Gesture Recognition

# **Problem Statement:**

The objective is to develop a cool feature in the smart-TV that can recognize the defined five different gestures performed by the user which will help users control the TV without using a remote

We have conducted various experiments using the deep learning models and the following are the results captured from these experiments

|  |  |  |  |
| --- | --- | --- | --- |
| Experiment | Model | Result | Explanation + Decision |
| 1 | Conv3D  Batch size = 128 | Train accuracy: 0.15 Val accuracy: 0.15 | The model is underfitting and not learning through epochs.  Next: reducing the batch size further |
| 2 | Conv3D  Batch size = 32 | Train accuracy: 0.20 Val accuracy: 0.18 | Reducing the batch size had no impact and no improvement seen in the model.  Next: adding more layers to increase the learning |
| 3 | Conv3D | Dimension mismatch Error | Error due to kernel size mismatch with the output of previous layers  Next: Reduce the kernel size of new layers |
| 4 | Conv3D | Train accuracy: 0.20 Val accuracy: 0.20 | No improvement as compared to model2  Next: Add Batch normalization layer after every CNN and dense layers |
| 5 | Conv3D | Train accuracy: 0.97 Val accuracy: 0.23 | Model is over-fitting on less data i.e. ablation data set.  Next: Train on full data |
| 6 | Conv3D | Train accuracy: 0.97 Val accuracy: 0.53 | Model is still over-fitting.  Next: Add dropouts to generalize the model |
| 7 | Conv3D  Dropout = 0.2 | Train accuracy: 0.97 Val accuracy: 0.69 | Slight improvement in the accuracies  Next: Increase the dropout from 0.2 to 0.5 |
| 8 | Conv3D  Dropout = 0.5 | Train accuracy: 0.95 Val accuracy: 0.51 | Validation accuracy dropped and model is over-fitted.  Next: revert to dropout of 0.2 and reduce a CNN layer to reduce the complexity |
| 9 | Conv3D  Dropout = 0.2 | Train accuracy: 0.97 Val accuracy: 0.81 | Model is good  Next: use Global Average Pooling instead of Flatten layer to see if it can be any better. |
| 10 | Conv3D | Train accuracy: 0.94 Val accuracy: 0.60 | Model is over-fitting.  Next: Model 9 has good score. Let us experiment with other architectures. |
| 11 | Time Distributed + GRU | Train accuracy: 0.94 Val accuracy: 0.57 | Model is over-fitted.  Next: Add dropouts after each layer to reduce over-fitting. |
| 12 | Time Distributed + GRU  Dropout = 0.2 | Train accuracy: 0.90 Val accuracy: 0.63 | Model is still over-fitting.  Next: Replace GRU with plain Dense layer and add global average pooling. |
| 13 | Time Distributed + Dense | Train accuracy: 0.91 Val accuracy: 0.58 | Model is still over-fitting.  Next: Let us try using a different architecture i.e. time distributed with convLSTM2D |
| Final Model | Time Distributed + ConvLSTM 2D | Train accuracy: 0.85 Val accuracy: 0.85 | The 37th epoch has generated this value and looks like a good model. Now Model 9 has better accuracy than model 14 however model 9 has 22,732,549 parameters but model 14 has only 13,781 parameters and hence chosen as final model. |

**Conclusion :** Model 14 (Final model) has good accuracy scores and less parameters compared to other models. Even though model 9 has a better accuracy scores there are considerably more parameters and hence Model 14 is chosen as the final model.