Control the Mountain-Car-v0 Gym env using a webcam

Study Mountain-Car-v0 Gym environment

This environment takes three actions 0,1, and 2.

0 is moving left, 1 is no movement, and two is moving right.

Create Dataset

The number of gestures to control the car will be three since there are three different actions in the Mountain-Car-v0 Gym environment.

Control of the mountain car will be done through a webcam, so; gestures must be rapid to change, and the webcam angle should always be the same concerning the hand.

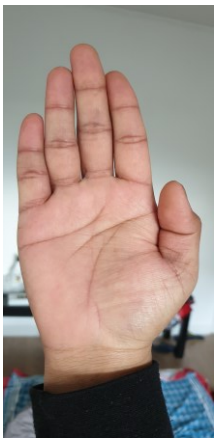

Fig. 1
Action = 1

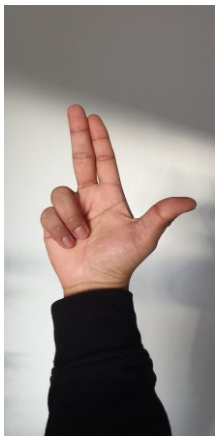

Fig. 2
Action = 2

Fig. 3
Action = 0

Pictures are taken from mobile phones and webcams to create a dataset. So that number of images can be increased, and different quality and angles can be included in the dataset.
The dataset includes images where the only gesture is present in the picture, i.e., Fig 1,2, and 3, and also where gesture and face/body are visible to create a more generalized dataset.

The risk involved with this dataset are:

Dataset is small.

It includes images of one person only and very minimum background.

It might fail when predicting where other objects/body parts are present in the image, along with the gestures.

Almost 20% of the images were used for validation.

Total Images = 159

Training images = 124

- Action 2 images = 44
- Action 1 images = 41
- Action 0 images = 39

Validation images = 35

- Action 2 images = 12
- Action 1 images = 12
- Action 0 images = 11

Training Model

Due to the small dataset, I preferred not to create a custom model from scratch.

With a small dataset, transfer learning provides an advantage and much better performance because it is already trained on a vast and general enough dataset.

To select a suitable pre-trained model, I looked for research papers to find out which model works better with gesture detection. And I found out that VGG16 is compatible with gesture detection from this paper.

https://link.springer.com/chapter/10.1007/978-981-16-7118-0_11

Not every model will provide the desired outcome with every dataset when using transfer learning to train models.

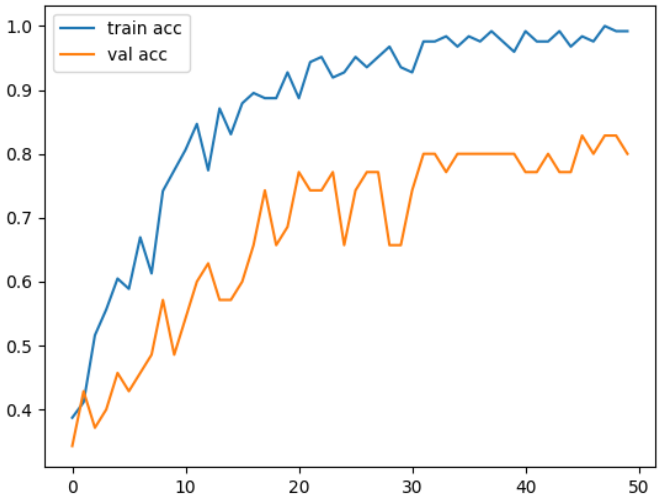

Fig. 4
Training Accuracy and Validation Accuracy

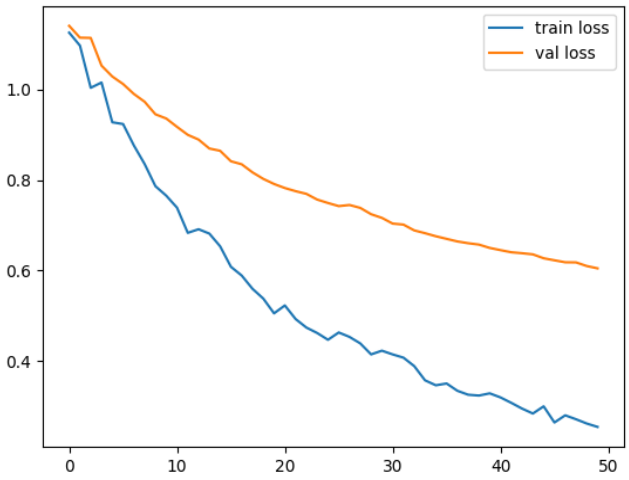

Fig. 5
Training Loss and Validation Loss

From Training Loss and Validation loss, it is concluded that the training dataset might be unrepresentative or the model is underfitting. Because training loss is decreasing continuously and has not become stable, the model can be improved using a more generalized dataset and increasing training time.

Loss curves also conclude that the model has trained at a reasonable learning rate, and its accuracy has increased concerning epochs and time.

Training and Validation Accuracy is promising enough. Thus, it is concluded that the model is ready for testing.

Step 4: Model Testing

The model was tested using a webcam in real-time for real-time. The model was accurate when there was only a gesture visible through the webcam, but when the face was also visible to the webcam model made false predictions.

