Problem Statement - Classifying five different hand gestures which correspond to specific commands to operate the television.

Approach - Classification can be achieved using two networks -

1. ***3D convolutional network(Conv3D)*** - An extension to the 2D convolutions, 3D conv takes input as a video. 3D convolutions apply a 3 dimensional filter to the dataset and the filter moves 3-direction (x, y, z) to calculate the low level feature representations. Their output shape is a 3 dimensional volume space such as a cube or cuboid. They are helpful in event detection in videos, 3D medical images etc.

2. ***Convolutions + RNN*** - Conv2D extracts the feature maps for each image in the sequence(video) and the sequence of these feature maps is then fed to RNN-based network. CNN layers extract the features and RNN layers are trained on the sequence of these extracted feature to classify them into one of the gestures using softmax at the output layer

Data generator -

- Images per sequence: Experimented with all 30 images and 15 images by picking alternate images.
- Ablation: Condition to check if experiment is ablation or not
- Total no. of batches: Obtained by dividing the total number of sequences (in train or validation set) by the batch size. If it is not completely divisible, handling the rest of the data in the last iteration.
- Image cropping: Since the dataset has images of two resolutions, 360x360 and 120x160, the images with unequal dimensions, here 120x160 are cropped to 120x120.
- Image resizing: The image sizes 120x120 and 100x100 are experimented with.
- Image normalization: The image data in each channel is normalized by dividing it by 255.

Experiments -

Experiment Number	**Model**	**Result**	**Decision + Explanation**	**Trainable Parameters**
CNN + RNN				
1.	Model0: Conv2D + LSTM Ablation=100 Images per sequence: 30 Image size: 120x120 Batch size: 64	OOM error	● Received OOM Error ● Kernel went down	10,280,709
2	Model0: Conv2D + LSTM Ablation=100 Images per sequence: 30	Model overfits Train acc: 1.0 Val acc: 0.40	● Started with all images in sequence ● Took 2 mins per epoch	10,280,709

3	Model0: Conv2D + LSTM Ablation=100 Images per sequence: 15	Model overfits Train acc: 1.0 Val acc: 0.28	• Reduced to 15 images per sequence, that is, considered alternate images. • Takes 1 min per epoch • **Training time** is much **lesser** with relatively the same learning. • Hence going ahead with 15 images per sequence	10,280,709
4	Model0: Conv2D + LSTM Complete dataset Images per sequence: 15	Model overfits Train acc: 0.9955 Val acc: 0.68	• Training on the complete dataset. • Model seems to have an overfit on it.	10,280,709
5	Model0_dr Conv2D + LSTM Added dropouts	Relatively better than previous one Train acc: 0.86 Val acc: 0.6	• Added dropout of 0.25 after LSTM and Dense layer. • Adding dropouts has a **regularising effect** on the model training process. • But, the model doesn't learn much after a point where train accuracy increases but validation accuracy stays the same at 0.6. • Using SGD already, hence considering a change in architecture of the model in the next experiment.	10,280,709
6	Model1 Conv2D + LSTM Add an extra TimeDistributed CNN layer	Better learning relative to the previous model. Train acc: 0.94 Val acc: 0.73	• Added another TimeDistributed CNN layer in the network. • With an added layer, the model has been able to **learn more features** and has learnt the data better. • Using LSTM	2,404,229
7	Model2 Conv2D + GRU	Train acc: 0.88 Val acc: 0.69	• **Lesser number of trainable parameters**	1,810,245

