Gesture Recognition – Deep learning

Problem Statement:
We need to develop a cool feature in the smart-TV that can recognise five different gestures performed by theuser 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.#	Model	Hyper Parm	Result	Decision + Explanation

1	Conv3D	Batchsize = 32
		Augumentation = False
		Input size = 120 X 120
		le_rate = 0.001
		optimizer = Adam
			categorical_accuracy: 0.9947 loss: 0.0219 - val_categorical_accuracy: 0.5547 val_loss: 0.8311	• The training accuracy increases rapidly, reaching very high values (close to 100%).
				• However, the validation accuracy plateaus around 50-60%, indicating significant overfitting.
				• The loss on the training data decreases very well, but the validation loss does not decrease as well, and even increases at some points. This is a very strong indicator of overfitting.
				Used DataAugmentation in next step

2	Conv3D	Batchsize = 32
		Augumentation = True
		Input size = 120 X 120
		le_rate = 0.001
		optimizer = Adam
			categorical_accuracy: 0.9901 - loss: 0.0539 - val_categorical_accuracy: 0.5312 - val_loss: 0.9033 -	• The model still heavily overfits, even with the data augmentation.
				• The validation loss is still not decreasing in a consistent manner.
				Increase the image size in next step

3	Conv3D	Batchsize = 32
		Augumentation = True
		Input size = 224 X 224
		le_rate = 0.001
		optimizer = Adam
			categorical_accuracy: 0.9971 - loss: 0.0042 - val_categorical_accuracy: 0.5547 - val_loss: 1.7219	• Overfitting is still a major problem, even with increased image size and data augmentation.
				Used dropout in next layer

4	Conv3D	Batchsize = 32
		Augumentation = True
		Input size = 180 X 180
		le_rate = 0.001
		optimizer = Adam
		Added Dropout
			categorical_accuracy: 0.8699 - loss: 0.5045 - val_categorical_accuracy: 0.3672 - val_loss: 1.2164	• **Overfitting Persists:** Despite the dropout, the training accuracy still increases faster than the validation accuracy, indicating overfitting.
				• **Validation Loss Fluctuation:** The validation loss starts to increase again after epoch 6, indicating that the model is not generalizing well.
				Use Batch Normalization and regularization

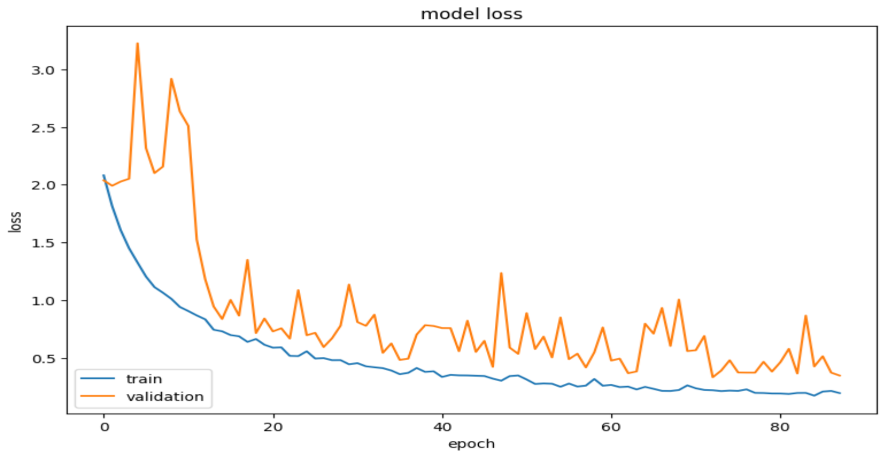
5	Conv3D	Batchsize = 32
		Augumentation = True
		Input size = 180 X 180
		le_rate = 0.001
		optimizer = Adam
		Removed Dropout, added Normalization and Regularization
			categorical_accuracy: 0.9999 -
			loss: 0.4735 -
			val_categorical_accuracy: 0.8828 -
			val_loss: 0.8097
				• **Improved Validation Accuracy:** The validation accuracy shows significant improvement, reaching 88.28% by the 10th epoch. This is a substantial improvement compared to previous models.
				• **Reduced Overfitting:** The gap between training and validation accuracy is smaller, indicating that the regularization and batch normalization are helping to reduce overfitting.
				• **Validation Loss Improvement:** The validation loss consistently decreases, which is a positive sign.
				Next try TimeDistribution and ConvLSTM

