M1 Statistical Machine Learning

ML for road signs recognition

Mohammed Akib Iftakher

Number: 20215282

Lecturer: Professor Lydie NOUVELIERE

Contents

Introduction

In this age of Artificial Intelligence, human dependence on technology is increasing. Numerous international corporations, like Google, Tesla, Uber, Ford, Audi, Toyota, Mercedes-Benz, and others, are utilizing improved technology to automate vehicles. They are attempting to improve the precision of autonomous or driverless cars. Self-driving automobiles, in which the vehicle itself acts as a driver and does not require human direction to operate on the road. It is not inappropriate to consider safety concerns, such as the possibility of significant machine accidents. However, no machine is more precise than humans. Numerous algorithms are being tested by researchers to assure 100 percent road safety and accuracy. Traffic Sign Recognition is one such algorithm discussed in this blog.

For a person to drive safely, he or she must obey numerous traffic signs such as traffic signals, turn left or right, speed restrictions, no passing of large vehicles, no entrance, children crossing, etc. To achieve accuracy, autonomous vehicles must also analyze these indications and make decisions. The process of identifying which class a traffic sign belongs to is referred to as Traffic signs classification. Using a convolutional neural network (CNN) and the Keras framework, a model has been developed in this Deep Learning project for the classification of traffic signs in a picture into multiple categories. Traffic sign categorization is the technique of automatically identifying roadside traffic indicators, such as speed limit signs, yield signs, and merge signs. The ability to automatically recognize traffic signs permits the development of "smarter" vehicles.

Self-driving cars require recognition of traffic signs in order to properly interpret and comprehend the roadway. Similarly, "driver alert" systems within vehicles must comprehend the surrounding roads in order to assist and protect drivers. One of the difficulties that computer vision and deep learning can handle is traffic sign recognition.

Library Installation

Multiple libraries have been installed in order to implement numerous functionalities. Out of these libraries, numpy and pandas are directly involved in the data manipulation where cv2, matplotlib, seaborn are associated with the visualization of the dataset. Library such as tensorflow is responsible behind the application of the deep learning model.

Dataset Download

The image dataset contains over 50,000 photographs of various traffic signs (speed limit, crossing, traffic signals, etc.). There are around 43 distinct classes in the dataset for picture categorization. Some classes of the collection have a small number of photos, whilst others contain a large number of images. It comprises two distinct files, train and test, with the train folder including classes and each category containing an assortment of photos.

Labeling Each class

In the table below, all the class name of the training dataset has been presented since the original classes are associated with the number.

Table 1: Class name representing each of the class

Class No	Class Name
0	Speed limit (20km/h)
1	Speed limit (30km/h)
2	Speed limit (50km/h)
3	Speed limit (60km/h)
4	Speed limit (70km/h)

5	Speed limit (80km/h)
6	End of speed limit (80km/h)
7	Speed limit (100km/h)
8	Speed limit (120km/h)
9	No passing
10	No passing veh over 3.5 tons
11	Right-of-way at intersection
12	Priority road
13	Yield
14	Stop
15	No vehicles
16	Veh > 3.5 tons prohibited
17	No entry
18	General caution
19	Dangerous curve left
20	Dangerous curve right
21	Double curve
22	Bumpy road
23	Slippery road
24	Road narrows on the right
25	Road work
26	Traffic signals
27	Pedestrians
28	Children crossing
29	Bicycles crossing
30	Beware of ice/snow
31	Wild animals crossing
32	End speed + passing limits
33	Turn right ahead
34	Turn left ahead
35	Ahead only
36	Go straight or right
37	Go straight or left
38	Keep right
39	Keep left
40	Roundabout mandatory
41	End of no passing
42	End no passing veh > 3.5 tons

Data Distribution

The bar graph below shows the overall data distribution of the training and testing data set. So, it can be seen that all the class from each dataset has unique number of individual files. In overall, the training dataset has around 39209 photos corresponding to the traffic sign where 12630 images are related to testing dataset.

