**Neural networks project**

**Hand Gesture Recognition case study**

**Approaches**

1. 3D Convolutional Network, or Conv3D
2. Convolutions + RNN (Conv2D + LSTM/GRU)

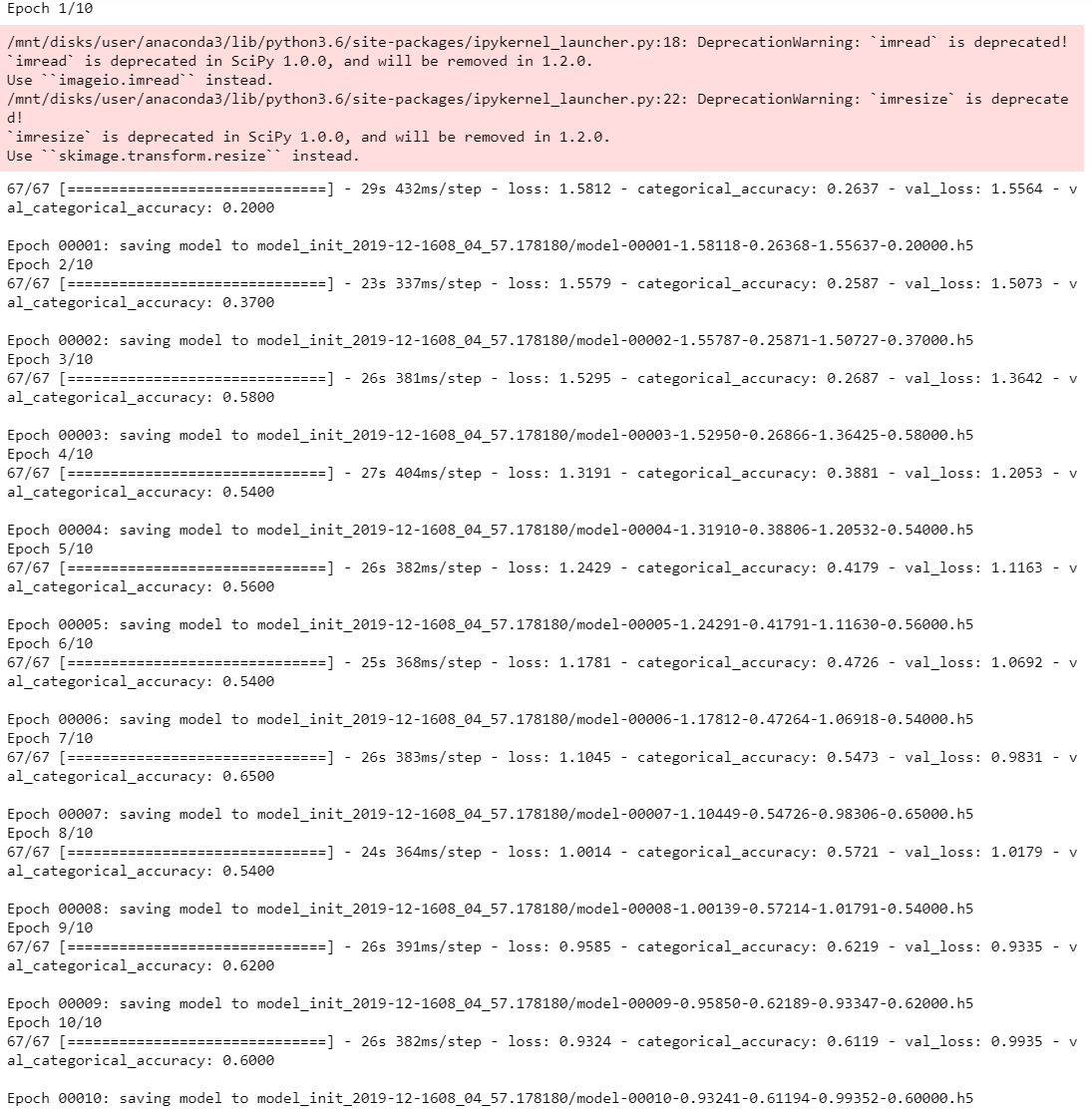
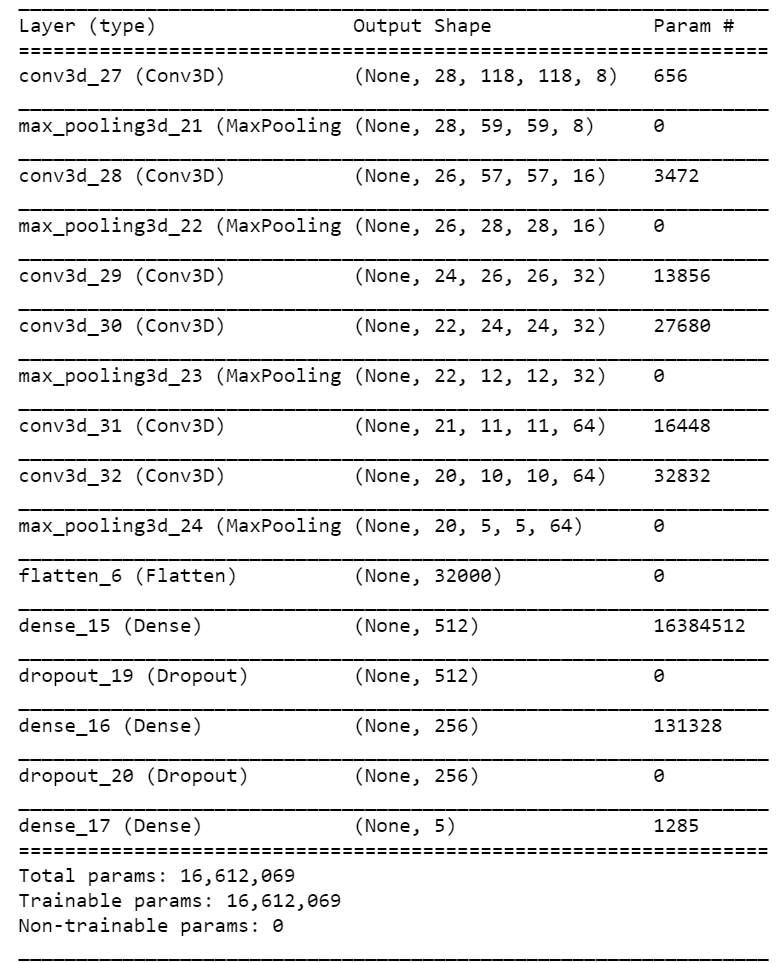
**Experiments conducted as below:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Result** | **Decision + Explanation** |
| 1 | Conv3D | loss: 0.9324 categorical\_accuracy: 0.6119 val\_loss: 0.9935 val\_categorical\_accuracy:0.6000 | No improvement in accuracy irrespective of increasing the data set with augmentation like color schemes using wrapaffine transformation from opencv.  16 million trainable parameters |
| 2 | Conv3D | loss: 1.6084 categorical\_accuracy: 0.2289 val\_loss: 1.6082 val\_categorical\_accuracy:0.2300 | Reduced the number of trainable parameters (5 million) + image normalization diving pixels by 255 |
| 3 | Conv3D | loss: 2.1638 categorical\_accuracy: 0.2338 val\_loss: 1.7714 val\_categorical\_accuracy:0.2000 | Further reduced number of trainable parameters to 900K |
| 4 | Conv3D | loss: 3.0842 categorical\_accuracy: 0.2189 val\_loss: 1.6290 val\_categorical\_accuracy:0.1800 | Further reduced number of trainable parameters to 800K |
