**Name: SM Arun Kumar**

**Batch: AIML C61 – 2024**

# Case Study: Gesture Recognition

## Generator Function:

1. img\_idx: Out of 30 frames, I have taken 15 images having every alternate images. ( i.e Odd numbered images starting from 1)
2. data\_multiplier: Code written to multiply the given data to increase the training set data. The final multiplied data is randomly sorted.
3. Data\_augmentation: Trained data is cropped at the top and bottom with 10 pixels. Further, options have been provided to add Gaussian Blur, Gaussian Noise, Contrast and Sharpness to the image OR retaining the original video frames. For a given video, one of augmentation will be added randomly.

The data augmentation can be controlled using a flag. This is mainly done to add more diversity to the training data to increase the robustness of the model.

1. The final resizing of the images are done for both training set and validation set.

## Generator Validation:

To validate the Generator function, small code has been written to check the output.

## Ablation Test:

This is code validates the Generator function and model fit on a small chunk of training data.

## Experiments with Conv3D:

1. Eperiment1:

Model1: Conv3D Base Model

Batch Size: 221

Decision: As mentioned in the video, kept a higher batch size that divisible by 663.

Result: After 1 Epoch run, model started running infinitely with jumping to the 2nd Epoch.

1. Experiment2:

Model1: Conv3D Base Model

Batch Size: 32, Optimizer: SGD

No of Batches: 21

Data used: 663

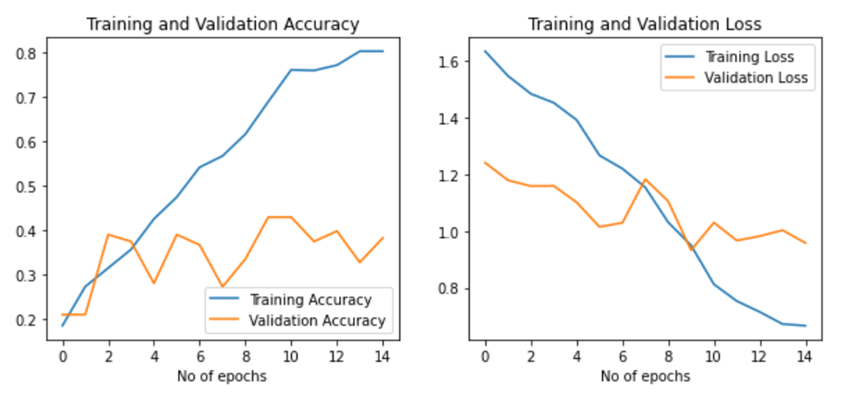
Trainable params: 47,728,005

Training accuracy: 80.21%

Validation accuracy: 38.28%

Decision: Kept lower Epoch with batch size 32.

Result: The convergence time is slow. There is lot of fluctuation in validation accuracy.



1. Experiment 3:

Model1: Conv3D Base Model

Batch Size: 32, Optimizer: Adam

Data used: 663

Trainable params: 47,728,005

Training accuracy:

Validation accuracy:

Decision: Changed the Optimizer to Adam.

Result: The convergence time is fast. The training data quickly converged to 98.66% at 6th Epoch and remained stable. There are no fluctuations observed on validation data, but the accuracy remained around 33 - 35%. This is clear sign of overfitting. The smoother output could be also attributed to batch size.

A graph of performance and training

Description automatically generated with medium confidence

1. Experiment 4:

Model1: Conv3D Base Model

Batch Size: 16, Optimizer: Adam

Data used: 623

Trainable params: 47,728,005

Training accuracy: 99.55%

Validation accuracy: 26- 46%

Decision: Batch size decreased from 32 to 16

Result: Training accuracy 98.96% and val accuracy: 69%. The training data accuracy reached about 99.55% at 8th Epoch. The validation accuracy is fluctuating between 26% to 46%. The validation loss started increasing. So, reducing the batch size deteriorated the model.

