**Approach Summary:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Exp.No.** | **Model** | **No. Of Trainable**  **Parameters** | **Result/Accuracy** | **Comment** |
| 1. | **Conv3D**   * **Batch\_size=64** * **Activation function = ‘relu’** * **Kernel\_size=(3,3,3)** * **Using alternate image frames** * **Epoch = 5** | 9,439,365 | **Got ResourceExhaustedError. This means we cannot experiment with batch size of 64 in Google Colab.** | **This means we cannot experiment with batch size of 64.** |
| 2. | **Conv3D**   * **Batch\_size=32** * **Activation function = ‘relu’** * **Kernel\_size=(3,3,3)** * **Using alternate image frames** * **Epoch = 5** | 9,439,365 | loss: nan -  categorical\_accuracy: 0.2066 - val\_loss: nan - val\_categorical\_accuracy: 0.1700 | **This model was also tried on Google colab. We tried to run for 5 epochs but got very low accuracy and the loss is nan.** |
| **3** | **Conv3D**   * **Batch\_size=32** * **Activation function = ‘relu’** * **Kernel\_size=(2,2,2)** * **Using alternate images** * **Epochs = 10** | 7,412,293 | categorical\_accuracy: 0.2016 val\_categorical\_accuracy: 0.1806 | **Model Under-fitting.**  **Changing the kernel size did not improve accuracy,** |
| **4.** | **Conv3D**   * **Batch\_size=32** * **Activation function = ‘elu’** * **Kernel\_size=(3,3,3)** * **Using alternate image frames** * **Epochs = 10** * **Learning rate = 0.001** * **Image normalized by dividing 255** * **Also decay 1e-6 is used in the SGD optimizer.**   **Also Learning rate decreases over the epoch** | 9,439,365 | loss: 0.9628 - categorical\_accuracy: 0.6576 - val\_loss: 6.8753 - val\_categorical\_accuracy: 0.1700 - lr: 1.2500e-04 | **There is very much difference in training and testing accuracy.** |
| **5.** | **Conv3D**   * **Batch\_size=32** * **Activation function = ‘elu’** * **Kernel\_size=(3,3,3)** * **Using alternate image frames** * **Epochs = 20** * **Learning rate = 0.01** * **Image normalized by dividing 255** * **Decay 1e-6 is removed in the SGD optimizer.**   **Also Learning rate decreases over the epoch** | 9,439,365 | loss: 0.6651 - categorical\_accuracy: 0.7496 - val\_loss: 4.4714 - val\_categorical\_accuracy: 0.2900 - lr: 1.9531e-05 | **The model is still not performing well by changing the learning rate and removing the decay rate from the SGD optimizer.** |
| **6.** | **Conv3D**   * **Batch\_size=32** * **Activation function = ‘elu’** * **Kernel\_size=(3,3,3)** * **Using alternate image frames** * **Epochs = 30** * **Learning rate = 0.01** * **Image normalized by subtracting the mean value**   **Also Learning rate decreases over the epoch** | 9,439,365 | Epoch 00030: ReduceLROnPlateau reducing learning rate to 0.00015624999650754035.  21/21 [==============================] - 95s 5s/step - loss: 0.1524 - categorical\_accuracy: 0.9367 - val\_loss: 0.7059 - val\_categorical\_accuracy: 0.7900 - lr: 3.1250e-04 | **The model is overfitting after changing the normalization of the image. Now the image is subtracted by the mean value.** |
| **7.** | **Conv3D**   * **Batch\_size=32** * **Activation function = ‘elu’** * **Kernel\_size=(3,3,3)** * **Using alternate image frames** * **Epochs = 30** * **Learning rate = 0.01** * **Image normalized by subtracting the mean value and dividing by standard deviation**   **Also Learning rate decreases over the epoch** | 9,439,365 | Epoch 00030: ReduceLROnPlateau reducing learning rate to 0.00015624999650754035.  21/21 [==============================] - 101s 5s/step - loss: 0.2022 - categorical\_accuracy: 0.9255 - val\_loss: 0.4387 - val\_categorical\_accuracy: 0.8450 - lr: 3.1250e-04 | **The model is slightly overfitting when we normalized the images by subtracting the mean and dividing by the standard deviation.** |
| **8.** | **Conv3D**   * **Batch\_size=32** * **Activation function = ‘elu’** * **Kernel\_size=(3,3,3)** * **Using alternate image frames** * **Epochs = 30** * **Learning rate = 0.01** * **Also Learning rate decreases over the epoch** * **Normalizing images by subtracting RGB image by [104, 117, 123] value (This is calculated from Image net competition VGG 16 architecture)** | 9,439,365 | Epoch 00030: saving model to model\_init\_2022-11-2005\_10\_49.405905/model-00030-0.20564-0.92459-0.37276-0.88000.h5  21/21 [==============================] - 94s 5s/step - loss: 0.2056 - **categorical\_accuracy: 0.9246** - val\_loss: 0.3728 - **val\_categorical\_accuracy: 0.8800** - lr: 3.9062e-05 | **No over-fitting or under-fitting.**  **Changing normalization of images gave good accuracy on both train and test set.** |
| **9.** | **Conv2D + GRU**   * **Using alternative 18 (84X84) images** * **LR = 0.01** * **Epochs =30** * **Normalizing images by subtracting RGB image by [104, 117, 123] value (This is calculated from Image net competition VGG 16 architecture)** | 733,957 | Epoch 00030: ReduceLROnPlateau reducing learning rate to 0.00015624999650754035.  21/21 [==============================] - 94s 5s/step - loss: 0.1452 - categorical\_accuracy: 0.9879 - val\_loss: 0.7297 - val\_categorical\_accuracy: 0.7400 - lr: 3.1250e-04 | **Model over-fitting.** |

