Motor AI Technical Challenge - Report

Task 1: Control the Mountain-Car-v0 Gym env using your webcam

- **Workflow :**
    1. Getting familiarized with Open AI gym
    2. Understanding the environment 'Mountain car v0'
    3. Running 'Mountain car v0' environment with random actions
    4. Gesture selection
    5. Dataset creation
    6. Preprocess dataset and dataflow from directory
    7. Model creation and optimization
    8. Improving dataset
    9. Controlling 'Mountain car v0' with real-time hand-gesture using laptop webcam

- **Gesture Selection:**

No Push -

Push Left -



Push Right-



- **Dataset creation**
    1. For creating the dataset the process used :
        - Webcam video feed
        - Tagging dataset through keyboard
        - Saving the tagged frame in specified label folders
    2. The first dataset performed **very poorly** with the model as it was:
        1. raw image
        2. small size : [train, test, validation] = [1111, 300, 75]
        3. unmasked
        4. without applying any filters
    3. The next dataset performed much better with the model as:
        1. Increased size [train, test, validation] = [2100, 303, 75]
        2. ROI
        3. Thresholding
        4. Masking
        5. Blurring
    4. Performance of the dataset is described with models in model selection summary

Motor AI Technical Challenge - Report

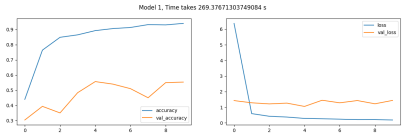
- **Preprocess dataset and dataflow from directory**
    - Normalization
    - Shearing (only for training data)
    - zooming (only for training data)
    - horizontal flip (only for training data)
    - I have used **ImageDataGenerator.flow_from_directory** for reading the data from the directory as labelled

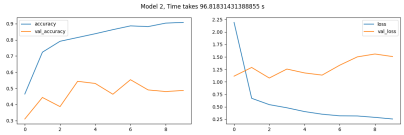
- **Model creation and optimization (also comparing dataset)**

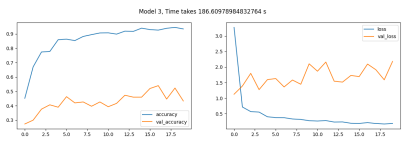
Data used - Dataset1:

1. raw image
2. small size : [train, test, validation] = [1111, 300, 75]
3. unmasked
4. without applying any filters

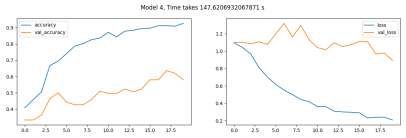
Model	Result/Accuracy	Comment

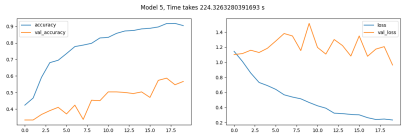
• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 1
  • **# Neurons in each layer :** [64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** False
• **Pooling applied :** None
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 10
• **Batch size :** 32
		• highly overfit
		  • Epoch 10 - loss: 0.1877 - accuracy: **0.9406** - val_loss: 1.4405 - val_accuracy: **0.5533**
		• computationally heavy
		• time taken for only 10 epochs = 269.38 s

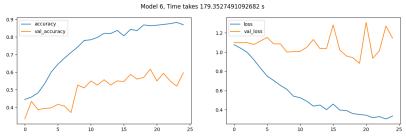
• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 1
  • **# Neurons in each layer :** [64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True (to address the overfitting)
• **Pooling applied :** MaxPool2D (to reduce computational load)
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 10
• **Batch size :** 32
		• still highly overfit (however, training accuracy reduced a lot)
		  • Epoch 10 - loss: 0.2565 - accuracy: **0.9082** - val_loss: 1.5068 - val_accuracy: **0.4867**
		• time taken for 10 epochs = **96.81831431388855 s** (thus we can increase the number of epochs)

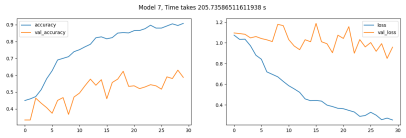
• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 1
  • **# Neurons in each layer :** [64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 20
• **Batch size :** 32
		• still highly overfit (deteriorated)
		  • Epoch 20 - loss: 0.1847 - accuracy: **0.9343** - val_loss: 2.1820 - val_accuracy: **0.4333**
		time taken for 20 epochs = **186.60978984832764 s**

