**1)Requirement Gathering Phase**

System Requirements

* [SYSFUN001] The system consists of an Embedded system, Machine Learning Application, and Hardware that has the capability to continuously capture the behavior of the driver, identify five different gestures of the driver behavior and perform certain actions accordingly such as send an alert to the subscriber in case of emergency or control the interfaces of the car.
* [SYSFUN002] The system turns ON the DC motor when the driver’s gesture is Victory symbol.
* [SYSFUN003] The system turns OFF the DC motor when the driver’s gesture is the Palm symbol.
* [SYSFUN004] The system sends a Text with GPS Co-Ordinates and a pre-recorded automated Call during an emergency like a heart attack.
* [SYSFUN005] The system makes the alarm to beep (through speakers) when the driver is drowsy.
* [SYSFUN006] The system has a user-configurable delay to perform the actions and alert the subscriber.
* [SYSFUN007] The system does not perform any action or send any alert in the condition of “No Gesture”.
* [SYSPER001] The system sends text alert within 20sec of occurrence of an emergency event.
* [SYSPER002]  The system sends a call alert within 30sec of occurrence of an emergency event.
* [SYSPER003] The system makes an alarm action within 5 sec of the gesture identified.
* [SYSPER004] The system controls the DC motor within 5sec of the gesture identified.

Embedded System Requirements

* [ESFUN001] The hardware of the embedded system consists of a Pi-camera that continuously captures the behavior of the driver, the speaker for an emergency alarm,  L293D IC to control DC Motor, mouse and keyboard to operate the system.
* [ESFUN002] The software of the embedded system consists of Miniconda and other important libraries like Tensorflow, Opencv, Keras, etc. to program in a high-level language such as python to build Machine Learning Application.
* [ESFUN003] The software of the embedded system consists of a function to extract frames from the capturing video and load it to the Machine Learning Application.
* [ESFUN004] The software of the embedded system consists of a Wi-Fi module and Twilio Interface that sends a text with GPS-Co-Ordinates and an emergency call to the Twilio Cloud.
* [ESPER001] The Embedded system extracts 1 frame per second.
* [ESPER002] The Embedded System loads the image to the Machine Learning Application within 2 sec after extracting the frame from the video.
* [ESPER003] The Embedded System transmits a text alert within 10 sec and calls alert within 15 sec to the Twilio cloud after receiving input from the Machine Learning Application.

Machine Learning Requirements

* [MLFUN001] The Machine Learning System consists of a Trained CNN classifier model that predicts five different gestures of driver behavior.
* [MLPERF001] The Machine Learning System should predict the gestures and behavior of the driver within 5 seconds.

Twilio Cloud Requirements

* [TWFUN001] The Twilio cloud consists of a free account registered with the subscriber number.
* [TLPERF001] The Twilio Cloud sends an alert to the subscriber within 5sec after receiving an alert from the embedded system.

|  |  |  |
| --- | --- | --- |
| **Layers** |  | **Structure** |
| Input Image |  | Shape(number of images)x(image width)x(image height)x(image color depth)  11895 x 200x200 pixels x3(RGB) |
| Conv 1 |  | Convolutional Layer: 16 filters of characteristics with receptive field of 3 x 3 x 3.  The Conv layer output volume had size [200x200x16]. Each of the 200\*200\*16 neurons in this volume was connected to a region of size [3x3x3] in the input volume. |
| Relu 1 |  | ReLu , Activation Function y = max(0,x) |
| Pool 1 |  | Max Pooling 2D with respective field of 2 x 2, strides 2 x 2 and padding 0 x 0  The layer output volume had size [100x100x16]. |
| Conv 2 |  | Convolutional Layer: 36 filters of characteristics with receptive field of 3 x 3 x 3.  The Conv layer output volume had size [100x100x36]. |
| Relu 2 |  | ReLu |
| Pool 2 |  | Max Pooling 2D with respective field of 2 x 2, strides 2 x 2 and padding 0 x 0  The layer output volume had size [50x50x36]. |
| Conv 3 |  | Convolutional Layer: 64 filters of characteristics with receptive field of 3 x 3 x 3.  The Conv layer output volume had size [50x50x64]. |
| Relu 3 |  | ReLu |
| Pool 3 |  | Max Pooling 2D with respective field of 2 x 2, strides 2 x 2 and padding 0 x 0  The Conv layer output volume had size [25x25x64]. |
| Conv 4 |  | Convolutional Layer: 64 filters of characteristics with receptive field of 3 x 3 x 3.  The Conv layer output volume had size [25x25x64]. |
| Relu 4 |  | ReLu |
| Pool 4 |  | Max Pooling 2D with respective field of 2 x 2, strides 2 x 2 and padding 0 x 0  The Conv layer output volume had size [14x14x64]. |
| Conv 5 |  | Convolutional Layer: 64 filters of characteristics with receptive field of 3 x 3 x 3.  The Conv layer output volume had size [14x14x64]. |
| Relu 5 |  | ReLu |
| Pool 5 |  | Max Pooling 2D with respective field of 2 x 2, strides 2 x 2 and padding 0 x 0  The Conv layer output volume had size [7x7x64]. |
| Flatten |  | **flatten** the output of the convolutional layers to create a single long feature vector. |
| Dense 1 |  | 128 neurons, Activation Function = relu  Each neuron receives input from all the neurons in the previous **layer**. The **layer** has a weight matrix W, a bias vector b, and the activations of previous **layer** a. |
| Dense 2 |  | N0\_of\_Classes = 5, Activation Function = SoftMax,  Loss = Categorical\_Crossentropy, optimizer=RMSprop |