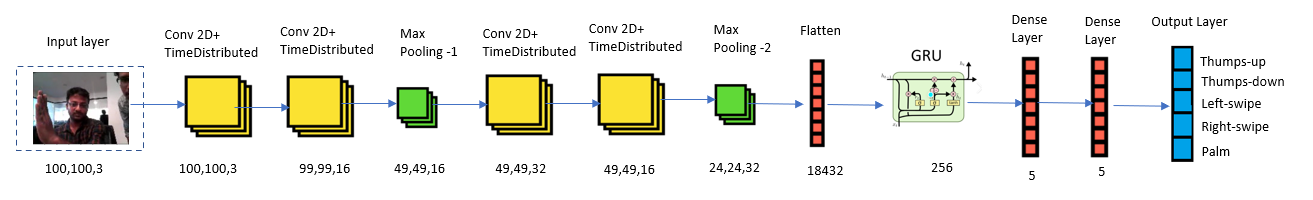
**Gesture Recognition**

**GOAL:** With given video clips, different classes such as thumps-up, thumps-down, left & right swipe and pause hand signal to control the video operation in Smart TV.

**Final Model:**

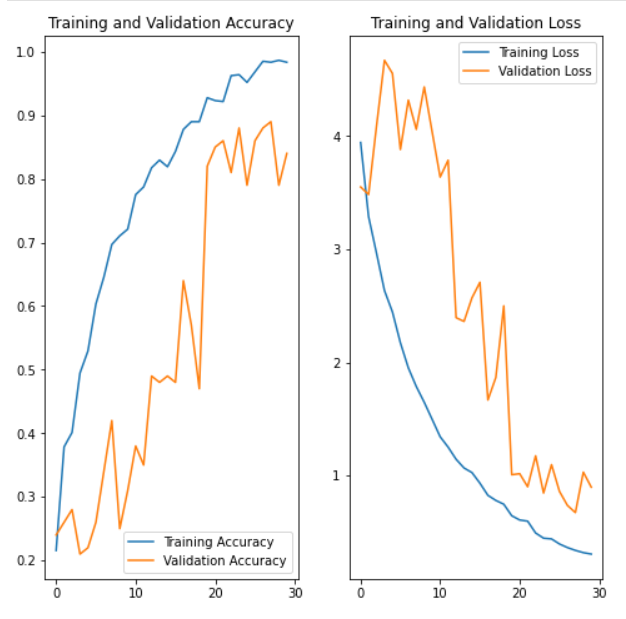
**Architecture:** CONV2D + Time Distributed + RNN



**Hyperparameter tuning:**

|  |  |  |
| --- | --- | --- |
| **SI NO** | **HYPERPARAMETERS** | **VALUES** |
| 1 | Image\_height & Width | 100,100,3 |
| 2 | No of CNN Layers | 4 |
| 3 | No of Pooling Layers | 2 \* Max Pooling |
| 4 | RNN Layers | Gated Recurrent Units |
| 5 | Regularization | Batch Normalisation + Dropouts |
| 6 | Activation Function | "SoftMax" in Output layer and "Relu" in rest of the layers |
| 7 | Optimizer | Adam |
| 8 | Kernel | 32,64,128,256 |

**Artifacts :**



**Experiment 1:**

**Architecture:** CONV2D + Time Distributed + RNN

|  |  |  |
| --- | --- | --- |
| **SI NO** | **HYPERPARAMETERS** | **VALUES** |
| 1 | Image\_height & Width | 200,200,3 |
| 2 | No of CNN Layers | 5 |
| 3 | No of Pooling Layers | 3 \* Max Pooling |
| 4 | RNN Layers | Gated Recurrent Units |
| 5 | Regularization | Batch Normalisation |
| 6 | Activation Function | "SoftMax" in Output layer and "Relu" in rest of the layers |
| 7 | Optimizer | Stochastic Gradient descent |
| 8 | Learning Rate | 0.01 |
| 9 | Batch Size | 128 |
| 10 | Ablation | 13 |
| 11 | Epoch | 20 |

**Result:**

InvalidArgumentError: Input to reshape is a tensor with 383385600 values, but the requested shape requires a multiple of 2304000

1. We have received multiple errors due to image resize and crop issues, we have fixed them one by one and finally the model was able to read all the images from generator as a batch.
2. We have chosen only few images, to be precise alternate images from the original dataset as still the information will be captured.
3. We handled image resize and crop through a new function.
4. Only 13 images were chosen as the initial ablation.
5. The remaining images from the batches were handled in such a way they will be sent in one batch to the model.

**Experiment 2:**

**Architecture:** CONV2D + Time Distributed + RNN

|  |  |  |
| --- | --- | --- |
| **SI NO** | **HYPERPARAMETERS** | **VALUES** |
| 1 | Image\_height & Width | 120,120,3 |
| 2 | No of CNN Layers | 4 |
| 3 | No of Pooling Layers | 2 \* Max Pooling |
| 4 | RNN Layers | Gated Recurrent Units |
| 5 | Regularization | Batch Normalisation |
| 6 | Activation Function | "SoftMax" in Output layer and "Relu" in rest of the layers |
| 7 | Optimizer | Stochastic Gradient descent |
| 8 | Learning Rate | 0.01 |
| 9 | Batch Size | 64 |
| 10 | Ablation | 10 |
| 11 | Epoch | 9 |

****

1. After changing Ablation count from 13 to 10, now the input shape error is arrested.
2. Now the Training accuracy Is too high and validation accuracy is too low, which is clear indication of Overfitting.

