Project – Deep Learning

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

I have used 2 architectures to solve this problem.

* CNN + RNN Stack
* Conv3D Architecture

With CNN+RNN stack I tried to extract features vectors from each image and these are fed to RNN to create model. With this setup I saw that training accuracy was very high and validation was very less , which is a clear case of overfitting.

In case of Conv3d, input is a video which is a sequence of 30 RGB images. Unlike 2dConv , we use 3D filters here which can be represented as (f1 \* f1 \*f1) \* C . Here value of C will be 3 as it is colored image and has 3 channels. This 3D filter will move in x y and z axis to convolve.

**Data Preparation**

We used 120\*120 image size

Cropping is done to recognize the gestures appropriately so that background noise doesn’t interfere in feature recognition

We normalized the image so that too bright or dark portion don’t affect our model.

TA : Training Accuracy. VA: Validation Accuracy

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Result** | **Decision + Explanation** |
| 1 | CNN+RNN stack (no dropout) | TA : 0.9881  VA : 0.6772 | Training accuracy was very high and validation accuracy was very less, which is a case of overfitting. So, I decided to add dropout |
| 2 | CNN+RNN stack + dropout | TA : 0.8844  VA : 0.5905 | I didn’t see any improvement in model so decided to use regularization. |
| 3 | CNN+RNN stack + dropout + batch normalization + regularization | TA: 0.9230  VA: 0.7095 | I saw improvement in model, but it still didn’t to very well. I decided to use Conv3D. |
| 4 | Conv3D layers + L2 Regularization + Batch Normalization + Dropout | TA: 0.7393  VA: 0.46 | I saw a in drop in training and validation accuracy which indicates that my model in underfitting. So I decided to use transfer learning. |
| 5 | Transfer learning + VGGNet + dropout, GRU layer | TA: 0.4978  VA: 0.5333 | Underfitting, we see training and validation accuracy is almost close but overall accuracy is very low. let’s try with another NN, mobilenet |
| 6 | Transfer learning + Mobilenet + dropout, GRU layer | TA: 0.9585  VA: 0.8571 | Wow, I saw improvement in model, both training and validation accuracy looks satisfactory and training time was less as compared to previous models. So I decided to play around with batch size and epoch. |
| Final Model | Transfer learning + Mobilenet + dropout, GRU layer + increased epoch(30) and batch size (20) | TA : 0.9735  VA: 0.89 | I saw increased training and validation accuracy. |

Final model

Chart, line chart

Description automatically generated

Total params: 3,444,485

Trainable params: 3,420,549

Non-trainable params: 23,936

Model: "sequential\_6"

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Layer (type) Output Shape Param #

=================================================================

time\_distributed\_47 (TimeDi (None, 15, 3, 3, 1024) 3228864

stributed)

time\_distributed\_48 (TimeDi (None, 15, 3, 3, 1024) 4096

stributed)

time\_distributed\_49 (TimeDi (None, 15, 1, 1, 1024) 0

stributed)

time\_distributed\_50 (TimeDi (None, 15, 1024) 0

stributed)

gru\_5 (GRU) (None, 64) 209280

dropout\_16 (Dropout) (None, 64) 0

dense\_16 (Dense) (None, 32) 2080

dropout\_17 (Dropout) (None, 32) 0

dense\_17 (Dense) (None, 5) 165

=================================================================

Total params: 3,444,485

Trainable params: 3,420,549

Non-trainable params: 23,936

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_