**Gesture Recognition Experiments Write-up**

This exercise was to do Gesture Recognition using 2 methods

* **Convolution 2D followed by RNN**
* **Convolution 3D**

We experimented extensively with both the approaches by running the code over and over again for various experiments. This clearly shows that we need to do a lot of experimentation to fine-tune the hyper parameters to continously improve the accuracy of the model.

**We have put table & some graphs at the end that indicate how learning rate and drop-outs impacted the model.**

Overall it was a great learning to see how we can use CNN and RNN can be used to analyze images as well as videos

Tabulated results below:

1. Field experiment details shows the hyperparameter value for the experiment
2. Field Model is the model for which the experiment was run
3. Field result shows the error/accuracy that was obtained after running the experiment.
4. Field decision + explanation what was the next decision taken to mover forward

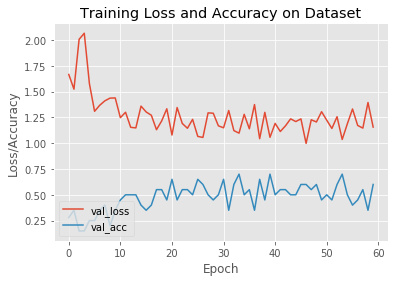
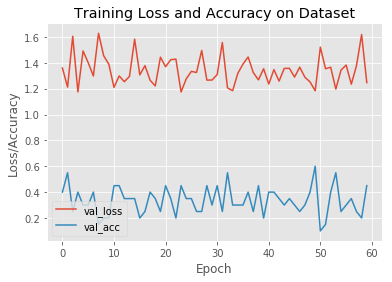
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Model** | **Experiment Details** | **Result** | **Decision + Explanation** |
| 1 | Conv3D | No cropping of images | Throws Generator error | Crop the images correctly to size 120 by 120, try to overfit on less amount of data |
| 2 | Conv3D | Batch size = 100 | Throws Memory Error | Overloading the memory. Reduce the batch size to 90 |
| 3 | Conv3D | no of images = 15 | Accuracy: 0.21 | Increase the no of images to be used to increase accuracy Used no of images = 30 (all images) |
| 4 | Conv3D | Epoch = 60 | Accuracy: 0.3 | Increased the no of epochs from 30 to 50 to ensure more learning from various combinations of the data |
| 5 | Conv3D | Add batch normalization after each layer | Accuracy: 0.346 | Batch Normalization helped normalize the inputs layer by adjusting and scaling the activation. |
| 6 | Conv3D | Add max pooling layer after each layer | Accuracy: 0.4 | Max pooling helped remove the dimensionality and allowed for assumptions to be made about the features to be contained. It helped removed overfitting |
| 7 | Conv3D | Additional feature map added 4 feature maps layers [8,16,32, 64] | Accuracy: 0.52 | Additional feature maps layers helped improved accuracy and identification of the gesture |
| 8 | Conv3D | Additional dense layer added. 2 dense layers [264, 128] | Accuracy: 0.58 | Additional dense layer helped improve accuracy further |
| 9 | Conv3D | Kernel = [1,3,3] for last 2 layers | Accuracy: 0.62 | Improved the accuracy further as more feature details were getting uncovered. |
| 10 | Conv3D | Learning rate = 0.1 /0.4/0.45 | Accuracy: 0.4 to 0.8 | Accuracy fluctuating from 0.4 to 0.8 indicating that the model was highly unstable and hence may not perform well on real data |
| **11** | **Conv3D**  **Final Model** | **Learning rate = 0.35** | **Validation Accuracy: 0.65**  **Training Accuracy: 0.58** | **Decent accuracy with steady learning rate as shown in the output generated by the program. Model is more stable as compared to other learning rate** |
| 12 | Conv2D+ GRU | First run | Accuracy: 0.635 | Batch Normalization helped normalize the inputs layer by adjusting and scaling the activation. |
| 13 | Conv2D+ GRU | Add batch normalization after each layer | Accuracy: 0.74 | Batch Normalization helped normalize the inputs layer by adjusting and scaling the activation. |
| 16 | Conv2D+ GRU | Additional GRU layer added. 2 GRU layers [128, 64] | Accuracy: 0.78 | Additional GRU layer helped improve accuracy further |
| 18 | Conv2D+ GRU | Learning rate = 0.1 /0.4/0.45 | Accuracy: 0.4 to 0.8 | Accuracy fluctuating from 0.4 to 0.8 indicating that the model was highly unstable and hence may not perform well on real data |
| 19 | Conv2D+ GRU | Learning rate = 0.4 |  |  |
| **20** | **Conv2D+ GRU** | **Learning rate = 0.35** | **Validation Accuracy: 0.65**  **Training Accuracy: 0.58** | **Decent accuracy with steady learning rate as shown in the output generated by the program. Model is more stable as compared to other learning rate** |

**Conv3d Model Graphs**

**Learning Rate = 0.45 Learning Rate = 0.40**

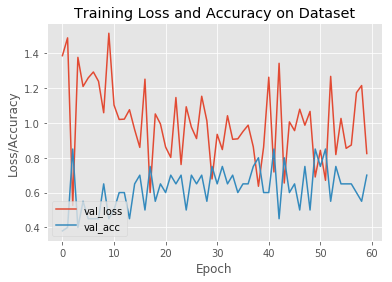
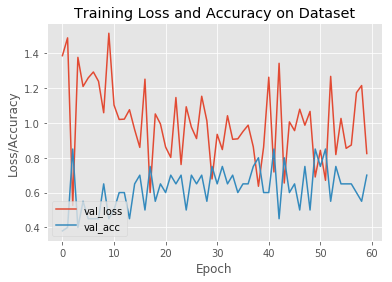
 

**Learning Rate = 0.35 Learning Rate = 0.35 & Drop out added in 2nd last layer**

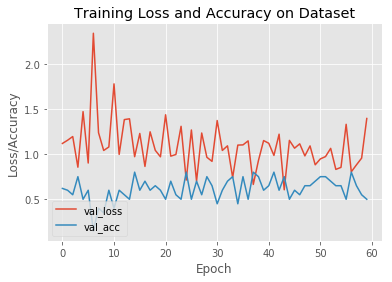
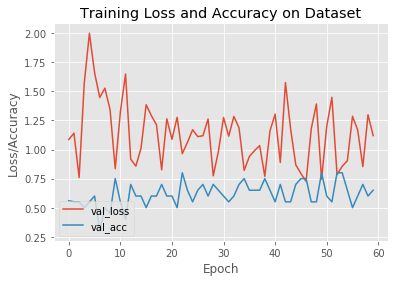
****

**Conv2d + RNN Model**

**Learning Rate = 0.45 Learning Rate = 0.40**

**Learning Rate = 0.35 Learning Rate = 0.35 & Drop out added in 2nd last layer**

 ****

**Learning Rate = 0.35 & Drop out added in 2nd last layer & GRU layer = [128, 64]**

