## **Project- Gesture Recognition**

* **Siddhant Sambit**
* **Tina Ghosh**

# 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.

* Thumbs up :  Increase the volume.
* Thumbs down : Decrease the volume.
* Left swipe : 'Jump' backward.
* Right swipe : 'Jump' forward.
* Stop : Pause the movie.

# Observations

* It was observed that as the Number of trainable parameters increase, the model takes much more time for training.
* A large batch size can throw *GPU Out of memory error,* and thus here we had to play around with the batch size till 40.When we have tried with hyperparameter frame\_to\_image =20 and batch\_size = 50 and image size = 160\*160 we got ResourceExhaustedError (OOM when allocating tensor Shape [50,16,20,160,160] )
* Increasing the batch size greatly reduces the training time but this also has a negative impact on the model accuracy. If we want the model to be ready in a shorter time span, choose larger batch size else we should choose lower batch size if we want the model to be more accurate.
* If we execute the model with more epoch there is a tendency of overfitting the model. So we have standardized epoch size = 15 for all model.
* For detailed information on the Observations and Matrix, please refer below chart.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **MODEL** | **EXPERIMENT** | **RESULT** | **DECISION + EXPLANATION** | **PARAMETERS** |
| **Conv3D** | **Finding the limit of hyperParameters/ batch\_size** | **OOM Error** | ***Model Build with below HyperParameter Combination :***  ***image height = 160***  ***image width = 160***  ***frames\_to\_sample=20,***  ***batch\_size=50,***  ***num\_epochs=3***  ResourceExhaustedError: OOM when allocating tensor with shape[50,16,20,160,160] and type float on /job:localhost/replica:0/task:0/device:GPU:0 by allocator GPU\_0\_bfc  So we have considered to build the base model using batch\_size = 40,frame\_to sample = 30,image\_height =160,image\_width=160] | 17,36,389 |
| **Model -1 (Base Model)** | **Training Accuracy : 0.98**  **Validation Accuracy : 0.93** | ***Model Build with below***  ***HyperParameter Combination :***  ***image height = 160***  ***image width = 160***  ***frames\_to\_sample=20,***  ***batch\_size=40,***  ***num\_epochs=15***  ***filtersize=(3,3,3),***  ***dense\_neurons=64,***  ***dropout=0.25***  **Model is Overfitting .**  **Let’s try with modify drop out to 0.5 and increase neuron density and keeping batch-size =20 .** | **11,17,061** |
| **Model -2** | **Training Accuracy : 0.83**  **Validation Accuracy : 0.87**  ***(Epoch considered – 7 to 14)*** | ***Model Build with below***  ***HyperParameter Combination :***  ***image height = 160***  ***image width = 160***  ***frames\_to\_sample=20,***  ***batch\_size=20,***  ***num\_epochs=15***  ***filtersize=(3,3,3),***  ***dense\_neurons=256,***  ***dropout=0.50***  **Val\_loss didn’t improve from .35713 .**  **Let’s lower the learning rate to 0.0002 reducing filter Size to (2,2,2, and batch\_size = 30 ).** | **36,38,981** |
| **Model -3** | **Training Accuracy : 0.86**  **Validation Accuracy : 0.81**  ***(C***onsidering ***Epoch:14 and epoch 15 )*** | ***Model Build with below***  ***HyperParameter Combination :***  ***image height = 120***  ***image width = 120***  ***frames\_to\_sample=20,***  ***batch\_size=30,***  ***num\_epochs=15***  ***filtersize=(2,2,2),***  ***dense\_neurons=256,***  ***dropout=0.50***  ***Optimiser learning rate =.0002***  **Overfitting has reduced but accuracy hasn't improved. *Let's trying adding more layers and batch-size =20*** | **17,62,613** |
| **Model -4** | **Training Accuracy : 0.85**  **Validation Accuracy : 0.93** | ***Model Build with below***  ***HyperParameter Combination :***  ***image height = 120***  ***image width = 120***  ***frames\_to\_sample=20,***  ***batch\_size=20,***  ***num\_epochs=15***  ***filtersize=(2,2,2),***  ***dense\_neurons=256,***  ***dropout=0.50***  ***There is a bit performance improvement. But the model is taking almost double time than earlier to train because of extra layer So it impact the performance.***  ***Let's try adding dropout at convolution Layers.*** | **25,56,533** |
| **Model -5** | **Training Accuracy : 0.69**  **Validation Accuracy : 0.25** | ***Model Build with below***  ***HyperParameter Combination :***  ***image height = 120***  ***image width = 120***  ***frames\_to\_sample=20,***  ***batch\_size=20,***  ***num\_epochs=15***  ***filtersize=(3,3,3),***  ***dense\_neurons=256,***  ***dropout=0.50***  **Overfitting Increase, adding dropouts has further reduced validation accuracy drastically as model doesn’t seems to be generalized well.**    **Let's try to reduce the parameters in number of neurons , image size and drop out with no extra convolution layer and no filter.** | **25,56,533** |
| **Model -6** | **Training Accuracy : 0.86 Validation Accuracy : 0.99** | ***Model Build with below***  ***HyperParameter Combination :***  ***image height = 100***  ***image width = 100***  ***frames\_to\_sample=20,***  ***batch\_size=20,***  ***num\_epochs=15***  ***dense\_neurons=128,***  ***dropout=0.25***  **With less number of trainable parameter less memory is used with so it is also overfitting. Validation accuracy of 99.25% .** | **6,96,645** |
| **CNN+LSTM** | **Model -7** | **Training Accuracy : 0.84**  **Validation Accuracy : 0.89** | **CNN - LSTM model - we get a best validation accuracy of 89%. Build the model with CNN2\_RNN**  **Here We are considering epoch 14 as in epoch 15 we are seeing again overfitting scenario is arising . Also the val\_loss is 0.311 which is pretty much ok to considered.** | **1,657,445** |

After doing all the experiments, we finalized **Model 7– CNN+LSTM**, which performed well.

**Reason:**

* (Training Accuracy: 85%, Validation Accuracy: 89%).
* Number of Parameters (1,657,445) less according to other models’ performance with no overfitting.
* Learning rate gradually decreasing after some Epochs.
* model-00014-0.42250-0.85294-0.31116-0.89140.h5 is the final model for your reference.