**Gesture Recognition – Deep learning**

**Problem Statement:**

We need to develop a cool feature in the smart-TV that can recognize five different gestures performed by the user which will help users control the TV without using a remote.

The following table consists of the experiments done to build a model to predict the gestures from the given data set.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Exp No.** | **Model** | **Hyper-params** | **Result** | **Decision + Explanation** |
| 1 | Conv3D | image\_height=160  image\_width=160  frames\_to\_sample=20  batch\_size=40  num\_epochs=15  Total params: 1,117,061  Trainable params: 1,116,325  Non-trainable params: 736 | Train Data:  Accuracy: 0.9737  Loss: 0.1253  Test Data:  Accuracy: 0.2600  Loss: 3.6642 | The training process shows that while the model is learning effectively from the training data (as evidenced by decreasing training loss and increasing training accuracy), it is not performing well on the validation set.  The increasing validation loss and constant low validation accuracy indicate overfitting and poor generalization.  **This model clearly shows overfitting**. |
| 2 | Conv3D | image\_height=160  image\_width=160  frames\_to\_sample=20  batch\_size=20  num\_epochs=25  Total params: 3,638,981  Trainable params: 3,637,477  Non-trainable params: 1,504 | Train Data:  Accuracy: 0.9013  Loss: 0.3034  Test Data:  Accuracy: 0.3600  Loss: 2.9202 | Reduced batch size.  The model performs well in Training Data but fails to generalize to Test Data.  **This model clearly shows overfitting**. |
| 3 | Conv3D | image\_height=120  image\_width=120  frames\_to\_sample=16  batch\_size=20  num\_epochs=25  Total params: 2,556,533  Trainable params: 2,554,549  Non-trainable params: 1,984 | Train Data:  Accuracy: 0.8875  Loss: 0.3440  Test Data:  Accuracy: 0.8500  Loss: 0.5590 | Cropped the image.  Added More Layers,filters and Dropout  The final validation accuracy of 85.00% shows that the model correctly predicted the class labels for 85% of the samples in the validation set. This high accuracy reflects the model's strong performance and effective learning.  The model has demonstrated excellent performance, achieving both a low validation loss and high validation accuracy by the end of training. This suggests that the model has effectively learned from the data and generalized well to the validation set.  We will evaluate other models. |
| 4 | Conv3D | image\_height=120  image\_width=120  frames\_to\_sample=16  batch\_size=20  num\_epochs=15  Total params: 2,556,533  Trainable params: 2,554,549  Non-trainable params: 1,984 | Train Data:  Accuracy: 0.8892  Loss: 0.3047  Test Data:  Accuracy: 0.2200  Loss: 3.5469 | Adding Dropout at Convolution Layer  This didn’t give any better result than previously tried model.  **This model clearly shows overfitting**. |
| 5 | Conv3D | image\_height=120  image\_width=120  frames\_to\_sample=16  batch\_size=20  num\_epochs=25  Total params: 504,709  Trainable params: 503,973  Non-trainable params: 736 | Train Data:  Accuracy: 0.8022  Loss: 0.5262  Test Data:  Accuracy: 0.2400  Loss: 2.8721 | Reduce No of Params No Improvement Reducing the number of parameters. Model Still Overfits. |
| 6 | ConvLSTM | image\_height=120  image\_width=120  frames\_to\_sample=18  batch\_size=20  num\_epochs=20  Total params: 1,657,445  Trainable params: 1,656,453  Non-trainable params: 992 | Train Data:  Accuracy: 0.9646  Loss: 0.1014  Test Data:  Accuracy: 0.9000  Loss: 0.3890 | It demonstrates strong performance with high validation accuracy and low validation loss, indicating good generalization.  However, **the model also shows signs of overfitting and some instability in later epochs.** |
| 7 | ConvLSTM | image\_height=120  image\_width=120  frames\_to\_sample=18  batch\_size=5  num\_epochs=25  Total params: 1,657,445  Trainable params: 1,656,453  Non-trainable params: 992 | Train Data:  Accuracy: 0.9542  Loss: 0.1342  Test Data:  Accuracy: 0.8900  Loss: 0.3579 | Reduced Batch Size  While training accuracy consistently improves, validation accuracy and loss exhibit some fluctuations, indicating potential overfitting or variance in validation performance. The model's performance stabilizes with high accuracy and low loss towards the end of the training. |
