Gesture Recognition

Group Facilitator: Jamie Paul Parkinson

Group Member: Sérgio Cláudio Fontes

Date: 16th of November 2022

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 recognise five different gestures performed by the user which will help users control the TV without using a remote.

Goals of the project

Build a model to recognise 5 hand gestures:

- 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

The Generator function

The generator takes a batch of videos as input without any error to help the model to support better large amounts of data. It's also responsible for data cropping, resizing and normalization. When running each experiment we pass in desired resolution, frames, batch size and epoch into the generator for it to process in a reusable manner.

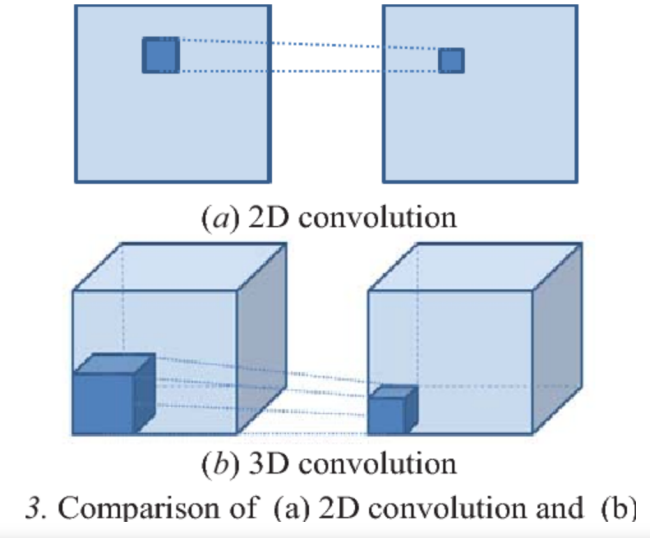
Data Preparation

- **Resizing & Cropping of Images:** Mainly done to ensure the network only focuses on the important parts of the image and not the background which can be distracting and is not relevant.
- **Normalization:** Ensuring statistical distributions across the samples are the same.

Types of architecture usable for video analysis by deep learning

1) 3D Convolutional Neural networks - Conv3D

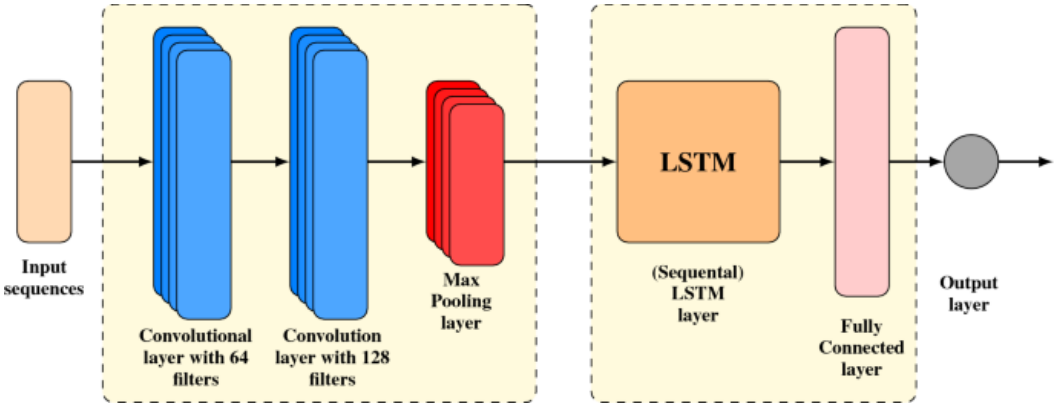
3D Convolutions are a natural extension to typical 2D convolutions where the kernel slides in 3 dimensions instead of just 2 to calculate the feature representations. Their output is respectively a 3d volume space such as a cuber. They are most useful in videos such as this case or medical image however not limited to just videos as they can also be utilized in images. The diagram below shows the difference between typical 2d convolutional networks and 3d ones.



3. Comparison of (a) 2D convolution and (b)

2) CNN + RNN Architecture

Features are extracted by the Conv2D network for each image and of which a sequence of said features are then fed into the RNN Based network. The output of said architecture is a regular softwax layer. The diagram below shows the basic architecture of a CNN+RNN Network.