| 5 | Conv3D | loss: 3.0853 categorical\_accuracy: 0.2040 val\_loss: 1.6408 val\_categorical\_accuracy:0.1700 | Kept trainable parameters to 800K added batchnormalization and dropouts at after every convolution layer |
| 6 | Conv3D | loss: 1.0920 categorical\_accuracy: 0.5423 val\_loss: 0.9913 val\_categorical\_accuracy:0.6900 | Kept trainable parameters to 800K. Added batchnormalization every layer and dropouts only at last convolution layer and FC layers.  Also horizontal flipping of images added |
| 7 | Conv3D | loss: 0.7513 categorical\_accuracy: 0.7114 val\_loss: 0.8170 val\_categorical\_accuracy:0.7200 | Same as model 6,  1. Last convolution layer dropout changed to batch normalization and decreased dropouts from 0.5 to 0.3 |
| 8 | ConvLSTM | loss: 12.0284 categorical\_accuracy: 0.2537 val\_loss: 13.2168 val\_categorical\_accuracy:0.1800 | Used pretrained VGG16 except the last layers (include\_top=False) and LSTM using time distributed layer.  19 million trainable parameters. |
| 9 | ConvLSTM | Out of memory error | Used pretrained VGG16 except the last layers (include\_top=False) and LSTM using time distributed layer.  15 million trainable parameters. |
| Final Model | Conv3D | loss: 0.7513 categorical\_accuracy: 0.7114 val\_loss: 0.8170 val\_categorical\_accuracy:0.7200 | Same as experiment #7 |