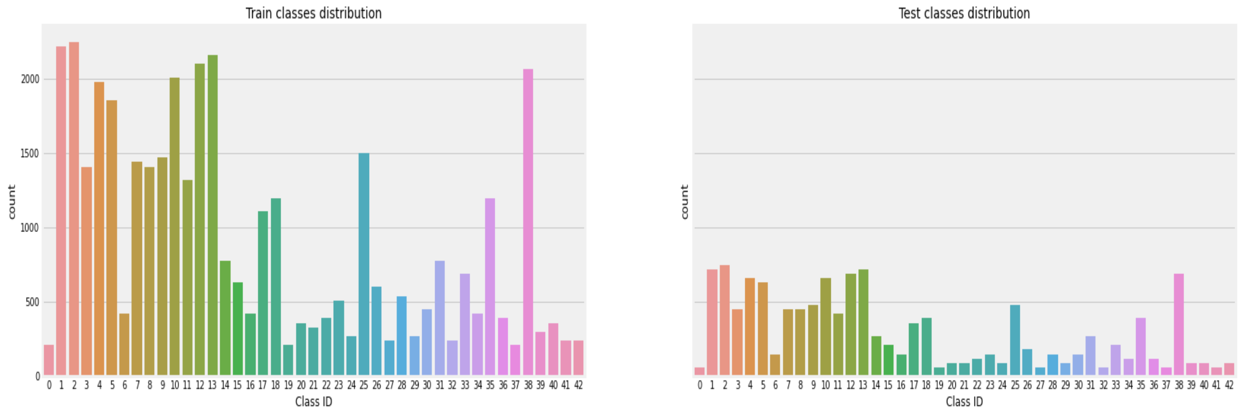

Figure 1: Data distribution of the Train and Test set

Image size distribution

If the original datasets are being observed, it can be noticed that all the images have different dimension. For some the images have higher height or width and some of them have lower height and width. The Multivariate charting below is showing trend of heigh and width. The red line represents the training dataset and blue one represents the testing dataset.

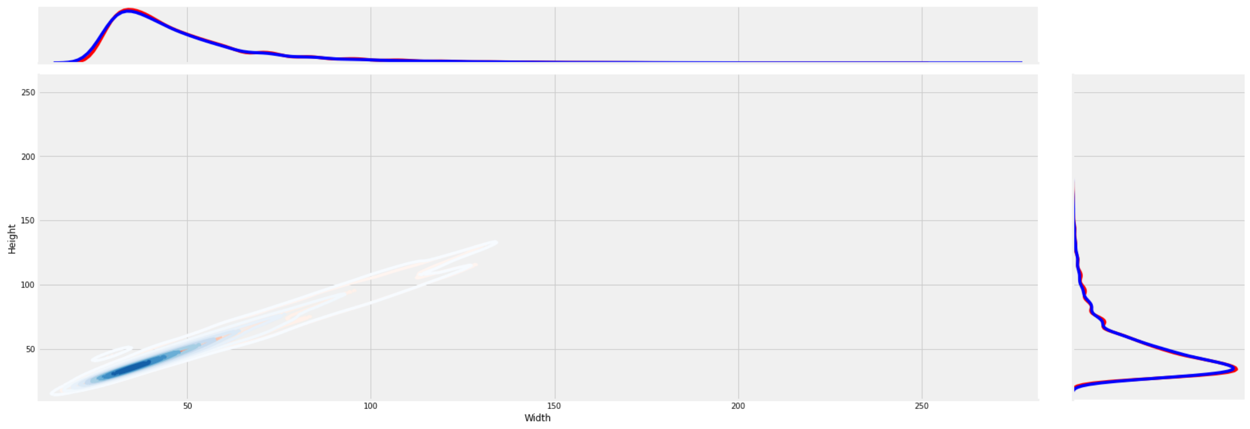

Figure 2: Multivariate charting to illustrate the heigh and width of the datasets.

Target class visualization

The visuals below represent each of the dataset's linked classes. To offer appropriate context, it is vital to illustrate the photographs with their labels. It will assist the reader in comprehending the dataset.

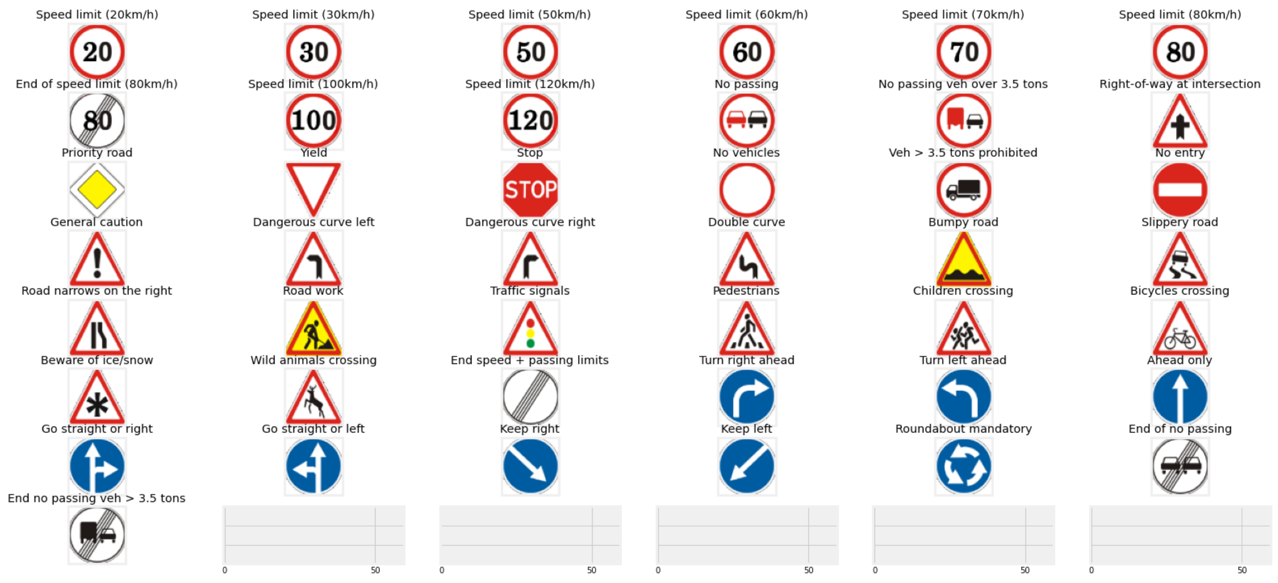

Figure 3: The images associates with each target class.