**Final Model:**Experiment 8 is the final model as it is giving good accuracy on both train and test set.

* Normailizing imaes by dividing 255 and some other techniques mentioned in experiment 1 to 7 was not giving good accuracy. Then used technique which is used in Image net competition in VGG – 16 architecture. In these model subtracted all images by [104, 117, 123] value.
* Model is able give good accuracy on both training and test set.
* Also we took all five different action for prediction and model is able to detect all 5 correctly.
* Model is traind on 30 epoch for batch size of 32.

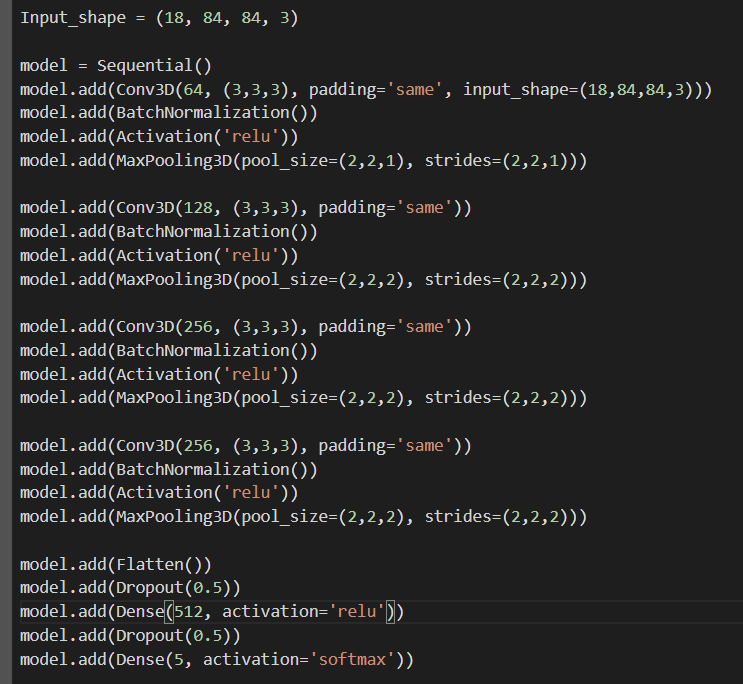
**Detailed Approach:  
Exp. 1 : Base Conv3D model**

Initially started in Google colab with Conv3D layers.

