## Problem Statement

Imagine you are working as a data scientist at a home electronics company which manufactures state of the art smart televisions. You want to develop a cool feature in the smart-TV that can recognise five different gestures performed by the user which will help users control the TV without using a remote

The gestures are continuously monitored by the webcam mounted on the TV. Each gesture corresponds to a specific command:

* Thumbs up: Increase the volume
* Thumbs down: Decrease the volume
* Left swipe: 'Jump' backwards 10 seconds
* Right swipe: 'Jump' forward 10 seconds
* Stop: Pause the movie

Each video is a sequence of 30 frames (or images)

**Understanding the Dataset**

The training data consists of a few hundred videos categorised into one of the five classes. Each video (typically 2-3 seconds long) is divided into a sequence of 30 frames(images). These videos have been recorded by various people performing one of the five gestures in front of a webcam - similar to what the smart TV will use.

**Model Overview**

| **Model** | **Model Type** | **Number of parameters** | **Augment Data** | **Model Size(in MB)** | **Highest Validation accuracy** | **Corres-ponding Training accuracy** | **Observations** |
| --- | --- | --- | --- | --- | --- | --- | --- |

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| conv\_3d1\_model | Conv3D | | 1,117,061 | | No | NA | 74% | 99% | | Model is over-fitting. Augment data using cropping |
| conv\_3d2\_model | | Conv3D | | 3,638,981 | Yes | 43.8 | 66% | 80% | Model is not over-fitting. Next we will try to reduce the parameter size. Moreover since we see minor oscillations in loss, let's try lowering the learning rate to 0.0002 | | |
| conv\_3d3\_model | | Conv3D | | 1,762,613 | Yes | 21.2 | 78% | 81% | Model has stable results .Also we were able to reduce the parameter size by half. Let's trying adding more layers at the same level of abstractions | | |
| conv\_3d7\_model | | Conv3D | | 504,709 | Yes | 6.15 | 77% | 85% |  | | |
| rnn\_cnn1\_model | | CNN-LSTM | | 1,657,445 | Yes | 21 | 81% | 94% | Model is over-fitting. Let’s try reducing the number of layers in next iteration | | |

## Models with More Data Augmentation

| **Model Name** | **Model Type** | **Number of parameters** | **Augment Data** | **Model Size(in MB)** | **Highest validation accuracy** | **Corresponding Training accuracy** |
| --- | --- | --- | --- | --- | --- | --- |
| conv\_3d10\_model | Conv3D | 3,638,981 | Yes | 43.8 | 86% | 86% |
| conv\_3d11\_model | Conv3D | 1,762,613 | Yes | 21.2 | 78 % | 79 % |

### Mobilenet model is considered as its parameter size is less compared to Inception and Resnet models

| **Model Name** | **Number of parameters** | **Augment Data** | **Model Size(in MB)** | **Highest validation accuracy** | **Corres-ponding Training accuracy** | **Observations** |
| --- | --- | --- | --- | --- | --- | --- |
| rnn\_cnn\_tl\_model | 3,840,453 | Yes | 20.4 | 78% | 93% | For this experiment, Mobilenet layer weights are not trained. Validation accuracy is very poor. So let’s train mobilenet layer’s weights as well |
| rnn\_cnn\_tl2\_model | 3,692,869 | Yes | 42.3 | 95% | 99% | We get a better accuracy on training mobilenet layer’s weights as well. |