We have conducted multiple experiments on Conv2D+RNN and Conv3D network in parallel, primary objective of the experiments conducted is to build an efficient neural network which results in highest accuracy on training and validation data with less number of parameters.

Problem statement states that the model should be efficient in terms of accuracy and take less storage space as it is going to run on a webcam ,Hence final decision of our model selection is based on the factors below -:

1. Training and Validation accuracy
2. Model size
3. Training time

**Experiment with Generator**

We decided to build a common generator which can be used with both Conv2D+RNN and Conv3D network for experimentation.

We followed the following steps in sequence based on results achieved during experimentation.

1. Firstly, all the images (30) were considered for experiment and normalization of images was done by dividing the image array by 255.
2. Number of images reduced to 15. Indexes used are-:

img\_idx = [0,2,4,6,8,10,12,14,16,18,20,22,24,26,28]

1. Updated Normalization technique , Instead of division by 255 we followed percentile approach. Percentile technique resulted in better results for both models.

batch\_data[folder,idx,:,:,0] = (image[:,:,0]-np.percentile(image[:,:,0],5))/(np.percentile(image[:,:,0],95)-np.percentile(image[:,:,0],5))

batch\_data[folder,idx,:,:,1] = (image[:,:,1]- np.percentile(image[:,:,1],5))/(np.percentile(image[:,:,1],95)-np.percentile(image[:,:,1],5))

batch\_data[folder,idx,:,:,2] = (image[:,:,2]- np.percentile(image[:,:,2],5))/(np.percentile(image[:,:,2],95)-np.percentile(image[:,:,2],5))

**Experiment on CNN+RNN**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Parameters** | **Result** | **Decision + Explanation** |
| 1 | Conv2D + RNN | Pretrained Conv2d-VGG16  batch\_size = 5 img\_size = 120,120 img\_indx = 0-29 (30) normalization = Division by 255 epochs = 2  Learning Rate = 0.001 Trainable params: 306,821 callbacks = [checkpoint] | validation\_accuracy = 23% validation\_loss = 1.6082 | Experiment conducted with batch size of 5 and normalization of images was done by dividing the image array by 255.  Accuracy came out to be 23%, hence decided to increase the batch size and reduce the number of images. |
| 2 | Conv2D + RNN | Pretrained Conv2d-VGG16  batch\_size = 15 img\_size = 120,120 img\_indx = 15 normalization = Division by 255 epochs = 2 Learning Rate = 0.01 Trainable params: 306,821 callbacks = [checkpoint] | validation\_accuracy = 20% validation\_loss = 1.5708 | Second experiment with 2 epochs resulted in accuracy of 20%. Decided to increase the number of epochs and normalization technique. |
| 3 | Conv2D +RNN | Pretrained Conv2d-VGG16  batch\_size = 15 img\_size = 120,120 img\_indx = 15 normalization = percentile epochs = 20 Learning Rate = 0.001 Trainable params: 306,821 callbacks = [checkpoint] | validation\_accuracy = 22% validation\_loss = 1.6091 | Accuracy came out to be 22% after 20 epochs, decided to introduce LRReduceonPlateau |
| 4 | Conv2D+RNN | Pretrained Conv2d-VGG16  batch\_size = 15 img\_size = 120,120 img\_indx = 15 normalization =percentile epochs = 20 Learning Rate = 0.01 Trainable params: 306,821 callbacks = [checkpoint] | validation\_accuracy = 47% validation\_loss = 1.5160 | Accuracy improved from 20% to 47% after adding LRReduceonPlateau.  Started with initial learning rate 0.01.  Decided to change image resizing approach. |
| 5 | Conv2D+RNN | Pretrained Conv2d-VGG16  batch\_size = 15 img\_size = 120,120 img\_indx = 15 normalization = percentile epochs = 50 Learning Rate = 0.01 Trainable params: 306,821 callbacks = [checkpoint] | validation\_accuracy = 63% validation\_loss = 1.1749 | Resizing of image and increase in number of epochs resulted in improvement of validation accuracy from 47% to 63%. |
| 6 | Conv2D+RNN | Pretrained Conv2d-VGG16  batch\_size = 15 img\_size = 120,120 img\_indx = 15 normalization = percentile epochs = 50 Learning Rate = 0.01 Trainable params: 306,821 callbacks = [checkpoint] | validation\_accuracy = 65% validation\_loss = 1.01384 | Added one more GRU layer along with dropout resulted in improved accuracy of 65%.  Decided to experiment with VGG16 as pre-trained model |
| 7 | Conv2D+RNN  (Final Model) | Pretrained Conv2d-VGG16  batch\_size = 15 img\_size = 120,120 img\_indx = 15 normalization = percentile epochs = 50 Learning Rate = 0.01 callbacks = [checkpoint] | Training \_accuracy: 0.95 val\_loss: 0.52  Val\_accuracy: 0.79 | Training , Validation accuracies and loss improved, best model resulted in -**Training \_accuracy: 0.95 val\_loss: 0.52**  **Val\_accuracy: 0.79**  **Model size : 65 MB** |

