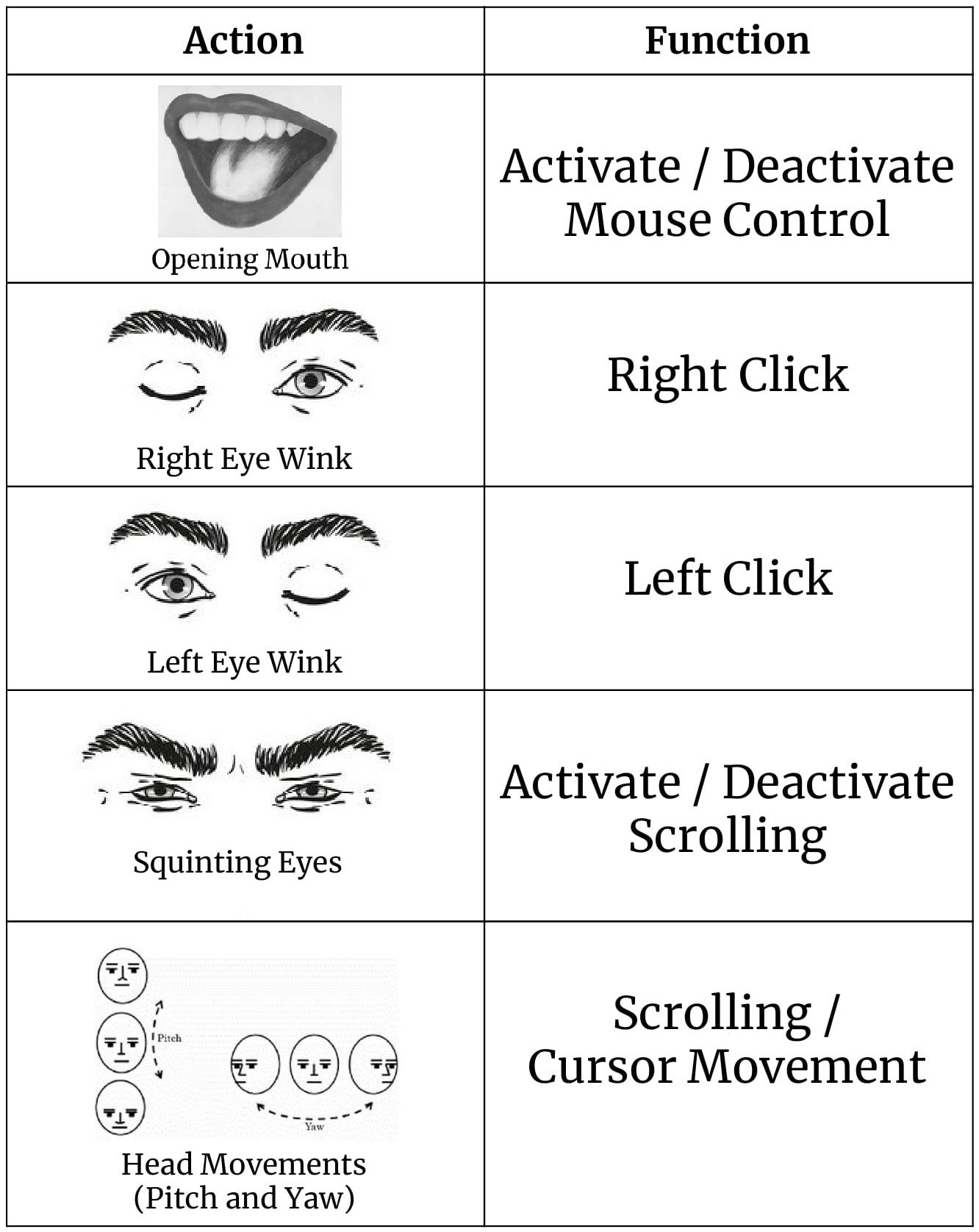
**Usage**



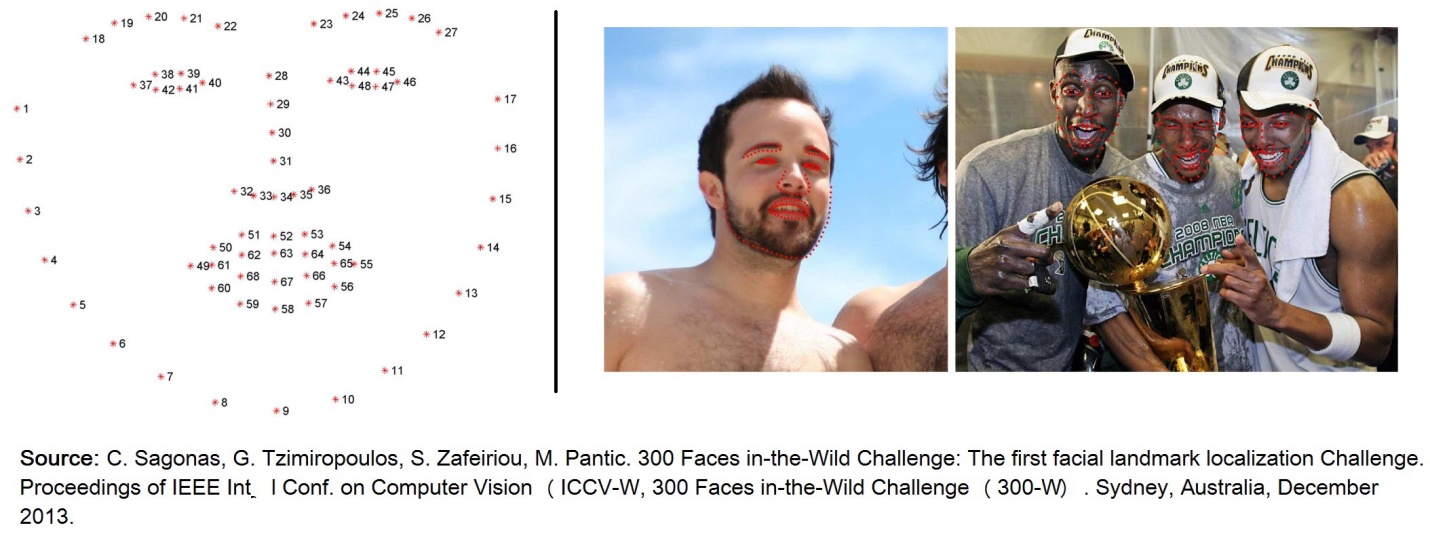
## Libraries Used

* Numpy — 1.13.3
* OpenCV — 3.2.0
* PyAutoGUI — 0.9.36
* Dlib — 19.4.0
* Imutils — 0.4.6

# How It Works

This project is deeply centered around predicting the facial landmarks of a given face. We can accomplish a lot of things using these landmarks. From detecting eye-blinks [3] in a video to predicting emotions of the subject. The applications, outcomes, and possibilities of facial landmarks are immense and intriguing.0

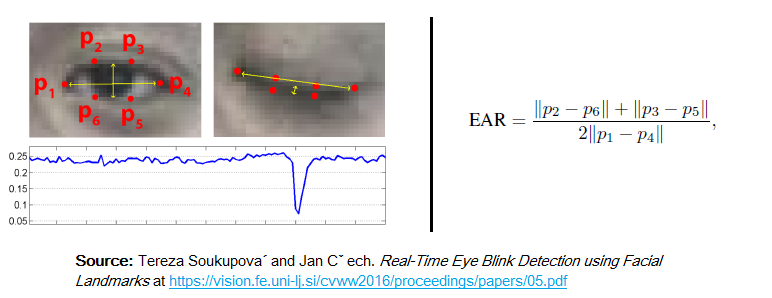
[Dlib](http://dlib.net/" \t "_blank)’s prebuilt model, which is essentially an implementation of [4], not only does a fast face-detection but also allows us to accurately predict 68 2D facial landmarks. Very handy.



Using these predicted landmarks of the face, we can build appropriate features that will further allow us to detect certain actions, like using the eye-aspect-ratio (more on this below) to detect a blink or a wink, using the mouth-aspect-ratio to detect a yawn etc or maybe even a pout. In this project, these actions are programmed as triggers to control the drowsiness. [PyAutoGUI](http://pyautogui.readthedocs.io/" \t "_blank) library was used to move the cursor around.

## Eye-Aspect-Ratio (EAR)

You will see that Eye-Aspect-Ratio [1] is the simplest and the most elegant feature that takes good advantage of the facial landmarks. EAR helps us in detecting blinks [3] and winks etc.

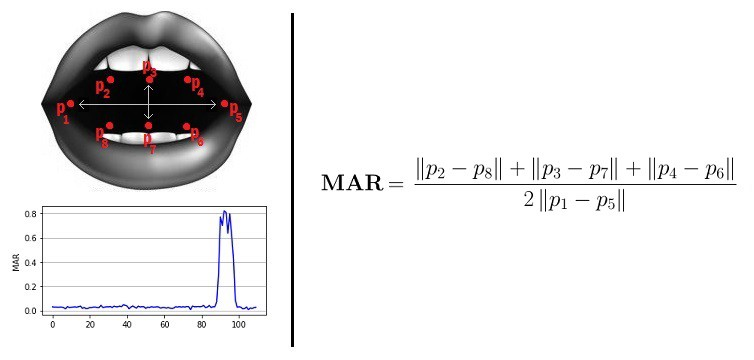


You can see that the EAR value drops whenever the eye closes. We can train a simple classifier to detect the drop. However, a normal if condition works just fine. Something like this:

If EAR<=SOME\_THRESHOLD: EYE\_STATUS = 'CLOSE'