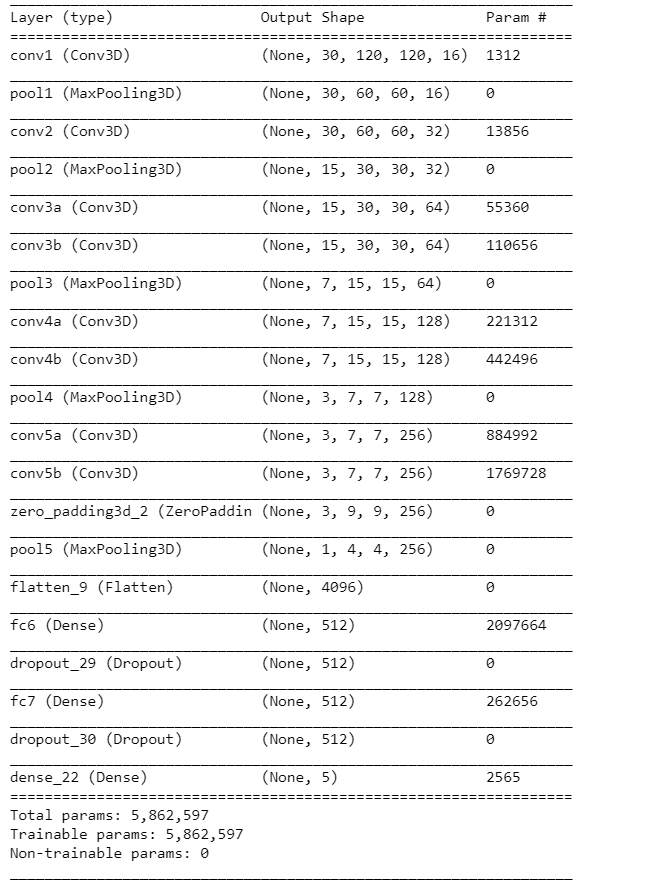
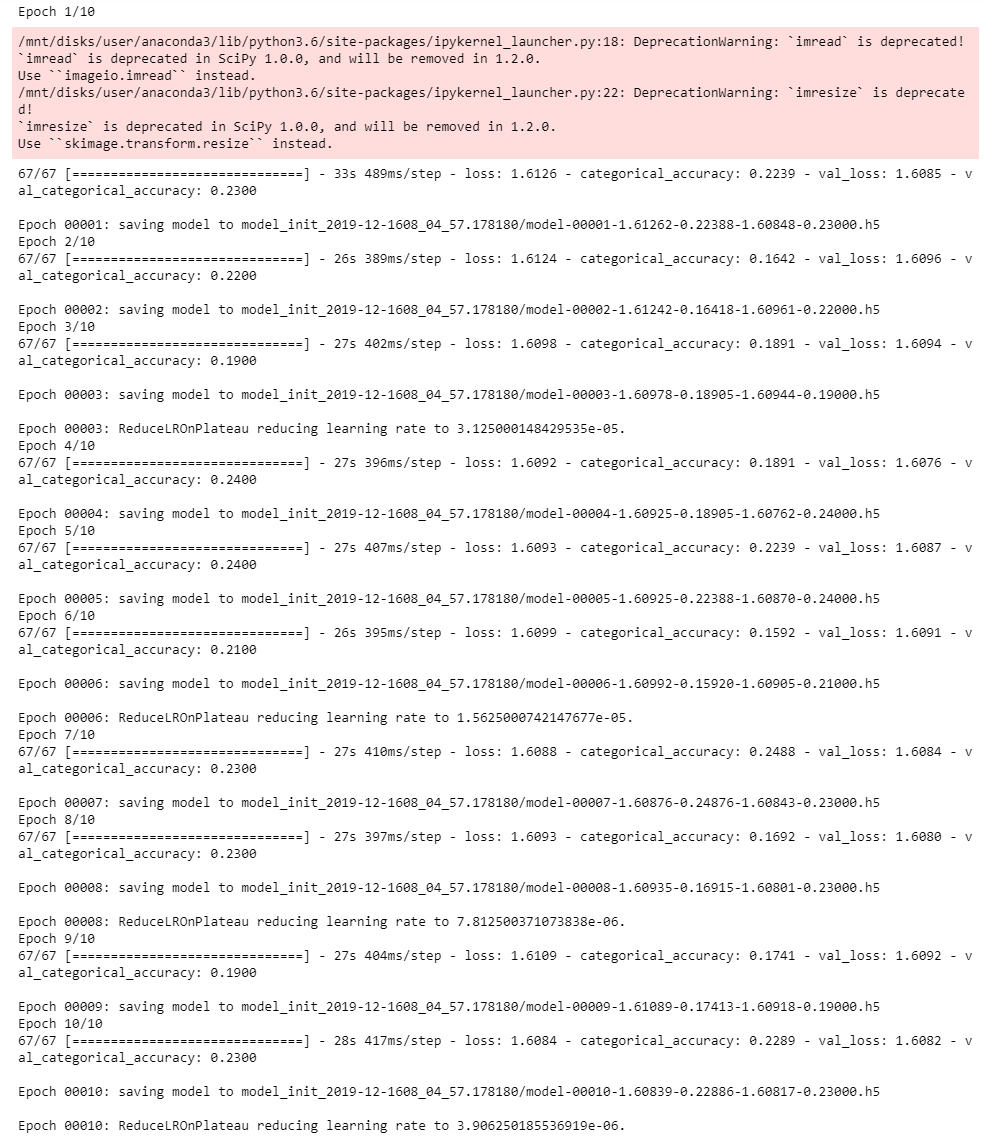
**Data processing**

