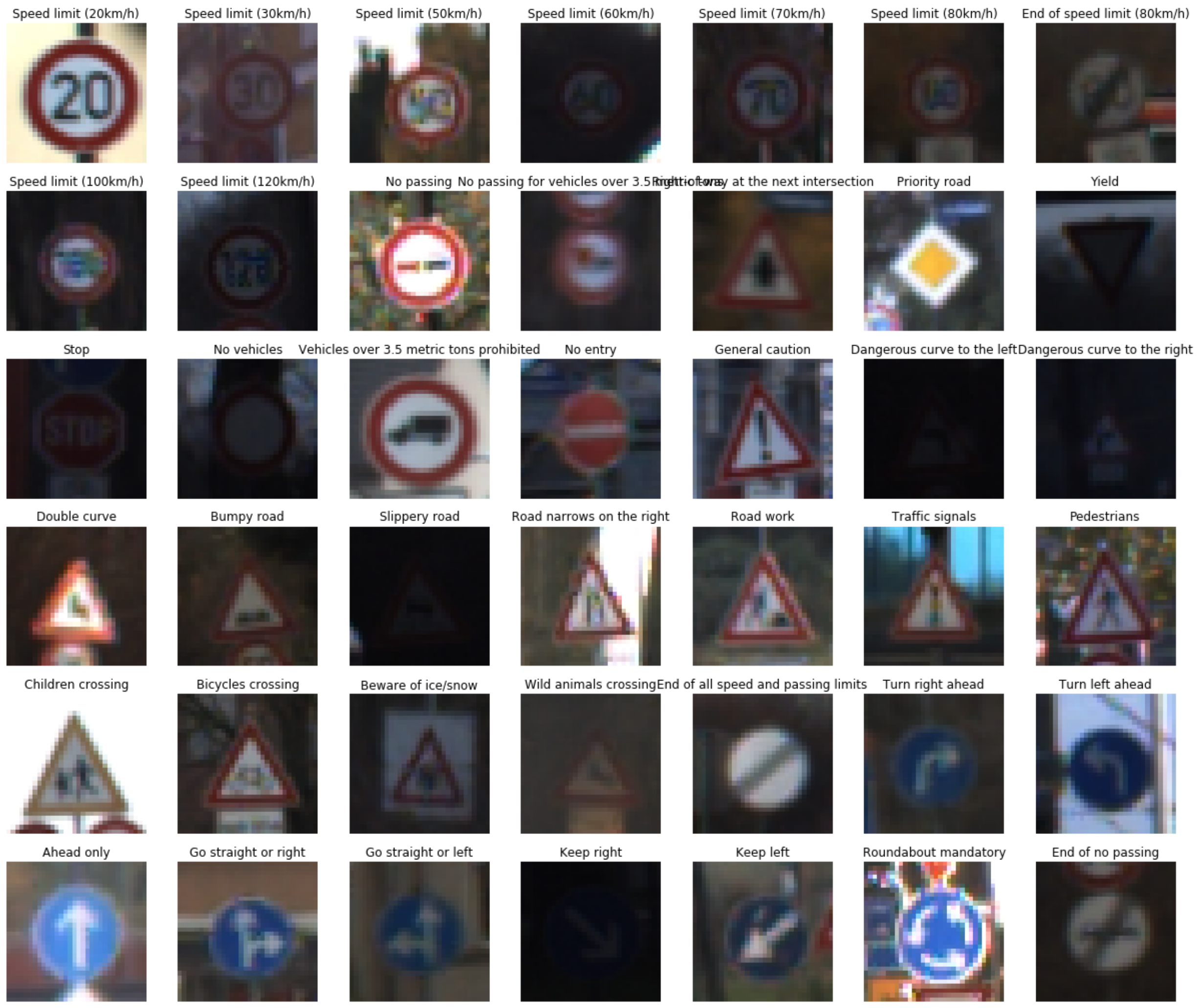
Dataset summary

The data set has 34799 training examples, 4410 validation examples, and 12630 testing examples. Each image is of shape (32, 32, 3), RGB colored. There are 43 classes in total.

Exploratory Visualization

Here is a traffic sign board with a random image selected for each different class in the dataset:



The class labels are not evenly distributed. The top ones have close to 2K samples per class.

	label_enum	label	count
0	2	Speed limit (50km/h)	2010
1	1	Speed limit (30km/h)	1980
2	13	Yield	1920
3	12	Priority road	1890
4	38	Keep right	1860
5	10	No passing for vehicles over 3.5 metric tons	1800

The bottom ones have only ~200 per class.