* Used Batch size = 64, Epoch = 5 and activation ‘Relu’
* Out of 30 images used 18 alternate images for all experiments. img\_idx = [0,1,2,4,6,8,10,12,14,16,18,20,22,24,26,27,28,29]
* But got resource exhaust error.
* This means we cannot use a batch size of 64. We need to reduce the batch size to 32.
* Image normalization by dividing 255

**Exp. 2 : Changing Batch size**

* Now we changed the batch size of 32.
* Used activation function = ‘Relu’ , Kernel size = (3,3,3) , Epoch = 5
* Model loss is ‘nan’. It means the prediction is very high then actual value. We can try changing kernel size. We already using ‘SGD’ optimizer. Also the accuracy is very low
* Image normalization by dividing 255.



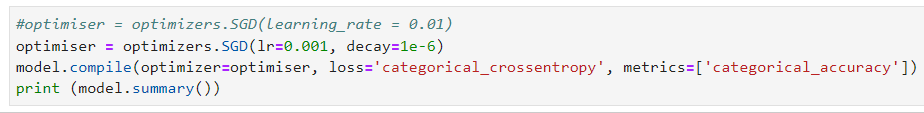
**Exp. 3 : Changing Kernel Size**

* Changing the kernel size from (3,3,3) to (2,2,2), Epoch = 10.
* No improvement in model performance.
* Model is under-fitting as both training and validation accuracy are poor.

**Exp. 4 : Changing activation function**

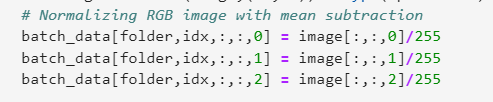
* Now, we changed activation function to ‘elu’ . From here all experiment tried on Jarvis lab AI platform.
* Batch size = 32, Kernel size (3,3,3), Epoch = 10, SGD Learning rate = 0.001, SGD Decay rate = 1e-6
* Model is still not performing well.





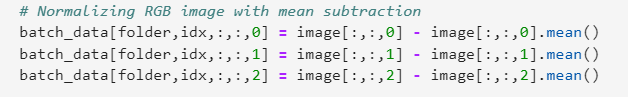
**Exp. 5 : Changing Learning rate and removing decay rate in SGD optimizer**

* Removed decay rate and changed learning rate to 0.01. Here we are dividing images by 255 for normalization.
* Model is still not working well but slightly improved in performance
* We tried it for 20 epochs.

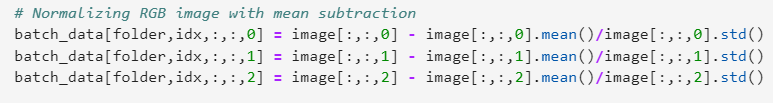


**Exp. 6 : Changing Image preprocessing**

* Here we changed the preprocessing of image.
* In this model we subtracted all images by mean value.
* The model is overfitting.

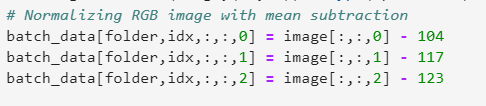
  
**Exp. 7 : Changing Image preprocessing**

* Here we again changed the preprocessing of the image
* Now we are subtracting the image by mean value and dividing by standard deviation.
* The performance is better now but the model is slightly overfitting
* We get an accuracy of 92% on the train and 84% on the test.



**Exp. 8 : Changing Image preprocessing(Final model)**

* Here, we used the technique used in the VGG 16 architecture of the Image net competition.
* We subtracted the image by [104, 117, 123] values.
* The model is very well on both tarin and test set. There is no overfitting and under fitting. The accuracy of model on train set is 92% and on test set 88%



**Exp. 9 : Conv2D+GRU model**

* Used Conv2D and GRU layer
* Model is overfitting.
* Train accuracy = 98% and test accuracy = 74%

