**Gesture Recognition using CNN/CNN-RNN**

**Problem Statement**

Imagine you are working as a data scientist at a home electronics company which manufactures state of the art smart televisions. You want 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 gestures are continuously monitored by the webcam mounted on the TV. Each gesture corresponds to a specific command:

1. Thumbs up: Increase the volume
2. Thumbs down: Decrease the volume
3. Left swipe: 'Jump' backwards 10 seconds
4. Right swipe: 'Jump' forward 10 seconds
5. Stop: Pause the movie

**Two Architectures: 3D Convs and CNN-RNN Stack**

After understanding and acquiring the dataset, the next step is to try out different architectures to solve this problem.

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.

Note:

1. You can use transfer learning in the 2D CNN layer rather than training your own CNN
2. GRU (Gated Recurrent Unit) or LSTM (Long Short Term Memory) can be used for the RNN

The other popular architecture used to process videos is a natural extension of CNNs - a 3D convolutional network. In this project, we will try both these architectures.

**Image processing and Generator**

We found that the dataset contains two categories of images:

1. 120x160 images
2. 360x360 images

We crop the 120x160 images to 120x120, to make them 1:1 aspect ratio. The other category of images already adheres to 1:1 aspect ratio. We further resize these images to decrease time taken to train the model.

The Generator works in batches, feeding images into the model for training/validation. We perform the image processing steps within this generator.

**Implementation**

**1. 3D Convolutional Network, or Conv3D**

1. The architecture is described below:
2. 3x3 filter was used, as it represented the best trade-off between accuracy and speed of training
3. The optimizer used is the SGD optimizer.

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment No.** | **Model** | **Result** | **Decision / Explanation** |
| **1** | **ImageSize**: original  (120x120) and (360x360)  **BatchSize:**16  **Epochs:**20  **Layers:**4 | Out of memory Error | Reduce Image Size to 120x120 for all images |
| **2** | **ImageSize**: 120x120  **BatchSize:**16  **Epochs:**20  **Layers:**4 | Accuracy : 0.61  Very slow training time | 1. Reduce image size further to 90x90 2. Reduce image size any further degrades images quality beyond acceptable levels |
| **3** | **ImageSize**: 90x90  **BatchSize:**16  **Epochs:**20  **Layers:**4 | Accuracy : 0.60  Slow training time, but improved on experiment 2 | Increase BatchSize |
| **4** | **ImageSize**: 90x90  **BatchSize:** 64  **Epochs:**20  **Layers:**4 | Accuracy : 0.53 | Higher batch size decreasing  Optimal batch size 16  Increase no. of epochs |
| **5** | **ImageSize**: 90x90  **BatchSize:** 16  **Epochs:** 60  **Layers:**4 | Accuracy : 0.77 | Increase Layers to 5 |
| **6** | **ImageSize**: 90x90  **BatchSize:** 16  **Epochs:** 10  **Layers:** 5 | Accuracy : 0.4  Extremely slow | Can’t continue towards 60 epochs for lack of time. |
| **Final Model** | **ImageSize**: 90x90  **BatchSize:** 16  **Epochs:** 60  **Layers:**4 | Accuracy : 0.77 | Increase Layers to 5 |

**2.VGG16+GRU**

As part of transfer learning used VGG16 as the base model and added a GRU layer along with output softmax layer, setting the VGG16 trainable as TRUE.

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment No.** | **Model** | **Result** | **Decision / Explanation** |
| **1** | **ImageSize**: 90x90  **BatchSize:**15  **Epochs:**20  **VGG Trainable:**True | Accuracy: 0.57 | the accuracy is not good, as the vgg trainable is set as False |
| **2** | **ImageSize**: 90x90  **BatchSize:**30  **Epochs:**20  **VGG Trainable:**True | Accuracy : 0.59  Slow training time | the accuracy has slightly improved but, took longer time, the accuracy was not good as the vgg trainable is set as False |
| **3** | **ImageSize**: 90x90  **BatchSize:**10  **Epochs:**40  **VGG Trainable:**True | Accuracy: 0.61  Very slow training time | Increase in epochs improved the accuracy but still not good. |
| **4** | **ImageSize**: 90x90  **BatchSize:** 10  **Epochs:**50  **VGG Trainable:**True | Runtime Error OS Error: No space left on device | the hdd space of the GPU in nimblebox is too less.  Need to decrease the no. of epochs |
| **Final Model** | **ImageSize**: 90x90  **BatchSize:** 10  **Epochs:**14  **VGG Trainable:**True | Accuracy : 0.89  Training is faster as compared to other experiments. | Decrease in epochs and setting the VGG trainable as true improved the accuracy. |