Problem statement:

A home electronics company which manufactures state of the art smart televisions. It wants to develop a cool feature in the smart-TV that can recognise five different gestures performed by the user which will help users to control the TV. The gestures are continuously monitored by the webcam mounted on the TV. 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

Each video is a sequence of 30 frames (or images).

Approach:

To identify the gesture, a number of experiments were conducted. Models were trained with different image sizes (for ex: 120x120, 150x150) with different number of epochs and varying batch sizes. Common architectures used during training the model were:

1. Conv3d architecture
2. LSTM architecture
3. GPU architecture.

A number of models were built by using the above architectures. Results and outcomes are listed below with Decision and explanation.

At first conv3d architecture was tested with different number of hyper-parameters to find the best model and then the same layers and hyper parameters were taken to experiment to LSTM/GPU architecture to validate the model.

Output:

Architecture-1:

* Different models were trained with different hyper parameters tuned in batch\_size, epochs with image size

|  |  |  |  |
| --- | --- | --- | --- |
| Exp. Nb. | Model | Result | Decision/Explanation |
| Architecture-1)   * 5 layer each with 8, 16, 32, 64, 128 neurons were created followed by with two dense layers 1000, 500 each with drop out of 0.5 * In initial 3 runs, model was trained with varying number of batch sizes, epochs to see the better fit of hyperparameters. | | | |
| **Model-1**  **Conv3D** | Conv3D  Image size: 120  Batch size: 40  epochs: 15 | Accuracy:  Training: 87.89  Validation: 28.83  Validation loss: 5.5734 | Initial run to see if architecture works. In initial run model should overfit. |
| **Model-2**  **Conv3D**  Change in hyperparameter  Batch-size (40) | Conv3D:  Image size: 120  Batch size: 40  epochs: 25 | Accuracy:  Training: 88.93  Validation: 50.00  Validation loss: 2.33 | As the model was trained with more number of epochs model performed better. Training, validation accuracy went up and validation loss improved as well. |
| **Model-3**  **Conv3D**  Change in hyperparameter  Batch-size (30) | Conv3D:  Image size: 120  Batch size: 30  epochs: 25 | Accuracy:  Training: 66.67  Validation: 67.50  Validation loss: 1.12 | **Compared to model 2:**  Training accuracy is down, validation accuracy is up. If model is trained with less number of epochs model training accuracy should be high and as the model is trained with more number of epochs validation accuracy/training accuracy should be high. |
| **Model-5**  **Conv3D**  Change in hyperparameter  Batch-size (50) | Conv3D:  Image size: 120  Batch size: 50  epochs: 25 | Accuracy:  Training: 74.18  Validation: 21.33  Validation loss: 3.6693 | **Compared to model 2:** training/validation accuracy and validation loss is down. Compared with model 2 only change was made in hyper parameter (batch-size: 50) |
| **Model-7**  **Conv3D**  Change in hyperparameter  Epochs (50) | Conv3D:  Image size: 120  Batch size: 40  epochs: 50 | Accuracy:  **Training: 88.58**  **Validation: 85.00**  **Validation loss: 0.5372** | **Compared to model 2:**  As the model is trained with more number of epochs, model training/validation accuracy should improve and validation loss should goes down. |
| Approach:   * Conduct the same exercise with image size: 150x150 and with different hyper parameter. | | | |
| **Model-4**  **Conv3D** | Conv3D:  Image size: 150  Batch size: 40  epochs: 25 | Accuracy:  Training: 87.54  Validation: 55.00  Validation loss: 1.98 | **Compared to model 2:**  When model is trained with image size: 150x150 result is comparable.  Training/validation accuracy and validation loss are in range with image size:120x120. |
| **Model-6**  **Conv3D** | Conv3D:  Image size: 150  Batch size: 50  epochs: 25 | Resource exhaustion (OOM) |  |
| **Model-8**  **Conv3D**  hyperparameterturned (epochs:50) | Conv3D:  Image size: 150  Batch size: 40  epochs: 50 | Accuracy/Loss:  **Training: 84.43**  **Validation: 85.00**  **Validation loss: 0.3722** | **Compared to model-4:**  When trained with higher number of epochs training/validation accuracy and validation loss are in range. Results are similar. |
| Architecture-2:   * 5 layer each with 8, 16, 32, 64, 128 neurons were created followed by with two dense layers 1000, 500 each with drop out of 0.5 * Introduce drop-out (0.25) after 32, 64 & 128 layer. **(Change compared to architecture-1)** | | | |
| **Model-9:**  **Conv3D** | Conv3D:  Image size: 120  Batch size: 40  epochs: 25 | Accuracy/Loss:  Training: 70.93  Validation: 35.00  Validation loss:11.48 | **Compared to model-2:**  When drop-out layer was introduced after few layers accuracy went down. Possible reasons could be the feature loss and neuron layers are not dense enough.  Architecture-1 is better compared to architecture-2 when trained with same hyper parameters. |
