**Summary:**  
There were lot of approaches used and all the approaches/models were trained on GPU enabled remote machine (*Google Colab*). All the models were trained on similar preprocessing techniques so that not much deviation is observed with so many combinations between hyper parameters and model approach selection. The best observed model (Model 8, in our case) seemed to have performed better than the rest. We feel the accuracy can further be increased by improving the initial pro-processing techniques.

**Experimental Models:**

1. **Model 1:**

* Architecture used :- Conv2D + LSTM (Transfer Learning with ResNet50)
* Hyper Parameters :-
  + Number of epochs:- 10
  + Batch Size:- 20
* Result :-
  + Categorical Training Accuracy :- 0.4593
  + Validation Accuracy:- 0.1200
* Decision :-  
  We froze ResNet trainable layers and increased the epochs to 20.

1. **Model 2**

* Architecture used :- Conv2D + LSTM (Transfer Learning with ResNet50)
* Hyper Parameters :-
  + Number of epochs:- 20
  + Batch Size:- 20
* Result :-
  + Categorical Training Accuracy :- 0.3703
  + Validation Accuracy:- 0.27
* Decision:-
  + Removed additional layer
  + Added Reshape layer
  + Updated number of epochs back to 10

1. **Model 3**

* Architecture used :- Conv2D + LSTM (Transfer Learning with ResNet50)
* Hyper Parameters :-
  + Number of epochs:- 10
  + Batch Size:- 20
* Result :-
  + Categorical Training Accuracy :- 0.6994
  + Validation Accuracy:- 0. 2200
* Decision :--
  + Changed optimizer
  + Increased dropout value
  + Replaced Callback ReduceLROnPlateau with DecayLR

1. **Model 4**

* Architecture used :- Conv2D + LSTM (Transfer Learning with ResNet50)
* Hyper Parameters :-
  + Number of epochs:- 10
  + Batch Size:- 20
* Result :-
  + Categorical Training Accuracy :- 0.8553
  + Validation Accuracy:- 0.3700
* Decision :--
  + Added a Dropout Layer

1. **Model 5**

* Architecture used :- Conv2D + LSTM (Transfer Learning with ResNet50)
* Hyper Parameters :-
  + Number of epochs:- 10
  + Batch Size:- 20
* Result :-
  + Categorical Training Accuracy :- 0.8618
  + Validation Accuracy:- 0.3400
* Decision :--
  + Added Batch Normalisation Layer

1. **Model 6**

* Architecture used :- Conv2D + LSTM (Transfer Learning with ResNet50)
* Hyper Parameters :-
  + Number of epochs:- 10
  + Batch Size:- 20
* Result :-
  + Categorical Training Accuracy :- 0.8391
  + Validation Accuracy:- 0.3700
* Decision :--
  + Added Batch Normalization Layer
  + Using Conv3D approach

1. **Model 7**

* Architecture used :- Conv3D
* Hyper Parameters :-
  + Number of epochs:- 10
  + Batch Size:- 20
* Result :-
  + Categorical Training Accuracy :- 0.2126
  + Validation Accuracy:- 0.1900
* Decision :--
  + It takes quite a lot of time to train, tried another approaches but the results were not much improved.
  + Halting training

1. **Model 8 (Final Model)**

* Architecture used :- Conv2D + GRU
* Hyper Parameters :-
  + Number of epochs:- 5
  + Batch Size:- 20
* Result :-
  + Categorical Training Accuracy :- 0.8482
  + Validation Accuracy:- 0.3800
* Decision :--
  + This was the rather the most convenient approach.

**Challenges Faced So far**

* 1. There were hurdles in improving the validation accuracy.
  2. Most of the times, the remote notebook ran out of memory and got killed.
  3. Keras do not have inbuilt model for Conv3D and hence Transfer learning could not be used in Conv3D approach.