Experiments & Results

Experiment	Model	Result	Decision + Explanation
1	**Conv3D:** **Normal quality image** 3 convolutional layers using Relu activation function and MaxPooling3D, with a SoftMax activation function in the end 317557 training parameters from a total of 317669 parameters - batch size 4	Accuracy: 37% Val Accuracy: 56%	**No image size change as the starting point. Now let's try lowering the resolution of images.**
2	**Conv3D** **Low Image Resolution (50x50)** 3 convolutional layers using Relu activation function and MaxPooling3D, with a SoftMax activation function in the end 96373 training parameters from a total of 96485 parameters	Accuracy: 44% Val Accuracy: 52%	**Reduce the size of the image considering low resolution cameras. Better results on test data but a small drop on validation accuracy. We will try drop it further to see if we can get any increase in accuracy**
3	**Conv3D** **Further Lower Image Resolution (25x25)** 3 convolutional layers using Relu activation function and MaxPooling3D, with a SoftMax activation function in the end 41077 training parameters from a total of 41189 parameters	Accuracy: 39% Val Accuracy: 56%	**Reduce even more the size of the image considering low resolution cameras. Drop on accuracy but an improvement on validation accuracy to the levels of initial image quality. We will now try with CNN - LSTM Model. The number of parameters is expected to increase in 4x**
4	**CNN - LSTM** **Low Image Resolution (25x25)** LSTM RNN model	Accuracy: 26% Val Accuracy: 19%	**Number of training parameters increased 3,5x but accuracy dropped 33% and validation accuracy dropped from 56% to 19% being less effective. We will**

	3 Time Distributed layers using Relu activation function Conv2D and MaxPooling2D, with a SoftMax activation function in the end 142549 training parameters from a total of 142613 parameters		**go back to our Conv3D Approach but with another optimiser (ADAM)**
5	**Conv3D with ADAM Optimiser:** **Normal quality image - Batch size 4 - Image Resolution (50x50)** 3 convolutional layers using Relu activation function and MaxPooling3D, with a SoftMax activation function in the end 96,373 training parameters from a total of f 96,485 parameters	Accuracy: 54% Val Accuracy: 70%	**Baseline test with ADAM optimiser instead of the standard SGD. Results were slightly better than previous on average. We will now try to increase batch size.**
6	**Conv3D with ADAM Optimiser** **Normal quality image - Batch size 8 - Image Resolution (50x50)** 3 convolutional layers using Relu activation function and MaxPooling3D, with a SoftMax activation function in the end 96373 training parameters from a total of 96485 parameters	Accuracy: 42% Val Accuracy: 60%	**Increase in batch size to check if any effect on accuracy - results show a decrease by around 10% we'll revert the batch size change and try using decreasing dropout and frames.**
7	**Conv3D with ADAM Optimiser** **Normal Image Quality - Decrease Frames and Dropout Rate - Batch size 4** 3 convolutional layers using Relu activation function and MaxPooling3D with a SoftMax Activation in the end. Dropout	Accuracy: 87% Val Accuracy: 76%	**Decreasing dropout rate to 25% and frames to see effect on accuracy whilst still using ADAM Optimiser. This showed a significant increase in accuracy and validation accuracy to 87% and 76% respectively. Next experiment we will increase epochs to see if we can further increase accuracy.**

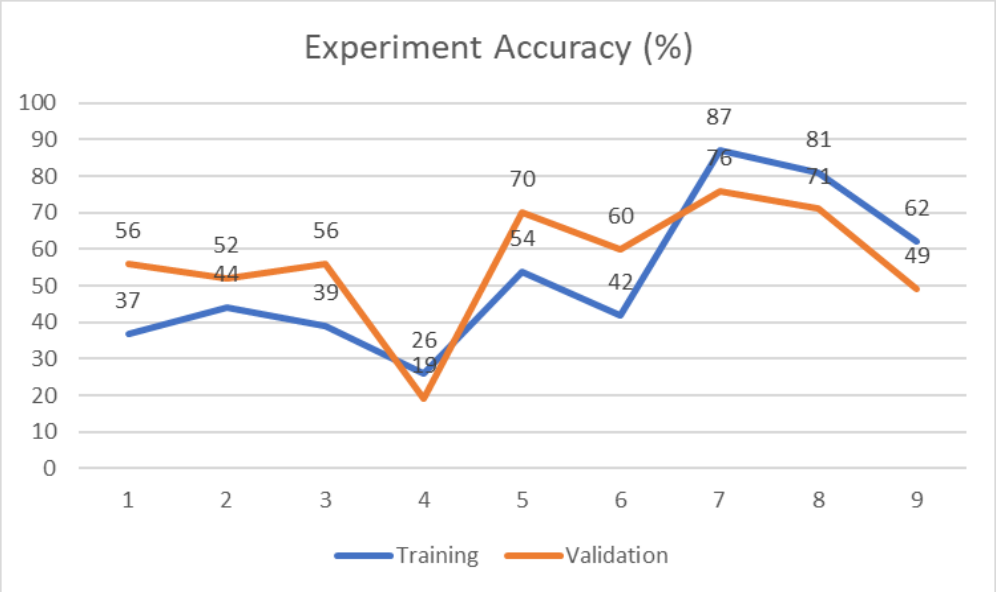
	Rate reduced to 25% and frames to 10 96,373 training parameters from a total of 96,485 parameters		**Reducing dropout rate also helped fix some fitting issues.**
8	**Conv3D with ADAM Optimiser** **Normal Image Quality - Decrease Frames, Dropout Rate, Batch Rate 4 with HIgher Epochs** 3 convolutional layers using Relu activation function and MaxPooling3D with a SoftMax Activation in the end. Dropout Rate reduced to 25% and frames to 10 with total epochs at 30 96,373 training parameters from a total of 96,485 parameters	Accuracy: 81% Val Accuracy: 70%	**Carrying on from experiment 7, increase epochs to see if we can further increase accuracy however this was not the case. We will now attempt to use CNN - LSTM with Adam Optimiser**
9	**CNN - LSTM with ADAM Optimiser** **Normal Image Quality - Decrease Frames, Dropout Rate, Batch Rate 4** 3 convolutional layers using Relu activation function and MaxPooling3D with a SoftMax Activation in the end. Dropout Rate reduced to 25% and frames to 10 142,549 training parameters from a total of 142,613 parameters	Accuracy: 62% Val Accuracy: 49%	**Using LSTM alongside the previous optimisations found didn't show any increase in accuracy.**

