git

```

Or directly download the zip.

### Step 2:

Download the file <b>shape\_predictor\_68\_face\_landmarks.dat</b><a href = "https://drive.google.com/file/d/1CYMKQn5cy9Fg-lbOI0Sp2lUl5k8gdXrI/view?usp=sharing"> here.</a> Make sure you download it in the same folder.

### Step 3:

Install all the system requirments by:

```bash

pip install -r requirements.txt

```

### Step 4:

After the system has been setup. Run the command:

```bash

python app1.py

```

### Step 5:

Open your browser and in the search bar type:

<b>localhost:8000</b> as this port is mostly used by flask.

In case, this port is not available in your system, flask will try to use another port. The port number will be displayed in the command prompt.

So, type in the same port number in that case as:

<b>localhost:<port\_number></b>.

After all these steps have been completed successfully, you will see a web page opening up in the browser. Now you are free to explore the system.

## Working Details:

The basic thing about drowsiness detection is pretty simple. 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. Ofcourse, we don't need all the landmarks, here we need to extract only the eye and the mouth region.

Now, after extraxting the landmarks we calculate the <b>Eye Aspect Ratio (EAR)</b> as:

```python

def eye\_aspect\_ratio(eye):

# Vertical eye landmarks

A = dist.euclidean(eye[1], eye[5])

B = dist.euclidean(eye[2], eye[4])

# Horizontal eye landmarks

C = dist.euclidean(eye[0], eye[3])

# The EAR Equation

EAR = (A + B) / (2.0 \* C)

return EAR

```

The eye region is marked by 6 coordinates. These coordinates can be used to find whether the eye is open or closed if the value of EAR is checked with a certain threshold value.<br>

![blink\_detection\_plot](https://user-images.githubusercontent.com/35571958/87878670-62d41400-ca03-11ea-8b96-fc4344c61a21.jpg)

In the same way I have calculated the aspect ratio for the mouth to detect if a person is yawning. Although, there is no specific metric for calculating this, so I have taken for points, 2 each from the upper and lower lip and calculated the mean distance between them as:

```python

def mouth\_aspect\_ratio(mouth):

A = dist.euclidean(mouth[13], mouth[19])

B = dist.euclidean(mouth[14], mouth[18])

C = dist.euclidean(mouth[15], mouth[17])

MAR = (A + B + C) / 3.0

return MAR

```

<b>Note: Learn more about dlib</b> <a href = "http://dlib.net/">here.</a>

## Results:

The GUI has been created using basic HTML, CSS and JavaScript and we have used Flask to render the python code into the website. Tkinter has also been used in order to make things simpler. It has 2 buttons: Run and Exit. The GUI looks like:

![df01ae7c-afc9-4676-b95b-b6cec592ddf0 (online-video-cutter com) (1)](https://user-images.githubusercontent.com/35571958/87902089-589f2d80-ca76-11ea-9eda-a53a83662721.gif)

The outputs of the working system detecting drowsiness is shown as: <br>

![frame\_yawn1](https://user-images.githubusercontent.com/35571958/87904322-ab2f1880-ca7b-11ea-97d2-82f9dd0c318a.jpg) ![Screenshot (405)](https://user-images.githubusercontent.com/35571958/87904406-dd407a80-ca7b-11ea-982d-1852e2228765.png)

Also, in order to keep a proof of the moment when the person was sleeping or yawning, we kept a seperate folder where those frames are stored as: <br>

![Screenshot (408)](<https://user-images.githubusercontent.com/35571958/87904688-7e2f3580-ca7c-11ea-839b-c049bace332f.png>)

Streaming using Phone Camera

We have used and Android App available for free in Play Store, named IP Webcam. It can be downloaded from this <a href = "https://play.google.com/store/apps/details?id=com.pas.webcam&hl=en\_IN">link</a>. After downloading it, open the app and scroll down to the option <b>Start Server</b>. It will look like: <br>

<img src = "https://user-images.githubusercontent.com/35571958/88623867-83673280-d0c3-11ea-9efd-63559024c0bd.jpg">

After starting the server, an IP will be displayed on the screen. Open the file <b>android\_cam.py</b>. In <b>line 36</b> put the given IP.

```python

url = "http://<YOUR\_IP\_HERE>/shot.jpg"

```

<b>Also, make sure that the phone and PC/Laptop is connected to the same network.</b>

Then, run the system in the same way as mentioned above. Click on the <b>Run Using Phone Cam</b> button to see the results:<br>

<img src = "https://user-images.githubusercontent.com/35571958/88624933-7b0ff700-d0c5-11ea-87da-3f6bf1516cc3.png">

Also, in order to toggle between the front and back camera, type the IP upto "http://<YOUR\_IP\_HERE>" in the search bar of yor browser and explore the page which will look like this: <br>

<img src = "https://user-images.githubusercontent.com/35571958/88626505-5f5a2000-d0c8-11ea-88f0-e1d4481eb9d9.png">

Also, we have tried plotting the MAR and EAR graph Vs. Time in order to make the working clearer to the audience. The graph looks like: <br>

<img src = "https://user-images.githubusercontent.com/35571958/88627012-42721c80-d0c9-11ea-860a-51b7a1f2961b.png">