* Resize
* Cropping
* Data agumentation
  + Affine transformation - Augmentation of images using openCV warpAffine
  + Flipping images horizontally – Using numpy.flip
  + Adding Gaussian noise
  + Illumination

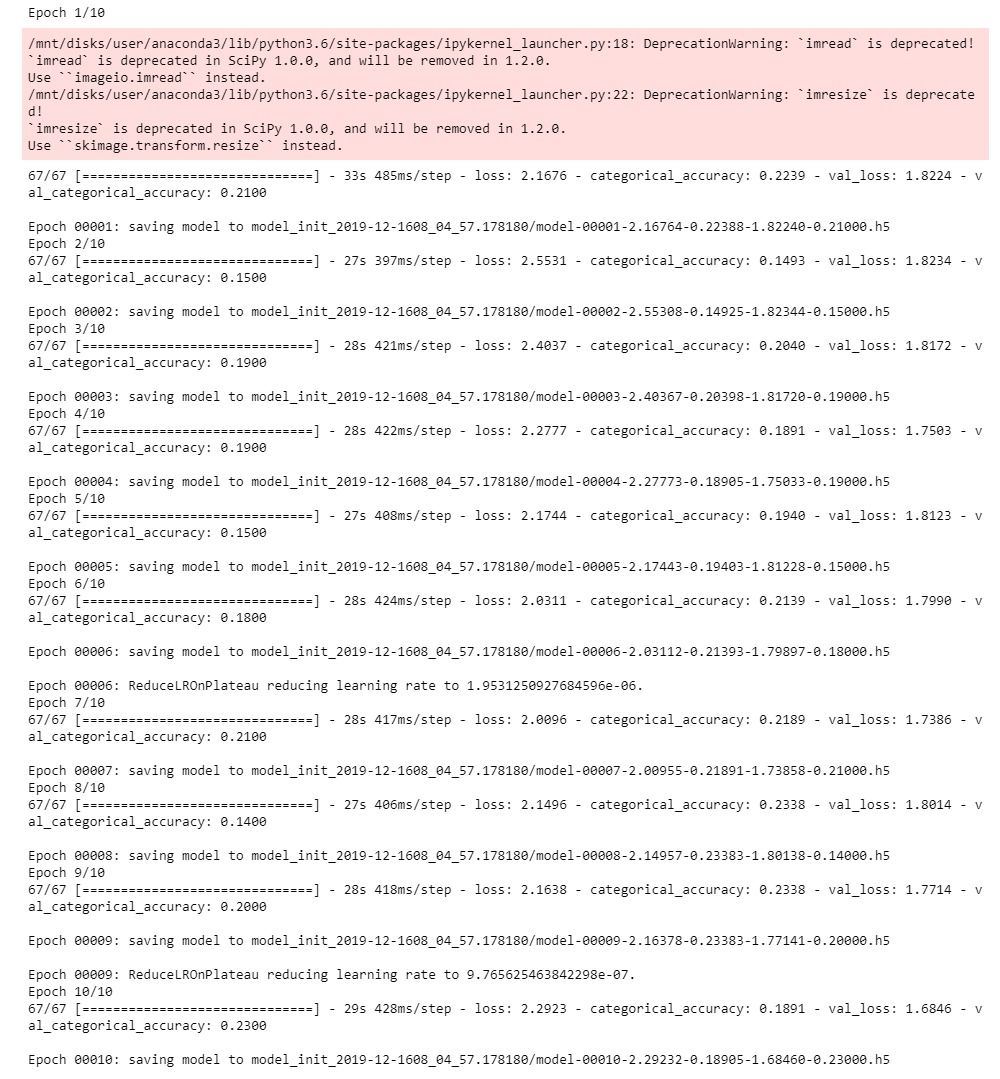
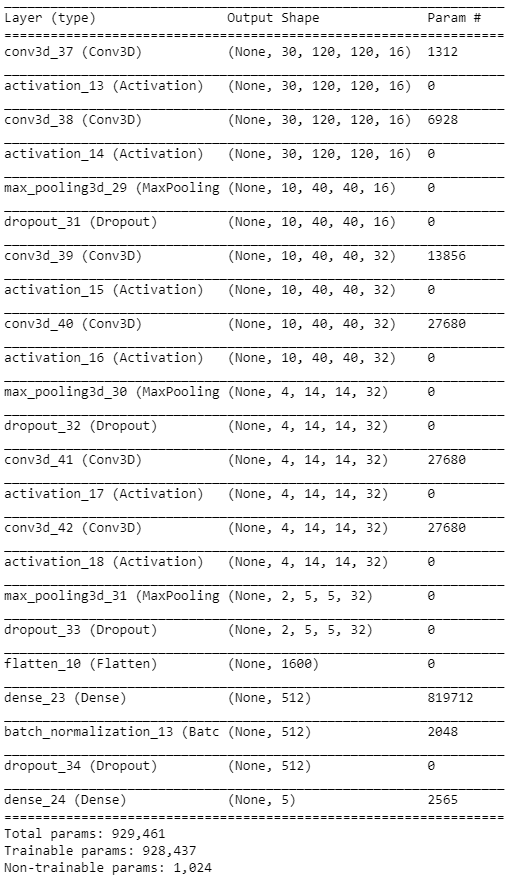
**Experiment #1:**