Summary

On this assignment we have tried to address the problem of gesture recognition. During the experiments we have attempted to find the best fit, combining different hyperparameters and model architectures:

- Conv3D vs CNN + RNN architecture
- Standard vs Low resolution images (100px, 50px and 25px)
- 50% vs 30% sequence size (number of frames per video)
- 50% vs 25% dropout rates
- Different batch sizes (8 and 4)
- SGD vs ADAM optimizer

The accuracy and validation accuracy results obtained during the experiments in a chronological order can be visualized on the image below:



Observations throughout our experiments:

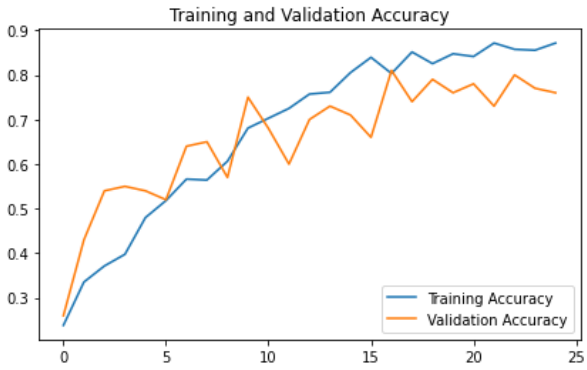
- More trainable parameters lead to an increase in training time overall and per epoch
- Increase in epoch did not necessarily lead to an increase in accuracy.
- Increasing batch size might have sped up training times; it however led to a decrease in accuracy which meant it was not worth the trade off for quicker results.

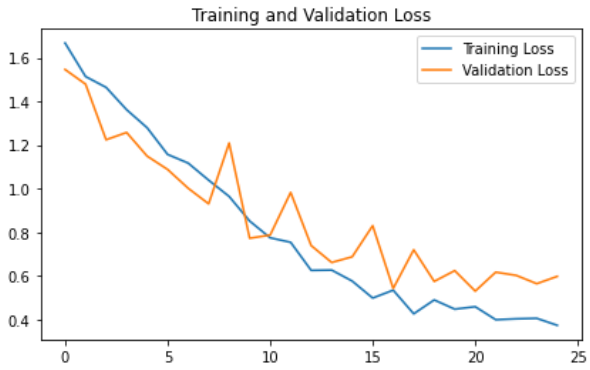
Recommended Experiment

After running all of our 9 experiments, we came to the conclusion that Experiment Number 7 - ADAM + Dropout + Frames performed the best.

Reasons are as follows:

- Training Accuracy: 87% with Validation Accuracy at 76%
- Training Params at 96,485 which lead to faster training time
- Learning rate decreases gradually





Suggestions for Improvement

- **Transfer Learning:** Due to time constraints and resource usage we were unable to conduct transfer learning using pre-trained models such as ResNet50 to carry out initial feature identification and then passing it onto RNN for sequencing following to the SoftMax layering for gesture classification.
- **Understanding and quality of Samples provided:** Better control of samples provided would benefit the overall training due to the vast difference in lighting, actor and camera resolution that the footage was taken with.
- **Tuning of Layers and Hyperparameters:** Trying out different combinations of layers and activation functions within the model would help further to increase accuracy but however we were limited with time and resources as each experiment took at least an hour for every tweak, test and final run.