Problem statement:

A home electronics company which manufactures state of the art smart televisions. It wants to develop a cool feature in the smart-TV that can recognise five different gestures performed by the user which will help users to control the TV. 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).

Google drive link to download the best model:

As the model size is 80MB, link is shared with google drive of the best model.

https://drive.google.com/file/d/1vtTk_Tot9DGtcNcnUi2WV5TsXCcEvgDQ/view?usp=sharing

Approach:

To identify the gesture, a number of experiments were conducted. Models were trained with different image sizes (for ex: 120x120, 150x150) with different number of epochs and varying batch sizes. Common architectures used during training the model were:

1. Conv3d architecture
    a) Architecture-1 (with Adam as optimizer)
    b) Architecture-1 variant with SGD as an optimizer
    c) Architecture-2 (introduce dropout with few layers)
    d) Architecture-3 (With different layers and architecture)

2. LSTM architecture
3. GPU architecture.
4. **Conv3d architecture** Data generator with affine transformation.

A number of models were built by using the above architectures. Results and outcomes are listed below with Decision and explanation.

At first conv3d architecture was tested with different number of hyper-parameters to find the best model and then the same layers and hyper parameters were taken to experiment to LSTM/GPU, and other architectures to validate the model.

Outcome:

Model performance with 120x120 image size tends to perform better compared to 150x150 and conv3D model performed better compared to LSTM/GPU.

Model Training:

Exp. Nb.	Input size	Result (Best accuracy is listed below)	Decision/Explanation
Architecture-1: - 5 layer each with 8, 16, 32, 64, 128 neurons were created followed by with two dense layers 1000, 500 each with drop out of 0.5 - In initial 3 runs, model was trained with varying number of batch sizes, epochs to see the better fit of hyperparameters.			
Model-1 **Conv3D**	**Conv3D** Image size: 120 Batch size: 40 epochs: 15	Accuracy: Training: 87.89 Validation: 28.83	**Initial run:** to see if architecture works. Model should overfit with less number of epochs. Initial run is to prove our architecture and model validation.

Model-2 Hyper-parameter tuning (change epochs) Change in hyperparameter Batch-size (40)	**Conv3D:** Image size: 120 Batch size: 40 epochs: 25	Accuracy: Training: 88.93 Validation: 50.00	As the model was trained with more number of epochs model should perform better. Training, validation accuracy went up and validation loss improved as well.
Model-3 **Conv3D** Change in hyperparameter Batch-size (30)	Conv3D: Image size: 120 Batch size: 30 epochs: 25	Accuracy: Training: 66.67 Validation: 67.50	**Compared to model 2:** When trained with batch size: 25, Training accuracy is down. With few epochs (i.e., 25) training accuracy should be high. **We should continue with model-2 as our base model and try to train with more number of epochs:**

Model-5 **Conv3D** Change in hyperparameter Batch-size (50)	Conv3D: Image size: 120 Batch size: 50 epochs: 25	Accuracy: Training: 74.18 Validation: 21.33 Validation loss: 3.6693	**Compared to model 2:** Training/validation accuracy is less. So, as a result we can conclude that batch-size: 40 tends to perform better when trained with epochs: 25 **We should continue with model-2 as our base model and try to train with more number of epochs.**
Model-7 **Conv3D** Change in hyperparameter Epochs (50)	Conv3D: Image size: 120 Batch size: 40 epochs: 50	Accuracy: Training: 88.58 Validation: 91.67 Validation loss: 0.3487	**Compared to model 2:** As the model is trained with more number of epochs: Training/validation accuracy is high. Validation loss is very less.

			We should continue with Model-7 as our base model.
<u>Approach:</u> Image cropping (150x150) - Conduct the same exercise with same set of hyper parameters			
Model-4 **Conv3D**	Conv3D: Image size: 150 Batch size: 40 epochs: 25	Accuracy: Training: 89.62 Validation: 50.00 Validation loss: 1.9805	**Compared to model 2:** When model is trained with image size: 150x150 result is comparable. Training/validation accuracy and validation loss are in range with image size:120x120. Model should be trained with more number of epochs and compared with model-7

