**Model Experiments Details:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sl no** | **Model** | **Architecture** | **Result** | **Decision + Explanation** |
| 1 | Conv 3D | 1 input layer  1 output layer  2 hidden layers  2 dense layers with drop out value 0.5  Filter size = (3,3,3) | Accuracy :  Train = 47%  Val = 28% | Batch size = 10  No of epochs = 10  Learning rate = 0.0001.  Since, the accuracy was very low, we decided to increase the no of hidden layers. |
| 2 | Conv 3D | 1 input layer  1 output layer  4 hidden layers  2 dense layers with drop out value 0.5  Filter size = (3,3,3) | Accuracy :  Train = 43%  Val = 57% | Batch size = 50  No of epochs = 10  Learning rate = 0.0001.  Since, the accuracy can be further improved, we decided to increase no of epochs. |
| 3 | Conv 3D | 1 input layer  1 output layer  4 hidden layers  2 dense layers with drop out value 0.5  Filter size = (3,3,3) | Accuracy :  Train = 78%  Val = 75% | Batch size = 50  No of epochs = 40  Learning rate = 0.0001.  Since, the accuracy can be further improved, we decided to check for different learning rate and filter size. |
| 4 | Conv 3D | 1 input layer  1 output layer  4 hidden layers  2 dense layers with drop out value 0.5  Filter size = (4,4,4) | Accuracy :  Train = 72%  Val = 73% | Batch size = 50  No of epochs = 40  Learning rate = 0.0001.  Changing the filter size didn’t lead to better results. Hence, we reverted back to filter size (3,3,3) and tried to increase learning rate. |
| 5 | **Conv 3D** | 1 input layer  1 output layer  4 hidden layers  2 dense layers with drop out value 0.5  Filter size = (3,3,3) | **Accuracy :**  **Train = 81%**  **Val = 81%** | Batch size = 50  No of epochs = 50  Learning rate = 0.001.  Tried tweaking lr and batch sizes and epochs. We couldn’t improve further on the accuracy. Hence, decided to freeze the model at **accuracy 81%. Best epoc = 46.**  For Accuracy and Loss plot refer figure 1 below. |
|  |  |  |  |  |
| 6 | Conv 2D + LSTM | 1 input layer  1 output layer  2 hidden layers  2 dense layers with drop out value 0.25  Filter size = (3,3) | Accuracy:  Train = 23%  Val = 20% | Batch size = 10  No of epochs = 10  Learning rate = 0.0001.  Since, accuracy was very low we tried to increase the no of layers. |
| 7 | Conv 2D + LSTM | 1 input layer  1 output layer  4 hidden layers  2 dense layers with drop out value 0.25  Filter size = (3,3) | Accuracy:  Train = 100%  Val = 67% | Batch size = 30  No of epochs = 30  Learning rate = 0.0001.  Since, model was over-fitting. We decided to try for GRU over LSTM. |
| 8 | Conv2D + GRU | 1 input layer  1 output layer  4 hidden layers  2 dense layers with drop out value 0.50  Filter size = (3,3) | Accuracy:  Train = 86%  Val = 62% | Tried with multiple batch size, epochs and learning rate. But, still faced overfitting. Hence, decided to try for transfer learning with ResNet. |
| 9 | ResNet |  | Nimblebox crashed | Not able to download the model and Nimblebox crashed. Hence, decided to try with MobileNet. |
| 10 | MobileNet |  | OOM Error | Throws OOM(Out Of Memory) Error for batch size = 50. Hence, decided to reduce batch size. |
| 11 | **MobileNet** |  | Accuracy:  Train = 81%  Val = 81% | Batch size = 10  No of epochs = 25  Learning rate = 0.0001  We freeze the model at **accuracy 81%.**  Good accuracy at **epoc =16**  For Accuracy and Loss plot refer figure 1 below. |
| **Final** | **Conv 3D or MobileNet** |  | **Accuracy:**  **Train = 81%**  **Val = 81%** | Both **Conv 3D** and **MobileNet** give a **accuracy of 81%** and has the **.h5 file size around 45MB**. So we can use any one of the above.  Since, we had the constraint to upload zip file size of <=50MB. We have uploaded only Conv 3D .h5 file.  The .h5 file for MobileNet can be found in the below mentioned google drive. |

**Note: The python code file in the submission folder contains only the 2 successful models selected (i.e., Conv 3D and MobileNet models). Code file to other tried models can be found in the google drive** <https://drive.google.com/file/d/1vODfc7bohbco3UUbdsQEqiH6pO7k9_go/view?usp=sharing>.

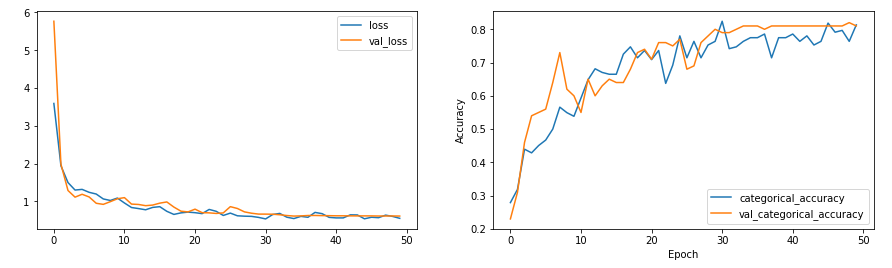


Figure1



Figure 2