**Experiment Summary**

# 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 recognise 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' backwards 10 seconds.

• Right swipe : 'Jump' forward 10 seconds.

• Stop : Pause the movie.

Below is the summary of different experiments conducted to derive a suitable model :-

Common Hyper parameters used in model building :-

* **ReduceLROnPlateau**(monitor='val\_loss', factor=0.2, verbose=1, patience=4)
* **ModelCheckpoint**(monitor='val\_loss', verbose=1, save\_best\_only=True, save\_weights\_only=False, mode='auto', save\_freq='epoch')
* **EarlyStopping**( patience=6, min\_delta=0.001, monitor="val\_loss", restore\_best\_weights=True)

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment  Number** | **Model** | **Result** | **Decision + Explanation** |
| 1 | Conv 3D Model with 16, 32, 64, 128 filters, followed by max pool 3D  + 128 dense nodes  + 64 dense nodes  + Dropouts   * image size 160 by 160 * sample frame – 30 * epoch – 20 * batch size - 20 | **Training Accuracy:** 0.3889  **Validation Accuracy:** 0.4625 | * Not much learning * Overfitting   Next steps   * Will increase complexity of model in next experiment |
| 2 | Conv 3D Model with 16,16, 32,32, 64, 128 filters, followed by max pool 3D  + 128, 128 dense nodes   * image size 160 by 160 * sample frame – 30 * epoch – 50 * batch size - 20 | **Training Accuracy:** 0.9847  **Validation Accuracy:** 0.7875 | * Train accuracy has improved * Overfitting is present   Next steps   * Reduce number of consecutive conv2D layers * Add more dropouts |
| 3 | Conv 3D Model with 16,16, 32, 64, 128 filters, followed by max pool 3D  + 128 dense nodes   * image size 160 by 160 * sample frame – 20 * epoch – 50 * batch size - 20 | **Training Accuracy:** 0.9407  **Validation Accuracy:** 0.8375 | * Train accuracy is good * Validation accuracy has improved as well * Overfitting has reduced   Next steps   * Try LSTM |
| 4 | Time distributed Conv 2D Model with 16, 32, 64, 128, 256 filters, each followed by max pool 2D  + 128 LSTM  + 128 dense nodes   * image size 160 by 160 * sample frame – 30 * epoch – 20 * batch size - 20 | **Training Accuracy:** 0.9241  **Validation Accuracy:** 0.6375 | * Huge Overfitting   Next steps   * Need to add more hyper parameters |
| 5 | Time distributed Conv 2D Model with 16, 32, 64, 128 filters, each followed by max pool 2D  + 128 LSTM  + 128, 128 dense nodes   * image size 160 by 160 * sample frame – 30 * epoch – 20 * batch size - 20 | **Training Accuracy:** 0.9463  **Validation Accuracy:** 0.5750 | * Training accuracy has improved * Still huge overfitting   Next steps   * Will try other layer combination |
| 6 | Time distributed Conv 2D Model with 16, 32, 64, 128 filters, followed by max pool 2D  + 128 GRU  + 128 dense nodes   * image size 160 by 160 * sample frame – 30 * epoch – 20 * batch size - 20 | **Training Accuracy:** 0.9037  **Validation Accuracy:** 0.5750 | * Huge Overfitting   Next steps   * Need to add more hyper parameters |
| 7 | Time distributed Conv 2D Model with 16, 32, 64 filters, followed by max pool 2D  + 128 GRU  + 128 dense nodes   * image size 160 by 160 * sample frame – 30 * epoch – 20 * batch size - 20 | **Training Accuracy:** 0.9907  **Validation Accuracy:** 0.7125 | * Training accuracy has improved * Overfitting is still there   Next steps   * Will try other layers in RNN |
| 8 | Time distributed Conv 2D with 16, 32, 64 filters, followed by max pool 2D  + 128 ConvLSTM2D  + GlobalAveragePooling2D  + 64 dense nodes   * image size 160 by 160 * sample frame – 30 * epoch – 50 * batch size - 20 | **Training Accuracy:** 0.8566  **Validation Accuracy:** 0.7875 | * Train accuracy is good * Some overfitting is observed   Next steps   * Try to tune hyper parameters further |
| 9 | Conv3D 32, 64 filters followed by max pool 3D  + GlobalAveragePooling3D  + 128 dense nodes   * image size 160 by 160 * sample frame – 30 * epoch – 50 * batch size - 20 | **Training Accuracy:** 0.8031  **Validation Accuracy:** 0.7875 | * Training accuracy is good enough * Overfitting has also reduced   Nest steps   * Will try to improve accuracy |
| 11 | TimeDistributed MobileNet  + BatchNormalization  + GRU  + 64 dense nodes   * image size 160 by 160 * sample frame – 30 * epoch – 50 * batch size - 5 | **Training Accuracy:** 0.9407  **Validation Accuracy:** 0.9000 | * Training and validation accuracy is around 90% |