**Problem statement:**

In this problem, we need to design a deep learning neural network which will detect user’s hand gestures. There are 5 gestures which need to be identified.

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

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)**.

About the data:

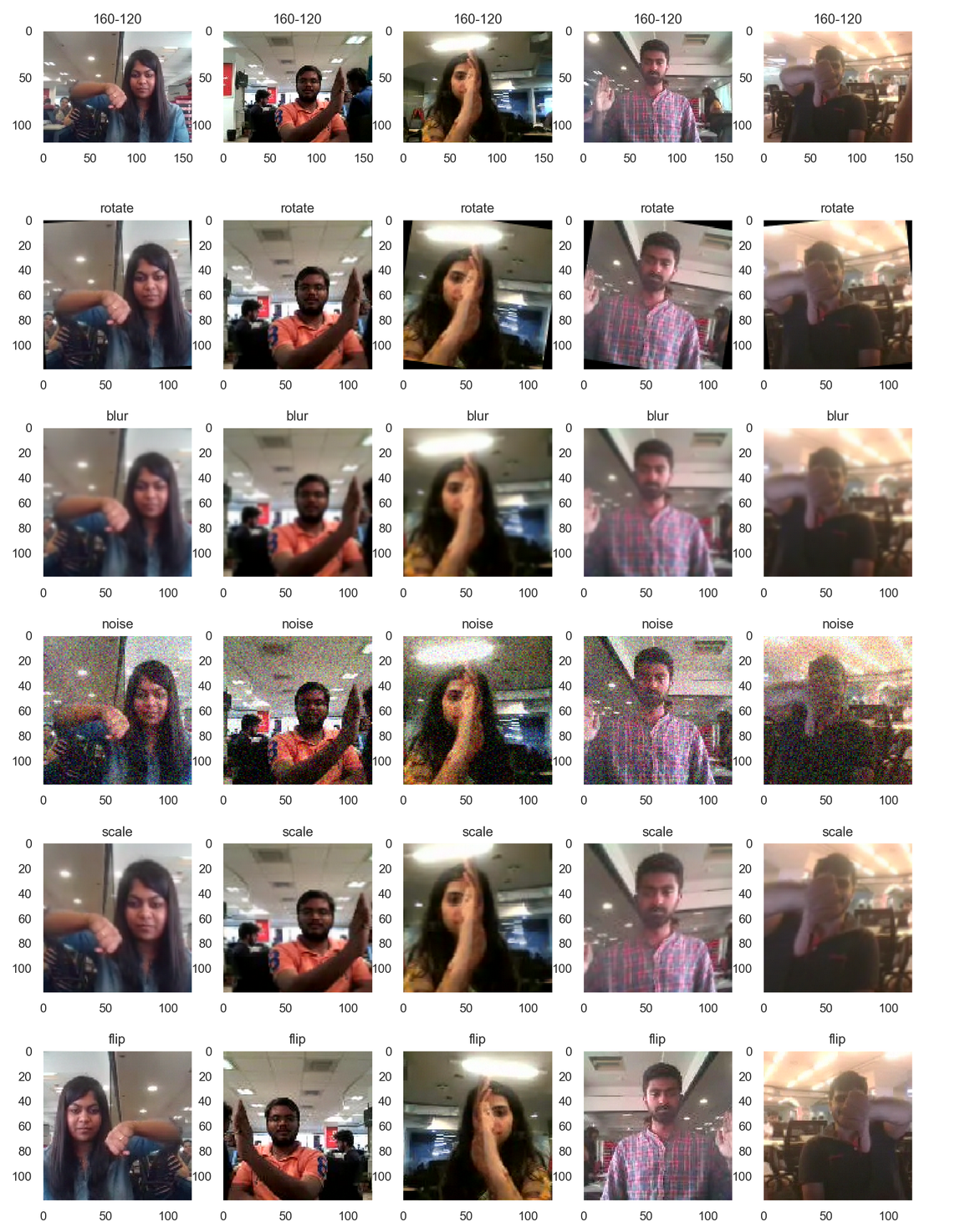
|  |  |  |
| --- | --- | --- |
| Training | Class | Number of Videos (folders) |
| 0 | 136 |
| 1 | 137 |
| 2 | 130 |
| 3 | 137 |
| 4 | 123 |

|  |  |  |
| --- | --- | --- |
| Validation | CLASS | NUMBER OF VIDEOS (FOLDERS) |
| 0 | 18 |
| 1 | 23 |
| 2 | 22 |
| 3 | 21 |
| 4 | 16 |

We can see that the data is evenly distributed among the classes. The images in the dataset have two dimensions: **160X120 and 360X360**

We will resize the images to a common size before feeding to the model.