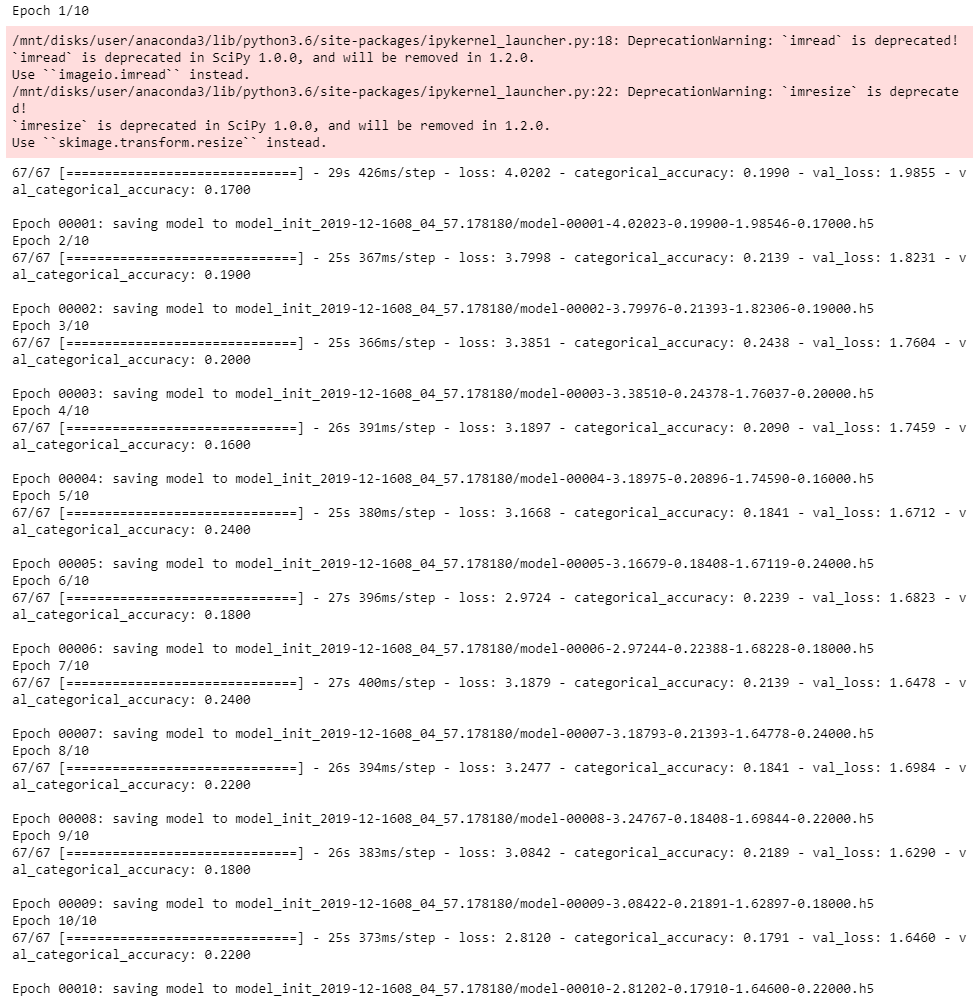
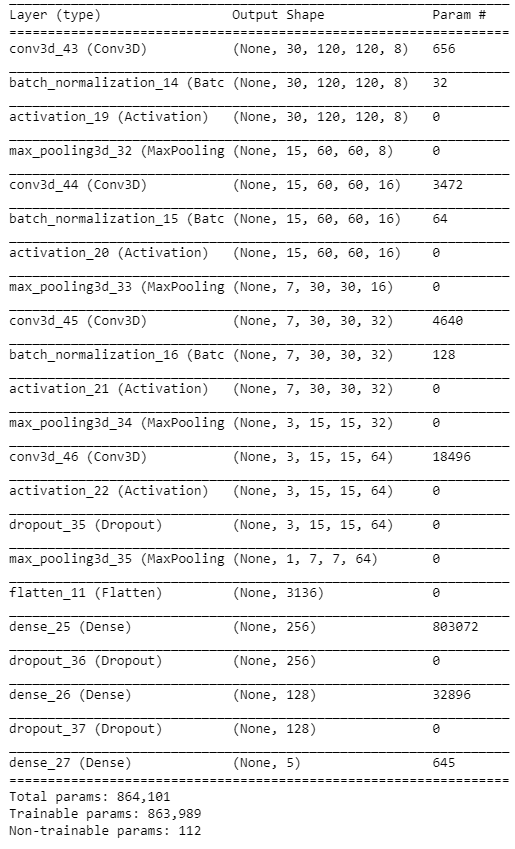
**Experiment #2:**

