Neural Networks Project - Gesture Recognition

Prepared by : Avdhesh Singh Chouhan, Jennis Marie Vicente-Feliciano

Problem Statement

A home electronics company which manufactures state of the art smart televisions, wants to develop a cool feature for a smart TV which is to recognize 5 different hand gestures which helps users control the TV without remote control.

The gestures are continuously monitored by the webcam mounted on the TV. Each gesture corresponds to a specific command:

- 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 movie

Dataset:

663 videos recorded for training, 30 frames each. There are 100 videos of 30 frames each to validate the accuracy of the solution.

Solution : 3D convolution

Experiments:

Experiment Number	Model	Result	Decision + Explanation
1	Conv3D	Accuracy: 0.4139 Validation accuracy: 0.60	Four layers in model (16, 32, 64, and 128 filters). Batch normalization used for each layer without dropouts. Batch size: 30. Number of Epoch: 1 Images used : 10/30 Image size: 160, 160 LR = tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, verbose=1, patience=4)
2	Conv3D	Accuracy: 0.7224 Validation accuracy: 0.27	Four layers in model (16, 32, 64, and 128 filters). Batch normalization used for each layer with dropout Batch size: 50. Number of Epoch: 5 Images used : 15/30 Image size: 160, 160 optimiser = optimizers.Adam(lr=0.01) LR = tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, verbose=1, patience=4, min_lr=0.001)
3	Conv3D	Accuracy: 0.5173 Validation accuracy: 0.14	Four layers in model (16, 32, 64, and 128 filters). Batch normalization used for each layer. Two dense layer with Dropout (128, 0.50 and 64, 0.25) Batch size: 13. Number of Epoch: 2 Images used : 10/30 Image size= 160,160
4	Conv3D	Accuracy: 0.5083 Validation accuracy: 0.41	Four layers in model (16, 32, 64, and 128 filters). Batch normalization used for each layer. Two dense layer with Dropout (128, 0.50 and 64, 0.75) Batch size: 13. Number of Epoch: 10 Images used : 10/30 Image size= 160,160

5	Conv3D	Accuracy: 0.5616 Validation accuracy: 0.57	Four layers in model (32, 32, 64, and 128 filters). Batch normalization used for each layer. Two dense layer with Dropout (128, 0.50 and 512, 0.25) Batch size: 13. Number of Epoch: 10 Images used : 10/30 Image size= 160,160
6	Conv3D	Accuracy: 0.7816 Validation accuracy: 0.27	Four layers in model (32, 32, 64, and 128 filters). Batch normalization used for each layer. Two dense layer with Dropout (128, 0.50 and 512, 0.25) Batch size: 25. Number of Epoch: 10 Images used : 10/30 Image size= 160,160
7	Conv3D	Accuracy: 0.7816 Validation accuracy: 0.27	Four layers in model (32, 32, 64, and 128 filters). Batch normalization used for each layer. Two dense layer with Dropout (128, 0.50 and 512, 0.25) Batch size: 25. Number of Epoch: 10 Images used : 10/30 Image size= 100,100
8	Conv3D	Accuracy: 0.9648 Validation accuracy: 0.25	Four layers in model (32, 32, 64, and 128 filters). Batch normalization used for each layer. Two dense layer with Dropout (128, 0.50 and 512, 0.25) Batch size: 25. Number of Epoch: 10 Images used : 10/30 Image size= 100,100
9	Conv3D	Accuracy: 0.8869 Validation accuracy: 0.30	Four layers in model (32, 32, 64, and 128 filters). Batch normalization used for each layer. Two dense layer with Dropout (128, 0.50 and 512, 0.25) Batch size: 25.

			Number of Epoch: 10 Images used : 8/30 Image size= 120,120
10	Conv3D	Accuracy: 0.8899 Validation accuracy: 0.24	Four layers in model (16, 32, 64, and 128 filters)(3,3,3). Batch normalization used for each layer. Two dense layer with Dropout (128, 0.50 and 512, 0.25) Batch size: 25. Number of Epoch: 10 Images used : 8/30 Image size= 120,120
11	Conv3D	Accuracy: 0.9321 Validation accuracy: 0.26	Four layers in model (16, 32, 64, and 128 filters)(3,3,3). Batch normalization used for each layer. Two dense layer with Dropout (64, 0.25 and 128, 0.25) Batch size: 25. Number of Epoch: 10 Images used : 8/30 Image size= 120,120 tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, verbose=1, patience=5, min_lr=0.0001)
12	Conv3D	Accuracy: 0.8869 Validation accuracy: 0.78	Four layers in model (16, 32, 64, and 128 filters)(3,3,3). Batch normalization used for each layer. Two dense layer with Dropout (64, 0.25 and 128, 0.25) Batch size: 20. Number of Epoch: 18 Images used : 14/30 Image size= 100,100 tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, verbose=1, patience=5, min_lr=0.0001)
13	Conv3D	Accuracy: 0.8431 Validation accuracy: 0.70	Four layers in model (16, 32, 64, and 128 filters)(3,3,3). Batch normalization used for each layer. Two dense layer with Dropout (128, 0.50 and 512, 0.25) Batch size: 25.

			Number of Epoch: 10 Images used : 14/30 Image size= 100,100 optimiser = optimizers.Adam(lr=0.01) tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, verbose=1, patience=4, min_lr=0.0001)
14	Conv3D	Accuracy:0.9170 Validation accuracy : 0.69	Five layers in model (16, 32, 64, and 128, 256 filters). Batch normalization used for each layer with dropout Batch size: 20. Number of Epoch: 20 Images used : 14/30 Image size: 120, 120 optimiser = optimizers.Adam(lr=0.01) tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, verbose=1, patience=4, min_lr=0.0001)

Conclusion :

Model chosen: 12

Choosing the 12th model gives best results on the training and validation set : Accuracy: 0.8869, Validation accuracy: 0.78
Hyperparameters for best model :

Four layers in model (16, 32, 64, and 128 filters)(3,3,3).
Batch normalization used for each layer.
Two dense layer with Dropout (64, 0.25 and 128, 0.25)
Batch size: 20.
Number of Epoch: 18
Images used : 14/30
Image size= 120,120

tf.keras.callbacks.ReduceLROnPlateau(monitor='val_loss', factor=0.2, verbose=1, patience=5, min_lr=0.0001)