**Images with different augmentations**



**Graph Colors:**

Training Loss Validation Loss

Training Accuracy Validation Accuracy

|  |  |  |  |
| --- | --- | --- | --- |
| E# | Model | Performance | Decision & explanation |
| 1 | CNN model  CONV 3D: (8,16,32,64)  Dense layer: (64) | A screenshot of a social media post  Description automatically generated  Training accuracy:21%  Validation accuracy: 23%. | The model did not run for 20 epochs as it was stopped using early stopping because there was no improvement in **validation loss (val\_loss).**  There was no significant change with epochs. So we decided to try different optimizer. We tried ADAM in the next experiment. |
| 2 | CNN model  CONV 3D: (8,16,32,64)  Dense layer: (64) | A close up of a map  Description automatically generated  Training accuracy:100%  Validation accuracy: 70%. | As we can see from the graph that the loss decreased significantly as the number of epochs increased. However, after around 6 epochs, the **training accuracy** started to increase and reached 1 signifying that the model was over fitting. The **validation accuracy** started to decrease. Thus, we decided to introduce batch normalization and dropout for our next experiment. |
| 3 | CNN model  CONV 3D: (8,16,32,64)  Dense layer: (64)  BN and DO | A close up of a map  Description automatically generated  Training accuracy:45%  Validation accuracy: 55%. | As can we see from the graph, the performance (**validation accuracy**) of the model decreased after introducing batch normalization and dropout. So, we decided to increase the complexity of the model by increasing the number of parameters. We increased the number of dense layers. |
| 4 | CNN model  CONV 3D: (8,16,32,64)  Dense layer: (128,64)  BN and DO | A close up of a map  Description automatically generated  Training accuracy:76%  Validation accuracy: 69%. | As we can see that the model is performing good (**validation accuracy**) with such less parameters. However, to improve further, we decided to introduce more regularization. |
| 5 | CONV 3D: (8,16,32,64)  Dense layer: (128,64)  BN and DO | A close up of text on a white background  Description automatically generated  Training accuracy:60%  Validation accuracy: 64%. | We can see that the performance of the model reduced. We decided to drop blur augmentation for the next experiment to check if it has any impact on the model performance. |
|  |  |  |  |

doesn’t state that ConvLSTM will give you better results than Conv3D. The explanation should be as detailed as possible so that the logic behind the decision is conveyed. Also, there are a lot of things you can experiment with in the generator function and elsewhere. Please do not forget to specify the exact metric values, here Accuracy which drives your decision.

You can draw inspiration from the concepts taught in the Industry demo in CNNs to experiment with the data and different architectures.

|  |  |  |  |
| --- | --- | --- | --- |
| **Experiment Number** | **Model** | **Result** | **Decision + Explanation** |
| **1** | **Conv3D** | **Throws Generator error** | **Crop the images correctly, try to overfit on less amount of data** |
| **2** | **Conv3D** | **Model not trainable as a lot of parameters** | **Reduce the size of the image/Reduce the number of layers** |
| **3** | **Conv3D** | **Accuracy: 0.21** | **Increase the amount of trainable data/ reduce the filter size** |
|  |  |  |  |
|  |  |  |  |
| **2** | **Conv3D** | **Accuracy: 0.32** | **Reduce Cropping** |
| **3** | **Conv3D** | **Accuracy : 0.38** | **………………** |
|  |  |  |  |
| **l-1th** | **Conv3D** | **Accuracy: 0.45** | **Try ConvLSTM as Conv3D not giving desired accuracy** |
| **lth** | **ConvLSTM** | **Accuracy: …….** | **…………..** |
|  |  |  |  |
| **Final Model** | **……………….** | **………….** | **…………………** |