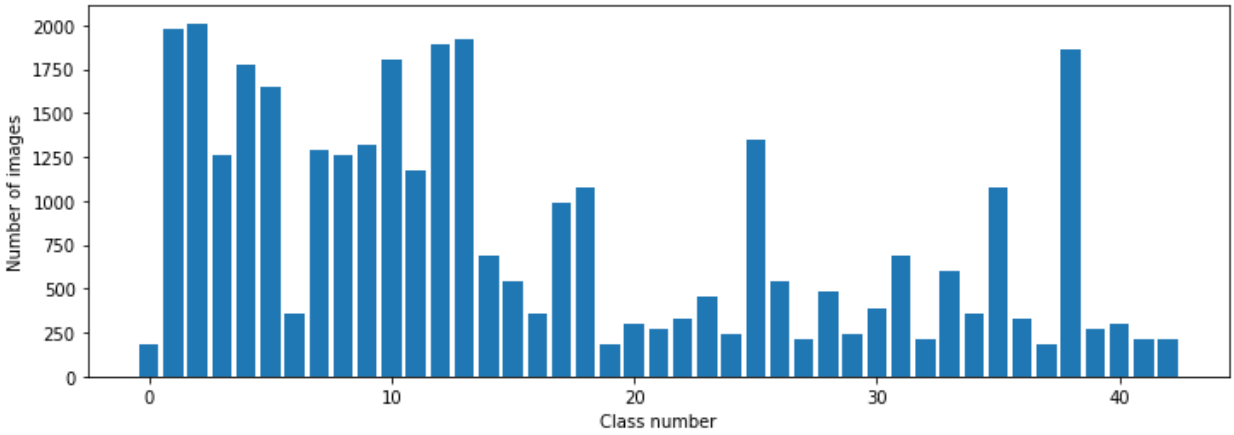
Ayanangshu Das
Self Driving Car ND-Term 1 P2
Mar 27, 2017

Report- Traffic Sign Classifier

Exploring the data set

Exploring the data set provided the following insights:

1. Number of images in each classes of the training set is skewed



2. Some of the images have different orientation-some are rotated

Inference: Images need to be augmented to gain accuracy in validation and test set

Processing the training set images

1. Used Gaussian blur to add blurry vision to the images
2. Added sharpened version
3. Added rotated version

This resulted in

1. The augmentation step added more variation to the training data set. This is more likely to perform better on unseen examples

2. The skewness in data could be resolved as some of the classes which didn't have that many images are now having more images

The Model Architecture

My model, Ayan_Net is a LeNet architecture with the following changes

Layer 1. This is a convolutional layer with output 26x26x32. Then a relu activation is applied to it. Finally 2x2 maxpooling is applied to generate the output tensors. The purpose of this step is to identify edges or elementary properties in the traffic signs

Layer 2. This is the second convolutional layer with o/p 26x26x256 with a 2x2 maxpool and relu layer in-between

Layer 3. This is a convolutional layer. Maxpooling outputs a tensor 6x6x256 which is flattened at the end of this layer

Layer 4 and 5: These are a fully connected layer. Layer 5 is the output layer w/ 43 length tensor which is the number of classes for the current data set.

The first convolutional layer identifies elementary properties of the training images and subsequent layers tries to form complex structures/functions using these elementary properties to classify training dataset.

My experiment shows that deeper first convolutional layer is better for traffic signal classification than LeNet architecture as it has many more involved elementary properties.

Model Training

I used a batch size of 128. I tried with a different epoch values between 10 and 100 at different stages of my experimentation. I tried with different learning rates. I used AdamOptimizer with base learning rate of .0005. I tried with learning rate of .0003 but have stick to .0005 for the last few iterations.

