Neural Networks Project – Gesture recognition

Submitted by: - Richard Samuel & Hareesh Rajasekharan

**Problem Statement**

A home electronics company which manufactures state of the art smart televisions would want to develop a cool feature in the smart-TV that can recognise five different gestures performed by the user which will help users control the TV without using a remote.

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

**Understanding 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.

**Objective: -**

The objective of this case study is to create and train the model on the ‘train’ and ‘val’ folders. The final model’s performance would be evaluated on the ‘test’ set.

**Architectures used: -**

For analysing videos using neural networks, two types of architectures are used commonly. One is the standard CNN + RNN architecture in which you pass the images of a video through a CNN which extracts a feature vector for each image, and then pass the sequence of these feature vectors through an RNN. The other popular architecture used to process videos is a natural extension of CNNs - a 3D convolutional network.

**Here in this case study, we are using 3D convolutional model.**

Convolutions in three dimensions are a natural extension of the two-dimensional convolutions you are currently familiar with. As with 2D convolution, you move the filter in two directions (x and y) with 3D convolution. three-way filtering (x, y and z). In this scenario, the input to a 3D convolutional neural network is a movie.

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Result** | **Decision + Explanation** |
| **Model B Trial: 1** | **Conv3D** | **Categorical Accuracy: 29.85**  **Validation Accuracy: 37** | **Initialized the values X,Y,Z**  **Number of Frames - 30**  **Image Width - 60**  **Image Height – 60**  **Based on the trial 1 result, experimenting on the batch size.** |
| **Model B Trial: 2** | **Conv3D** | **Categorical Accuracy: 65.17**  **Validation Accuracy: 70** | **Increasing the batch size to 20** |
| **Model B Trial: 3** | **Conv3D** | **Categorical Accuracy: 82.59**  **Validation Accuracy: 76** | **Increasing the batch size to 30** |
| **Model B Trial: 4** | **Conv3D** | **Categorical Accuracy: 87.45**  **Validation Accuracy: 69.50** | **Increasing the batch size to 40** |
| **Model B Trial: 5** | **Conv3D** | **Categorical Accuracy: 99.21**  **Validation Accuracy: 67.50** | **Changing the optimizer to ADADELTA** |
| **Model B Trial: 6** | **Conv3D** | **Accuracy: 99.81** | **Initializing the number of Epoch to 20** |
| **Model C Trial: 7** | **Conv3D** | **Categorical Accuracy: 37.31**  **Validation Accuracy: 50** | **Total Params: 2061157**  **Trainable Params: 2060133**  **Non-Trainable Params: 1024**  **Reverting the batch size to 10 and adjusting the images per frame, height and width**  **X = 30**  **Y = 120**  **Z = 120** |
| **Model D Trial: 8** | **Conv3D** | **Categorical Accuracy: 41.79**  **Validation Accuracy: 41** | **Introducing filters, dense & softmax layer.** |
| **Model D Trial: 9** | **Conv3D** | **Categorical Accuracy: 28.36**  **Validation Accuracy: 38** | **Changing the image width and height and introducing classes and channel.**  **Y = 64**  **Z = 64**  **Classes = 5**  **Channel = 1**  **Batch size is reduced to 5** |
| **Model F Trial: 10** | **Conv3D** | **Categorical Accuracy: 67.66**  **Validation Accuracy: 74** | **Changing the image width and height. Classes and channel values remains the same.**  **Y = 64**  **Z = 64**  **Classes = 5**  **Channel = 1**  **Dense also increased.**  **Batch size increased 10**  **Epoch = 30** |
| **Model G Trial: 11** | **Conv3D** | **Categorical Accuracy: 40.80**  **Validation Accuracy: 47** | **Reducing number of Epoch to 10** |
| **Model H Trial: 12** | **Conv3D** | **Categorical Accuracy: 51.74**  **Validation Accuracy: 44** | **Dense removed** |
| **Model I Trial: 13** | **Conv3D** | **Categorical Accuracy: 53.23**  **Validation Accuracy: 58** | **Adding additional layers** |
| **Final Model Trial: 1** | **Conv3D** | **Categorical Accuracy: 57.21**  **Validation Accuracy: 66** | **Adjusting the filters and dense.**  **Introduced shape**  **Adjusting the Kernel size**  **Batch size = 10**  **Number of epochs = 20** |
| **Final Model Trial: 2** | **Conv3D** | **Categorical Accuracy: 63.18**  **Validation Accuracy: 51** | **Batch size = 32**  **Number of epochs = 20** |
| **Final Model Trial: 3** | **Conv3D** | **Categorical Accuracy: 96.75**  **Validation Accuracy: 75** | **Batch size = 64**  **Number of epochs = 20** |
| **Final Model Trial: 4** | **Conv3D** | **Categorical Accuracy: 96.14**  **Validation Accuracy: 73.50** | **Batch size = 40**  **Number of epochs = 20** |

**After doing all experiments, we selected the "Final Model- Trial-4 - CNN3D," which performed admirably.**