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated

1. Experiment 5:

Model 2: Conv3D Base Model with Dropout

Batch Size: 32, Optimizer: Adam

Data used: 623

Trainable params: 47,728,005

Training accuracy: 88.69%

Validation accuracy: 28.12%

Decision:

Result:

A graph of performance indicators

Description automatically generated with medium confidence

1. Experiment 6:

Model 2: Conv3D Base Model with Dropout

Batch Size: 16, Optimizer: Adam

Data used: 623

Trainable params: 47,728,005

Training accuracy: 91.96%

Validation accuracy: 33.04%

Decision:

Result:

A graph of training and training accuracy

Description automatically generated

1. Experiment 7:

Model 3: Conv3D Base Model with Dropout and Batch Normalization

Batch Size: 32, Optimizer: Adam

Data used: 623

Trainable params: 21,726,789

Non-Trainable params: 192

Training accuracy: 27-32%

Validation accuracy: 28%

Decision: Added Batch Normalization along with Drop out to handle overfitting.

Result:

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated

1. Experiment 8:  
   Model 4: Conv3D with stack same filters back-to-back

Batch Size: 32, Optimizer: Adam

Data used: 623

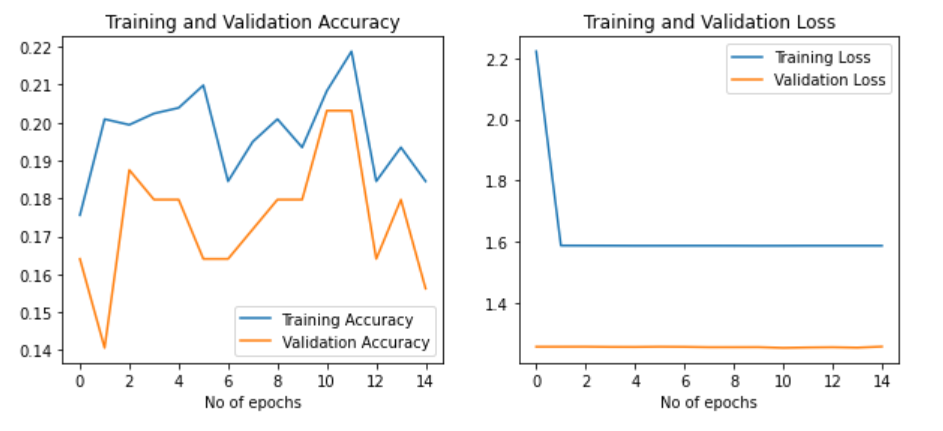
Trainable params: 27,951,589

Training accuracy:

Validation accuracy:

Decision: The model4 is trained by stacking Conv3D 32 and 64 back to back to check the performance

Result: Lot of fluctuation observed with training and validation accuracy. The training and validation loss is constant. Discarding this model



1. Experiment 9:  
   Model 5: Conv3D with increased frames. 32, 64 and 128

Batch Size: 32, Optimizer: Adam

Data used: 623

Trainable params: 6,841,605

Training accuracy:

Validation accuracy:

Decision: Added additional layer 128 filter to extract more features for better performance of the model.

Result: The model is underfitting. The loss is steadily decreasing.

A graph of a line and a line graph

Description automatically generated with medium confidence

1. Experiment 10:  
   Model 5: Conv3D with increased frames. 32, 64 and 128

Batch Size: 32, Optimizer: Adam

Data used: 1326

Trainable params: 6,841,605

Training accuracy: 97.40%

Validation accuracy: 44.53

Decision: Added additional layer 128 filter to extract more features for better performance of the model.

Result:

A graph of a training and training accuracy

Description automatically generated with medium confidence

1. Experiment 11:

Model 6: Conv3D with a fully connected layer of 64 filters.