** **

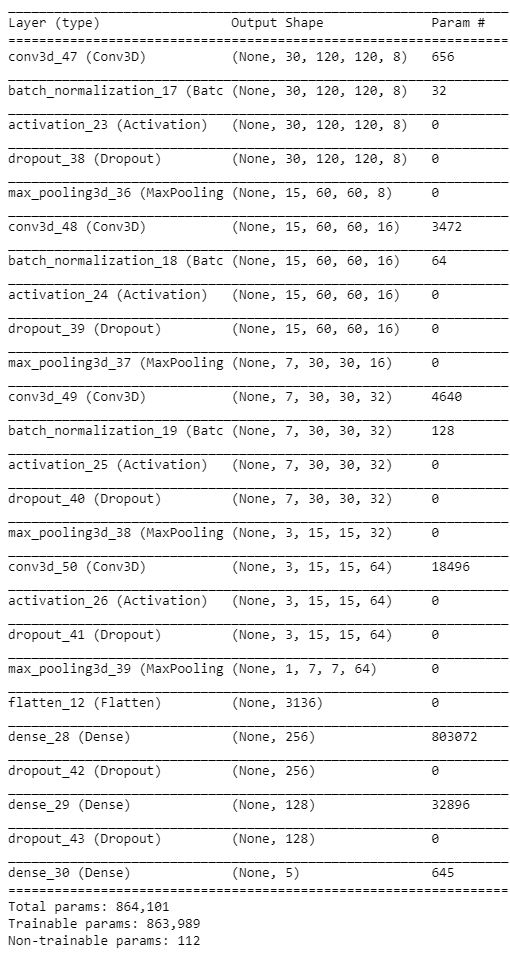
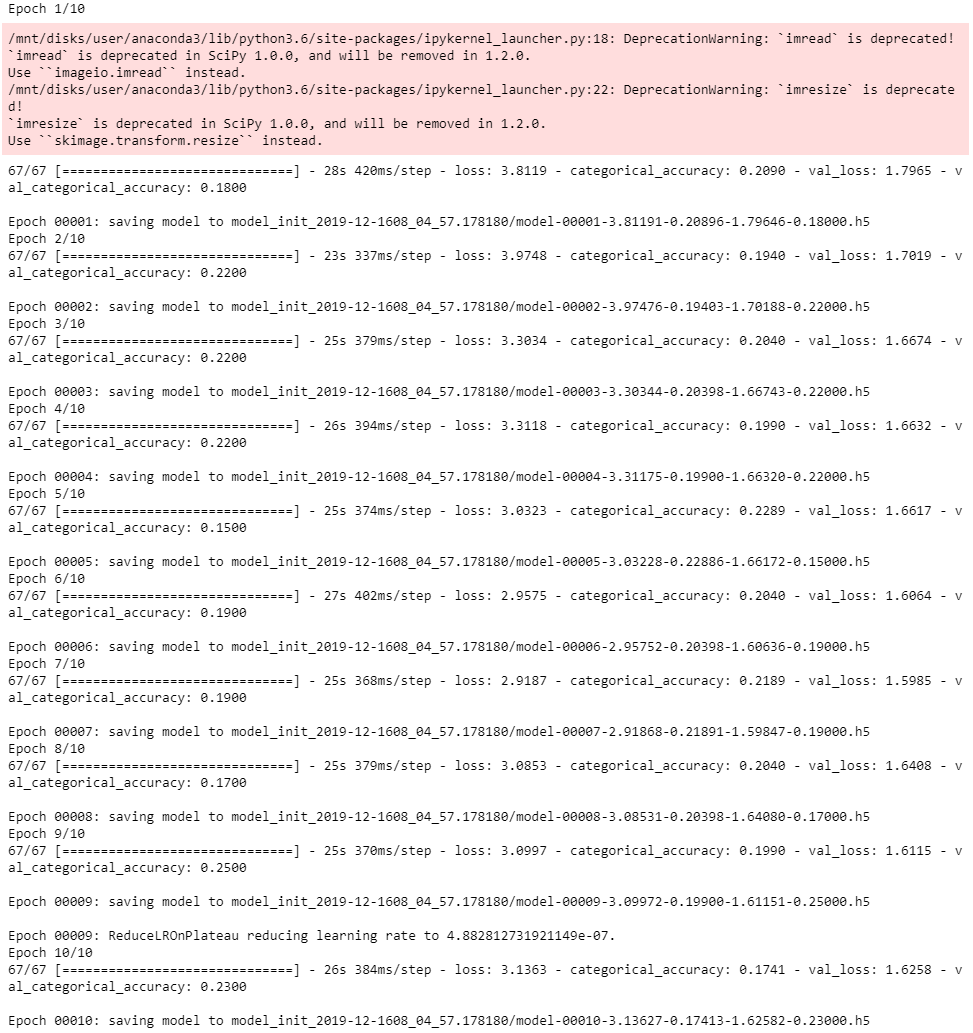
**Experiment #3:**