Visualizing The Dataset

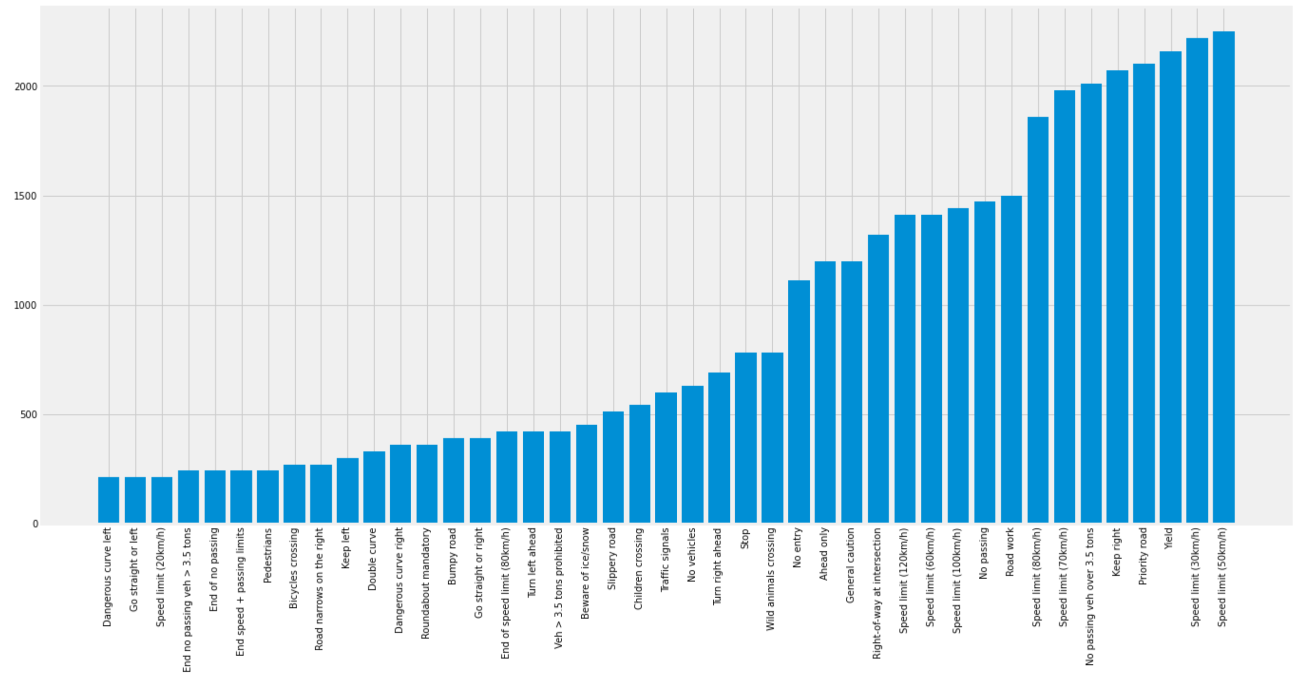

Figure 4: The dataset in ascending order.

Visualizing 25 random images from test data

In this particular section, random images have been presented below to be familiar with the test data. Here, the labels are missing.

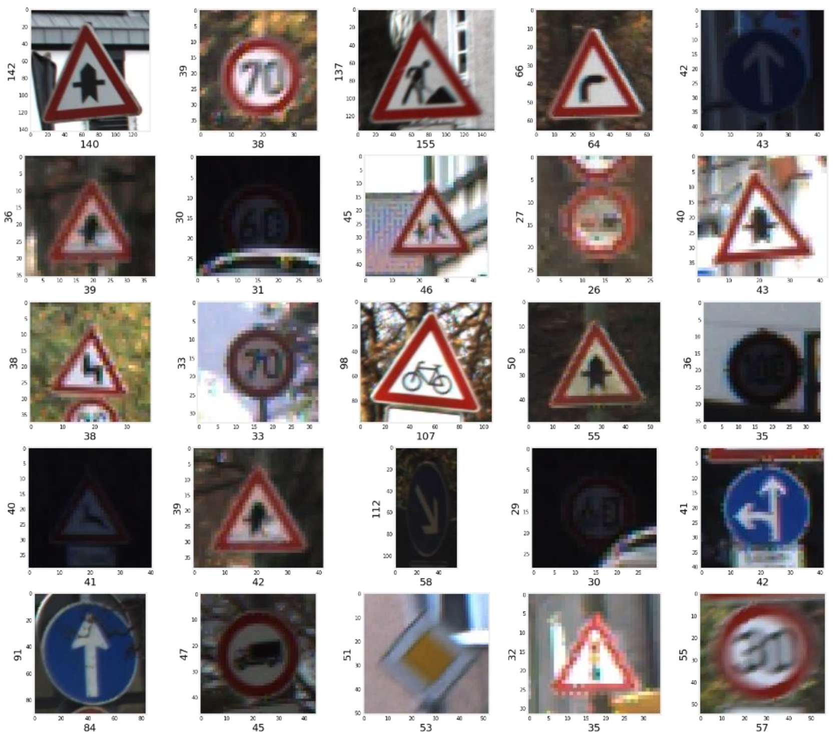

Figure 5: 25 random images of the test data

Visualization of Samples

Moreover, images associated with the training sets are presented below with its own label.

