CNN + RNN MODEL BUILDING

# Model Strategy

1. CNN with 2d convolutions through transfer learning
2. Model used: MobileNetV2 with **include\_top = False**
3. GRU RNN with flattening and 1 dense layer and 1 output layer
4. MobileNetV2 was used mainly because:
   1. it’s a Lite model and will give faster results
   2. its accuracy is comparable to that of ResNet models
5. GRU was also chosen because of the lower number of gates and comparable performance to LSTM
6. The whole CNN model was first created and TimeDistributed and taken as input (in 1 step) to the RNN model
7. Upon this, the GRU was added
8. Finally, 1 dense layer with RELU activation and 1 output layer with 5 outputs (corresponding to 5 gestures) was added

# Experiments Strategy

1. A number of experiments were conducted with the Validation Accuracy being the main metric.
2. The Validation Loss was also checked for the run models, just to ensure that it shows consistent values
3. Following parameters were changed and different models tested:
   1. No. of neurons in the Dense Layer of CNN
   2. No. of neurons in the Dense Layer of GRU
   3. No. of units of GRU
   4. The Learning Rate initial value
   5. Regularization (added and removed – outputs of Regularized model not shared here)
   6. Different batch-sizes (20, 30, 40, 60, 90). Finally all the experiments shared here, have been run with a batch-size of 90.
   7. Different expts were run for different no. of epochs and some were discarded after a small no. of epochs

# Best Model:

* MobileNetV2, 96x96, not\_trainable, Dense 64, No Batch Normalization (BN)
* GRU, 64, Dense 64
* Dropouts: None
* Batch Size: 90
* Images/sequence: 15
* Epochs: 15
* Augmentation: Affine, Noise (uniform)
* LR start: not specified

Metrics:

Cat Acc (Training): 0.8461 (10th Epoch)

Cat Acc (Val) : 0.80 (4th Epoch)

Cat Acc (Training) at 4th Epoch: 0.7523 (here, Val Acc was max 0.80)

Cat Acc (Val) at 10th Epoch: 0.7700 (here, Training Acc was max 0.8461)

Cat Loss (Training, 4th Epoch): 0.6752

Cat Loss (Val, 4th Epoch) : 0.5690

Cat Loss (Training, 10th Epoch): 0.3812

Val Loss (Val, 10th Epoch) : 0.6524

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train Acc** | **Val Acc** |  | **Train Loss** | **Val Loss** |
| 0.3876 | 0.68 |  | 1.4283 | 1.0427 |
| 0.602 | 0.71 |  | 1.0128 | 0.7658 |
| 0.6987 | 0.66 |  | 0.8289 | 0.833 |
| 0.7406 | 0.76 |  | 0.697 | 0.6347 |
| 0.7885 | 0.77 |  | 0.5948 | 0.6318 |
| 0.7908 | 0.76 |  | 0.541 | 0.6417 |
| 0.8211 | 0.79 |  | 0.4734 | 0.598 |
| 0.8398 | 0.8 |  | 0.434 | 0.6725 |
| 0.8447 | 0.74 |  | 0.4028 | 0.7252 |
| 0.8594 | 0.76 |  | 0.3785 | 0.6792 |
| 0.8645 | 0.78 |  | 0.3655 | 0.6552 |
| 0.8624 | 0.76 |  | 0.3487 | 0.7462 |
| 0.8641 | 0.77 |  | 0.3415 | 0.6884 |
| 0.8758 | 0.77 |  | 0.3308 | 0.7052 |
| 0.8702 | 0.78 |  | 0.3261 | 0.7227 |