****

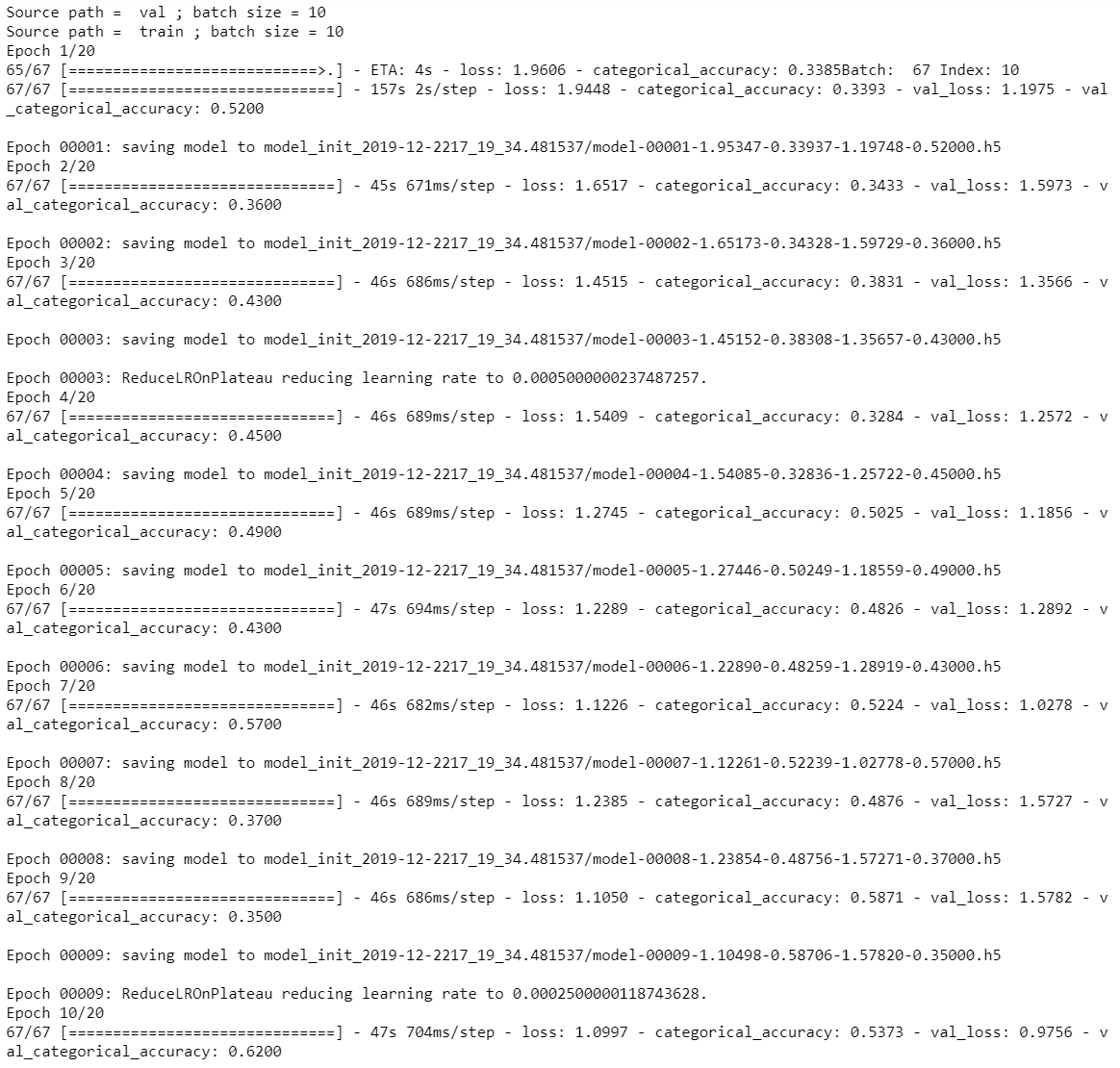
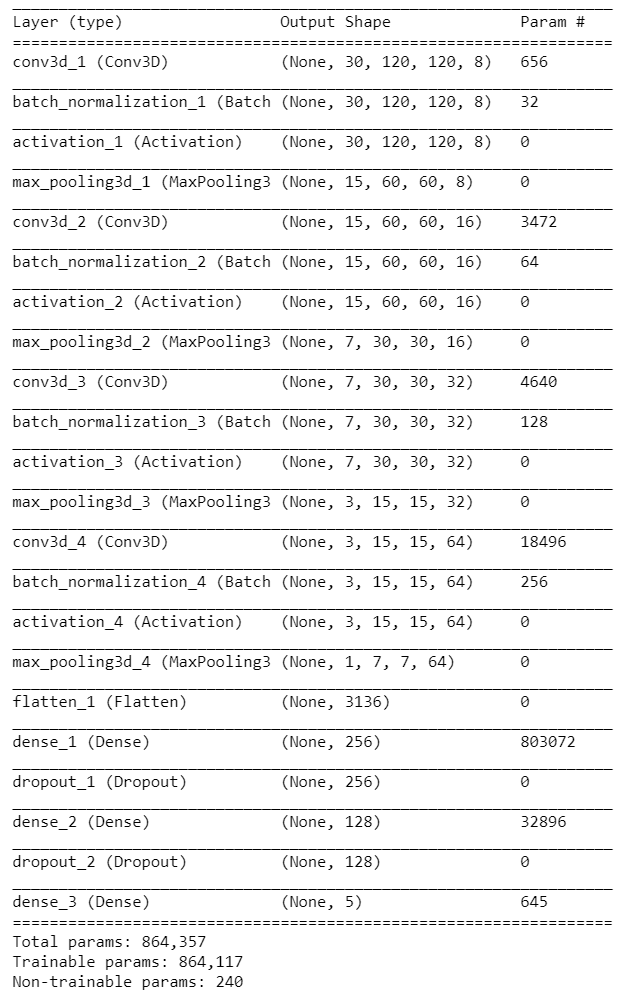
**Experiment #4:**