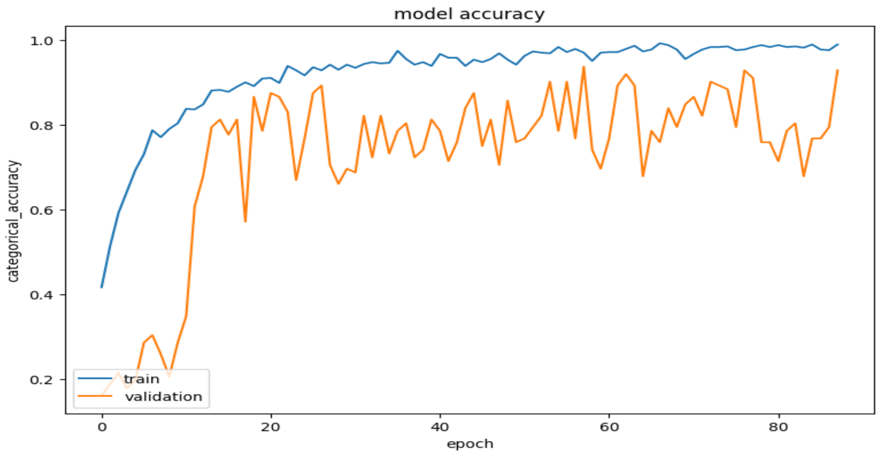
6	Time Distributed + ConvLSTM	Batchsize = 32
		Augumentation = True
		Input size = 180 X 180
		le_rate = 0.001
		optimizer = Adam
		added Normalization and Regularization
			categorical_accuracy: 0.5796 -
			loss: 1.0137 -
			val_categorical_accuracy: 0.1172 -
			val_loss: 1.4067
				• **Validation Loss Plateau:** After the first epoch, the validation loss plateaus and starts to increase. This indicates that the model is not generalizing well.
				• **Overfitting:** The training accuracy is higher than the validation accuracy, indicating overfitting.

Final Model	Time Distributed + ConvLSTM	Batchsize = 16
		Augumentation = True
		Input size = 180 X 180
		learning_rate = 0.0005
		optimizer = Adam
		added Normalization and Regularization
		Learning Rate Scheduling: Cosine annealing for dynamic learning rate.
		Early Stopping: To prevent overfitting.
			categorical_accuracy: 0.9751 -
			loss: 0.2357 -
			val_categorical_accuracy: 0.9018 -
			val_loss: 0.3312
				…………………

Analysis of the Output of Final model:

1. **High Training Accuracy:**
   - The training accuracy is very high, consistently above 95%, and often above 98%. This indicates the model is fitting the training data very well.
2. **Fluctuating Validation Accuracy:**
   - The validation accuracy fluctuates significantly, ranging from around 67% to 93%. This suggests the model's performance on unseen data is inconsistent.
3. **Validation Loss Improvement and Fluctuations:**
   - The validation loss has improved over time, reaching a low of 0.33121.
   - However, similar to the validation accuracy, the validation loss also fluctuates, indicating the model is not consistently generalizing well.
4. **Overfitting:**
   - The large gap between training accuracy (very high) and validation accuracy (fluctuating) strongly suggests overfitting.
   - The model is memorizing the training data rather than learning generalizable patterns.





Conclusion:

The Model built with Time distributed Conv2D and ConvLSTM2D (Final Model) gave better results compared to all the other models and also the model has very least number of parameters compared to other models