**Experiment 3:**

**Architecture:** CONV2D + Time Distributed + RNN

|  |  |  |
| --- | --- | --- |
| **SI NO** | **HYPERPARAMETERS** | **VALUES** |
| 1 | Image\_height & Width | 120,120,3 |
| 2 | No of CNN Layers | 4 |
| 3 | No of Pooling Layers | 2 \* Max Pooling |
| 4 | RNN Layers | Gated Recurrent Units |
| 5 | Regularization | Dropouts |
| 6 | Activation Function | "SoftMax" in Output layer and "Relu" in rest of the layers |
| 7 | Optimizer | Stochastic Gradient descent |
| 8 | Learning Rate | 0.01 |
| 9 | Batch Size | 64 |
| 10 | Ablation | 10 |
| 11 | Epoch | 9 |

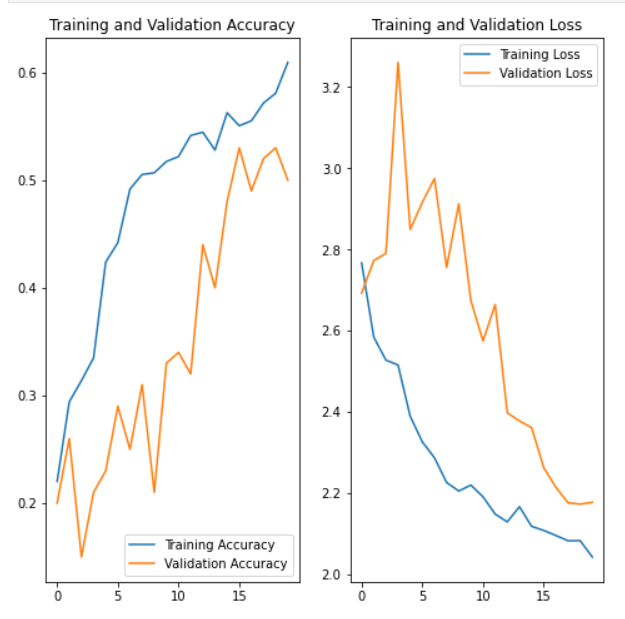
****

1. Our model performs well with training dataset but not with validation data, which is indication of Overfitting.
2. To control Overfitting, this time we added Dropout’s regularization along with Batch Normalization.
3. Now the validation accuracy is comparatively better than previous.

**Experiment 4:**

**Architecture:** CONV2D + Time Distributed + RNN

|  |  |  |
| --- | --- | --- |
| **SI NO** | **HYPERPARAMETERS** | **VALUES** |
| 1 | Image\_height & Width | 120,120,3 |
| 2 | No of CNN Layers | 4 |
| 3 | No of Pooling Layers | 2 \* Max Pooling |
| 4 | RNN Layers | Gated Recurrent Units |
| 5 | Regularization | Dropouts |
| 6 | Activation Function | "SoftMax" in Output layer and "Relu" in rest of the layers |
| 7 | Optimizer | Stochastic Gradient descent |
| 8 | Learning Rate | 0.001 |
| 9 | Batch Size | 64 |
| 10 | Ablation | 10 |
| 11 | Epoch | 20 |

****

1. After tuning learning rate and increasing epoch, now the model performance is much better than previous.
2. To note, Increase in epoch results in increase in training time.
3. This time the validation performance went near to 50 %, when training accuracy was 60 %.

**Experiment 5:**

**Architecture:** CONV2D + Time Distributed + RNN

|  |  |  |
| --- | --- | --- |
| **SI NO** | **HYPERPARAMETERS** | **VALUES** |
| 1 | Image\_height & Width | 100,100,3 |
| 2 | No of CNN Layers | 4 |
| 3 | No of Pooling Layers | 2 \* Max Pooling |
| 4 | RNN Layers | Gated Recurrent Units |
| 5 | Regularization | Batch Normalization + Dropouts |
| 6 | Activation Function | "SoftMax" in Output layer and "Relu" in rest of the layers |
| 7 | Optimizer | Adam |
| 8 | Learning Rate | 0.001 |
| 9 | Batch Size | 64 |
| 10 | Ablation | 10 |
| 11 | Epoch | 30 |

****

1. This time the validation performance went up to 70% and training accuracy was above 80%.
2. We have used “Adam” optimizer replacing SGD, and modified the input image size to 100.100, 3.
3. Changed to 30 epochs

**Experiment 6:**

**Architecture:** CONV2D + Time Distributed + RNN

|  |  |  |
| --- | --- | --- |
| **SI NO** | **HYPERPARAMETERS** | **VALUES** |
| 1 | Image\_height & Width | 100,100,3 |
| 2 | No of CNN Layers | 4 |
| 3 | No of Pooling Layers | 2 \* Max Pooling |
| 4 | RNN Layers | LSTM |
| 5 | Regularization | Batch Normalization + Dropouts |
| 6 | Activation Function | "SoftMax" in Output layer and "Relu" in rest of the layers |
| 7 | Optimizer | Adam |
| 8 | Learning Rate | 0.001 |
| 9 | Batch Size | 64 |
| 10 | Ablation | 10 |
| 11 | Epoch | 30 |



1. Tried replacing GRU with LSTM as LSTM do have more gates than GRU.
2. It was found model took additional time to complete the epoch.
3. After modifying kernel size and placed dropouts before activation layer, the model accuracy on train and test were ~90 and ~89 percent respectively in the final model.

===================================END====================================