Motor AI Technical Challenge - Report

• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 3
  • **# Neurons in each layer :** [32, 64, 64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 20
• **Batch size :** 32
		• still overfit (improved)
		  • Epoch 20 - loss: 0.2096 - accuracy: **0.9262** - val_loss: 0.8911 - val_accuracy: **0.5800**
		• time taken for 20 epochs = **147.6206932067871 s**

• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 3
  • **# Neurons in each layer :** [64, 128, 128]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 20
• **Batch size :** 32
		• still overfit (deteriorated)
		  • Epoch 20 - loss: 0.2337 - accuracy: **0.9046** - val_loss: 0.9628 - val_accuracy: **0.5667**
		• time taken for 20 epochs = **224.3263280391693 s (deteriorated)**

• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 4
  • **# Neurons in each layer :** [32, 64, 64, 64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 25
• **Batch size :** 32
		• still overfit (improved)
		  • Epoch 25 - loss: 0.3349 - accuracy: **0.8695** - val_loss: 1.1449 - val_accuracy: **0.5967**
		• time taken for 25 epochs = **179.3527491092682 s**

• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 4
  • **# Neurons in each layer :** [32, 64, 64, 64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 516
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 30
• **Batch size :** 32
		• still overfit (deteriorated)
		  • Epoch 30 - loss: 0.2533 - accuracy: **0.9082** - val_loss: 0.9587 - val_accuracy: **0.5867**
		• time taken for 30 epochs = **205.73586511611938 s (deteriorated)**

Comment : Since changing different hyper-parameters of the model is not improving the accuracy, it could be concluded that we need to take care of the dataset:

A. Increase the number of images in the dataset - (not done as that will be computational expensive)

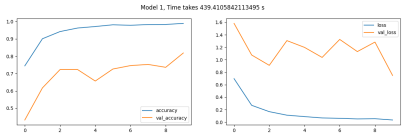
B. Preprocess the video to create the dataset

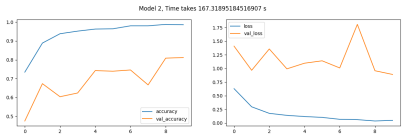
Motor AI Technical Challenge - Report

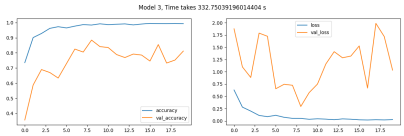
Data used - Dataset2:

1. Increased size [train, test, validation] = [2100, 303, 75]
2. Region of Interest
3. Thresholding
4. Masked
5. Blurred with Gaussian Blur

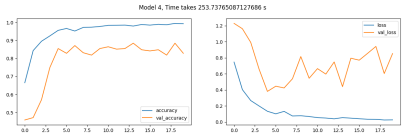
Model	Result/Accuracy	Comment

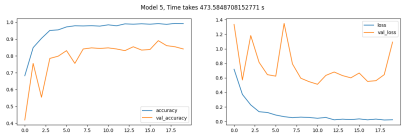
• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 1
  • **# Neurons in each layer :** [64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** False
• **Pooling applied :** None
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 10
• **Batch size :** 32
		• delayed convergence for validation data
		• Epoch 10 - loss: 0.0387 - accuracy: **0.9900** - val_loss: 0.8265 - val_accuracy: **0.8119**
		• computationally heavy
		• time taken for only 10 epochs = 401.2909851074219 s

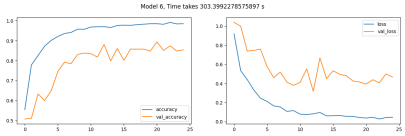
• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 1
  • **# Neurons in each layer :** [64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True (to address the overfitting)
• **Pooling applied :** MaxPool2D (to reduce computational load)
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 10
• **Batch size :** 32
		• delayed convergence for validation data
		• Epoch 10 - loss: 0.0514 - accuracy: **0.9838** - val_loss: 0.7508 - val_accuracy: **0.8350**
		• time taken for 10 epochs = **158.1661729812622 s** (thus we can increase the number of epochs)

