First we will try 3D convolutional network (`Conv3D) model architecture because they are less complex than CNN-RNN. First we will train with small amount of data to see if at all the model learns.

Since the data contains different size of images like some are `120 x 180` and some are `360 x 360` we will resize all the images to size `120 x 120` using customised data generator function to standardise and feed a consistent size to the model.

To start with we will use batch_size of `100` , `20` `epochs` and all the `30` frames/images available per video. The metric used here is Categorical Accuracy.

Experiment Number	Model	Result	Decision + Explanation
1	**Conv3D**	**Out of memory error**	**1) The device is not able to handle the large data size in a batch which depends on the `no_of_images` per video, `size` of images and `batch_size`. We will reduce all to smaller values and if required increase them step by step in next experiments to increase performance. 2) Reduce `batch_size` to `32`, `no_of_images` to `15` and image size to `80 x 80` and see if the memory issue resolves.**
2	**Conv3D**	**Training Accuracy: 1.0000 Validation Accuracy: 0.4500**	**1) The model is learning but overfitting and training accuracy 1 indicates model memorising train data due to small train data. 2) Use full training data.**
3	**Conv3D**	**Training Accuracy: 1.0000 Validation Accuracy: 0.7100**	**1) Though the validation accuracy has increased there is still overfitting. 2) To regulate the model, add Dropout layer before output layer.**
4	**Conv3D**	**Training Accuracy: 0.7074 Validation Accuracy: 0.6300**	**1) There is considerable reduction in overfitting but the accuracies have dropped. 2) For model to learn more features add third conv2d layer with 64 filters.**
5	**Conv3D**	**Training Accuracy: 0.8959 and Validation Accuracy: 0.7300**	**1) The accuracies have increased but there is overfitting. 2) We will try reducing filter size/kernel_size to (2,2,2)**
6	**Conv3D**	**Training Accuracy: 0.4992 Validation Accuracy: 0.6100.**	**1) Poor performance so will revert to previous filter size and try Dropout layer after flatten layer to reduce overfitting.**

7	**Conv3D**	**Training Accuracy: 0.7843** **Validation Accuracy: 0.7200**	**1) Overfitting is reduced but accuracies decreased.** **2) We will try cropping images and see if accuracies increase due to reduction in data noise.**
8	**Conv3D**	**Training Accuracy: 0.8552** **Validation Accuracy: 0.8200**	**1) Best performance till now.** **2) Will now try increasing the image size to 100 x 100 to see if it improves performance.**
9	**Conv3D**	**Training Accuracy: 0.7195** **Validation Accuracy: 0.6900.**	**1) The performance degraded so will revert back to previous model.** **2) Next will try increasing `no_of_images per video` from 15 to 20 to feed more information for learning and see if accuracy increases.**
10	**Conv3D**	**Training Accuracy: 0.8673** **Validation Accuracy: 0.7700.**	**1) Not better than previous model so will revert back.** **2) Will now try reducing batch-size from `32 to 20` and see if it improves performance.**
11	**Conv3D**	**Training Accuracy: 0.7843** **Validation Accuracy: 0.7700.**	**1) Lower accuracies than previous model so will revert back.** **2) Will now try CNN-RNN architecture to see if it improves performance.**
12	**Conv2D-ConvLSTM2D**	**Training Accuracy: 0.6501** **Validation Accuracy: 0.6100**	**1) Not better than Conv2D models.**
Final Model	**Conv3D**	**Training Accuracy: 0.8552** **Validation Accuracy: 0.8200** **Total Parameters: 480,517 (1.83 MB)**	**We choose model built in Experiment #8 as our best model.**