Solution Approach

I used LeNet architecture with image augmentations. I initially played with the learning rate and could achieve only 75% accuracy on the validation set. Later I worked on the architecure making convolutional layer tensors deep and adding one more fully connected layer increased the accuracy to 91%. I tried varying the dropout which did not help. Finally I added one more convolutional layer which gave the final push and I could achieve the following:

Accuracy on training set : 98.8%

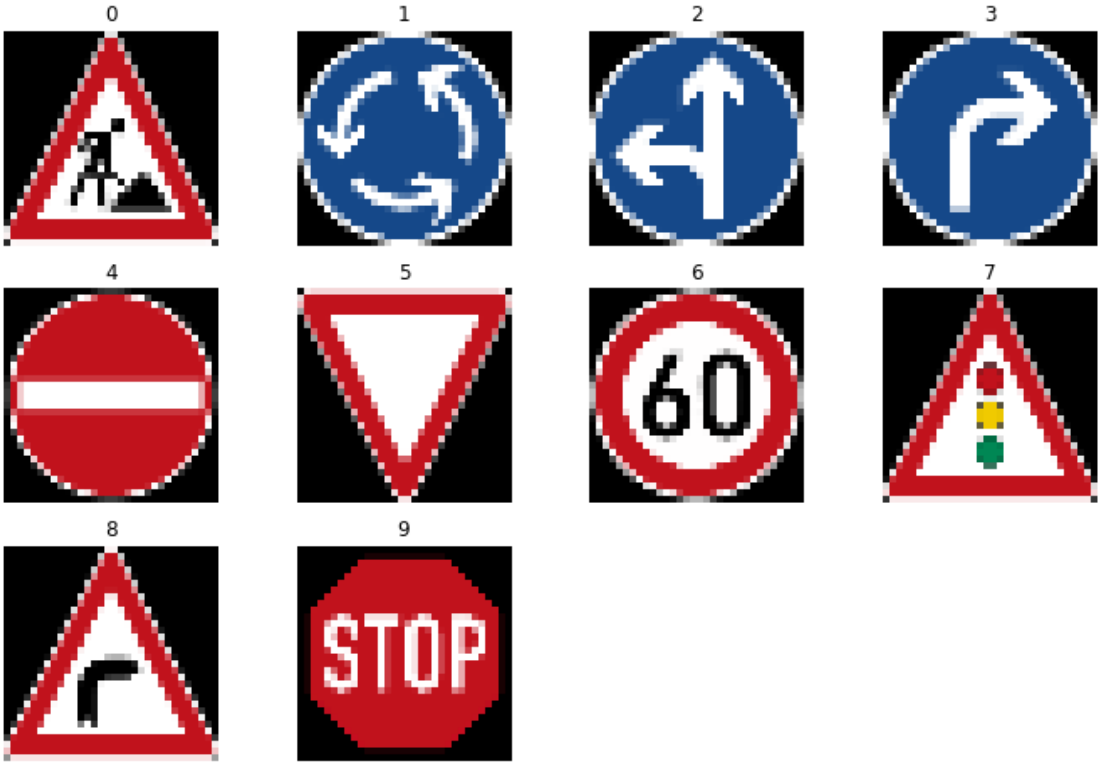
Accuracy on validation set: 95.5%

Accuracy on test set: 95%

Testing on new images

I downloaded two sets of images from internet. One set had 10 images and the other set had 12 images. I had to resize and augment them as the image size was large and crisp.

Set 1:

