**Gesture Recognition**

**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 recognize 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)

**About the Dataset**

The training data consists of a few hundred videos categorized 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. The data contains a 'train' and a 'val' folder with two CSV files for the two folders.

**Findings on different Model**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Experiment Number | Model Description | Number of Parameter | Best Validation Accuracy | Best Training Accuracy | Decision + Explanation |
| 1A | 2D Convolution model with Vanilla RNN as the RNN model | 723173 | 22.73 | 19.97 | * Vanilla RNN has yield to a very low accuracy percentage * No Dropout and Batch Normalization layers are used. |
| 1B | 2D Convolution model with GRU as the RNN model - Dropout after each layer and no Batch Normalization | 1082149 | 31.45 | 24.55 | * Using GRU has increased the training percentage as compared to Vanilla RNN. * Still accuracy percent is low as the model has no Batch Normalization layers |
| 1C | 2D Convolution model with GRU as the RNN model - Dropout after dense layer and added Batch Normalization | 1083621 | 70.19 | 71.35 | * Adding both Batch Normalization and Dropouts after Convolution and Dense layers has increased the accuracy percentage drastically. * Also, validation accuracy is almost same as training accuracy, so there is no overfitting |
| 1D | 2D Convolution model with GRU as the RNN model - Using L1 regularization | 1083621 | 58 | 75 | * Adding L1 Regularization on the Dropout layer has increased the training accuracy. * The Validation accuracy has reduced a lot and it shows the model is over fitted. |
| 1E | 2D Convolution model with GRU as the RNN model - Using L2 regularization | 1083621 | 23 | 72 | * Adding L2 Regularization on the Dropout layer has increased the training accuracy. * The Validation accuracy has drastically reduced a lot and it shows the model is highly over fitted. |
| 1F | 2D Convolution model with LSTM as the RNN model | 523797 | 60.91 | 87.78 | * Using LSTM as RNN layer has increased the accuracy * The model is however over fitted as the validation accuracy is very low as compared to training accuracy |
| 1G | 2D Convolution model with Bidirectional LSTM as the RNN model | 623253 | 72.73 | 99.73 | * Using bi-directional LSTM has drastically improved the accuracy percentage. |
| 2A | **3D Conv model with default arguments and batch\_size=11** | **900805** | **84** | **83** | * **This is our best model as Training and Validation accuracy is almost same, there is almost no overfitting.** |
| 2B | 3D Conv model with image dimesion 100\*100 and batch\_size=20 | 687813 | 79 | 87 | * This model takes lesser number of parameters, but Validation accuracy is not that high. |
| 2C | 3D Conv model without DropOuts | 900805 | 80 | 97 | * Training accuracy is very high * Model is however overfit as Validation accuracy is low |
| 2D | 3D Conv model wit DropOuts and without Batch Normalization | 899077 | 76 | 90 | * Training accuracy is very high * Model is however overfit as Validation accuracy is low |