# Summary:

## Optimizer ,loss and Accuracy:

- This is a multiclass classification problem with 5 classes. Hence in all the models, the loss function is used as a Categorical Cross Entropy.

- The classes are almost balance ; hence metric Categorical Accuracy is used to measure the performance of the models.

- Adam optimizer is used in all the models ,as in most of the scenario it works better compare to other optimizer like SGD with Momentum, RMSPROP,ADADELTA and ADAGRADE.

## Preprocessing:

- In custom generator, resize function is used and dimension( height, width) of image kept as (120,120).

- I have experimented with dimension (60,60), too much data loss happened hence model did not perform well.

- Image dimension (160,160) takes lot of time to process with existing infrastructure and resource(RAM and GPU ). With batch size 40 environment ran into GPU error.

- The image normalization kept as [-1,1] by dividing image size with 127.5 (image/127.5 -1)

- Models with these architectures did not perform well with image normalization [0,1] i:e by dividing image/255.

## Model Architecture:

- Filters/kernels are feature detectors .It subsequently detects edges, lines, object parts gradually in the layers.

- In conv3D layers, we have experimented with kernel/filter size as (2,2,2) and (3,3,3).Higher order filters are not used as it creates the bottle neck with performance and model convergence and execution time. Lower order filters reduces the no of parameters to be trained.

- padding "same" is used in every conv3D layer as it adds zero evenly to left/right and up and down to avoid loss of information during conv operation.

- In few of the convolution layers in Batch normalization is used to avoid internal covariate shift.

- RELU activation function is used in input and middle layers .In the output layer SOFTMAX activation function is used as we need to predict 5 classes(multiclass classification) .

- Maxpoling is used with pool size as (2,2,2).

- In few of the layers dropout is used to avoid overfitting ,dropout 0.25 and 0.50 are used based on no of parameters.

- With Conv2D and RNN(LSTM +GRU),I have added TIMEDISTRIBUTED layer on top of conv2D. As for each video frame we need to extract features and would feed these to LSTM or GRU layer.

- In the callback, we have used Modelcheckpoint and ReduceLRonplateau.

- Model checkpoint observes the validation loss and saves the model.

- ReduceLRonplateau has patience =2,validation loss is monitored.

## Fit/Generator Method:

- steps per epoch is calculated as training length//batch size. sometimes I have taken more steps per epoch (67) [(663/10) +1] to increase the training efficiency.

- Num of epoch experimented with 10, due to infrastructure/resource bottleneck(GPU and CPU) in Google colab as well as Jarvis.

- Two callback was used (MODEL-CHECKPOINT and REDUCE-LR-ON-PLATEU) .

- Modelcheckpoint helps in saving the model and weights at a frequency of each improved epoch.

- REDUCELRONPLATEU helps in reducing learning rate when model is stopped improving.

## Model Experiment :

| Experiment Number | Model | Result | Decision + Explanation |
|---|---|---|---|
| 1. Conv3D | Epoch :10<br>Batch Size :10<br>Kernel size (2,2,2) and (3,3,3) with 8,16,32,64 numbers up to conv layer 4. | Training Accuracy:83.44%<br>Validation Accuracy:76%<br>Training Loss: 0.41<br>Validation Loss:0.69 | - Resize image to (120,120)<br>- Normalize to -1 to 1 (image/127.5 -1).<br>- No of parameters 76,893 |