Model-6 **Conv3D**	Conv3D: Image size: 150 Batch size: 50 epochs: 25	Resource exhaustion (OOM)	
Model-8 **Conv3D** hyperparameter turned (epochs:50)	Conv3D: Image size: 150 Batch size: 40 epochs: 50	Accuracy/Loss: Training: 84.08 Validation: 85.00 **Validation** loss: 0.3635	**Compared to model-7:** When trained with same set of hyper-parameters results are identical to image size (120x120). In case of image-size: 120x120 accuracy training/validation is higher and validation loss is less. **We continue with model-7 as out base model.**

Architecture-2:

- Introduce dropout(0.25) at few layers compared to architecture-1.
- Number of layers remain same

Model-9: **Conv3D**	Conv3D: Image size: 120 Batch size: 40 epochs: 25	Accuracy/Loss: Training: 70.93 Validation: 25.00 Validation loss: 11.4840	**Compared to model-2:** When drop-out layer was introduced after few layers' accuracy went down. Possible reasons could be the feature loss and neuron layers are not dense enough. Architecture-1 is better compared to architecture-2 when trained with same hyper parameters.

Architecture-3:

- Train different conv3D architecture and compare with architecture-1

Model-10 **Conv3D**	Conv3D: Image size: 120 Batch size: 40 epochs: 25	Accuracy/Loss: Training: 89.27 Validation: 45.00 Validation loss: 1.9392	**Compared to model 2:** When trained with same set of hyper parameters, results are in range. We should try to train the model with higher number of epochs (50) and see the output.
Model-10-variant **Conv3D** Hyperparameter tuned (epochs:50)	Conv3D: Image size: 120 Batch size: 40 epochs: 50	Accuracy/Loss: Training: 89.27 Validation: 81.67 Validation loss: 0.4952	**Compared to model 7:** When model was trained with higher number of epochs. Model seems to perform better but compared to model-2 accuracy training/validation is less and validation loss is slightly higher.

			As a result, we continue with Model -7 as our base model.
Architecture-4: - Variant of architecture 1. Optimizers used here is SGD			
Model-11 **Conv3D + SGD optimizer**	Conv3D: (SGD) Image size: 120 Batch size: 40 epochs: 50	Accuracy/Loss: Training: 81.31 Validation: 85.00 Validation loss: 0.5227	**Compared to model 7:** When trained with same set of hyper parameters and layers. Results are comparable with Model 7. In model 7 accuracy tends to be slightly better and validation loss is less. Based on accuracy and validation loss we continue with model-7 **As a result, we continue with Model -7 as our base model.**

Model-12 **LSTM**	**LSTM**: Image size: 120 Batch size: 40 epochs: 50	Accuracy/Loss: Training: 88.93 Validation: 75.00 Validation loss: 0.6490	**Compared to model 7:** Model seems to be overfitting with higher validation loss. Model can be experimented with several different architectures to reduce the overfitting. With current set of layers & hyper parameters, model-7 seems to be better. **As a result, we continue with Model -7 as our base model.**
Model-13	**GPU**: Image size: 120 Batch size: 40 epochs: 50	Accuracy/Loss: Training: 97.58 Validation: 85.00 Validation loss: 0.5572	**Compared to model 7 & model 12:** Model seems to be overfitting. Model can be experimented with different set of hyperparameters and

			input batch sizes to reduce the overfitting. With current set of layers & hyper parameters, model-7 seems to be better. **As a result, we continue with Model -7 as our base model.**

Architecture-7: (Conv3D) – with data augmentation.

- Conv3d architecture was tried with different generator and cv2.wrapaffine() transformation was used to try out the model

- Same architecture-1 was used with different generator to train the model.
- As with conv3d architecture it was noticed that model performs better with batch-size:40, and higher number of epochs (50). Model can be tried out same number of hyperparameters and results can be evaluated.

Model-14 **Conv-3D + Data augmentation**	**Conv3D**: (Data augmentation) Image size: 120 Batch size: 40 epochs: 50	Accuracy/Loss: Training.: 84.43 Validation: 83.33 Validation loss:0.4966	**Compared to model 7:** Training, validation accuracy and validation loss is in range with compared to model-7 even with different generators outputs are nearly the same. **However, we continue with model-7 as our base model with given training data.**
Final Model:			
Final Model: **Conv 3D**	**Conv3D**: Image size: 120 Batch size: 40 epochs: 50	Accuracy: **Training: 88.58** **Validation: 91.67** **Validation loss: 0.**3487	Conv3d architecture tends to perform better compared with LSTP & GPU architecture.