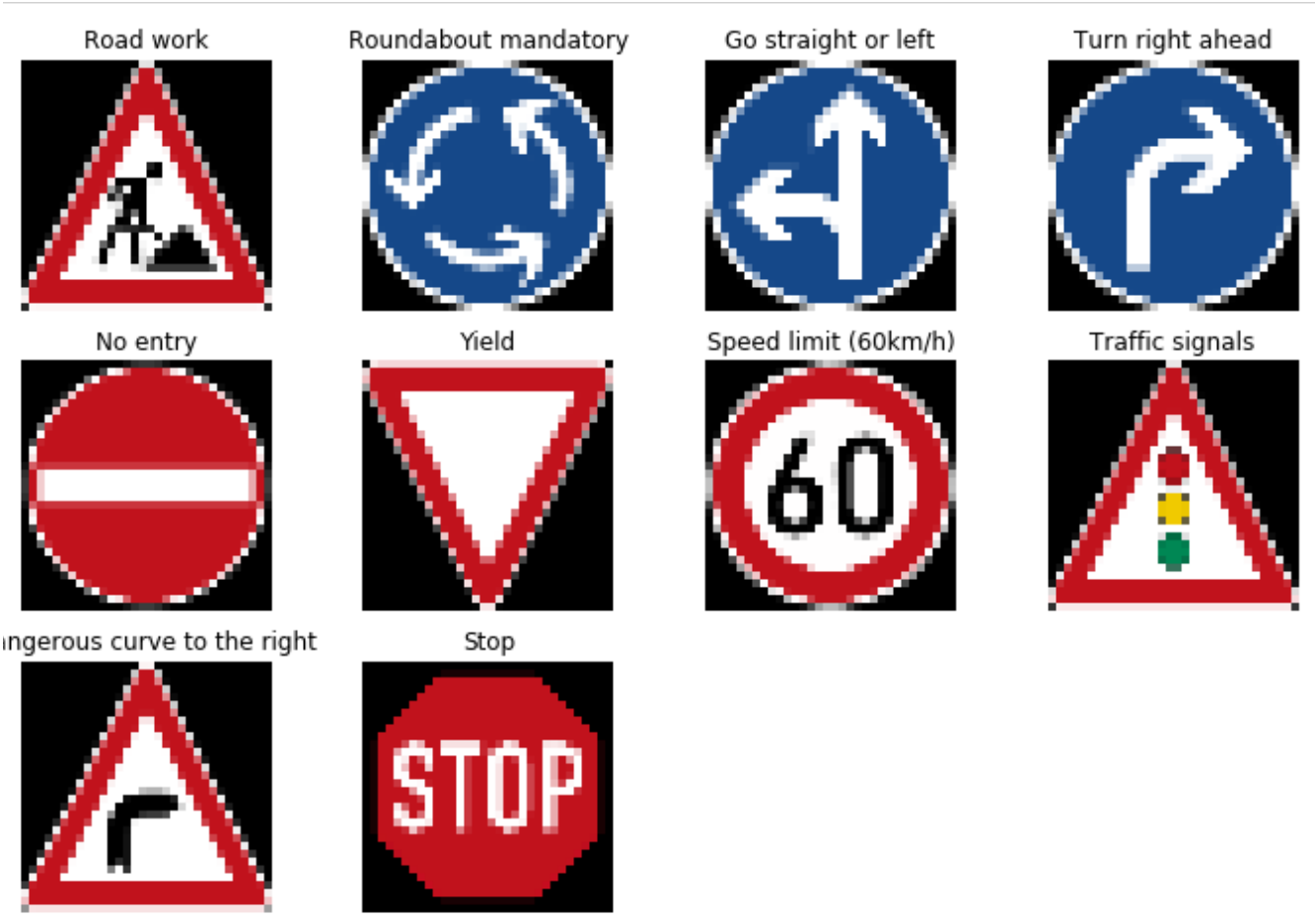


Set 2:


0
1
2
3
60
4
5
6
7
40
80
8
9
10
11
20

Performance of model on new images

The model did predict with an accuracy of 95.04% on the set 1.



The model could not predict as good in set 2 with accuracy is 85%

Keep right

Speed limit (60km/h)

Yield

Go straight or left

Priority road

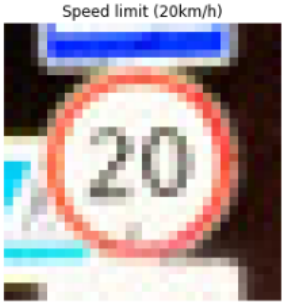
Speed limit (30km/h)

Go straight or right

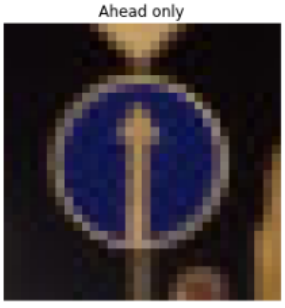
Speed limit (80km/h)

Speed limit (70km/h)

Speed limit (20km/h)



Ahead only



No entry

It missed the 40 Km/h speed and predicted 30 Km/h. It predicted the No U-turn wrongly. “No vehicles” presented difficulties and was misclassified. It seems to me that due to the scarcity of the training data for these classes, it might have predicted wrongly.