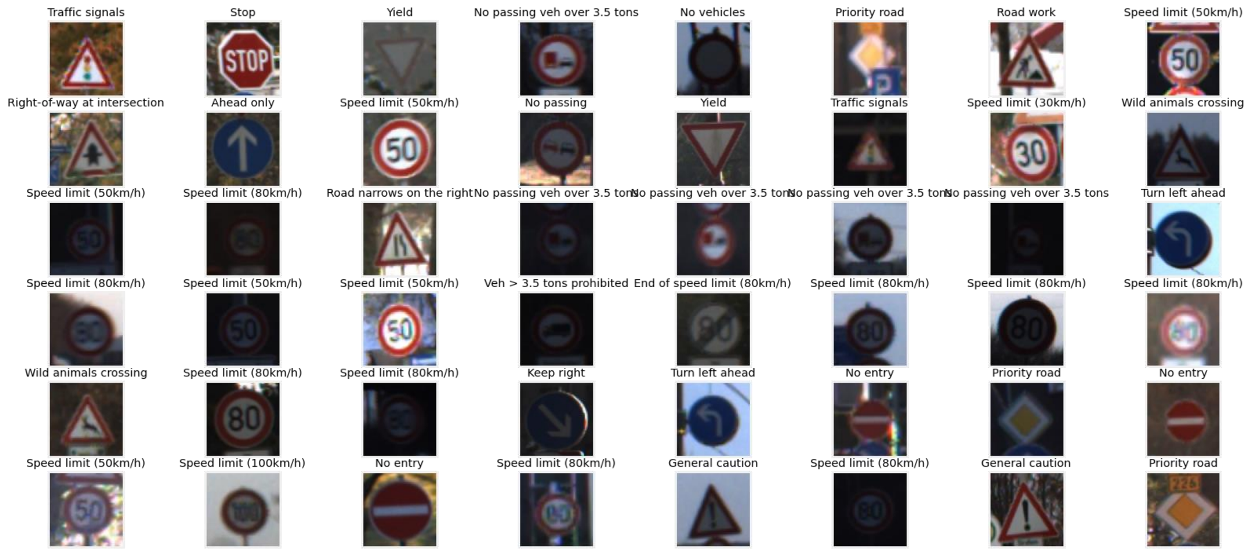

Figure 6: Visualization of the training set

Obtaining Training Data

In this part of the stage, the training dataset has been shuffled so that the randomization can be increased in the model.

Splitting the Dataset

A strategy for measuring the performance of a machine learning algorithm is the train-test split. It can be used for any supervised learning technique and can be utilized for classification or regression tasks. A dataset is divided into two subsets as part of the technique. The training dataset is the first subset, which is used to fit the model. The second subset is not used to train the model; instead, the dataset's input element is given to the model, which then makes predictions and compares them to the predicted values. The test dataset is the name given to the second dataset.

Train Dataset: Dataset for fitting the machine learning model.

Test Dataset: Used to assess how well the machine learning model fits.

The goal is to estimate the machine learning model's performance on new data that was not used to train the model.

Designing the Convolutional Neural Network Architecture

In this stage, a CNN has been designed to implement the recognition system.

Convolution Layer

This layer is a fundamental building piece in the convolution procedure. It uses a convolution technique to identify several visual features using a supplied image. It essentially scans the full pixel grid and computes dot product. A filter or kernel is nothing more than a feature from numerous characteristics that identify in an image input. For instance, in the case of edge detection, a separate filters for curves, blurring, and image sharpening may have exist. As progress makes deeper into the network, more complicated characteristics can be identified.

Pooling Layer

This layer is utilized for the down sampling of features. It diminishes the dimensionality of a huge image while preserving the image's essential characteristics. It reduces the number of computations and weights. Depending on the need, one can select between Maximum or Average pooling. Max pooling extracts the largest value from the feature map, whereas average extracts the average value from all pixels.

Activation Function

This layer adds nonlinear characteristics to the network. It aids in determining which data should be further processed and which should not. The input signal to the activation function is the weighted sum of the input signals, yielding one output signal.

Without activation, the output signal from the function would be a linear function with limited sophisticated learning capabilities. The activation function types include Sigmoid, Tan H, ReLU, Identity, and Binary Step. Sigmoid function is commonly employed in backpropagation; its range is 0 to 1, whereas TanH's range is -1 to 0. Sigmoid function optimization is straightforward. ReLU is a widely used activation function with a range of 0 to infinity.

Flattening Layer

The output of the pooling layer is a 3D feature map, whereas the data must be sent to the fully connected layer as a 1D feature map. This layer turns a 3x3 matrix into a one-dimensional list.