| Architecture-3:   * 4 layer each with 16, 32, 64, 128 neurons were created followed by with two dense layers 1000, 500 each with drop out of 0.5 * Drop initial layer 8 (and use only 4 layers) **compared to architecture 1.** Keep the rest of the parameters same. | | | |
| **Model-10**  **Conv3D** | Conv3D:  Image size: 120  Batch size: 40  epochs: 25 | Accuracy/Loss:  Training: 89.97  Validation: 35.00  Validation loss: 2.0271 | **Compared to model 2:**  When trained with same number of hyper parameters, results are in range and model tends to over-fit slightly. Model should be trained with more number of epochs to see the output. |
| **Model-10- variant**  **Conv3D**  Hyperparameter tuned (epochs:50) | Conv3D:  Image size: 120  Batch size: 40  epochs: 50 | Accuracy/Loss:  Training: 88.58  Validation: 70.00  Validation loss: 0.8010 | **Compared to model 4:**  When model was trained with higher number of epochs though training/validation accuracy went up and validation loss went down.  But compared to model-4, model-10 did not perform very well. It’s due to the drop in initial layer 8. |
| Architecture-4:   * Variant of architecture-1 with SGD optimizer. Everything remains same. * As we noticed the model performs better with epochs=50. Experiment was conducted with 50 epochs * As with conv3d architecture it was noticed that model performs better with batch-size:40, and higher number of epochs (50). Model can be tried out same number of hyperparameters and results can be evaluated. | | | |
| **Model-11**  **Conv3D + SGD optimizer** | Conv3D: (SGD)  Image size: 120  Batch size: 40  epochs: 50 | Accuracy/Loss:  Training: 81.31  Validation: 0.5227  Validation loss: 85.00 | **Compared to model 7:**  When SGD optimizer was tried with same number of parameters results obtained are good and in range, but model training/validation accuracy is better in Adam optimizer |
| Architecture-5: (Conv2d + LSTM)   * 5 layer each with 8, 16, 32, 64, 128 neurons were created followed by with two dense layers 256, 128 each with drop out of 0.25. * Model is trained with directly higher number of epochs. * As with conv3d architecture it was noticed that model performs better with batch-size:40, and higher number of epochs (50). Model can be tried out same number of hyperparameters and results can be evaluated. | | | |
| **Model-12**  **LSTM** | **LSTM**:  Image size: 120  Batch size: 40  epochs: 50 | Accuracy/Loss:  Training: 88.93  Validation: 75.00  Validation loss: 0.6490 | **Compared to model 7:**  Model training/validation accuracy and validation loss is good when accuracy and loss plot is observed.  Model should be tried out with different architectures and more. Hyper parameter tuning for better accuracy. |
| Architecture-6: (Conv2d + GPU)   * 5 layer each with 8, 16, 32, 64, 128 neurons were created followed by with two dense layers 256, 128 each with drop out of 0.25. * Model is trained with directly higher number of epochs. * As with conv3d architecture it was noticed that model performs better with batch-size:40, and higher number of epochs (50). Model can be tried out same number of hyperparameters and results can be evaluated. | | | |
| **Model-13**  **GPU** | **GPU**:  Image size: 120  Batch size: 40  epochs: 50 | Accuracy/Loss:  Training: 97.58  Validation: 81.67  Validation loss: 0.5572 | **Compared to model 7 & model 12:**  Compared with Model-12, we can notice that model performance is better compared to LSTM architecture.  Results are comparable with Model 7, but with same layers and parameters, Conv3D architecture perform better. |
| Architecture-7: (Conv3D) – with data augmentation.   * Conv3d architecture was tried with different generator and cv2.wrapaffine() transformation was used to try out the model * Same architecture-1 was used with different generator to train the model. * As with conv3d architecture it was noticed that model performs better with batch-size:40, and higher number of epochs (50). Model can be tried out same number of hyperparameters and results can be evaluated. | | | |
| **Model-14**  **Conv-3D + Data augmentation** | **Conv3D**: (Data augmentation)  Image size: 120  Batch size: 40  epochs: 50 | Accuracy/Loss:  Training. : 80.97  Validation: 81.67  Validation loss: 0.4966 | **Compared to model 7:**  Training, validation accuracy and validation loss is in range. Even with different generators outputs are nearly the same. |
| **Final Model:** | | | |
| **Final Model:**  **Conv 3D** | **Conv3D**:  Image size: 120  Batch size: 40  epochs: 50 | Accuracy:  **Training: 84.43**  **Validation: 85.00**  **Validation loss: 0.3722** | Conv3d architecture tends to perform better compared with LSTP & GPU architecture. |