**Problem Statement**: Recognizing five different gestures performed by the user helping users to control TV without remote.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Input parameters** | **Number of parameters** | **Highest Validation accuracy** | **Corresponding Training accuracy** | **Result** | **Decision + Explanation** |
| 1 | Conv3D | x=30  y=160  z=160  batch\_size=40  num\_epochs=30 | 3562053 | - | - | Out of memory issue | Crop the image size and re-try. |
| 2 | Conv3D | x=11  y=120  z=120  batch\_size=30  num\_epochs=30 | 2128453 | 0.4750 | 0.3043 | After reducing image size OOM issue gots resolved. | After changing and lowering image size ,  Model is executing without OOM error but train and validation accuracy needs to be improved. |
| 3 | Conv3D | x=11  y=120  z=120  batch\_size=64  num\_epochs=30 | 7722373 | 0.3333 | 0.9733 | Model is overfitted. | Include drop outs and change the batch size.Also, try to change no. of images fed to model. |
| 4 | Conv3D | x=16  y=120  z=120  batch\_size=40  num\_epochs=30 | 15095173 | 0.8167 | 0.8270 | Models look good as the difference between training and validation accuracy is very less. | It is very clear that selection of image id's plays an important role in training and validation accuracy. Also, changing the dropout value resolved the overfitting issue. |
| 5 | Conv3D | x=16  y=120  z=120  batch\_size=40  num\_epochs=50 | 15095173 | 0.70 | 0.7474 | After doing analysis of all epochs of model\_7: Difference between validation and training accuracy is increasing which indicates overfitting. | Let’s reduce the number of images fed into the network. Also, included dropout layers before flattening. |
| 6 | Conv3D | x=11  y=120  z=120  batch\_size=40  num\_epochs=30 | 7722373 | 0.73 | 070 | Better results | The training loss is higher because we have made it artificially harder for the network to give the right answers by including more dropouts. However, during validation all of the units are available, so the network has its full computational power - and thus it might perform better than in training.  Lets try CNN-RNN Model. |
| 7 | Conv2D +RNN | x=16  y=120  z=120  batch\_size=40  num\_epochs=30  lstm\_cells=128  dense\_neurons=128  dropout=0.25 | 1657445 | 0.65 | 0.95 | Model is overfitting | Model is over-fitting. Let’s try reducing the number of layers in next iteration along with increasing dropout value from 0.25 to 0.4 |
| 8 | Conv2D +RNN | x=16  y=120  z=120  batch\_size=40  num\_epochs=30  lstm\_cells=128  dense\_neurons=128  dropout=0.4 | 1657445 | 0.78 | 0.82 | Better results. | After changing as mentioned above, training accuracy got reduced but overfitting issue clearly got resolved as the difference between training and validation accuracy got reduced. |

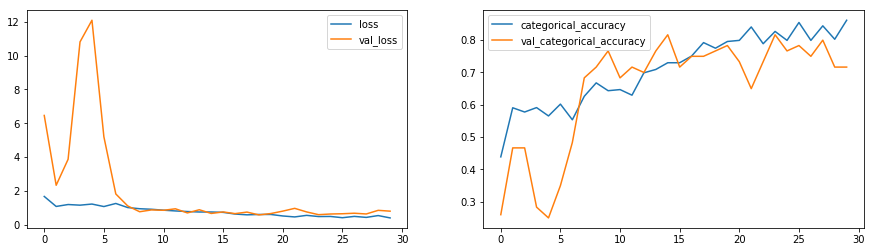
**Important Observations:**

* We might start with a large value like 0.1, then try exponentially lower values: 0.01, 0.001, etc. When we start training with a large learning rate, the loss doesn’t improve and probably even grows while we run the first few iterations of training.
* Overfitting issues can be resolved by including dropouts in the model.
* Selection of image Id’s list plays an important role in accuracy.
* Model might get overfitted if the number of layers are more.
* Image size also plays an important role and can cause Out of Memory issues if combination of image size+no.of images+ batch\_size etc is at higher end. Resizing/cropping can help to resolve OOM issues along with changing batch\_size.

**Final Model Details:**

* Selected Adam optimizer.
* Learning\_rate=0.001
* dropout=0.4
* Selected around 16 images out of 30 to feed into the neural network.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Conv3D | x=16  y=120  z=120  batch\_size=40  num\_epochs=30 | 15095173 | 0.8167 | 0.8270 | Models look good as the difference between training and validation accuracy is very less. | It is very clear that selection of image id's plays an important role in training and validation accuracy. Also, changing the dropout value resolved the overfitting issue. |

****