Fully connected Layer

This is the layer where classification occurs. It reaches a classification conclusion based on the end result of convolution or polling layer by layer. Weights connect each input to its corresponding output. It integrates the features into additional properties that more precisely forecast the classes.

For building, sequential model has been used from keras library. Then the layers are being added to make convolutional neural network. In the first 2 Conv2D layers 16 and 32 filter are being used and the kernel size is (3,3).

In the MaxPool2D layer the pool size (2,2) has been kept which means it will select the maximum value of every 2 x 2 area of the image. By doing this dimensions of the image will reduce by factor of 2. In dropout layer, dropout rate = 0.25 have been kept that means 25% of neurons are removed randomly. The batch normalization layer has been added to solve issue such as internal covariate shift.

These 4 layers again with some change in parameters. Then flatten layer has been applied to convert 2-D data to 1-D vector. This layer is followed by dense layer, dropout layer and dense layer again. The last dense layer outputs 43 nodes as the traffic signs are divided into 43 categories in our dataset. This layer uses the softmax activation function which gives probability value and predicts which of the 43 options has the highest probability.

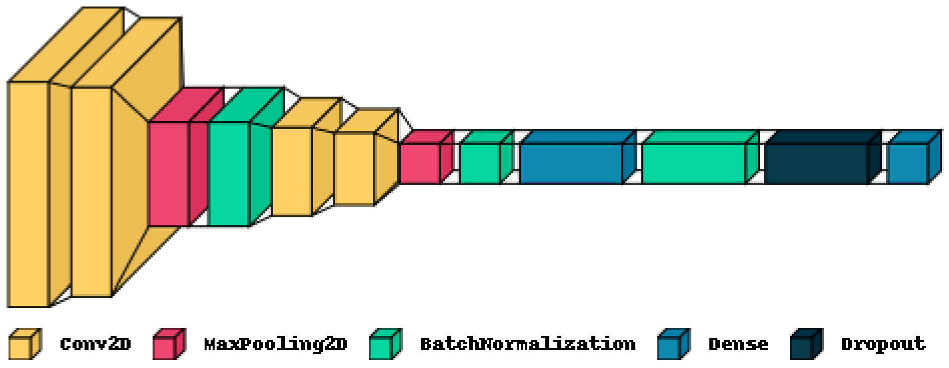

Figure 7: Visualization of the layers.

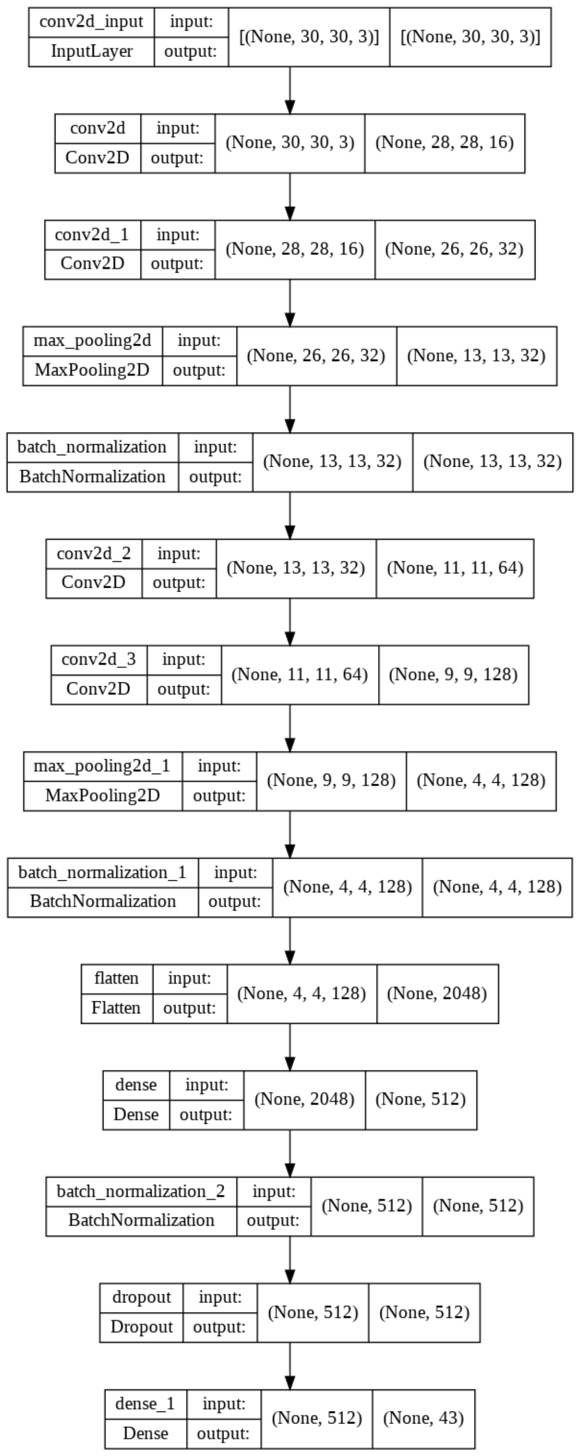
conv2d_input | InputLayer | input: [(None, 30, 30, 3)] | output: [(None, 30, 30, 3)]
conv2d | Conv2D | input: (None, 30, 30, 3) | output: (None, 28, 28, 16)
conv2d_1 | Conv2D | input: (None, 28, 28, 16) | output: (None, 26, 26, 32)
max_pooling2d | MaxPooling2D | input: (None, 26, 26, 32) | output: (None, 13, 13, 32)
batch_normalization | BatchNormalization | input: (None, 13, 13, 32) | output: (None, 13, 13, 32)
conv2d_2 | Conv2D | input: (None, 13, 13, 32) | output: (None, 11, 11, 64)
conv2d_3 | Conv2D | input: (None, 11, 11, 64) | output: (None, 9, 9, 128)
max_pooling2d_1 | MaxPooling2D | input: (None, 9, 9, 128) | output: (None, 4, 4, 128)
batch_normalization_1 | BatchNormalization | input: (None, 4, 4, 128) | output: (None, 4, 4, 128)
flatten | Flatten | input: (None, 4, 4, 128) | output: (None, 2048)
dense | Dense | input: (None, 2048) | output: (None, 512)
batch_normalization_2 | BatchNormalization | input: (None, 512) | output: (None, 512)
dropout | Dropout | input: (None, 512) | output: (None, 512)
dense_1 | Dense | input: (None, 512) | output: (None, 43)