| **Using Augmentation Technique on ConvLSTM Model** | | | | |
| 8 | ConvLSTM | image\_height=120  image\_width=120  frames\_to\_sample=18  batch\_size=10  num\_epochs=25  Total params: 1,657,445  Trainable params: 1,656,453  Non-trainable params: 992 | Train Data:  Accuracy: 0.9760  Loss: 0.0716  Test Data:  Accuracy: 0.9200  Loss: 0.2891 | The model consistently improves in both training and validation metrics throughout the epochs. This indicates effective learning and generalization.  The validation accuracy reaches a high value of ~92.00% by the end of training, and the validation loss stabilizes, suggesting that the model performs well on new, unseen data.  While the training metrics are very high, it’s crucial to ensure that the model doesn’t overfit to the training data. The validation metrics being high as well indicates good generalization.  **This Can be a good model.** |
| 9 | ConvLSTM  + GRU | image\_height=120  image\_width=120  frames\_to\_sample=18  batch\_size=20  num\_epochs=20  Total params: 2,573,925  Trainable params: 2,573,445  Non-trainable params: 480 | Train Data:  Accuracy: 0.9958  Loss: 0.0331  Test Data:  Accuracy: 0.8000  Loss: 0.6159 | The gap between training accuracy/loss and validation accuracy/loss suggests that the model is overfitting. The model performs exceptionally well on training data but less consistently on validation data, especially in the later epochs.  **Model Overfits**. |
| 9.1 | ConvLSTM  + GRU | image\_height=120  image\_width=120  frames\_to\_sample=18  batch\_size=15  num\_epochs=20  Total params: 2,573,925  Trainable params: 2,573,445  Non-trainable params: 480 | Train Data:  Accuracy: 0.9709  Loss: 0.1140  Test Data:  Accuracy: 0.8600  Loss: 0.4673 | Reduced batch size  Training loss continues to decrease, while validation loss shows a more erratic behavior, with some improvements but also some periods of stagnation.  The validation loss shows significant improvements early on but then plateaus and even slightly worsens toward the end of training.  **The model performs exceedingly well on training data but does not generalize as effectively to unseen data.** |
| 10 | ConvLSTM  + GRU | image\_height=120  image\_width=120  frames\_to\_sample=18  batch\_size=15  num\_epochs=25  Total params: 5,048,677  Trainable params: 5,048,197  Non-trainable params: 480 | Train Data:  Accuracy: 0.9837  Loss: 0.8392  Test Data:  Accuracy: 0.9100  Loss: 1.0703 | Using Bidirectional GRU and Regularization techniques  The gap between training and validation accuracy and loss suggests potential overfitting. The model performs well on training data but slightly less so on validation data. |
| 11 | ConvLSTM  + GRU | image\_height=120  image\_width=120  frames\_to\_sample=16  batch\_size=5  num\_epochs=20  Total params: 3,693,253  Trainable params: 3,669,317  Non-trainable params: 23,936 | Train Data:  Accuracy: 0.9932  Loss: 0.0181  Test Data:  Accuracy: 0.9900  Loss: 0.0303 | With Trainable weights of Transfer Learning  The model trained effectively, as seen by the progressive improvement in both training and validation accuracy, alongside decreasing loss values.  The model reached high accuracy rates with low validation loss, indicating good performance and generalization ability.  Adjustments to the learning rate proved beneficial in fine-tuning the model's learning process, allowing it to converge effectively without overfitting.  **The results suggest that the model is well-suited for its classification task, maintaining robust performance on unseen data.** |

**Conclusion on Final Model :**

1. Based on the Experiment we observed that Model – 11 (GRU with Transfer learning) is the better model to chose as it’s better on Accuracy and Loss.

2. Even we can consider the Model – 8 as it also has good Accuracy and Loss. It can be chosen for its potential advantages in terms of computational efficiency, training time especially if the slight trade-off in performance is acceptable. The no of params also less in Model – 8 compared to others as well.

**I am submitting the Model-8 as I also have issue while loading the model-11.**

***NOTE: I am submitting the ‘.keras’ file as my local has upgraded versions of Keras and Tensorflow. Also I faced issue using the Jarvis Lab.***