The classifier had high accuracy on the test set(95%), but 85% on the new captured images. The result is good but more can be achieved with genralization. The underfitting of the classifier on the new captured images indicates that it is overfitting the training set. We are able to make better predictions on the test set as they have some similarities with the training set.

Some possible way to overcome this is to have more variation in the training data.

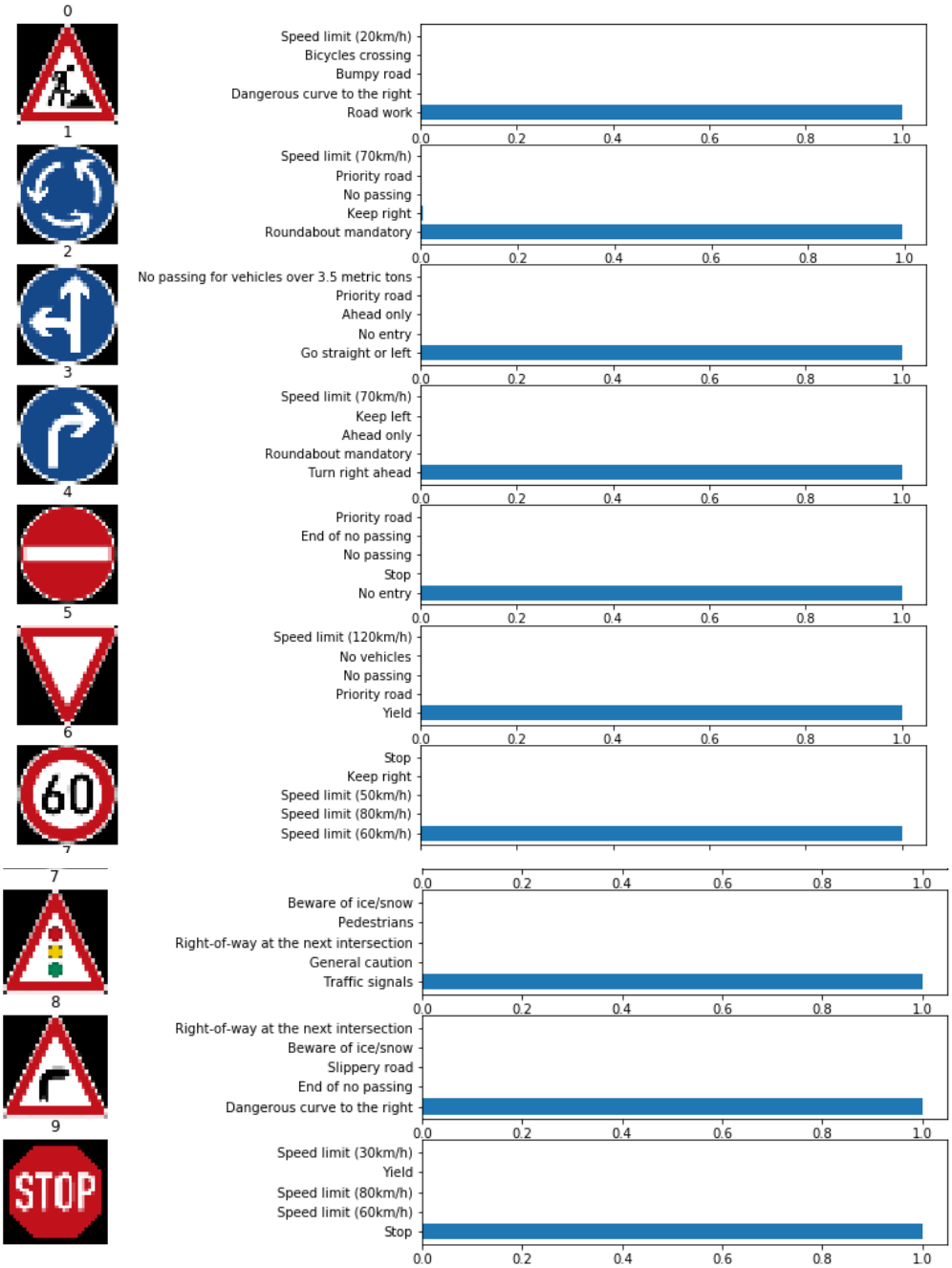
Softmax probabilities

```
TopKV2(values=array([[  1.00000000e+00,   0.00000000e+00,   0.00000000e+00,
          0.00000000e+00,   0.00000000e+00],
       [  9.70188081e-01,   2.97815017e-02,   1.93184169e-05,
          1.10675855e-05,   2.15976765e-08],
       [  1.00000000e+00,   3.31753158e-33,   7.00228783e-35,
          2.76848399e-36,   2.34439677e-37],
       [  1.00000000e+00,   7.99264277e-23,   8.80474388e-27,
          1.40850422e-32,   5.36812638e-34],
       [  1.00000000e+00,   1.30143483e-27,   4.39246216e-28,
          6.54603417e-32,   1.36136234e-32],
       [  9.72619295e-01,   2.68841479e-02,   2.34401334e-04,
          1.42706529e-04,   5.26764379e-05],
       [  9.79891837e-01,   1.72549859e-02,   1.87750626e-03,
          5.13888954e-04,   2.72938691e-04],
       [  1.00000000e+00,   3.54273472e-10,   1.88285761e-13,
          2.29508950e-16,   3.82690081e-17],
       [  5.65974236e-01,   4.30498242e-01,   2.74279970e-03,
          3.32261348e-04,   2.73447338e-04],
       [  8.08735013e-01,   1.10391110e-01,   6.92010000e-02,
          1.09491665e-02,   6.67046348e-04],
       [  6.10301554e-01,   3.89698505e-01,   1.14199743e-08,
          5.80037060e-12,   1.39694040e-12],
       [  9.99999404e-01,   6.16461307e-07,   2.18353100e-13,
          8.85088973e-16,   3.29328607e-16]], dtype=float32),
indices=array([[38,  0,  1,  2,  3],
       [ 3,  5,  2, 10,  9],
       [13, 12,  8, 15,  2],
       [37, 39, 35, 40, 36],
       [12, 15, 13, 18, 38],
       [ 1, 38, 18,  5, 12],
       [36, 35, 38, 18, 40],
       [ 5,  3, 10,  7,  6],
       [ 4, 15,  8, 13,  7],
       [ 0,  4,  1,  5,  9],
       [35, 36, 34, 38,  3],
```