Figure 8: The Keras deep learning framework is used to build a Convolutional Neural Network (CNN) for traffic sign classification.

Increasing the amount of data and training the model

The model is now being trained. The model will be trained using the 'fit()' function with the following parameters: training data (X train), validation data (X val), and the number of epochs.

The number of epochs determines how many times the model cycles over the data. The model will improve over time as we run additional epochs, up to a point. After then, the model will no longer improve with each epoch. The number of epochs in our model will be fixed to 20.

Analyzing the model

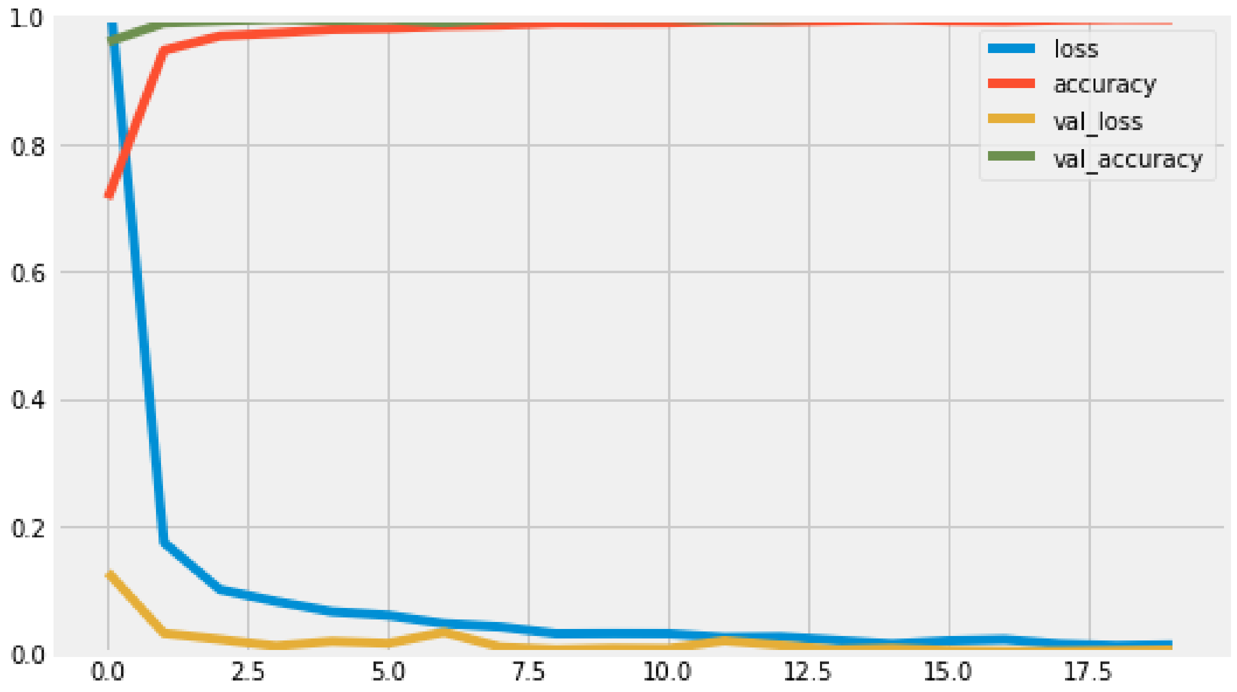

Figure 9: Performance illustration of the model.\

From the line graph, it is obvious that the accuracy rate is almost 100% (Actual value is around 99%) and loss is almost 0% (actual value 1%)

Loading test data and evaluating metrics

In order to compare the considered approaches, the basic metrics applied for the evaluation of classification problems: accuracy (A), precision (P), recall (R) and F1 score.