| | | | |
|---|---|---|---|
| | | | - Usage of batch normalization to avoid internal covariate shift.<br>- Started with small kernel size(2,2,2) as experiments, it gives better result. |
| 2. Conv3D | Epoch :10<br>Batch Size :20<br>Kernel size (2,2,2) with 8,16,32,64 numbers up to conv layer 4. | Training Accuracy:99.50%<br>Validation Accuracy:78.50%<br>Training Loss: 0.07<br>Validation Loss:.0.57 | Increased the batch size ,but the validation loss remains same i:e 0.57 .<br><br>No of parameters 76,893 |
| 3. Conv3D | Epoch :10<br>Batch Size :30<br>Kernel size (2,2,2) with 8,16,32,64 numbers up to conv layer 4. | Training Accuracy:100%<br>Validation Accuracy:79.0%<br>Training Loss: 0.03<br>Validation Loss: 0.47 | A larger batch size will give us better gradient and helps to prevent jumping around.<br><br>Validation loss slightly decreased from 0.57 to 0.47<br><br>No of parameters 76,893 |
| 4.Conv3D | Epoch :10<br>Batch Size :64<br>Image size- (120,120)<br>Kernel size (3,3,3) with 32,64 numbers up to conv layer 4. | Training Accuracy:100%<br>Validation Accuracy:75.50%<br>Training Loss: 0.01<br>Validation Loss: 0.47 | Validation loss did not improve from 0.47126.<br><br>No of parameters 76,893 |
| 5. Conv3D | Epoch :10<br>Batch Size :30<br>Image size- (60,60)<br>Kernel size (3,3,3) with 32,64 numbers up to conv layer 4. | Training Accuracy:57.85%<br>Validation Accuracy:57.0%<br>Training Loss: 0.97<br>Validation Loss:.0.57 | - No of parameters 259,045<br>- ReduceLROnPlateau reduced learning rate to 0.0005. |
| 6 .Conv3D | Epoch :10<br>Batch Size :30 | Training Accuracy:35.56% | No of parameters 908,725. |

| | | | |
|---|---|---|---|
| | Image size-(120,120)<br>Kernel size (2,2,2) with 16,32,64,128 numbers up to conv layer 4. | Validation Accuracy:46.0%<br>Training Loss: 1.46<br>Validation Loss: 1.94 | 4 Convolution layers are added.<br><br>Dense layer has 128 neurons. Batch normalization and dropout (0.25) are used to avoid overfitting |
| 7. Conv3D | Epoch :10<br>Batch Size :20<br>Image size-(120,120)<br>Kernel size (2,2,2) with 16,32,64,128 numbers up to conv layer 4. | Training Accuracy:40.80%<br>Validation Accuracy:60.50%<br>Training Loss: 1.47<br>Validation Loss: 1.11 | No of parameters 908,725.<br>Steps per Epoch on Tranining:34 |
| 8.CNN(Conv2D with LSTM | Epoch :10<br>Batch Size :20<br>Image size-(120,120)<br>Kernel size (3,3) with 16,32,64,128 numbers up to conv layer 4. | Training Accuracy:89.04%<br>Validation Accuracy:25.00%<br>Training Loss: 0.43<br>Validation Loss: 2.63 | No of training parameters: 1,720,997.Timedistributed layer added on top of conv2D. 4 Convolution layers are added,64 LSTM layers used . |
| 9. CNN(Conv2D with LSTM | Epoch :64<br>Batch Size :20<br>Image size-(120,120)<br>Kernel size (3,3) with 16,32,64,128 numbers up to conv layer 4. | Training Accuracy:89.04%<br>Validation Accuracy:28.00%<br>Training Loss: 0.39<br>Validation Loss: 2.46 | No of training parameters: 1,720,997 |
| 10. CNN(Conv2D with GRU | Epoch :10<br>Batch Size :20<br>Image size-(120,120)<br>Kernel size (3,3) with 16,32,64,128 numbers up to conv layer 4. | Training Accuracy:47.0%<br>Validation Accuracy:36.00%<br>Training Loss: 1.37<br>Validation Loss: 1.66 | No of training parameters: 5,547,877 |
| Final Model | Epoch :10<br>Batch Size :20 | Training Accuracy:84.31% | Total params: 76,893<br>Trainable params: 76,653 |

| | | | |
|---|---|---|---|
| | Image size-(120,120)<br>Kernel size (3,3,3) and(2,2,2) (with 16,32,64, numbers up to 3 conv layer . | Validation Accuracy:83.00%<br>Training Loss: 0.47<br>Validation Loss: 0.59 | Non-trainable params: 240 |