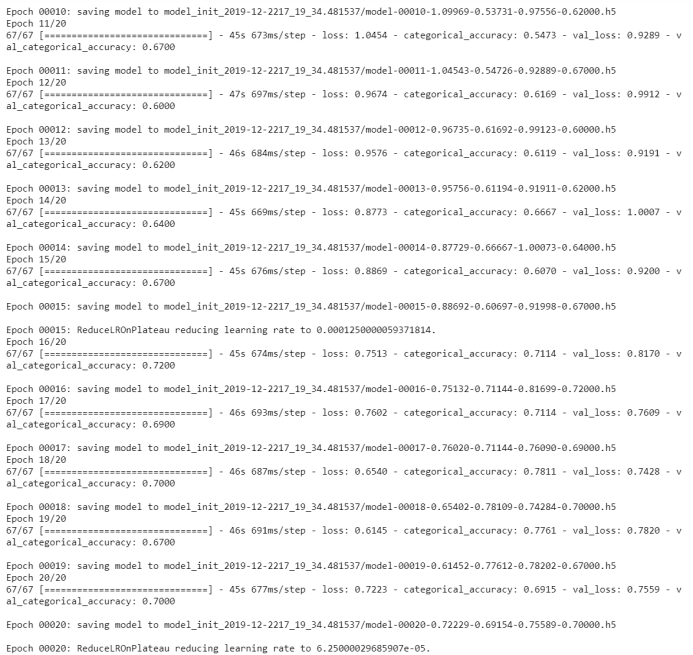
****

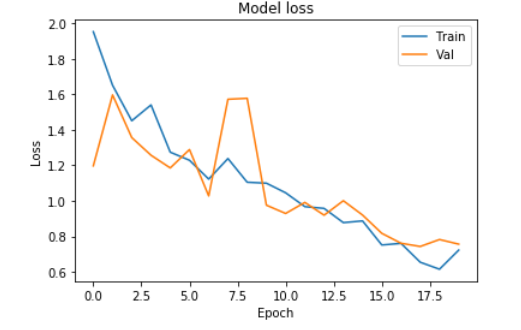
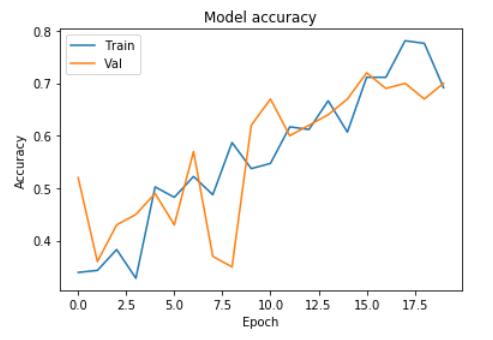
**Experiment #5:**

** **

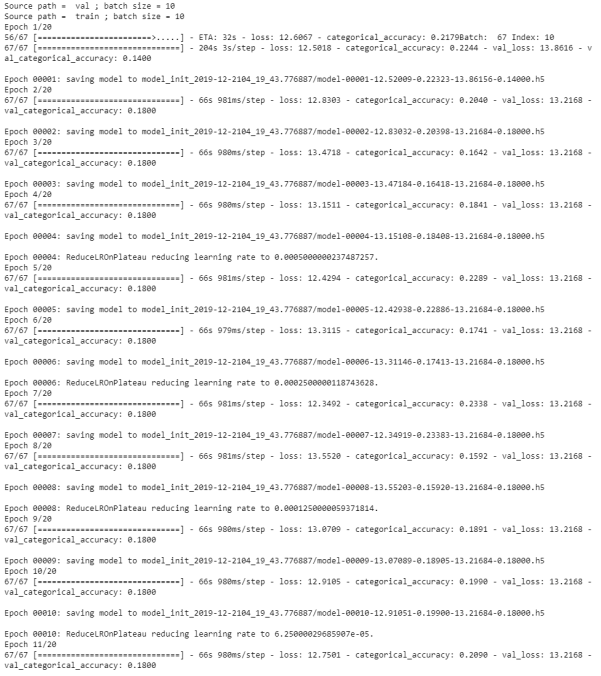
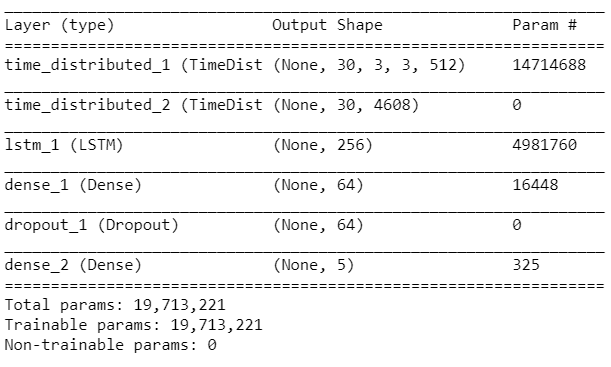
**Experiments #6 (final model with Conv3D):**

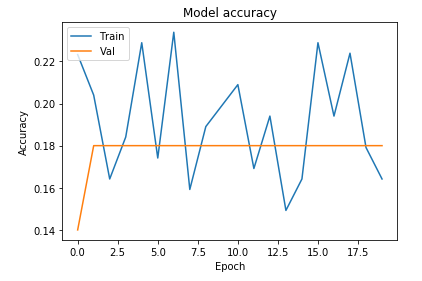
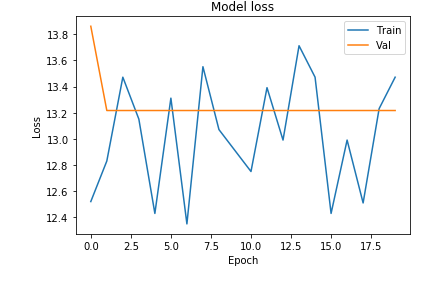




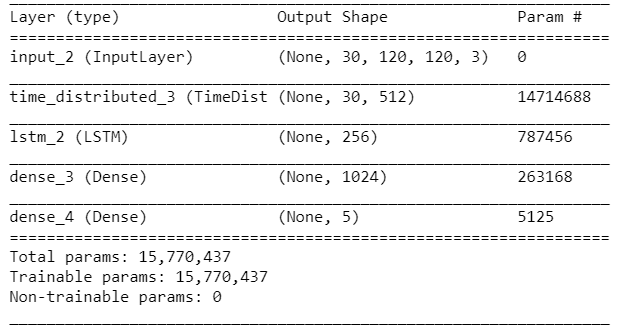


**Experiment #8 (includes pre-trained VGG16):**

****

** **

**Experiment #9 (includes pre-trained VGG16):**

****

OOM – out of memory error