A = $\frac{TP+TN}{TP+TN+FP+FN}$

P = $\frac{TP}{TP+FP}$

R = $\frac{TP}{TP+FN}$

F1 = $\frac{2*R*P}{R+P}$

Here,

True Positive (TP), False Positive (FP), False Negative (FN) and True Negative (TN).

TP being the number of correct classifications of the traffic sign, FN being erroneous classification of traffic sign, FP being erroneous classification of traffic sign.

Table 2: Metric values

Test Data accuracy	98.15518606492478
Precision	98.2440116267321
Recall	98.15518606492478
F1 score	98.11570579048382

The Confusion Matrix in Visual Form

Given a convolutional neural network (CNN), a confusion matrix shows where the model is getting confused, i.e. which classes the model predicts correctly and which classes the model predicts incorrectly.

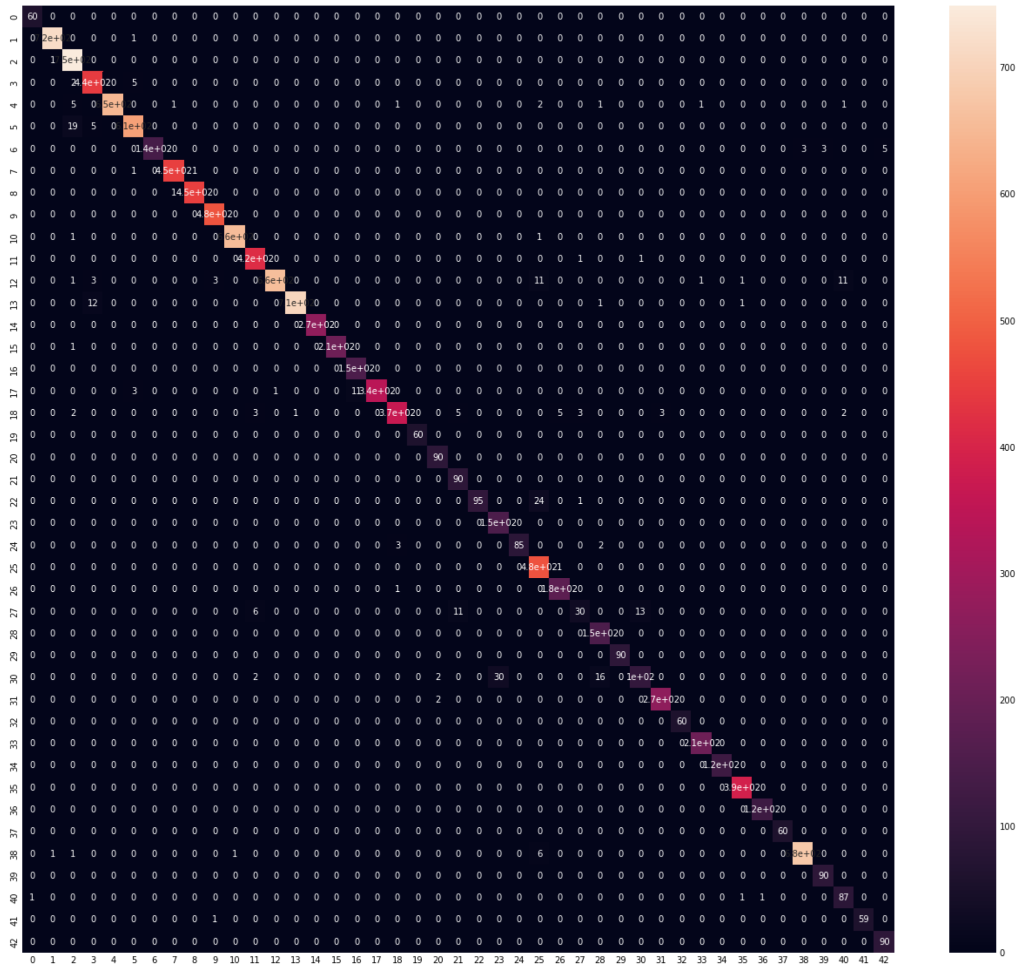

Figure 10: Confusion Matrix of the Model.

