Experiment Tracker

A write-up on the different experiments tried out for Gesture Recognition Case Study.

The metrics used throughout the project are:

- categorical_crossentropy_loss
- categorical_accuracy

Note:

- Since the project was run on local GPU, the batch sizes maybe smaller than expected.
- Some of the Experiments contained multiple trials. Only the best results for each experiment are shown below.

Experiment 1

Model: Conv3D

Decision + Explanation

- Completed the generator without handling partial batch.
- Chose frames as every third image from all frames.
- Resized all images to 120x120. Cropped non-square images.
- Model architecture:
  - 3 sets of Conv3D -> MaxPooling3D -> BatchNormalization
  - GlobalAveragePooling3D -> Flatten
  - Dense -> Dropout
- Ran it on a sample of 30 train videos and 30 val videos
- Found max batch size that worked was 8.

Result:

- Code runs but model doesn't train.

Experiment 2

Model: Conv3D

Decision + Explanation:

- Added code to handle partial batch in Generator.
- Ran it on full data

Result:

- Train Accuracy: 0.55
- Validation Accuracy: 0.46
- Models trains but underfits.

Experiment 3

Model: Conv3D

Decision + Explanation:

- Reduced Learning Rate.

Result:

- Scores improved but still underfits.
- Train Accuracy: 0.62
- Validation Accuracy: 0.65

Experiment 4

Model: Conv3D

Decision + Explanation:

- Increased number of Conv layers.

Result:

- Memory Exhausted Error.

Experiment 5

Model: Conv3D

Decision + Explanation:

- Went back to model in Experiment 3.
- Added one more Dense layer and increased dense units.
- Switched to same padding

Result:

- Scores improved but still underfits.
- Train Accuracy: 0.66
- Validation Accuracy: 0.80

Experiment 6

Model: TimeDistributed(Conv2D)+GRU

Decision + Explanation:

- Model architecture:
  - 3 sets of TimeDistributed(Conv2D) -> MaxPooling2D -> BatchNormalization
  - TimeDistributed(Flatten)
  - 2 sets of Dense -> Dropout
- Max working batch size was 4.

Result:

- Error due to Input shape being different. Conv3D expects `(width, height, frames, channels)`. TimeDistributed(Conv2D) expects `(frames, width, height, channels)`.

Experiment 7

Model: TimeDistributed(Conv2D)+GRU

Decision + Explanation:

- Fixed issue with shapes.

Result:

- Error due to not applying TimeDistributed to MaxPooling.

Experiment 8

Model: Conv3D

Decision + Explanation:

- Removed MaxPooling layers and changed the strides to 2 in Conv2D get similar results.
- Increased frames count from 10 to 18 (out of 30) to add more data.
- Added logic to incorporate grayscale images and calculated scores with grayscale images.

Result:

- Scores improved but still underfits.
- Train Accuracy: 0.74
- Validation Accuracy: 0.84

Experiment 9

Model: TimeDistributed(Conv2D)+GRU

Result:

- Error due to not mismatch in sizes of expected input and actual input to GRU unit. This was caused due to input having 10 frames while model expected full 30.

Experiment 10

Model: TimeDistributed(Conv2D)+GRU

Decision + Explanation:

- Corrected frames count in model creation code. Model was expecting 30 frames while the input data had 10.
- Increased Dropout to rectify the overfitting.
- Changed strides back to 1 and added TimeDistributed MaxPooling2D and BatchNormalization.

Result:

- Model Overfit.
- The model takes a long time to converge.
- Train Accuracy: 0.97
- Validation Accuracy: 0.80

Experiment 11

Model: TimeDistributed(Pretrained-Conv2D)+GRU

Decision + Explanation:

- Replaced custom Conv2D layers with a pre-trained MobileNetV2.

Result:

- Scores look good

- The model converges in less iterations.
- Train Accuracy: 0.89
- Validation Accuracy: 0.86
- Looks like a good candidate for final model.

Experiment 12

Model: TimeDistributed(Pretrained-Conv2D)+GRU

Decision + Explanation:

- Changed logic for image resize. Resized 300x300 images to 160x160. Padded 120x160 images to bring them to 160x160.
- Some information was getting cropped out due to initial resize method. Padding the smaller images helped improve the score a little more.

Result:

- Train Accuracy: 0.90
- Validation Accuracy: 0.90
- This is the final model:
  - Has acceptable scores.
  - Does not Over/Under fit.
  - Has a reasonable number of parameters.
  - Convergence is faster and stable.