Batch Size: 32, Optimizer: Adam

Data used: 623

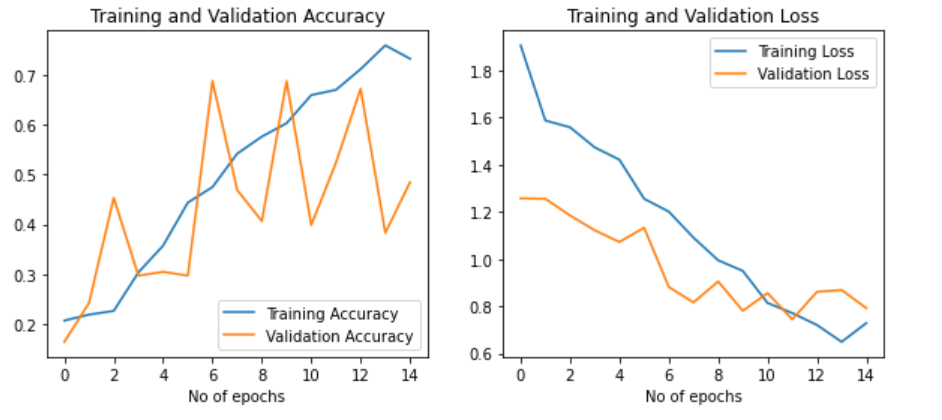
Trainable params: 2,241,541

Training accuracy: 73.21%

Validation accuracy: 38-67%.

Decision: Changed the MaxPool layer and FC with 64 Layers

Result: The training data reached an accuracy of 73.21% for 12th Epochs. The validation accuracy is fluctuating between 38 to 67%. The training and validation loss has steadily decreased.



1. Experiment 12:

Model 6: Conv3D with a fully connected layer of 64 filters.

Batch Size: 32, Optimizer: Adam

Data used: 623

Trainable params: 2,241,541

Training accuracy: 75.30%

Validation accuracy: 46-54%

Decision: Reduced the Adam learning rate to 0.0001. Increased the Epoch to 25 due to slow convergence.

Result: The huge fluctuation observed in Experiment 11 is handled post decreasing the LR to 0.0001.

A graph of performance indicators

Description automatically generated with medium confidence

1. Experiment 13:

Model 6: Conv3D with a fully connected layer of 64 filters.

Batch Size: 32, Optimizer: Adam

Data used: 1326

Trainable params: 2,241,541

Training accuracy: 80%

Validation accuracy: 50%

Decision: Increased the data to check if the model can generalize well.

Result: The training accuracy increased but validation accuracy is around 50%. The training accuracy reached 80% little earlier 15th Epoch as compared to experiment 12 with limited dataset.

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated

## Experiments with Conv2D+LSTM/GRU:

1. Experiment 14:

Model 7: Conv2D +LSTM.

Batch Size: 32, Optimizer: Adam

Data used: 623

Trainable params: 8,712,133

Training accuracy: 98.66%

Validation accuracy: 50%

Decision: New architecture Conv2D+RNN

Result: The training accuracy reached 98.66% at 8th epoch and it remains stable. The validation accuracy is about 50%. The validation loss is around 0.91

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated

1. Experiment 15

Model 8: Conv2D +LSTM with droput

Batch Size: 32, Optimizer: Adam

Data used: 623

Trainable params: 8,712,133

Training accuracy: 98.66%

Validation accuracy: 50%

Decision: Added a dropout layer to handle the overfitting.

Result: The training accuracy reached 99.55% at 8th epoch and it remains stable. The validation accuracy is about 64 to 67%. The validation loss is around 0.9981.

A graph of a training and training accuracy

Description automatically generated with medium confidence

1. Experiment 16

Model 8: Conv2D +LSTM with dropout

Batch Size: 32, Optimizer: Adam

Data used: 1326

Trainable params: 8,712,133

Training accuracy: 98.66%

Validation accuracy: 50%

Decision: For model8, added additional data with data augmentation.

Result: With increased dataset, the model is overfitting.

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated

1. Experiment 17

Model 8: Conv2D +LSTM with dropout

Batch Size: 32, Optimizer: Adam

Data used: 623

Trainable params: 8,712,133

Training accuracy: 93.45%

Validation accuracy: 66.41%

Decision: For model8, with data augmentation.

Result: With data augmentation, able to achieve 66.41% of validation accuracy.