Test Data Prediction

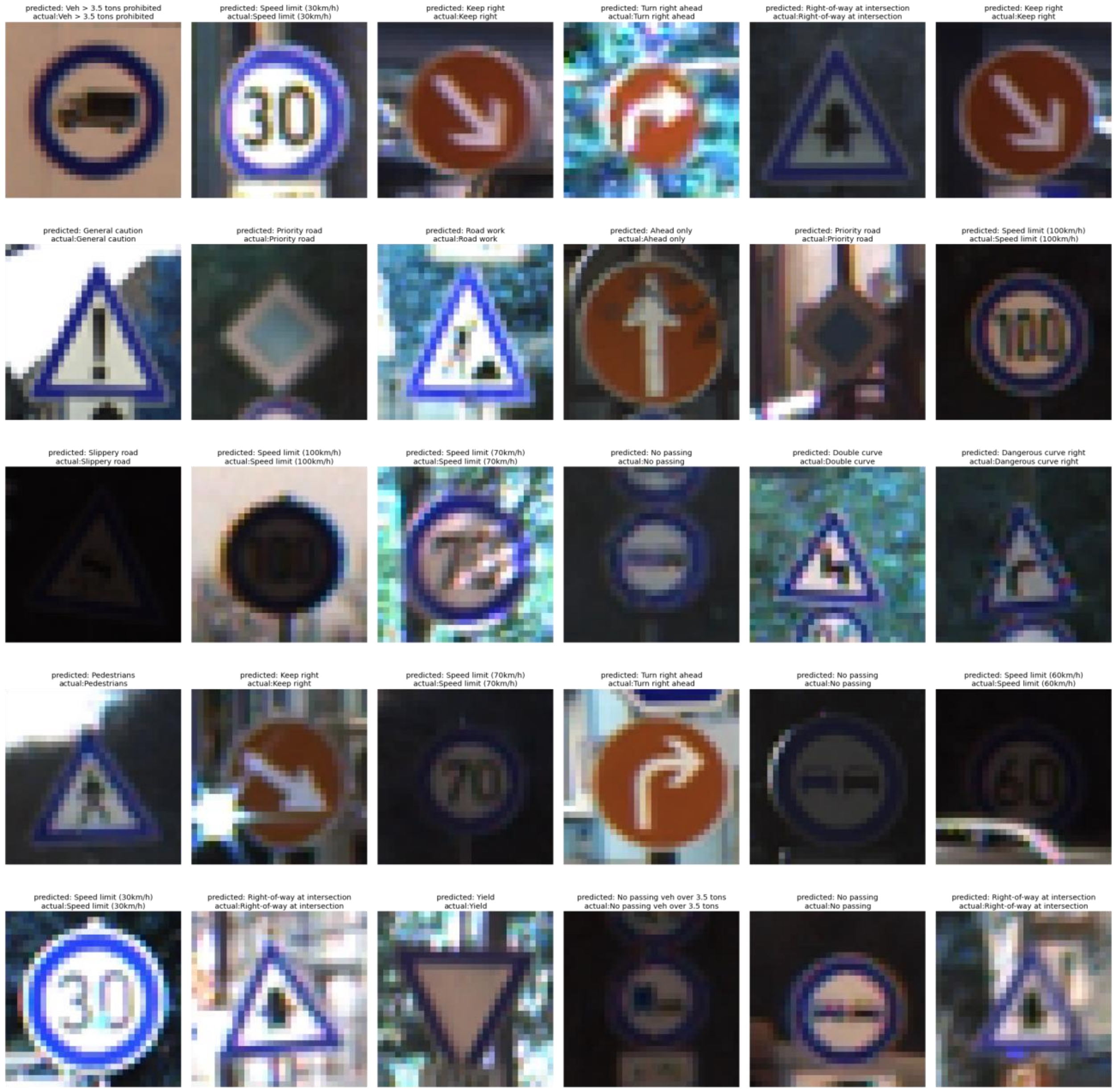
predicted: Veh > 3.5 tons prohibited
actual:Veh > 3.5 tons prohibited
predicted: Speed limit (30km/h)
actual:Speed limit (30km/h)
predicted: Keep right
actual:Keep right
predicted: Turn right ahead
actual:Turn right ahead
predicted: Right-of-way at intersection
actual:Right-of-way at intersection
predicted: Keep right
actual:Keep right
predicted: General caution
actual:General caution
predicted: Priority road
actual:Priority road
predicted: Road work
actual:Road work
predicted: Ahead only
actual:Ahead only
predicted: Priority road
actual:Priority road
predicted: Speed limit (100km/h)
actual:Speed limit (100km/h)
predicted: Slippery road
actual:Slippery road
predicted: Speed limit (100km/h)
actual:Speed limit (100km/h)
predicted: Speed limit (70km/h)
actual:Speed limit (70km/h)
predicted: No passing
actual:No passing
predicted: Double curve
actual:Double curve
predicted: Dangerous curve right
actual:Dangerous curve right
predicted: Pedestrians
actual:Pedestrians
predicted: Keep right
actual:Keep right
predicted: Speed limit (70km/h)
actual:Speed limit (70km/h)
predicted: Turn right ahead
actual:Turn right ahead
predicted: No passing
actual:No passing
predicted: Speed limit (60km/h)
actual:Speed limit (60km/h)
predicted: Speed limit (30km/h)
actual:Speed limit (30km/h)
predicted: Right-of-way at intersection
actual:Right-of-way at intersection
predicted: Yield
actual:Yield
predicted: No passing veh over 3.5 tons
actual:No passing veh over 3.5 tons
predicted: No passing
actual:No passing
predicted: Right-of-way at intersection
actual:Right-of-way at intersection

Figure 11: Prediction results of the CNN model.

Conclusion

CNN is good at recognition, and accuracy and recognition rate may be enhanced through hyper parameter adjustment. As a consequence, CNN has been employed for traffic sign identification in the suggested scheme to create a warning traffic sign detection system for drivers. In this project, a CNN model has been designed to identify traffic signs and classify them with 99% accuracy.