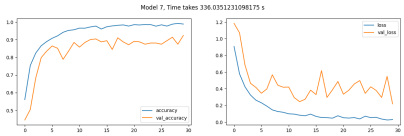
• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 1
  • **# Neurons in each layer :** [64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 20
• **Batch size :** 32
		• decent model
		• Epoch 20 - loss: 0.0250 - accuracy: **0.9929** - val_loss: 1.0348 - val_accuracy: **0.8119**
		• time taken for 20 epochs = **332.75039196014404 s**

Motor AI Technical Challenge - Report

• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 3
  • **# Neurons in each layer :** [32, 64, 64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 20
• **Batch size :** 32
		• decent model(improved)
		• Epoch 20 - loss: 0.0254 - accuracy: **0.9924** - val_loss: 0.8529 - val_accuracy: **0.8284**
		• time taken for 20 epochs = **253.73765087127686 s**

• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 3
  • **# Neurons in each layer :** [64, 128, 128]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 20
• **Batch size :** 32
		• decent model(improved)
		• Epoch 20 - loss: 0.0234 - accuracy: **0.9929** - val_loss: 1.0923 - val_accuracy: **0.8416**
		• time taken for 20 epochs = **473.5848708152771 s (deteriorated)**

• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 4
  • **# Neurons in each layer :** [32, 64, 64, 64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 256
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 25
• **Batch size :** 32
		• decent model(improved)
		• Epoch 25 - loss: 0.0470 - accuracy: **0.9852** - val_loss: 0.4671 - val_accuracy: **0.8548**
		• time taken for 25 epochs = **303.3992278575897 s**

• **Convolution Model Metrics**
  • **Model used :** Conv2D
  • **# layers used :** 4
  • **# Neurons in each layer :** [32, 64, 64, 64]
  • **Activation function :** relu
  • **Kernel size :** 2
• **Dropout :** True
• **Pooling applied :** MaxPool2D
• **Dense Layer Metrics**
  • **# Nuerons :** 516
  • **Activation function :** relu
• **Optimizer:** 'adam'
• **Epochs :** 30
• **Batch size :** 32
		• decent model(improved)
		• Epoch 30 - loss: 0.0296 - accuracy: **0.9886** - val_loss: 0.2186 - val_accuracy: **0.9241**
		• time taken for 30 epochs = **336.0351231098175 s (deteriorated)**

Comment : The preprocessed dataset works pretty well

Motor AI Technical Challenge - Report

- **Controlling 'Mountain car v0' with real-time hand-gesture using laptop webcam**

———-Pseudo code---------
1. initializing the environment
2. load model
3. loop for each episode
   a. resetting env
   b.video processing
      i. open video
      ii. image processing
      iii. interpreting result using model
      iv. print output on screen
      v. close video
----------end----------

- **Analysis of the challenge solution**

1. **Gesture :** selected keeping in mind the similarity of these gestures with traffic signs
2. **Dataset :**
    1. Choice :
        1. compared on different models to check its performance
        2. preprocessed dataset(the one with thresholding, masking, blurring) to reduce the computational expense
        3. Dataset2
    2. Performance :
        1. the dataset does a decent job to train the model efficiently
    3. Scope of improvement
        1. The masking is done on the basis of colour
        2. performs poorly if having a complicated background
        3. can be improved by changing the masking strategy or increasing the size of the dataset
3. **Model :**
    1. Choice :
        1. comparing accuracies
        2. Model 7
    2. Performance :
        1. good accuracy : Epoch 30 - loss: 0.0296 - accuracy: **0.9886** - val_loss: 0.2186 - val_accuracy: **0.9241**
    3. Scope of improvement
        1. Can be compared by changing few more hyper-parameters like the activation function, optimizer, batch size
4. **Environmental Control**
    1. Performance :
        1. Works well because of decently trained model
    2. Scope of improvement
        1. If the model could be trained with huge dataset, the control can me seamless

- **References**

1. **https://github.com/chasinginfinity/number-sign-recognition/blob/master/collect-data.py**
2. **Udemy Course - 'Autonomous Cars: Deep Learning and Computer Vision in Python'**
3. **Udemy Course 'Machine Learning A-Z™: Hands-On Python & R In Data Science'**