Below is the list of experiments done to Train a Neural network for **Gesture Recognition**.

There are **5 Gestures** which we want the Neural network to learn –

- 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

As part of this experiment, we have trained 2 different model types –

- Convolution 3D
- CNN(2D) + RNN

Below table shows all the experiments done on both the model types:

Experiment Number	Model	Result	Decision + Explanation
Convolution 3D			
1	Conv3D	Throws Generator error	Crop the images correctly, try to overfit on less amount of data
2	Conv3D	Model not trainable as a lot of parameters	Reduce the size of the image/Reduce the number of layers

3	Conv3D
	Images used = 18, 100x100
	Batch size = **64**
	Layers = 3 conv3d layers, 2 Dense
	Activation = Relu
	Epoch = 30
	Optimiser = **SGD**
		OOM when allocating tensor with shape[64,64,18,100,100]	Throws Memory error with 64 batch size

4	Conv3D
	Images used = 15, 84x84
	Batch size = **16**
	Layers = 3 conv3d layers, 2 Dense
	Activation = Relu
	Epoch = 30
	Optimiser = **SGD**
		Train Accuracy: 0.73
		Validation Accuracy: 0.60
			Not Overfitting but also not great accuracy. We can probably do better.

5	Conv3D Images used = 18, 100x100	InternalError: Could not synchronize CUDA stream: CUDA_ERROR_ILLEGAL_ADDRESS: an illegal memory access was encountered	

6	Conv3D
	Images used = 15, 84x84
	Batch size = **32**
	Layers = 3 conv3d, 2 Dense
	Activation = Relu
	Epoch = 30
	Optimiser = **RMSProp**
		Train Accuracy: 0.8703
		Validation Accuracy: 0.56
		Overfitting
			Lets try smaller batch size and images in next run. Also, lets try experimenting with activation function.

7	Conv3D Batch size:**8** Image used: 8, 42 x 42 Epochs:10 Activation: **elu**	Train accuracy :0.71 Val. Accuracy : 0.66	Reducing Batch size and image size and using ELU activation has controlled overfitting but accuracy is still not very good.

	Dropout:0.25 (throughout)		Lets try different activation function.
8	Conv3D Batch size:**16** Image used: 12, 72 x 72 Epochs:15 Activation: **tanh** Dropout:0.25 (throughout)	Train accuracy :0.89 Val. Accuracy : 0.70 **Overfitting**	Using **tanh** has improved accuracy but still overfitting. Lets trying adding another layer and increasing the images and size.
9	Conv3D Batch size:**10** Image used: 20, 120 x 120 Epochs:20 Layers: Added 1 more layer of Conv3D Hyperparameters: Used Batch normalisation in Dense layers as well this time. Activation: **RELU** Dropout:0.25 (only in Dense layer)	After 20 epochs: Train accuracy :0.9050 Val. Accuracy : 0.7900 **Best accuracy:** Epoch=13 Train accuracy :0.8627 Val. Accuracy : 0.8300	We have achieved good accuracy in this model. In 13th epoch we can see that both train and validation accuracies are close to each other. This is an Optimal model and is submitted for evaluation.
CNN + RNN			
10	**Conv2D+GRU** Pre-trained model = **VGG16** Images used = 15 , 84x84 Batch size = 16 Layers = 5 (Timedist(Cnn2d),, GRU, GRU, Dense, Dense) Activation = Relu and softmax Epoch = 30	Train Accuracy: 0.1862 Validation Accuracy: 0.1700	Not a good model as accuracy is very less.
11	**Conv2D+GRU** Pre-trained model = **VGG16** Images used = 15 , 84x84 Trainable weights = False Batch size = 32 () Layers = 6 (Timedist(Cnn2d),, GRU, GRU, GRU, Dense, Dense) Activation = Relu and softmax Epoch = 30 Dropout layer = Removed	Train Accuracy: 0.2620 Validation Accuracy: 0.27 ()	Adding additional GRU layer and removing Dropout has improved accuracy a little 16 batch size has offered lower generalization Dropout removed to check if these layers have generalized the model too much)
12	**Conv2D+GRU** **Using a different Pre-trained model to check if we can get better results:** Pre-trained model = **InceptionV3** Images used = 15 , 84x84 Trainable weights = False Batch size = 32 (16 batch size has offered lower generalization) Layers = 5 (Timedist(Cnn2d),, GRU, GRU, Dense, Dense) Activation = Relu and softmax (No change) Epoch = 30 Dropout layer = added a dropout layer **Using a different Optimizer: RMSProp()**	Train Accuracy: 0.7134 Validation Accuracy: 0.4700 Best result: Epoch = 23 Train Accuracy: 0.7255 Validation Accuracy: 0.5300	using RMSProp and Inceptionv3 model has given better results in terms of accuracy than VGG16 but is overfitting
13	**Conv2D+GRU** Pre-trained model = **InceptionV3** Images used = 15 , 84x84 Trainable weights = False Batch size = 32 (16 batch size has offered lower generalization) Layers = 5 (Timedist(Cnn2d),, GRU, GRU, Dense, Dense)	Train Accuracy: 0.5400 Validation Accuracy: 0.2600	using SGD and removing the Dropout layer has reduced accuracy

	Activation = Relu and softmax (No change)
	Epoch = 30
	Dropout layer = **removed** dropout layer
	
	Using a different Optimizer: SGD()		

14	**Conv2D+GRU** Pre-trained model = **InceptionV3** Images used = 12, 75x75 (min for InceptionV3 module) Trainable weights = False Batch size = 16 Layers = 5 (Timedist(Cnn2d),, GRU, GRU, Dense, Dense) Activation = Relu and softmax (No change) Epoch = 30 Dropout layer = 0.25 **Using a different Optimizer: RMSProp()**	Train Accuracy: 0.9201 Validation Accuracy: 0.4600	Model is Overfitting. Lets try a different Pre-trained model.
15	**Conv2D+LSTM** Pre-trained model = **Mobilenet** Images used = 20, 120x120 Trainable weights = **True** Batch size = 10 Layers = 4 (Timedist(Cnn2d), LSTM, Dense, Dense) Activation = Relu and softmax (No change) Epoch = 20 Dropout layer = 0.25 (only on Dense layer) **Using a different Optimizer: Adam()**	Train Accuracy: 0.9864 Validation Accuracy: 0.8800	Model is Overfitting. We have definitely improved the accuracy of Validation data set but the model is till overfitting.