Gesture Recognition via cnn3d and ConvLSTM

02/07/24

Problem Statement:

Develop a deep learning model to predict 5 hand gestures to control a 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

The following table exhibits major experiments performed to build models to predict hand gestures.

Experiment #	Model	Model config / parameters	Model parameters	Result	Comment / Explanation
0	Conv3D	batch_size = 48			GPU memory overload error
CONV3D	**Baseline configuration** **Batch size: 8; Augmentation: 0 (False) ; Optimizer: Adam with 1e-3 learning rate** **Frames: 30; Image dim: 120x120x3; Epochs: 30; Early stopping: 10 (epochs);** **LR Scheduler : ReduceLROnPlateau: {patience: 4, min_lr: 5e-5, val-loss}** **Loss function: categorical cross entropy** **Model architecture configuration: Conv3D layers with filters shown below. Kernel size of 3 for filters and 2 for Maxpool is employed throughout. Architecture: [Conv-layer -> batch_norm -> maxpool] x N —> Flatten —> [Dense -> batch_norm] x N -> Dense(5, softmax)**				
1	Conv3D	Baseline Conv-filters: [8, 16]	221.22M	Train acc: 0.9517 Validation acc: 0.7500	Although the model performed reasonably well, the 221M parameters add space complexity. It is also overfitting. Add more convolution layers to reduce parameters.
2	Conv3D	Conv-filters: [16, 32, 64]	103.33M	Train acc: 0.8800 Validation acc: 0.9670	Model performance improved because of better features were extracted with using more filters in the convolution layers,

3	Conv3D	Conv-filters: [16, 32, 64, 128 20 Frames from 5 to 24	15.07M	Train acc: 0.9500 Validation acc:0.7600	No Augmentation Severally overfitting. Introduce augmentation.
4	Conv3D	Conv-filters: [16, 32, 64, 128 20 Frames from 5 to 24	15.07M	Train acc: Validation acc:	Augmenting with 50% of randomly sampled clips from train dataset.
5	Conv3D	Conv-filters: [16, 32, 64, 128]	22.45M	Train acc: 0.9945 Validation acc: 0.900	No Augmentation Lower parameters and accuracy are in a trade-off with Conv3D.
6	Conv3D	Conv-filters: [16, 32, 64, 128]	22.45M	Train acc: 0.9648 Validation acc: 0.7500	Augmenting with 50% of randomly sampled clips from train dataset.
CONV2D + ConvLSTM	**Baseline configuration (same as CONV3D)** **Batch size: 10; Augmentation: 0 (False) ; Optimizer: Adam with 1e-3 learning rate** **Frames: 30; Image dim: 120x120x3; Epochs: 30; Early stopping: 10 (epochs);** **LR Scheduler : ReduceLROnPlateau: {patience: 4, min_lr: 5e-5, val-loss}** **Loss function: categorical cross entropy** **Model architecture configuration: Conv3D layers with filters shown below. Kernel size of 3 for filters and 2 for Maxpool is employed throughout. Architecture: [Timedistributed(Conv2D] x N —> Time-distributed(Dense) -> BatchNorm -> Global Average Pooling 2D—> [Dense -> batch_norm] x N -> Dense(5, softmax)**				
1	Conv2D + Conv-LSTM	Baseline config clip_dim: 30x3x120x120	13,781	Train acc: 0.9216 Validation acc: 0.9400	- Validation accuracy (0.9400) surpassed train accuracy (0.9155) marginally - Achieved the result with only 13,781 parameters - Trained all 30 frames. In the next series of experiments, the frames are reduced, by taking from the middle of the video clip, to investigate performance.
2	Conv2D + Conv-LSTM	Clip_dim: 20x3x120x120 Clip indices from 5 to 24 in timeframe order	13,781	Train acc: 0.9216 Validation acc: 0.9500	- Validation accuracy of the model trained with 20 frames from the middle of the video clip is better than using all the 30 frames. The difference in accuracy is only 0.01 (0.95 vs. 0.94). - In the next experiment, the frames are reduced to 15.

3	Conv2D + Conv-LSTM	Clip_dim: 15x3x120x120 Clip indices from 7 to 21 in time frame order	13,781	Train acc: 0.8446 Validation acc: 0.900	Train and validation accuracy are 0.8446 and 0.9 respectively. The gap is more than the previous two experiments. The accuracies are increasing and had we ran beyond 30 epochs the values would have been closer to previous best. In the next experiment, the frames were increased to 17 and introduced augmentation on 25% of training clips. The train dataset will aggregate to 1.25x of the original.
4	Conv2D + Conv-LSTM	Clip_dim: 17x3x120x120 Clip indices from 7 to 23 in time frame order Augmented 25% of train images	13,781	Train acc: 0.9011 Validation acc: 0.9100	The performance difference is marginal (by +0.01) in comparison with 15-frames experiment 2
5	Conv2D + Conv-LSTM	Using 20 frames with 160x160 images. Frames are from 6 to 26. Augmented 50% of train images	13,781	Train acc: 0.9905 Validation acc: 0.900	Augmentation do not work with the sequential modeling
5	Conv2D + Conv-LSTM	No Augmentation LSTM layers increased from 8 to 16	25877	Train acc: 0.9200 Validation acc: 0.900	Adding more LSTM layers may have caused vanishing gradients.

Conclusions:

- The model built with Conv3D (experiment 2) outperformed on validation prediction accuracy and it comes with space complexity. Second in the list is train distributed Conv2D + ConvLSTM2D (experiment 2)
- Models built with 20 frames time-ordered between 5 and 25 gave better results compared to all the combinations.
- The Conv2D + ConvLSTM2D model is advised to be selected for several orders of lower parameters.