38	32	End of all speed and passing limits	210
39	27	Pedestrians	210
40	37	Go straight or left	180
41	19	Dangerous curve to the left	180
42	0	Speed limit (20km/h)	180

Preprocessing

To preprocess, first, I converted all images to grayscale because the 3 color channels triples the number of parameters but don’t necessarily give enough extra information. Then, I normalized by subtracting all pixels by 128 and dividing them by 128, such that the maximum and minimum value of the normalized input are 1 and -1 respectively. By doing so, the neural network does not need to deal with large inputs and hence possibly iterates with very small multiplicative weights.

Model

1st Approach: LeNet (91.8%)

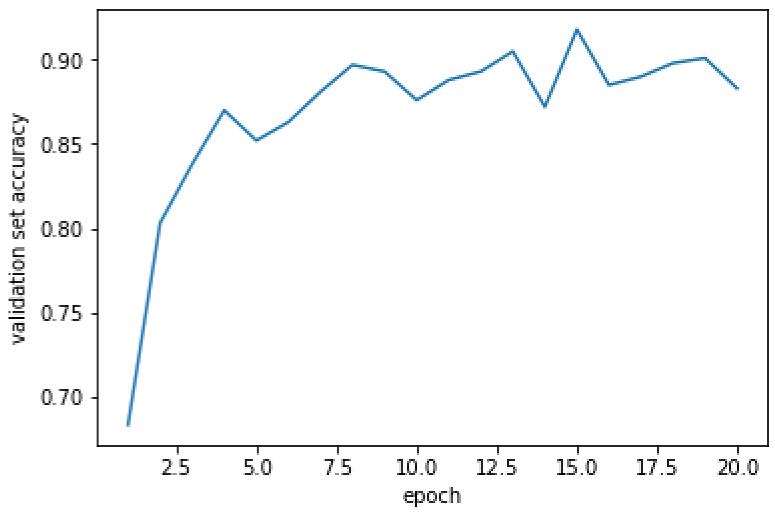
I started off by directly applying LeNet.

In the LeNet architecture, the 32x 32 images pass through a convolutional layer with 6 5x5 filters. The output is 28x28x6. After max pooling, the output dimension is 14x14x6. In a similar manner, it goes through a second convolutional iteration with 16 filters. The output after max-pooling is 5x5x16. We flatten this output to 400. Then it undergoes several fully connected layer, condensing to 120 neurons, then 84 neurons, and lastly 43 neurons with one corresponding to each output class.

It goes through 10 epochs of training, with a learning rate of 0.001. It uses the AdamOptimizer.

This gets an accuracy of ~89%.

From reading the training log, the validation set still kept increasing at epoch 10. This led me to believe that the model is underfitting. Therefore, I extended to 20 epochs. This leads to an accuracy of 91.8%. As seen from the plot below, the training accuracy is saturating at epoch 20.



To gain more insights, I obtained the precision and recall breakdown by each class. Some of the classes did particularly bad:

0 Speed limit (20km/h) - precision: 0.4 recall 0.07
24 Road narrows on the right - precision: 0.86 recall 0.2
27 Pedestrians - precision: 1.0 recall 0.5
32 End of all speed and passing limits - precision: 0.35 recall 0.27

Most of these classes have relatively few training data.

2nd approach: dropout (94.7%)

I applied the dropout technique on the two fully connected layers, with a drop out rate of 0.5. This technique is highly effective: it approves the accuracy from 91.8% to 94.7%.

In particular, it drastically improved the precision and recall of the classes that did the worst in the previous approach:

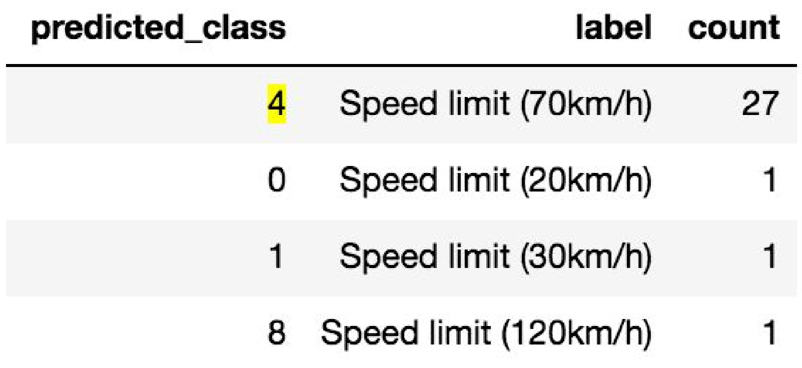
26 Traffic signals
BEFORE - precision: 0.68 recall 0.53
AFTER - precision: 0.79 recall 0.97