The model also gave a false prediction when no gesture was present in front of the webcam.

These false predictions can result from the small dataset to train the model and less training time. The model does not extract features precisely for gesture detection.

The model gave desired output when gestures were close enough to the webcam, and no other object was present in the image.

Step 5: A model visual explanation using the Grad-CAM

I have chosen Grad-Cam to interpret model prediction because it produces a coarse localization map highlighting the critical regions in the image for predicting the concept. Which helps to visualize precisely what the model looks at in the picture for a particular class prediction.

Grad-CAM can be used to better understand a model by providing insight into model failure modes.

Fig. 6 shows what the model looks at when detecting gesture for Action = 2.

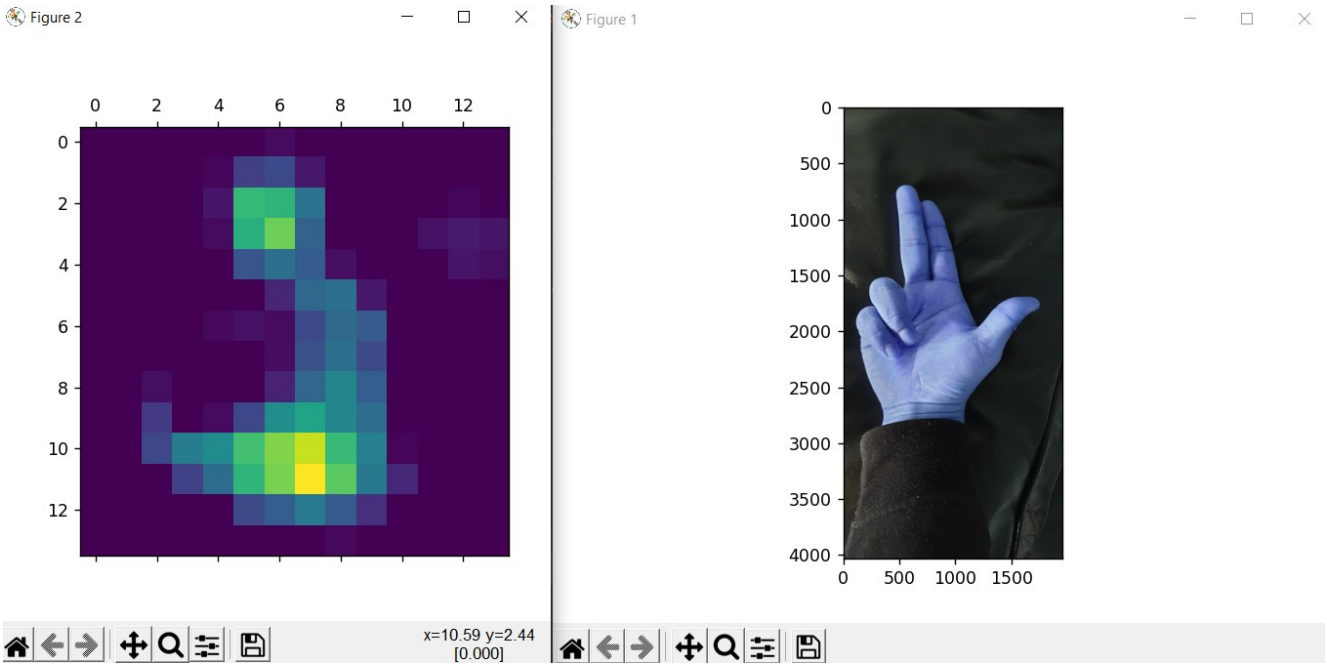

Fig. 6
Grad-CAM Vision

Several images of all classes were fed to the Grad-CAM function, and the results were analyzed. From these results, it was concluded that the model understands the difference between all three gestures when there was only a gesture present in the image; that's why the model gave accurate predictions.

Step 6: Control Mountain Car

Predicted classes by the model were given to Mountain-Car-v0 Gym env as action.

Control of the mountain car was a success, as seen in the video. The model predicted correctly, and those expected class values were given to the environment as action. The car showed changes in position and velocity according to the given action.

Code Flow/ Explanation

1. To create a dataset using a webcam, the create_dataset.py python file was created.
    1.1. Create_dataset function takes two arguments as input. The first is the path where the images will be stored, and the second is the number of images to be taken and saved.
2. To train model detection_model.py python file was created.
    2.1. VGG16 model is downloaded without the top (Prediction) layer with ImageNet weights.
    2.2. All the weights were frozen to stop them from training.
    2.3. New prediction layer is added with three output nodes.
    2.4. Model is created combining VGG16 model, new prediction layer, and trained prediction layer.
    2.5. Model training and validation accuracy and loss were analyzed.
3. To test the model live_detection.py python file was created.
    3.1. It converts webcam frames into images and feeds them as input images to the model. Model predictions were compared and analyzed.
4. To understand the model grad_cam.py python file was created.
    4.1. Grad-CAM function returns heatmap of what model looks at in the image.
5. To control the mountain car contro_mountain_car.py python file was created.
    5.1. This is an extension of the live_detection.py python file. Which takes the prediction class as input and feeds them to the Mountain-Car-v0 Gym environment as Action.