## Mouth-Aspect-Ratio (MAR)

Highly inspired by the EAR feature, I tweaked the formula a little bit to get a metric that can detect opened/closed mouth. Unoriginal but it works.



Similar to EAR, MAR value goes up when the mouth opens. Similar intuitions hold true for this metric as well.

**Prebuilt Model Details**

The model offers two important functions. A detector to detect the face and a predictor to predict the landmarks. The face detector used is made using the classic Histogram of Oriented Gradients (HOG) feature combined with a linear classifier, an image pyramid, and sliding window detection scheme.

The facial landmarks estimator was created by using Dlib’s implementation of the paper: *[One Millisecond Face Alignment with an Ensemble of Regression Trees by Vahid Kazemi and Josephine Sullivan](https://www.semanticscholar.org/paper/One-millisecond-face-alignment-with-an-ensemble-of-Kazemi-Sullivan/1824b1ccace464ba275ccc86619feaa89018c0ad" \t "_blank)*, CVPR 2014. And was trained on the iBUG 300-W face landmark dataset: C. Sagonas, E. Antonakos, G, Tzimiropoulos, S. Zafeiriou, M. Pantic. 300 faces In-the-wild challenge: Database and results. *[Image and Vision Computing (IMAVIS), Special Issue on Facial Landmark Localisation “In-The-Wild”. 2016](https://ibug.doc.ic.ac.uk/resources/facial-point-annotations/" \t "_blank)*.



You can get the trained model file from [http://dlib.net/files,](http://dlib.net/files," \t "_blank) click on **shape\_predictor\_68\_face\_landmarks.dat.bz2**. The model, .dat file has to be in the project folder.

**Note:** The license for the iBUG 300-W dataset excludes commercial use. So you should contact Imperial College London to find out if it’s OK for you to use this model file in a commercial product.