32 End of all speed and passing limits
BEFORE - precision: 0.35 recall 0.27
AFTER - precision: 0.9 recall 0.9

However, even with dropout, the model is still doing terribly in the following classes:

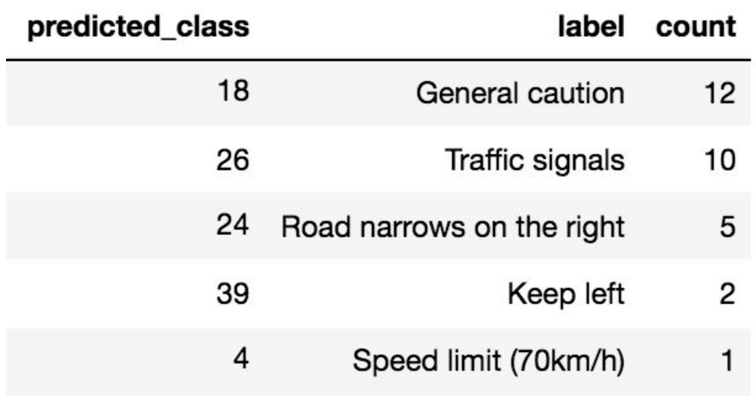
0 Speed limit (20km/h) - precision: 0.5 recall 0.03
24 Road narrows on the right - precision: 0.83 recall 0.17

Let's also take a look at what wrong labels the model predicts these classes to be. The "Speed limit (20 km/h)" is most likely to be predicted as 70 km/h. The two numbers look similar in shape, and the latter class has 10x more training data.



predicted_class	label	count
4	Speed limit (70km/h)	27
0	Speed limit (20km/h)	1
1	Speed limit (30km/h)	1
8	Speed limit (120km/h)	1

The "road narrow to the right" sign is most likely to be predicted as "General caution", followed by traffic signals.



predicted_class	label	count
18	General caution	12
26	Traffic signals	10
24	Road narrows on the right	5
39	Keep left	2
4	Speed limit (70km/h)	1

The two do look similar:

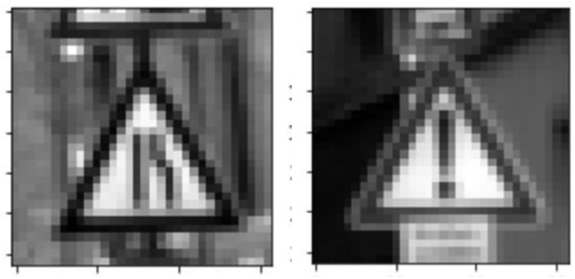


Image: left: “Road narrows on the right”, right: “general caution”.

It seems like the network is capturing the contour of traffic signs (triangular signs vs. circular signs) and rough shape of what’s on the signs well, but it is missing out some details.

3rd approach: bigger neural network (96.9%)

Given that the model is missing out some details, I decided to increase the number of filters. Separately, with drop out, the last hidden layer now is effectively 42 in size, and it connects to 43 classes. It worked for MINST dataset which predicts 10 classes, but does not quite make sense here. Thus, I decided to increase the network size for the fully the connected layers as well.

In the first convolutional layer, I increased the filter from 6 to 24. In the second consolutionary layer, I increased the filters to 72. The flattened output thus is 5x5x72 = 1800. Going through 3 fully connected layers, it condensed down to 900, 300, and eventually 43 neurons.

Now the accuracy improves to 96.9%.

In particular, it improves on the two classes it did worse at in the previous run:

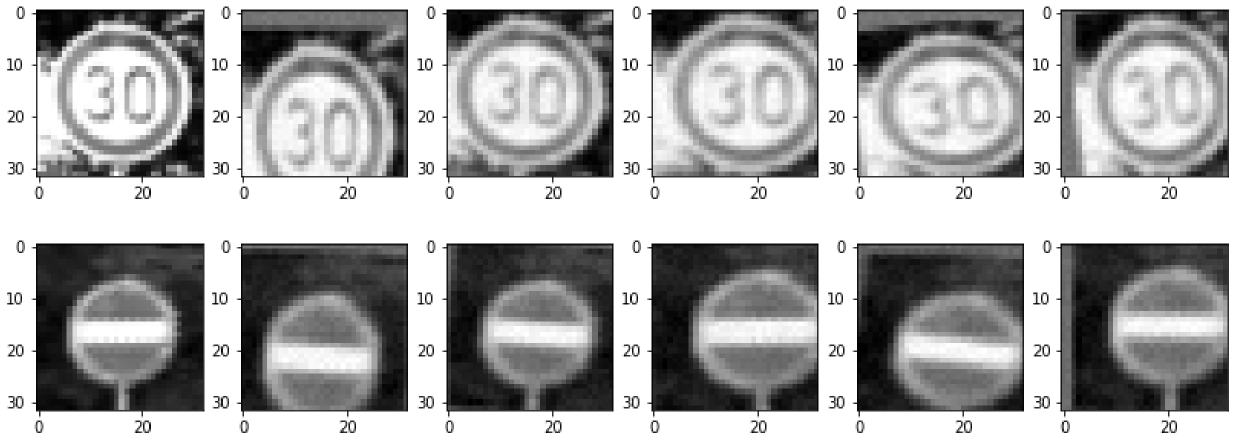
0 Speed limit (20km/h)
BEFORE - precision: 0.5 recall 0.03
AFTER - precision: 1.0 recall 0.8

24 Road narrows on the right
BEFORE - precision: 0.83 recall 0.17
AFTER - precision: 0.95 recall 0.63

4th approach: augment dataset (96-97%)

The performances on these small classes are still lackluster. Therefore I decided to augment the data set for these small classes. I did this by randomly rotating an angle between -10 and 10 degree, then randomly translating the x and y direction between 0 and 3 pixels, randomly zooming in on the images between 1x and 1.3x, and lastly adding some noise.

Shown below, the leftmost image in each row is the original image from the data set. The subsequent 5 images are the generated samples.



This increases the data set by 6000 data points. The accuracy is 97.0%. It is questionable whether it actually improved from the last run whose accuracy is 96.9%.

As a next step, now that I have a balanced dataset, I augmented the data for each sample in this fashion, and increased the dataset size in total by 3 times. The accuracy still didn’t improve (highest during training is 96.9% at 15th epoch.)

The final test accuracy is 94.9%.

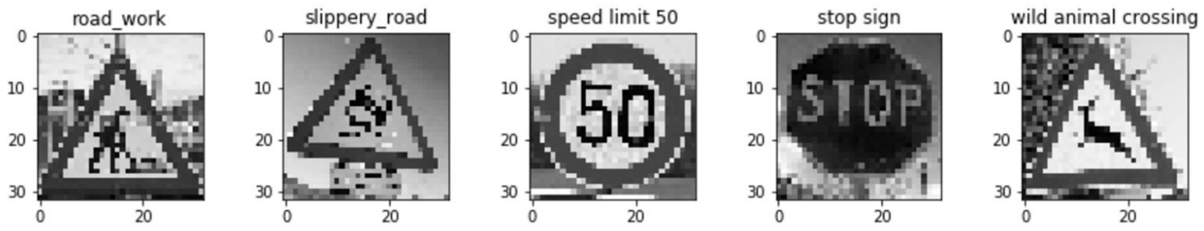
Testing model on new images

Acquiring new images

I downloaded 5 images from the web.

Here are the original images:

After centering, cropping, and preprocessing, here are the images fed into the network:

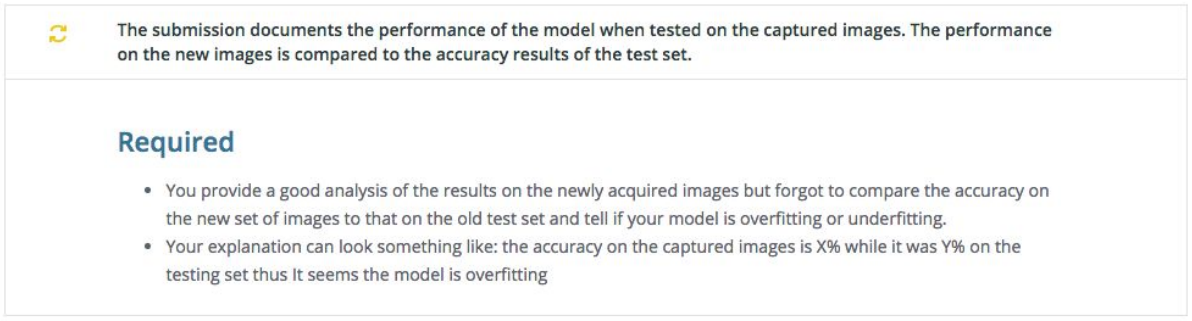


The slippery road image might be difficult to predict, because after resizing down to 32 x 32, much of the detail is lost and it is even hard for a human to tell. In addition, the model's precision on the "slippery road" class is 0.92, which is on the lower end.