**Experiment on Conv3D**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Parameters** | **Result** | **Decision + Explanation** |
| 1 | Conv3D | *batch\_size = 30*  *final\_img\_size=(120,120)*  *No cropping used*  *images = 30*  *epochs = 10*  *num\_classes = 5*  *learning\_rate = 0.001*  *validation\_split = 0.2*  *verbosity = 1*  *Three Con3d layers (64,32,16 kernels respectively)+dense layer(256)*  *optimizer = Adam*  *callbacks used* | *validation\_accuracy = 22% validation\_loss = 12.57*  *training\_accuracy -18.87%* | Experiment conducted with batch size of 30 and normalization of images was done by dividing the image array by 255.  Accuracy came out to be 22%, hence decided to use cropping ,reduce batch size and change number of convolution kernels. |
| 2 | Conv3D | *batch\_size = 10*  *final\_img\_size=(120,120) Cropped images*  *images = 15*  *epochs = 15*  *num\_classes = 5*  *learning\_rate = 0.001*  *validation\_split = 0.2*  *verbosity = 1*  *Three Con3d(32,64,128 kernels)+ dense layer(32)*  *optimizer = Adam* | *validation\_accuracy = 51% validation\_loss = 1.28*  *training\_accuracy -80.87%*  *Overfitting issue* | Accuracies improved but resulted in overfitting. Hence decided to increase batch size,use batch normalization along with drop out of 0.3 after dense layer. |
| 3 | Conv3D | *batch\_size = 15*  *final\_img\_size=(120,120) Cropped images*  *images = 15*  *epochs = 30*  *num\_classes = 5*  *learning\_rate = 0.001*  *validation\_split = 0.2*  *verbosity = 1*  *Three Con3d(32,64,128 kernels)+ dense layer(32)*  *optimizer = Adam* | *validation\_accuracy = 65% validation\_loss = 0.8*  *training\_accuracy -66.87%* | Results improved after adding batch normalization and dropout of 0.3.  Decided to reduce epochs and use SGD optimizer . |
| 4 | Conv3D  **(Final Model)** | *Same as above ,*  *Optimizer – SGD*  *Kernel size – 16,32,64* | *validation\_accuracy = 79% validation\_loss = 0.57*  *training\_accuracy -92.75%* | Training and validation accuracies improved by using SGD optimizer with learning rate of 0.001. |

**Final Model**

Both Conv2d+RNN and CONV3D models have given similar validation accuracy of 79%.

We have decided to select artifact generated by Conv3D architecture as our final model for submission because it best meets the business objective. Model generated by Conv3D is memory efficient and taking less training time and epochs.

Number of epochs taken by Conv3D to reach the convergence is 20, whereas CNN+RNN model is taking 50 epochs to produce similar results.