**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:

* 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).

Different architectures and different parameter tuning were experimented as part of the gesture recognition case study. The inferences, we drew as part of our experiment are as given below.

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Result** | **Decision + Explanation** |
| **1** | **Conv3D without data augmentation 100\*100 image size**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 25  Batch Size 40  **Model Parameters** Total params: 686,901  Trainable params: 686,037  Non-trainable params: 864 | **Categorical Accuracy : 98.34%**  **Validation Accuracy : 55%** | **We can clearly see the model overfitting. Hence we tried to augment the images to the training data set to see whether we are able to overcome overfitting problem** |
| **2** | **Conv3D with Data augmentation 100\*100 image size**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 25  Batch Size 40  **Model Parameters**  Total params: 686,901  Trainable params: 686,037  Non-trainable params: 864 | **Categorical Accuracy : 96.38%**  **Validation Accuracy : 45%** | **Even after augmenting the training data , we still see the overfitting problem exists. We tried to add additional Cov3D to the architecture to see whether there is any improvement in the model performance** |
| **3** | **Conv3D with additional layers and without data augmentation 100\*100 image size**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 25  Batch Size 40  **Model Parameters**  Total params: 562,853  Trainable params: 561,637  Non-trainable params: 1,216 | **Categorical Accuracy : 93.36%**  **Validation Accuracy : 45%** | **By adding additional layers, the model took considerable time to fit the model. And still the overfitting issue seems to be exist. Still we wanted to try whether we see some improvement with the data augmentation with the current architecture** |
| **4** | **Conv3D with additional layers and with data augmentation 100\*100 image size**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 25  Batch Size 40  **Model Parameters**  Total params: 562,853  Trainable params: 561,637  Non-trainable params: 1,216 | **Categorical Accuracy : 88.69%**  **Validation Accuracy : 40%** | **Here we see that the training accuracy is reducing considerably. And the validation accuracy is not what we expect it to be. Hence we will try to add dropout to conv3d layers to regularize the model outoout** |
| **5** | **Conv3D with additional layers with dropout to COV3D 100\*100 image size**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 25  Batch Size 40  **Model Parameters**  Total params: 562,853  Trainable params: 561,637  Non-trainable params: 1,216 | **Categorical Accuracy : 83.26%**  **Validation Accuracy : 28%** | **Again we see a drop in both training and validation accuracy and the model continues to over fit. In continuation with the same architecture, we tried to augment training data set and tried to run more number of epochs, with early stopping with a patience of 10.** |
| **6** | **Conv3D with additional layers with dropout to COV3D 100\*100 image size and early stopping**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 100  Batch Size 40  **Model Parameters**  Total params: 562,853  Trainable params: 561,637  Non-trainable params: 1,216 | **Categorical Accuracy : 84.09%**  **Validation Accuracy : 25%** | **No significant improvement in model performance. Hence we tried to reduce model parameters and changed the filter size of first two layers to 3\*3\*3** |
| **7** | **Conv3D with reduced layers with 120\*120 image size and data augmentation**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 50  Batch Size 30  **Model Parameters**  Total params: 230,949  Trainable params: 230,453  Non-trainable params: 496 | **Categorical Accuracy : 89.89%**  **Validation Accuracy : 75%** | **With this architecture we see a considerable improvement in both the categorical and validation accuracy. However there is still scope to improve the validation and training accuracy and reduce the overfitting problem. Hence we tried to fine tune this architecture to achieve our goal** |
| **8** | **Conv3D with reduced layers with 120\*120 image size and data augmentation and image rotation**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 30  Batch Size 30  **Model Parameters**  Total params: 230,949  Trainable params: 230,453  Non-trainable params: 496 | **Categorical Accuracy : 91.33%**  **Validation Accuracy : 79%** | **After we incorporated image rotation to the data augmentation techniques, we had been following so far, we see increase in both training and validation accuracy. However, there is still overfitting problem is what we observed.**  **We tried to use SGD optimizer with a momentum of 0.9 instead of Adam optimizer we had been using so far** |