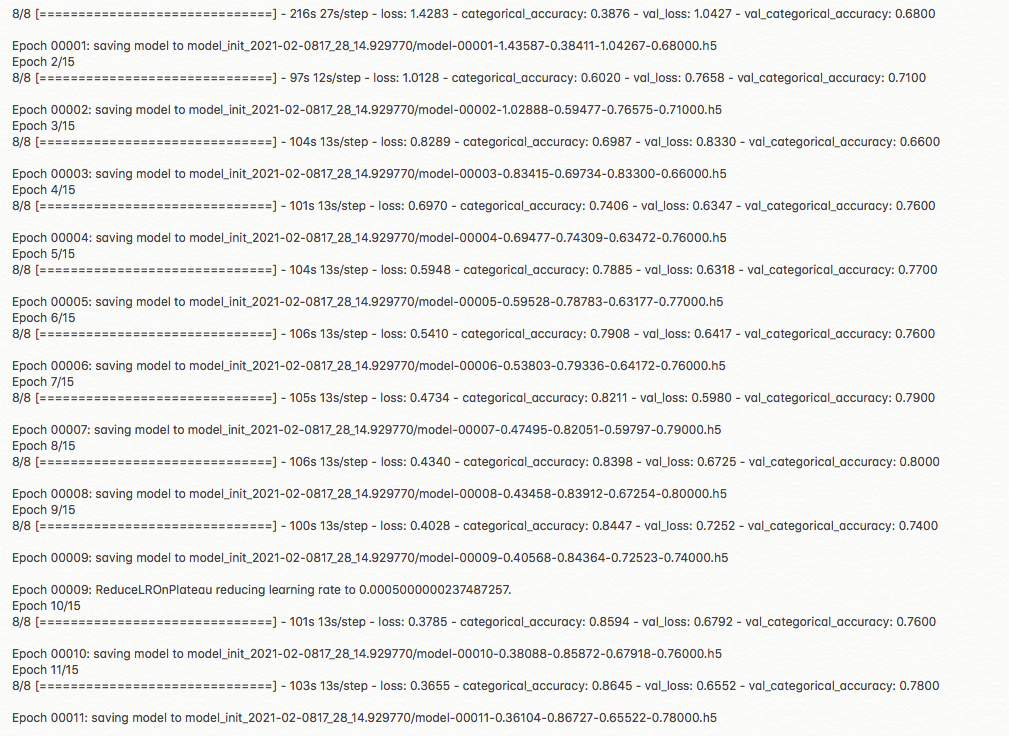
Fig 1: (Left): Training Accuracy & Validation Accuracy (Right): Training Loss & Validation Loss

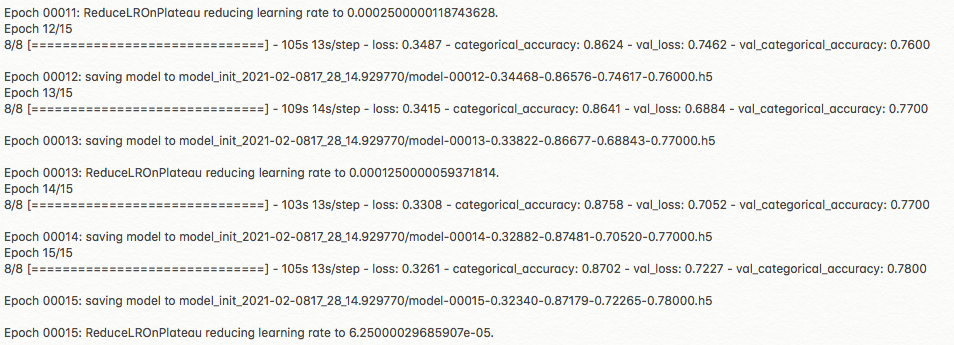
 

**Conclusion on the best model:**

1. We can see that there is no overfitting if we compare the training and validation accuracies
2. The max training and validation accuracies are close
3. In order to give more data for training, Augmentation is done is the form of 2 transforms: a **shear** Affine Transform and a noisy image, by adding uniform random noise to the image
4. However 2 surprising aspects were observed:
   1. The best results were obtained **only when** No Normalization was used and No Dropouts were used
   2. In this, as well as in other experiments, Validation loss and accuracy values **do not** increase/decrease monotonically
5. Another observation is that, in certain epochs towards the beginning, the validation accuracy is higher than the training accuracy! When we did some research/study on the Internet, it was found that this can happen since Augmentation is used only on the Training set and not on the Validation set.

Below is a screenshot of the outputs of this model. The h5 files of this model have been submitted with the assignment. This is not given for all models. The last h5 file in this set of screenshots is shared with the submission.