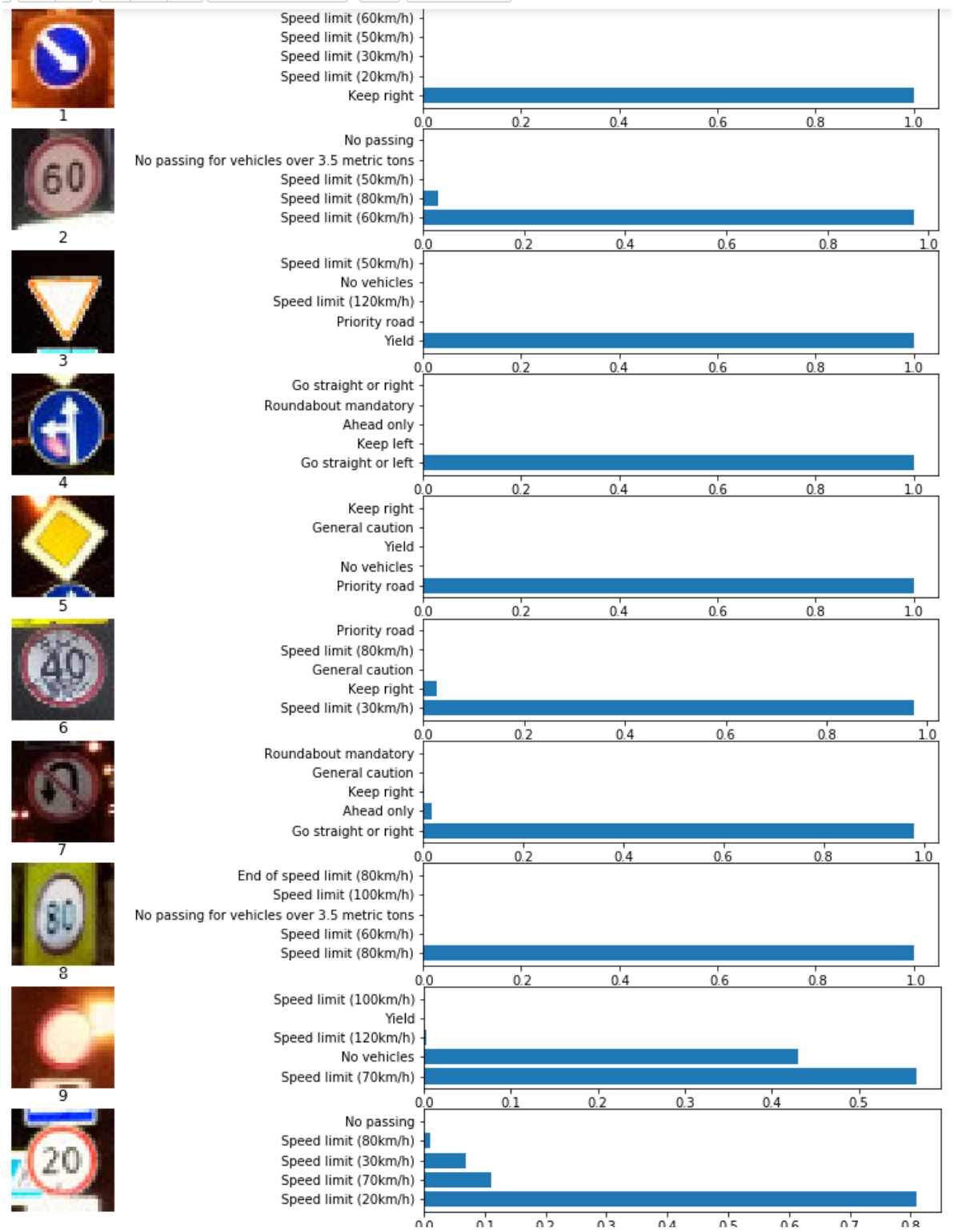
```
[17, 14,  9, 38, 41]], dtype=int32))
```

