**Gesture Recognition**

**Requirement:**

We want to develop a 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. The gestures are continuously monitored by the webcam mounted on the TV. Each gesture corresponds to a specific command as shown below:

| **Gesture** | **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 |

**Data:**

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). All the images in a particular video subfolder have the same dimensions but different videos may have different dimensions.

The task is to train a model on the 'train' folder which performs well on the 'val' folder as well.

**Results:**

We have broadly used two different neural network architectures here to train the models:

* Conv3D Model
* Conv3D+LSTM

We started with a base Conv3D model and experimented with the batch size and image parameters to select an optimal batch size for the model building and training. The selected batch size range is 20-40.

We have used a generator which takes a batch of videos as input and performs steps like cropping, resizing and normalization on the frame images.

Following table elaborates the different models tuned for accuracy and fit and the corresponding outcomes:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Model Type: Name** | **No. Parameters** | **Result** | **Decision + Explanation** |
|  | Conv3D: conv\_3d1 | 1,117,061 | High no. of training parameters and high memory utilization.  Batch size of 60 with 30 frames of image size 160x160 throws an error: ResourceExhaustedError  Batch size 40, Epochs 15, Image 160x160:  Train Accuracy: 0.92  Val Accuracy: 0.28 | This was the basic Conv3D model and no data augmentation was used. Model is clearly overfitting and the validation accuracy is very low. We need to update filter size and try to overfit on less amount of data. |
|  | Conv3D: conv\_3d2 | 3,638,981 | The val loss is high and accuracy is low which seems to improve with the no. of epochs.  Filter size of (3,3,3), 160x160 Image, Batch size: 20:  Train Accuracy: 0.79  Val Accuracy: 0.77 | The results seem to be improving with no. of epochs. We will next try to increase the amount of trainable data and reduce the filter size and crop the images. |
|  | Conv3D: conv\_3d3 | 1,762,613 | Filter size (2,2,2), image size 120x120, batch Size 30, Epochs 30.  Train Accuracy: 0.69  Val Accuracy: 0.63 | We were able to reduce the parameter size but the accuracy was compromised. We will further try to reduce the parameters by altering the network alongside improving the accuracy. |
|  | Conv3D: conv\_3d4 | 696,645 | Image 100x100, Batch 20, epochs 30.  The val losses have increased due to image cropping but the accuracy is improved.  Train Accuracy: 0.77  Val Accuracy: 0.74 | The val accuracy improves and the train accuracy plateaus with increasing the epochs. We will further try to reduce the parameters by updating the network neurons and update the image size for better accuracy. |
|  | Conv3D: conv\_3d5 | 230,949 | Image 120x120, Batch 20, Epochs 30.  Train Accuracy: 0.69  Val Accuracy: 0.67 | Reducing the network further to lower the number of parameters results in plateau of accuracies for both train and val sets to a low value. We will next try Conv+LSTM to compare the results. |
|  | CNN+LSTM: cnn\_rnn1 | 1,657,445 | The val accuracy is low and loss are high with scope for improvement.  Image 120x120, Batch 20, Epoch 20:  Train Accuracy: 0.85  Val Accuracy: 0.70 | This is a base CNN+LSTM model which shows average performance with overfitting. We will update the filters, image sizes and use more data for better training to counter overfitting. |
|  | CNN+LSTM: conv\_rnn\_7 | 3,638,981 | Improved val accuracy.  Filter (3,3,3), Image 160x160, Batch 20, Epochs 30.  Train Accuracy: 0.82  Val Accuracy: 0.74 | The val accuracy seems to be improving now. We will further try to reduce the filter size and alter the image for controlling the training parameters. |
|  | CNN+LSTM: conv\_rnn\_8 | 1,762,613 | Filter (2,2,2), Image 120x120, Batch 20.  Train Accuracy: 0.67  Val Accuracy: 0.65 | The parameters have reduced but the accuracies and the losses have been impacted. We will reduce the parameters by altering the network and updating the image and filters. |
|  | CNN+LSTM: conv\_rnn\_9 | 696,645 | Image 100x100, batch 20, epochs 30.  The val accuracy and loss improve with more epochs.  Train Accuracy: 0.84  Val Accuracy: 0.74 | The train and val accuracies improve and plateau with more epochs. We will try to further reduce the parameters by updating the network structure and will improve the image size to counter the high losses. |
|  | CNN+LSTM: conv\_rnn\_10 | 230,949 | Image 120x120, batch 20, epochs 30.  Train Accuracy: 0.65  Val Accuracy: 0.61 | The parameters have decreased but also the accuracies. We will next try transfer learning for better model performance. |
|  | Transfer Learning using MobileNet: rnn\_cnn\_tl | 3,840,453 | Basic transfer learning model. Image 120x120, Epochs 20.  Train Accuracy: 0.95  Val Accuracy: 0.67 | Validation accuracy is very poor. We will next train the MobileNet’s layer weights for better model performance. |
|  | Transfer Learning with GRU: rnn\_cnn\_tl2 | 3,693,253 | The val losses are better and model is stable. Image 120x120, Epochs 20.  Train Accuracy: 0.99  Val Accuracy: 0.94 | We get a better accuracy on training all MobileNet’s layer’s weights as well. The loss is also low, overall being a good model. |
| 1. Final Model | **Final Model:**  **Transfer Learning with GRU: rnn\_cnn\_tl2** |  | **Train Accuracy: 0.99**  **Val Accuracy: 0.94**  The final model selected from the analysis is “Transfer Learning with GRU:rnn\_cnn\_tl2” with the above performance. | The Conv3D performed best for model 4. (conv\_3d4) with:  train accuracy of 0.77  val accuracy of 0.74  for the low training parameters.  The CNN+LSTM model performed best with Model 9. (conv\_rnn\_9) with:  train accuracy of 0.84  val accuracy of 0.74 with low number of parameters.  The Transfer Learning with GRU had the highest accuracies as:  Train accuracy: 0.99  Val Accuracy: 0.94 |