Various experiments done

## Model 1

* MobileNetV2, 96x96, not\_trainable, Dense 64, No BN
* GRU, 64, Dense 64
* Dropouts 0.25
* Batch Size: 90
* Images/sequence: 15
* Epochs: 20
* Augmentation: None
* LR start = 0.1

Metrics:

Cat Acc (Training): 0.20

Cat Acc (Val) : 0.18

Cat Loss (Training): 1.6107

Cat Loss (Val). : 1.6101

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train Acc** | **Val Acc** |  | **Train Loss** | **Val Loss** |
| 0.1972 | 0.23 |  | 2.1263 | 1.6081 |
| 0.1929 | 0.21 |  | 1.623 | 1.6152 |
| 0.2035 | 0.23 |  | 1.6133 | 1.6054 |
| 0.1985 | 0.23 |  | 1.6124 | 1.6052 |
| 0.1825 | 0.21 |  | 1.6113 | 1.6062 |
| 0.2049 | 0.21 |  | 1.6119 | 1.6067 |
| 0.2049 | 0.18 |  | 1.6107 | 1.6101 |

### Conclusion on Model 1

* 1. This model has very low Training as well as Validation accuracy
  2. It could be because of a specifically initialized Learning Rate
  3. Further, adding dropouts to the previous model (the best performing model) appears to have resulted in this drastic change

## Model 2

* MobileNetV2, 96x96, not\_trainable, Dense 64, No BN
* GRU, 64, Dense 64
* Dropouts 0.25
* Batch Size: 90
* Images/sequence: 15
* Epochs: 20
* Augmentation: None
* LR start: not specified

Metrics:

Cat Acc (Training): 0.80

Cat Acc (Val) : 0.51

Cat Loss (Training): 0.5166

Cat Loss (Val) : 1.2907

### Conclusion on Model 2

1. The only difference between Model 1 and Model 2 is that the **Learning Rate** initial value is specified here
2. This has increased the accuracy quite significantly
3. Of course, in spite of this, the validation accuracy is not good
4. All through the experiments on RNN, we found that adding BN and dropouts, which were supposed to help reduce overfitting, in fact ended up not performing well at all.
5. The model that performed best was the one without any BN and dropouts!

## Model 3

* MobileNetV2, 96x96, not\_trainable, Dense 512, No BN
* GRU: 512 Dense: 512
* Dropouts: None
* Batch Size: 90
* Images/sequence: 15
* Epochs: 8
* Augmentation: Affine, noise
* LR start = not specified

Cat Acc (Training): 0.8765

Cat Acc (Val) : 0.6200

Cat Loss (Training): 0.3298

Cat Loss (Val) : 1.1870

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train Acc** | **Val Acc** |  | **Train Loss** | **Val Loss** |
| 0.4344 | 0.4 |  | 1.7048 | 1.3943 |
| 0.6953 | 0.43 |  | 0.7629 | 1.591 |
| 0.7567 | 0.38 |  | 0.6233 | 1.7102 |
| 0.8073 | 0.55 |  | 0.4797 | 1.3187 |
| 0.8328 | 0.41 |  | 0.4302 | 1.9518 |
| 0.832 | 0.47 |  | 0.394 | 1.8633 |
| 0.8525 | 0.52 |  | 0.3505 | 1.5354 |
| 0.8765 | 0.62 |  | 0.3298 | 1.187 |

### Conclusion on Model 3

1. Again, the model is somewhat okay, but increasing the no. of neurons in the dense layer is not helping to increase accuracy, on comparing this to the best model (since both these models don’t have BN and dropouts)
2. Again, a wavy pattern is seen in the Cat Loss and Accuracy for the validation set

## Model 4

* MobileNetV2, 96x96, not\_trainable, Dense 64, BN
* GRU, 64, Dense 64
* Dropouts 0.25
* Batch Size: 90
* Images/sequence: 15
* Epochs: 20
* Augmentation: Affine, Noise

Cat Acc (Training): 0.8178 (max is actually 0.8295, on Epoch 8)

