**Gesture Recognition Project**

This project is meant to recognize 5 hand gestures. We have tested different models including Conv3D and pretrained network using Resnet. The main model that we experimented with different parameters was the Conv3D model. The model was trained using a Generator that selects batches of different gestures where each gesture consists of 30 frames.

**Generator:** The Generator was created to handle the full train and val data sets. The datasets were divided into batches of fixed size. The remaining datapoints after selecting the full batches were handled using an extra batch of less size. The generator also handled images to standardize the size and the selected frames.

**Images Preprocessing:** After analyzing the images and frames, we found the following:

1. All Gestures are made of 30 frames (images).
2. All images are of 2 sizes (160x120x3) or (360x360x3). We resized all images to a common standard size of 120x120x3 after analyzing the images .
   1. Images of 360x360 were resized to 120x120
   2. Images of 160x120x3 we cropped the images based on the gesture:
      1. Left Swipe: Crop 10 pixels from left and 30 from right for right.
      2. Right Swipe: Crop 10 pixels from Right and 30 pixels from left.
      3. Thumb Up/Down and Stop: Crop 20 pixels from left and 20 pixels from right.
3. Regarding the frames, we considered 10 out of the 30 frames in our model. We noticed that in Left and Right Swipe, the middle frames cover most of the movements, while in Thumb UP/Down and Stop, the beginning frames are most critical.
   1. Left and right swipe, we selected the frames: [0,5,9,11,13,15,17,20,24,29]
   2. Thumb Up/Down and stop we selected [0,3,5,7,9,12,15,19,24,29]
4. We Normalized all images dividing the pixels values by 255 as the images are RGB.

**Batches:**

We have tested the generator with different batch sizes. Batch size of 5 is very slow. So we will use 15 -40 batch sizes.

**Model:**

We have tested the model with different hidden layers, convolutions, and epochs. The best was achieved using 2 hidden Conv3D layers, each followed by a Maxpool3D layer, followed by flatten and 2 dense layers. On testing the model we noticed over fitting with train accuracy reaching almost 1 while val accuracy has max around 70%. We added 2 dropouts of 0.15 and 0.1 and handle over fitting. This increased the val accuracy to over 75%. The best val accuracy we achieved was 80%, which is good.

Here is a summary of some of the test.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Result** | **Decision + Explanation** | **Parameters** |
| **0** | **Conv3D (batch size =5** | **Very slow model** | **Reduced the batch size and tested many sizes < 10.** |  |
| **1** | **Conv3D: Base Model - Batch Size = 40 and No. of Epochs = 15** | **Trained badly, suffered overfitting; Very high train accuracy, but val accuracy was less** | **Overfitting**  **Lets add some dropout layers and increase epochs** | 1117061 |
| **2** | **Conv3D: Adding dropout layers - Batch Size = 20 and No. of Epochs = 25 to base Model** | **Complicated the model and increased parameters without increasing accuracy** | **Val\_loss did’nt improve. Early stopping at 11 epoch** | 3638981 |
| **3** | **Conv3D: Reduce filter size to (2,2,2) and image res to 120 x 120, - Batch Size = 30 and No. of Epochs = 25** | **Model has a best validation accuracy of 20% and training accuracy of 70%. Need to early stop at 11 epoch.** | **Also we were able to reduce the parameter size by half the earlier model. Let's trying adding more layers** | 1762613 |
| **4** | **Conv3D: Adding more layers - Batch Size = 20 and No. of Epochs = 25** | **With more layers we dont see much performance improvement. We get a best validation accuracy of 36% .** | **Let's try adding dropouts at the convolution layers**[**¶**](http://localhost:8888/notebooks/UPGRAD/Course6/Module6%20-%20RNN%20in%20Python/Last_Model/gesture_recognition/Neural_Nets_Project.ipynb#With-more-layers-we-dont-see-much-performance-improvement.-We-get-a-best-validation-accuracy-of-36%-.-Let's-try-adding-dropouts-at-the-convolution-layers) | 2556533 |
| **5** | **Conv3D: Adding dropout at convolution layers** | **Adding dropouts has further reduced validation accuracy as the model doesn't seem to generalise well.** | **All the experimental models above have more than 1 million parameters. Let's try to reduce the model size and see the performance** | 2556533 |
| **6** | **Conv3D: Reducing the number of parameters** | **For the this low memory foot print model, we get the best validation accuracy of 25%** | **Let’s Reducing the number of parameters again and see result** | 696645 |
| **7** | **Conv3D: Reducing the number of parameters** | **For the this low memory foot print model the best validation accuracy of 34%** | **We see increase in accuracy after decreasing parameters** | 504709 |
|  |  |  |  |  |
| **Final Model** | **CNN- LSTM** | **Accuracy: 44%** |  |  |

