Gesture Recognition Case Study

Problem Statement:

This problem revolves around integrating gesture recognition into a smart TV, enhancing user interaction by eliminating the need for a remote. The goal is to build a 3D Conv model that will enable the TV to interpret five specific hand gestures captured by a webcam in real time and execute the corresponding commands. Below are the 5 gestures and commands:

- Thumbs up: Increase the volume.
- Thumbs down: Decrease the volume.
- Left swipe: Jump backwards 10 seconds.
- Right swipe: Jump forward 10 seconds.
- Stop: Pause the current playback.

About the Dataset:

The training data consists of a few hundred videos categorised into one of the five classes. Each video (typically 2-3 seconds long) is divided into a sequence of 30 frames(images). These videos have been recorded by various people performing one of the five gestures in front of a webcam - similar to what the smart TV will use.

Custom Generator:

- **Image Preprocessing**: The `crop_and_resize` function ensures that input images of various sizes are centrally cropped and resized to a consistent target size (120x120).
- **Batch Data Preparation**: The `generator` function shuffles the folder list, processes image frames (30 per folder), and normalizes pixel values to [0, 1].
- **Label Encoding**: It extracts and one-hot encodes class labels from folder metadata for use in categorical classification.
- **Batch Yielding**: Data is grouped into batches (or a smaller batch for remaining data) and yielded for training in a memory-efficient manner.
- **Generalization**: The generator handles dynamic folder and image counts, making it versatile for gesture recognition datasets.
- **Number of Epochs:** 30

Experiment Number	Model	Model Description	Result	Decision + Explanation
1	Conv3D	The model consists of 2 Conv3D layers with Batch Normalization and MaxPooling3D, followed by a Flatten layer and 2 Dense layers. Batch Size: 16	loss: 1.5524 categorical_accuracy: 0.3967 val_loss: 4.7615 val_categorical_accuracy: 0.4100	Result: Poor generalization; low accuracy on both training and validation sets. Next Step: Increase model capacity by adding a layer
2	Conv3D	The model consists of 3 Conv3D layers with Batch Normalization and MaxPooling3D, followed by a Flatten layer and 2 Dense layers. Batch Size: 16	loss: 0.1958 categorical_accuracy: 0.9216 val_loss: 1.0449 val_categorical_accuracy: 0.7400	Result: Overfitting; high training accuracy but lower validation accuracy. Next Step: Increasing dropout or adding an additional layer
3	Conv3D	The model consists of 4 Conv3D layers with Batch Normalization and MaxPooling3D, followed by a Flatten layer, 2 Dense layers and Dropout. Increasing the dropout rates in various layers. Batch Size: 16	loss: 0.12068 categorical_accuracy: 0.9517 val_loss: 0.7971 val_categorical_accuracy:0.8100	Result: Balanced performance with good generalization; validation accuracy of 81%. Next Step: Trying with CNN LSTM Model
4	ConvLSTM	The model consists 2 Conv2D Layers with TimeDistributed wrapper, BatchNormalization, MaxPooling2D and Flatten Layer after TimeDistributed for flattening each frame's feature maps, LSTM Layer with 128 units for sequence processing and a dropout layer is also present after the dense layer. Batch Size: 16	loss: 0.3938 categorical_accuracy: 0.8522 val_loss: 0.9038 val_categorical_accuracy: 0.6800	Result: LSTM-based model struggled to generalize; moderate validation accuracy (66%). Next Step: Adding more Conv2D layers before LSTM and tuning dropout values.
5	ConvLSTM	The model consists 3 Conv2D Layers with TimeDistributed wrapper, BatchNormalization,	loss: 0.1203 categorical_accuracy: 0.9638	Result: Moderate overfitting; validation accuracy lower than training accuracy.

		MaxPooling2D and Flatten Layer after TimeDistributed for flattening each frame's feature maps, LSTM Layer with 128 units for sequence processing and a dropout layer is also present after the dense layer. Batch Size**:** 16	val_loss: 0.8349 val_categorical_accuracy: 0.7400	Next Step: Adding an additional layer.
6	ConvLSTM	The model consists 4 Conv2D Layers with TimeDistributed wrapper, BatchNormalization, MaxPooling2D and Flatten Layer after TimeDistributed for flattening each frame's feature maps, LSTM Layer with 128 units for sequence processing and a dropout layers. Batch Size**:** 16	loss: 0.7193 categorical_accuracy: 0.7115 val_loss: 1.0299 val_categorical_accuracy: 0.6300	Result: Moderate overfitting; validation accuracy lower than training accuracy. Next Step: Adding an additional layer.
7	ConvGRU	The model consists 3 Conv2D Layers with TimeDistributed wrapper, BatchNormalization, and MaxPooling2D, Flatten Layer after TimeDistributed for flattening each frame's feature maps, GRU Layer with 128 units for sequence processing and increased Dropout Layers to address overfitting. Batch Size**:** 8 (Batch size reduced as started getting Memory Issues)	loss: 0.5642 categorical_accuracy: 0.7722 val_loss: 0.9183 val_categorical_accuracy: 0.7200	Result : Model's performance shows promising results. Next Step: Trying with Transfer Learning using MobileNetV2
8	Transfer Learning - MobileNet V2	Transfer Learning Using MobileNetV2 Batch Size**:** 8	loss: 0.1896 categorical_accuracy: 0.9306 val_loss: 0.2685 val_categorical_accuracy: 0.9100	This looks like the best model