**Problem Statement**

Imagine you are working as a data scientist at a home electronics company which manufactures state of the art smart televisions. You 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.

The gestures are continuously monitored by the webcam mounted on the TV. Each gesture corresponds to a specific command:

| **Gesture** | **Gesture Interpretation** |
| --- | --- |
| 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).

### Objectives:

1. **Generator**: The generator should be able to take a batch of videos as input without any error. Steps like cropping, resizing and normalization should be performed successfully.
2. **Model**: Develop a model that is able to train without any errors which will be judged on the total number of parameters (as the inference(prediction) time should be less) and the accuracy achieved. As suggested by Snehansu, start training on a small amount of data and then proceed further.
3. **Write up**: This should contain the detailed procedure followed in choosing the final model. The write up should start with the reason for choosing the base model, then highlight the reasons and metrics taken into consideration to modify and experiment to arrive at the final model.

### Understanding Dataset: The training data consists of a few hundred videos categorised 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 - like what the smart TV will use.  The data is in a zip file. The zip file contains a 'train' and a 'val' folder with two CSV files for the two folders. These folders are in turn divided into subfolders where each subfolder represents a video of a particular gesture. Each subfolder, i.e. a video, contains 30 frames (or images). Note that all images in a particular video subfolder have the same dimensions but different videos may have different dimensions. Specifically, videos have two types of dimensions - either 360x360 or 120x160 (depending on the webcam used to record the videos). Each row of the CSV file represents one video and contains three main pieces of information - the name of the subfolder containing the 30 images of the video, the name of the gesture and the numeric label (between 0-4) of the video.