**Generator with rotated images: We rotated the image slightly to feed the model and see result.**

**Below models have slightly rotated image**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Result** | **Decision + Explanation** |  |
| **9** | **Conv3D: (3,3,3) Filter & 160x160 Image resolution - similar to Model 2** | **For CNN model with augmentatio we get a best train accuracy of 70% and validation accuracy of 70%** | **Add dropouts layers to handle overfitting** | 3638981 |
| **10** | **Conv3D: (2,2,2) Filter & 120x120 Image resolution - similar to Model 3** | **Validation accuracy is decreased.** |  | 1762613 |
| **11** | **Conv3D: Similar to model 4** | **For CNN model with augmentation we get a best train accuracy of 75% and validation accuracy of 75%** | **20 or 25 epochs lead to best accuracy 75%. Let’s try adding dropouts** | 2556533 |
| **12** | **Conv3D: Similar to model 5** | **Model is overfitting badly!!. Very bad validation accuracy only 34%** | **Let’s reduce network parameters - Similar to Model 6** | 2556533 |
| **13** | **Conv3D: Similar to model 6** | **Got train accuracy of 77% and validation accuracy of 75%** | **Let’s further reduce network parameters - Similar to Model7** | 696645 |
| **14** | **Conv3D: Similar to model 7** | **Best result till now train accuracy of 75% and validation accuracy of 80%** | **For gesture recognition we need less parameters to get better results. Let’s CNN LSTM with GRU** | 504709 |
| **15** | **CNN LSTM with GRU - Similar to Model 8** | **We see that overfitting is considerably higher** | **Lets try transfer now** | 2573925 |
| **16** | **Using Mobile Net model due to its lightweight architecture and high speed performance as compared**  **# to other heavy-duty models like VGG16, Alexnet, InceptionV3 et** | **We are not training the mobilenet weights and we see validation accuracy is very poor. Let's train them as well and observe if there is performance improvement**[**¶**](http://localhost:8888/notebooks/UPGRAD/Course6/Module6%20-%20RNN%20in%20Python/Last_Model/gesture_recognition/Neural_Nets_Project.ipynb#We-are-not-training-the-mobilenet-weights-and-we-see-validation-accuracy-is-very-poor.-Let's-train-them-as-well-and-observe-if-there-is-performance-improvement) | **Lets try Transfer Learning with GRU and training all weights** | 3840453 |
| **17** | **Model 17 - Transfer Learning with GRU and training all weights** | **Awesome results! 99% Training accuracy and 95% validation accuracy :)** |  | 3693253 |
|  |  |  |  |  |
| **Final Model** | **Conv3D** | **Accuracy: 80%** |  |  |

**Final Result:** **After doing all the experiments, we finalized Model 14 - CNN, which performed well.**

Reason:

* (Training Accuracy: 75%, Validation Accuracy: 80%)
* Number of Parameters(696645)less according to other models perform
* Model performs well when the number of parameters is reduced

The best weights of CNN-LSTM: model-00025-0.62456-0.75339-0.63580-0.80000.h5 (8.2 MB)