A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated

1. Experiment 18

Model 9: Conv2D +LSTM with dropout

Batch Size: 32, Optimizer: Adam

Data used: 623

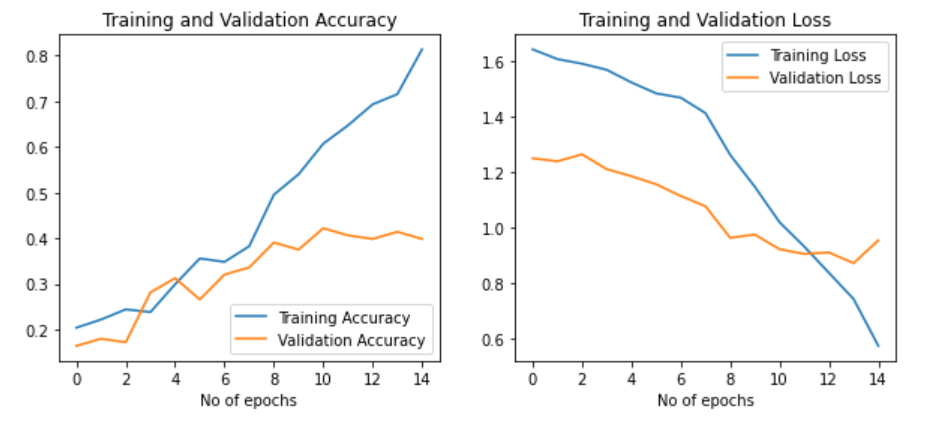
Trainable params: 4,359,621

Training accuracy: 81.40%

Validation accuracy: 40%

Decision: For model8, added additional data with data augmentation.

Result: The training data reached an accuracy of 81.40% and validation set accuracy is around 39-40% with 15 Epochs. The model is overfitting.



1. Experiment 19

Model 10: Conv2D +GRU

Batch Size: 32, Optimizer: Adam

Data used: 623

Trainable params: 3,275,269

Training accuracy: 97.62%

Validation accuracy: 46%

Decision: Tried a model with GRU

Result:

A graph of a training and training accuracy

Description automatically generated with medium confidence

## Transfer Learning:

1. Experiment 20

Model 11: MobileNet + Conv2D +GRU

Batch Size: 32, Optimizer: Adam with learning rate: 0.001

Data used: 623

Pre-Trained Model: Mobile Net with last 10 layers is retrained.

Trainable params: 3,774,213

Non-trainable params: 22,144

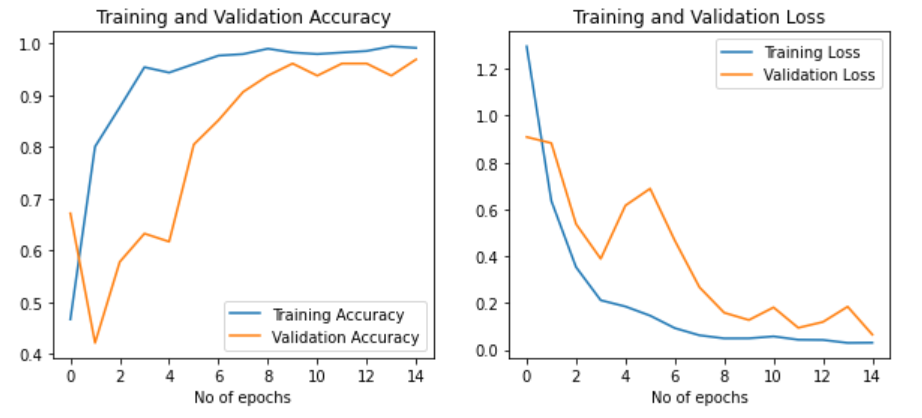
Training accuracy: 99.11%

Validation accuracy: 96.88%

Data augmentation: Only Cropping and Resizing is used.

Decision: Using Mobile Net to achieve a better result. All 30 images are used to train this model.

Result: With 15 Epochs, model has learnt the features. This is good git model with validation loss 0.0650



Note: With transfer learning and GRU architecture, tried other variations but with other variations could not achieve good accuracy.

## Conclusion:

The final model selected is Model 11 (MobileNet + Conv2D+GRU) with Experiment 20. The model is saved in .keras format as the latest tensor flow doesn’t support .h5.

My Model>> **model-00015-0.03051-0.99107-0.06500-0.96875.keras**

The model size is 94 MB, as upgrade portal is not allowing to upload data, I have pushed the model to my Github repository.

<https://github.com/smarunkumar/Gesture_Recognition>