### Result:

You can draw inspiration from the concepts taught in the Industry demo in CNNs to experiment with the data and different architectures.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No.** | **Model** | **Hyper Parameters** | **Result** | **Decision + Explanation** |
| 1 | Conv3D | Batch size: 100, Frames used: 30, image: 160x160, epoch:1 | Throws ResourceExhaustedError | Tried with multiple batch size, frames, and image size |
| 2 | Conv3D | Batch size: 40, Frames used: 30, image: 160x160, epoch:1 | Model was executed fine but needed more tuning | Need to experiment with parameters |
| 3 | Conv3D | Batch size: 30, Frames used: 20, image: 100x100, epoch:2 | Execution time: 1m  Val\_loss: 1.94 | Needs more tuning |
| 4 | Conv3D | Batch size: 20, Frames used: 30, image: 100x100, epoch:2 | Execution time: 2m  Val\_loss: 1.88 | Time increased by double because of number of frames increase |
| 5 | Conv3D | Batch size: 15, Frames used: 20, image: 160x160, epoch:2 | Execution time: 2m  Val\_loss: 1.74 | Same time even though frames decreased, because of increase in the size of image |
| 6 | Conv3D | Batch size: 15, Frames used: 30, image: 120x120, epoch:2 | Execution time: 1m  Val\_loss: 1.7 | Val\_los improved |
| 7 | Conv3D | Batch size: 15, Frames used: 15, image: 120x120, epoch:2 | Execution time: 1m  Val\_loss: 1.6 | Val\_los improved. The most impact on execution time is by image size, followed by frames and the batch size |
| 8 | Conv3D | Batch size: 40, Frames used: 20, image: 160x160, epoch:15, dropout:0.25 | Execution time: 8m  Val\_loss: 1.5  Total Params: 1117061 | Early stopping as there was no improvement in the model accuracy. Underfitting model |
| 9 | Conv3D | Batch size: 40, Frames used: 20 image: 160x160, epoch:25, dropuout:0.3, dense\_neurons: 256 | Execution time: 8m  Val\_loss: 1.5  Training accuracy: 80%  Validation Acc: 26%  Total Params: 1117061 | The model needs more tuning |
| 10 | Conv3D | Batch size: 40, Frames used: 20 image: 160x160, epoch:25, dropuout:0.3, dense\_neurons: 256, filter: (3,3,3) | Execution time: 8m  Training accuracy: 78%  Validation Acc: 31%  Total Params: 3637477 | Slight improvement but the model needs more tuning |
| 11 | Conv3D | Batch size: 40, Frames used: 20 image: 160x160, epoch:25, dropuout:0.3, dense\_neurons: 256, filter: (2,2,2) | Execution time: 7m  Training accuracy: 88%  Validation Acc: 16%  Total Params: 3433781 | Execution time and params decreased but accuracy downgraded |
| 12 | Conv3D | Batch size: 30, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (2,2,2) | Execution time: 5m  Training accuracy: 53%  Validation Acc: 21%  Total Params: 1762613 | The gap between training and validation decreased a bit |
| 13 | Conv3D | Batch size: 30, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (2,2,2), LR: 0.0002 | Execution time: 5m  Training accuracy: 55%  Validation Acc: 19%  Total Params: 1762613 | No increase in the accuracy, adding more layers |
| **Adding More layers** | | | | |
| 14 | Conv3D | Batch size: 20, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (3,3,3) | Execution time: 12m  Training accuracy: 40%  Validation Acc: 36%  Total Params: 2556533 | No Early stopping. There was considerable improvement in validation accuracy |
| 15 | Conv3D | Batch size: 30, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (2,2,2) | Execution time: 5m  Training accuracy: 47%  Validation Acc: 22%  Total Params: 1937893 | Early stopping. Highly overfitted model |
| 16 | Conv3D | Model 14 with dense:128 | Execution time: 13m  Training accuracy: 33%  Validation Acc: 40%  Total Params: 1702645 | Needs further improvement. Trying CNN LSTM |
| 17 | CNN + LSTM | Batch size: 20, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 128, lstm: 128 | Execution time: 12m  Training accuracy: 51%  Validation Acc: 53%  Total Params: 1657445 | We see that there is improve in the validation accuracy |
| **Adding Augmentation and running above models again** | | | | |
| 18 | Conv3D | Batch size: 20, Frames used: 20 image: 160x160, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (3,3,3) | Execution time: 19m  Training accuracy: 66%  Validation Acc: 22%  Total Params: 3638981 | Highly overfitted |
| 19 | Conv3D | Batch size: 30, Frames used: 16 image: 160x160, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (2,2,2) | Execution time: 12m  Training accuracy: 63%  Validation Acc: 25%  Total Params: 3433781 | Early stopping, Highly overfitted |
| 20 | Conv3D | Batch size: 30, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (2,2,2) | Execution time: 11m  Training accuracy: 51%  Validation Acc: 26%  Total Params: 1762613 | Parameters and time decreased, no better results |
| **Adding more layers + Augmentation** | | | | |
| 21 | Conv3D | Batch size: 30, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (2,2,2) | Execution time: 11m  Training accuracy: 51%  Validation Acc: 26%  Total Params: 1762613 | Even after adding more layers, no improvement observed |
| 22 | Conv3D | Batch size: 20, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (3,3,3) | Execution time: 11m  Training accuracy: 39.6%  Validation Acc: 18%  Total Params: 2556533 | No improvement |
| **Adding more layers + Augmentation + Dropouts in between layers** | | | | |
| 23 | Conv3D | Batch size: 20, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 256, filter: (3,3,3) | Execution time: 11m  Training accuracy: 31.6%  Validation Acc: 23%  Total Params: 2556533 | Early stopping, overfitting result improved a bit |
| **Basic layers + Augmentation** | | | | |
| 24 | Conv3d with different filter size | Batch size: 20, Frames used: 16 image: 120x120, epoch:25, dropuout:0.25, dense\_neurons: 128 | Execution time: 11m  Training accuracy: 64 %  Validation Acc: 22%  Total Params: 909637 | Badly Overfitting |
| **No early Stopping in train model** | | | | |
| 25 | Conv3d with different filter size | Batch size: 20, Frames used: 16 image: 120x120, epoch:25, dropuout:0.25, dense\_neurons: 64 | Execution time: 24m  Training accuracy: 57%  Validation Acc: 49%  Total Params: 504309 | Highly overfitting |
| 26 | CNN + LSTM with GRU | Batch size: 20, Frames used: 16 image: 120x120, epoch:25, dropuout:0.25, dense\_neurons: 128, lstm:128 | Execution time: 24m  Training accuracy: 82.6%  Validation Acc: 54%  Total Params: 2573925 | The validation accuracy has increased further but there is still overfitting issue. |
| **Final Model** | **CNN + LSTM** | **Batch size: 20, Frames used: 16 image: 120x120, epoch:25, dropuout:0.5, dense\_neurons: 128, lstm: 128** | **Execution time: 12m**  **Training accuracy: 51%**  **Validation Acc: 53%**  **Total Params: 1657445** | **Of all the models, this model has the best result and the number of parameters is also low.** |