	Same layers as before but replaced LSTM layer with GRU		• Relatively the same amount of learning as the previous one.	
8	Model2 Conv2D + GRU Same model as above but L2 regularization at Dense layer	Train acc: 0.88 Val acc: 0.69	• As there was some overfitting in the previous experiment, decided to add **L2 regularization.** • No improvement over previous one	1,810,245
9	Model3 Conv2D + LSTM Two layers as initial model but with bigger filters in second TimeDistributed layer	Train acc: 0.72 Val acc: 0.55	• **Increased filter size** in second TImeDistributed layer hoping to find features in bigger portion of image • Performed worse than previous ones	10,313,477
10	Model4 VGG16 + GRU **Transfer Learning**	Train acc: 0.75 Val acc: 0.67	• Using a pre-trained CNN model, VGG16 in the TimeDistributed layer. • Marking **all layers as non-trainable**	901,701
11	Model5 VGG16 + GRU Last 3 trainable layers of VGG16	Train acc: 0.96 Val acc: 0.77	• **Training the last three layers** of the VGG16 along with GRU and Dense layer • Decided this approach so that final layers of VGG16 are able to learn the problem specific features. • Has definitely improved over previous ones.	5,621,317
12	Model5 VGG16 + GRU Last 3 trainable with dropouts in GRU and Dense	Train acc: 0.84 Val acc: 0.7	• As there was some overfitting in the previous experiment, added a Dropout of 0.25 after GRU and Dense	5,621,317

			• Model isn't overfitting as much this time.	
13	Model5 VGG16 + GRU Last 3 trainable With L2 regularization in Dense layer	Train acc: 0.88 Val acc: 0.77	• Experimented by adding an L2 regularizer in the Dense layer and Dropout after GRU. • Model performs pretty well for the given problem statement. • Also, with relatively less number of parameters • Considering this for the **final model**.	5,621,317
Conv3D				
1	Conv3D_1 Ablation with 100 videos.	Number of epochs -10 categorical_accuracy: 0.9955 val_categorical_accuracy: 0.35	• Model overfits. • Decided to train on the whole date and added dropout layers to reduce the number of trainable parameters.	24,087,269
2	Conv3D_1	Model throws memory exhausted errors. Number of epochs -10	• GPU memory was loaded with huge data so received OOM error. • Reduced the batch size from 50 to 20.	24,087,653
3	Conv3D_1	Categorical_accuracy: 0.3499 Val_categorical_accuracy: 0.1700	• Train and validation accuracies remain constant and loss also remains constant after several changes. • Decided to try a different architecture as the model wasn't learning after multiple changes and also number of parameter were very high	23,964,533
4-	Conv3D_2 Ablation with 100 videos	Categorical_accuracy: 0.983 val_categorical_accuracy: 0.321	• Model overfits. • Training on the whole date and increasing the epochs to 20.	1,361,381

5	Conv3D_2	categorical_accuracy: 0.847 val_categorical_accuracy: 0.49	• Model accuracy increases and the loss decreases smoothly. • Decided to increase the feature maps and added dropouts after the convolutional layer.	5,512,901
6	Conv3D_2	categorical_accuracy0.80 val_categorical_accuracy: 0.67	• Model accuracy increases further. • Training loss and validation loss curves are overlapping and decrease smoothly.	5,490,077

Observations:

After conducting several experiments for solving the Gesture Recognition problem using Conv2D + RNN approach and Conv3D approach, the following observations have been made.

- Training the model over a complete sequence of 30 images takes almost twice as long compared to training a sequence of 15 images. But the performance isn't affected much.
- Using a batch size of 64 with image resolution 120x120 gave an OOMError, hence picked a smaller batch size of 32 and image resolution of 100x100.
- Ablation experiments gave a clear idea on whether the model will learn or not before training on the whole dataset.
- If there is overfitting, adding Dropouts helps in reducing the effects.
- An L2 regularizer also helps in reducing the overfitting problem, but needs experimenting.
- GRU performs as good as an LSTM but with lesser number of parameters
- Increasing the filter size didn't add advantage in this specific problem statement by keeping the TimeDistributed CNN layers the same.
- In transfer learning, the model overfits on using it directly. But by training the final layers makes it learn problem specific weights, and hence perform better.

Choice of Final Model:

Model5 **VGG16 + GRU** Last 3 trainable With L2 regularization in Dense layer	**Good accuracy for train and validation** Train acc: 0.88 Val acc: 0.77	• Pre-trained VGG16 by training the final 3 layers with some regularization in place. • Relatively less number of parameters	**Trainable parameters:** 5,62