Cat Acc (Val) : 0.65

Cat Loss (Training): 0.4630

Cat Loss (Val) : 1.1847

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train Acc** | **Val Acc** |  | **Train Loss** | **Val Loss** |
| 0.3868 | 0.52 |  | 1.7363 | 1.3059 |
| 0.5924 | 0.56 |  | 1.0321 | 1.1843 |
| 0.6702 | 0.6 |  | 0.8411 | 1.1423 |
| 0.6957 | 0.66 |  | 0.7806 | 1.141 |
| 0.7243 | 0.62 |  | 0.7017 | 1.2313 |
| 0.7549 | 0.59 |  | 0.6264 | 1.3034 |
| 0.7986 | 0.6 |  | 0.5396 | 1.3093 |
| 0.8014 | 0.62 |  | 0.519 | 1.2519 |
| 0.7944 | 0.63 |  | 0.5323 | 1.235 |
| 0.7993 | 0.61 |  | 0.5058 | 1.2312 |
| 0.8156 | 0.59 |  | 0.4826 | 1.2309 |
| 0.8051 | 0.61 |  | 0.5 | 1.2199 |
| 0.8182 | 0.63 |  | 0.4689 | 0.8182 |
| 0.8295 | 0.64 |  | 0.4618 | 1.2052 |
| 0.8065 | 0.65 |  | 0.4855 | 1.2015 |
| 0.8227 | 0.65 |  | 0.4531 | 1.196 |
| 0.8168 | 0.65 |  | 0.4776 | 1.1921 |
| 0.8243 | 0.65 |  | 0.4574 | 1.1891 |
| 0.8137 | 0.65 |  | 0.4903 | 1.1867 |
| 0.8178 | 0.65 |  | 0.463 | 1.1847 |

### Conclusion on Model 4

1. This is a very good model that showed the importance of adding BN and dropouts to reduce overfitting
2. This was the best model obtained by adding BN and dropouts
3. However, this model, while it was expected to perform the best, never performed that great
4. Undoubtedly, **overfitting was handled to an extent**. However, the validation accuracy was also low
5. Further, the validation accuracy just got stuck to, possibly, a local minimum, multiple times, in spite of ReduceLRonPlateau. It can be seen that the validation loss and accuracy curves are flat towards the end. Actually, the validation accuracy did not increase much at all

## Model 5

* MobileNetV2, 96x96, not\_trainable, Dense 64, No BN, No Dropouts
* GRU, 64, Dense 64
* Dropouts: None
* Batch Size: 90
* Images/sequence: 15
* Epochs: 8
* Augmentation: None
* LR start: not specified

Cat Acc (Training): 1.0 (max is actually 0.8295, on Epoch 8)

Cat Acc (Val) : 0.8 (on Epoch 4)

Cat Loss (Training): 0.0525

Cat Loss (Val) : 0.5789 (on Epoch 4)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Train Acc** | **Val Acc** |  | **Train Loss** | **Val Loss** |
| 0.4275 | 0.61 |  | 1.3919 | 1.1009 |
| 0.7382 | 0.69 |  | 0.8721 | 0.8302 |
| 0.857 | 0.76 |  | 0.5425 | 0.707 |
| 0.9285 | 0.8 |  | 0.355 | 0.5789 |
| 0.9642 | 0.78 |  | 0.2238 | 0.5908 |
| 0.9804 | 0.79 |  | 0.1291 | 0.5801 |
| 0.9923 | 0.77 |  | 0.0777 | 0.5563 |
| 1 | 0.8 |  | 0.0525 | 0.5559 |

### Conclusion on Model 5

1. This was the precursor to the best model we got. This can be considered to exhibit a significant amount of overfitting. However, its training accuracy is also pretty good, when compared to other models of CNN + RNN.
2. It can also be seen here, that the validation loss is reducing almost monotonically
3. Surprisingly, this does not use any augmentation, any dropouts and any BN
4. This is the reason this model has been shown here

## Model 6

* MobileNetV2, 96x96, Trainable (4/5th of the layers), Dense 64, No BN
* GRU, 64, Dense 64
* Dropouts: None
* Batch Size: 90
* Images/sequence: 15
* Epochs: 8
* Augmentation: Affine, Noise

Cat Acc (Training): 0.8620

Cat Acc (Val) : 0.6400

Cat Loss (Training): 0.3365

Cat Loss (Val) : 1.0968

### Conclusion on Model 6

1. The main reason this has been given is that, **4/5 of the MobileNetV2 layers were set as trainable here. However no major improvement was observed**
2. The loss and accuracy tables have not been given for this since it performed somewhat similar to Model 4
3. This model also had no BN and dropouts
4. But, training the readymade NN did not help much

# Other experiments

For CNN + RNN many other experiments were conducted, but have not been shared here since it would make for a very bulky document.

Following were tried:

1. Different values of Batch Size (20, 30, 40, 60) along with the 90 above
2. Different values of neurons in the Dense layers for both CNN and RNN
3. Different augmentation combinations:
   1. Shear + Noise
   2. Noise + Edge detected image
   3. Edge Detected image + Shear
4. Only the best one was chosen from these and that was Shear + Noise
5. Some of these were tried:
   1. With initialized LR values
   2. Default LR values

Since none of these showed any great performance, they have not been analysed here