**Project: Build a Traffic Sign Recognition Program**

Deadline: March 9th, 2023

Requirement: Video report (not more than 3 mins)

Video link and report to be shared via LMS

For testing purpose, use photo taken by you.

GIT link also to be shared in the video.

In this project, deep neural networks and convolutional neural network architecture (LeNet) to classify traffic signs needs to be used.

For model training and validation use “[German traffic sign dataset](https://bitbucket.org/jadslim/german-traffic-signs)” and later use the trained model to apply on traffic sign images taken from internet.

Steps followed:

1. Import Libraries and datasets
2. Perform image visualization

Dataset consists of 43 different classes

Images are 32\*32 pixels of 3 RGB channels.

Data loading for training, validation and testing

Image Shape: (32, 32, 3)

Training Set: 34799 samples

Validation Set: 4410 samples

Test Set: 12630 samples

1. Convert images to gray-scale and perform normalization

RGB shape for train: (34799, 32, 32, 3)

RGB shape for valid: (4410, 32, 32, 3)

RGB shape for test: (12630, 32, 32, 3)

Grayscale shape for train: (34799, 32, 32, 1)

Grayscale shape for valid: (4410, 32, 32, 1)

Grayscale shape for test: (12630, 32, 32, 1)

Graphical user interface

Description automatically generated with medium confidence

1. Build a deep convolutional network model using Keras

You may use the following architecture:

LeNet architecture

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Layer | Description | | Input | Output |
| 1 | Convolution | kernel: 5x5; stride:1x1; padding: valid | 32x32x3 | 28x28x6 |
| Max pooling | kernel: 2x2; stride:2x2; | 28x28x6 | 14x14x6 |
| 2 | Convolution | kernel: 5x5; stride:1x1; padding: valid | 14x14x6 | 10x10x16 |
| Max pooling | kernel: 2x2; stride:2x2; | 10x10x16 | 5x5x16 |
| Flatten | Input 5x5x16 -> Output 400 | 5x5x16 | 400 |
| 3 | Fully connected | connect every neural with next layer | 400 | 120 |
| 4 | Fully connected | connect every neural with next layer | 120 | 80 |
| 5 | Fully connected | output 43 probabilities for each label | 80 | 43 |

Hyper-parameter’s passed:

LEARNING\_RATE = 0.01

EPOCHS = 100

BATCH\_SIZE = 100

1. Compile and train deep convolutional network model
2. Assess trained model performance
3. Output

Text

Description automatically generated

And the trained model successfully able to classify the signs.

Graphical user interface, application

Description automatically generated

1. Improvement/ Future Scope