| **9** | **Conv3D with reduced layers with 120\*120 image size and data augmentation and image rotation and sgd optimier**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 30  Batch Size 30  **Model Parameters**  Total params: 230,949  Trainable params: 230,453  Non-trainable params: 496 | **Categorical Accuracy : 77.45%**  **Validation Accuracy : 71%** | **We no longer see overfitting problem. However we felt there is still scope for improving model accuracy.**  **We see that momentum plays a major role in helping to combat the overfitting problem. We wanted to explore the momentum wrt to Batch normalization and see whether it makes a difference** |
| **10** | **Conv3D with reduced layers with 120\*120 image size and data augmentation and image rotation and batch normalization**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 30  Batch Size 30  **Model Parameters**  Total params: 230,949  Trainable params: 230,453  Non-trainable params: 496 | **Categorical Accuracy : 77.83%**  **Validation Accuracy : 78%** | **We see almost training and validation accuracy are same and this seems to be a great model with Conv3D and its performance seems to be great compared to other variants we tried.**  **With more number of epochs run, we should definitely get a better performing model. However, we wanted to try other options before training this model for more number of epochs** |
| **11** | **LSTM with image size of 120**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 25  Batch Size 30  **Model Parameters**  Total params: 1,657,445  Trainable params: 1,656,453  Non-trainable params: 992 | **Categorical Accuracy : 60.63%**  **Validation Accuracy : 54%** | **The model seems to be underperforming. Both the training and validation accuracy were low even after 25 epochs run. However, we don’t see too overfitting problem here. We wanted to try GRU to see how it performs on the same model architecture** |
| **12** | **GRU with image size of 120**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 60  Batch Size 30  **Model Parameters**  Total params: 2,573,925  Trainable params: 2,573,445  Non-trainable params: 480 | **Categorical Accuracy : 53.24%**  **Validation Accuracy : 56%** | **Even after 60 epochs run, we don’t see appreciable increase in model performance with GRU architecture.**  **We wanted to try using transfer learning along with GRU to see how the model performs** |
| **13** | **Transfer Learning with VGG16 and GRU**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 30  Batch Size 30  **Model Parameters**  Total params: 14,980,421  Trainable params: 264,709  Non-trainable params: 14,715,712 | **Categorical Accuracy : 97.89%**  **Validation Accuracy : 74%** | **Since we froze the initial layers of vgg16 , the number of trainable parameters were considerable. We see that the model is performing extremely well on the training data while the validation accuracy is comparable but overfitting problem still exists.**  **We wanted to try the transfer learning with light weight model, since we wanted to use the same on the webcam which would be fit to the TV. So we choose mobilenet to see whether we can fine tune its parameters and get good accuracy** |
| **14** | **Transfer Learning with Mobile Net and GRU – SGD optimizer**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 30  Batch Size 20  **Model Parameters**  Total params: 4,284,613  Trainable params: 4,260,677  Non-trainable params: 23,936 | **Categorical Accuracy : 98.87%**  **Validation Accuracy : 98%** | **We see both the validation and training accuracy has improved very layers. In this model, we tried to make the last 9 layers weights as trainable and after much fine tuning, we got a very good validation and training accuracy.**  **However, we were not very happy with the number of trainable parameters. Since our model should fit into webcam we wanted to try and reduce the number of trainable parameters** |
| **15** | **Final Model - Transfer Learning with Mobile Net and GRU – SGD optimizer – reduced layers**  **Hyper parameters**  learning\_rate = 0.01  Number of epochs 15  Batch Size 20  **Model Parameters**  Total params: 3,446,725  Trainable params: 3,422,789  Non-trainable params: 23,936 | **Categorical Accuracy : 97.51%**  **Validation Accuracy : 93%** | **We see with mere 15 epochs, we got a model which is performing very well on both the training and validation accuracy.**  **The image size used was also 100\*100, which provides edge over the training speed.**  **Number of trainable parameters was also only mere 3.5 lakhs.**  **With all the experiments conducted we were able to achieve a model which was very light weight and able to perform well on both validation and training data set.** |