The softmax probabilites shows the model is 100% certain on its predictions. I think I need to do some more analysis to figure out why some of the predictions failed.

Set 1:



Set 2:


Speed limit (60km/h)
Speed limit (50km/h)
Speed limit (30km/h)
Speed limit (20km/h)
Keep right
1
No passing
No passing for vehicles over 3.5 metric tons
Speed limit (50km/h)
Speed limit (80km/h)
Speed limit (60km/h)
2
Speed limit (50km/h)
No vehicles
Speed limit (120km/h)
Priority road
Yield
3
Go straight or right
Roundabout mandatory
Ahead only
Keep left
Go straight or left
4
Keep right
General caution
Yield
No vehicles
Priority road
5
Priority road
Speed limit (80km/h)
General caution
Keep right
Speed limit (30km/h)
6
Roundabout mandatory
General caution
Keep right
Ahead only
Go straight or right
7
End of speed limit (80km/h)
Speed limit (100km/h)
No passing for vehicles over 3.5 metric tons
Speed limit (60km/h)
Speed limit (80km/h)
8
Speed limit (100km/h)
Yield
Speed limit (120km/h)
No vehicles
Speed limit (70km/h)
9
No passing
Speed limit (80km/h)
Speed limit (30km/h)
Speed limit (70km/h)
Speed limit (20km/h)