Deep Learning Course Project- Gesture Recognition

Problem Statement

As a data scientist at a home electronics company which manufactures state of the art smart televisions. We 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 will be continuously monitored by the webcam mounted on the TV.

The gesture will correspond to the following actions

- 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.

DATASET

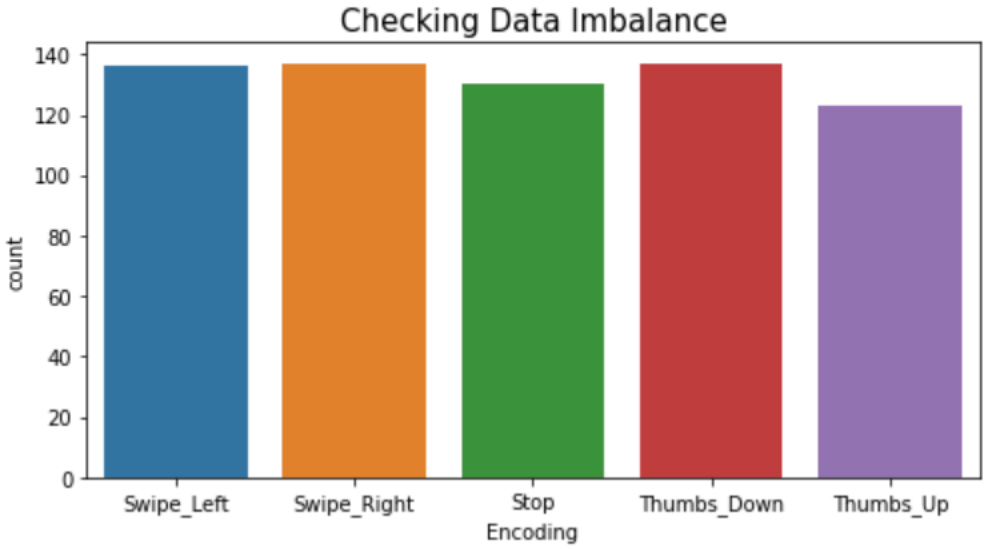
The training data consists of a few hundred videos categorized into one of the five classes. Each video (typically 2-3 seconds long) is divided into a **sequence of 30 frames(images)**. These videos have been recorded by various people performing one of the five gestures in front of a webcam - similar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

Task to be Performed

- We need to train different models using the training data set to predict/classify the actions performed in the video.
- We need to build and train two different types of models using Neural Networks.
- As per the given statement, we need to use **3D Convolution Models (3D CNN)** and **CNN + RNN architecture** in which we pass the images of a video through a CNN which extracts a feature vector for each image, and then pass the sequence of these feature vectors through an RNN.

- The conv2D network will extract a feature vector for each image, and a sequence of these feature vectors is then fed to an RNN-based network. The output of the RNN is a regular softmax.
- **Transfer learning** can be used in the 2D CNN layer rather than training our own CNN .
- In **Transfer learning** we can use the VGC Model or the Mobile Net architecture etc.
- The objective of the Model is to reach/train a Model which has the highest accuracy with the least number of parameters.

Data Imbalance Check



- The data seems to be nicely balanced in the training data set.
- We don't need to perform data balancing step in this data set.

Data Pre-processing

- ***Resizing* and *cropping* of the images.** This was mainly done to ensure that the NN only recognizes the gestures effectively rather than focusing on the other background noise present in the image.
- ***Normalization* of the images.** Normalizing the RGB values of an image can at times be a simple and effective way to get rid of distortions caused by lights and shadows in an image.

MODEL	EXPERIMENT	RESULT	DECISION + EXPLANATION	TRAINABLE PARAMETERS
Conv3D	1 (model)	OOM Error Batch Size - 25	Reduce the batch size and Reduce the number of neurons in Dense layer.(GPU Error)	-
	2 (model)	Training Accuracy : 0.99 Validation Accuracy : 0.70 *(Best weight Accuracy, Epoch: 17/50)*	Change the Architecture of the model as It is Overfitting	3,237,125
	3 (model1)	Training Accuracy : 0.76 Validation Accuracy : 0.74 *(Best weight Accuracy, Epoch: 50/50)* *(Optimizer: SGD)*	This model is able to generalize better than the previous model, but the accuracy is still low. Trying to get good accuracy.	30,152,677
	4 (model2)	Training Accuracy : 0.69 Validation Accuracy : 0.64 *(Best weight Accuracy, Epoch: 34/50)* *(Optimizer: Adam)*	Model is generalizing but the model accuracy has gone further down on changing the optimizer and learning rate. Increasing the Learning Rate, to reach the global minima.	30,152,677
	5 (model3)	Training Accuracy : 0.56 Validation Accuracy : 0.54 *(Best weight Accuracy, Epoch: 48/50)* *(Optimizer: Adam)*	***Reduced filters in the Dense layer, but did not see much performance improvement.***	15,222,501
	6 (model4)	Training Accuracy : 0.89 Validation Accuracy : 0.87 *(Best weight Accuracy, Epoch: 48/50)* *(Optimizer: Adam)*	Reduced Batch size and learning rate, which increased the accuracy as well as the generalizability of the model, also have comparatively smaller number of parameters. Best model with the Conv-3D approach.	3,057,381
CNN + GRU	8 (CNN_RNN)	Training Accuracy : 0.63 Validation Accuracy : 0.57	Initial CNN - GRU model. This model is clearly underfitting, so instead of Our own CNN we will use Transfer learning	3,159,621
	9 (Mobilenet Trainable)	Training Accuracy : 0.93 Validation Accuracy : 0.90	Transfer learning + GRU is giving the best results and it also has the lesser trainable parameters.	2,433,971
	10 (Mobilenet Non Trainable)	Training Accuracy : 0.94 Validation Accuracy : 0.62	This Model is clearly overfitting, so we will not go ahead with this Model	2,100,99

BEST MODEL : Transfer Learning (Mobile Net + GRU

- We can clearly see the Transfer Learning is giving the best results, it also has less parameters and this it can be used to solve our problem.
- We have used the MobileNet architecture since it has least number of trainable parameters and is also giving good accuracy on both training and validation data set.