Performance on new images

The neural net predicts all of them correctly except for the slippery road, which it predicted as wild animal crossing. The prediction accuracy is 80% -- though I wouldn't read too much into this number given the small sample size.

The first round of review gave me the following feedback:


The submission documents the performance of the model when tested on the captured images. The performance on the new images is compared to the accuracy results of the test set.

Required

- You provide a good analysis of the results on the newly acquired images but forgot to compare the accuracy on the new set of images to that on the old test set and tell if your model is overfitting or underfitting.
- Your explanation can look something like: the accuracy on the captured images is X% while it was Y% on the testing set thus It seems the model is overfitting

For completion purpose, I am here comparing the web images prediction accuracy of 80% to the test set accuracy of 95% and validation set accuracy of 97%: the web image prediction accuracy is lower. However, I reiterate my point above that such comparison is not too meaningful, as there are only 5 data points in the web image samples; the sample size is so tiny, and the variance would be huge (either 80% or 100% but nothing in between). Hence it is premature to draw the conclusion purely based on this that it is over- or under- fitting. Testing on 5 web downloaded images in my opinion is mostly for fun, and the result should not be read into seriously unless we plan to build up a big dataset from web images. To truly understand over- or under- fitting, we should compare the validation result against the test set result, which the model has never seen before and I only used it once in the very end of the training. Based on that, the model is slightly overfitting.

Model Certainty

Below are the top 5 softmax probabilities for each image.

First image (road work):

	label	probability
0	Road work	1.000000e+00
1	Beware of ice/snow	3.515332e-14
2	Dangerous curve to the right	8.044525e-22
3	Right-of-way at the next intersection	3.281440e-22
4	Road narrows on the right	8.412306e-23

Second image (slippery road):

	label	probability
0	Right-of-way at the next intersection	0.991443
1	Road work	0.007448
2	Beware of ice/snow	0.000941
3	Double curve	0.000144
4	Slippery road	0.000018

Unfortunately slipper road is only the 5th choice with a small probability. Comparing to the slippery road image from the training set, it appears that the sign is slightly different: the ones in the training set has the car more in the center and occupying a bigger area of the sign.

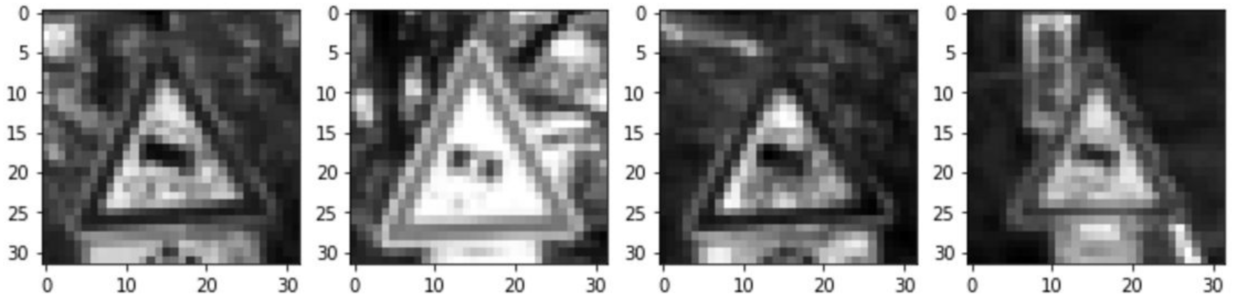


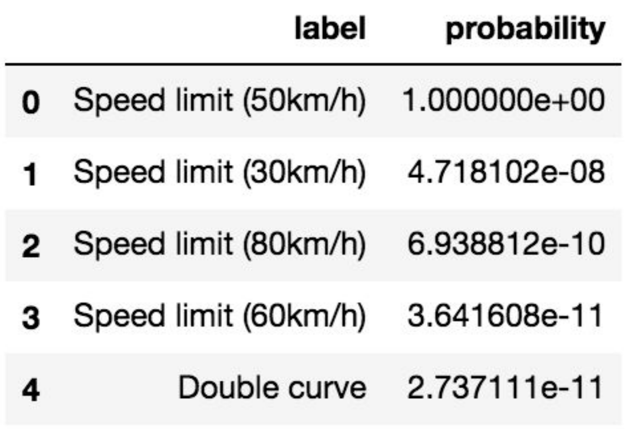
Image: Slippery road sign from training set

Third image (50 Km/ h speed limit)

	label	probability
0	Speed limit (50km/h)	1.000000e+00
1	Speed limit (30km/h)	4.718102e-08
2	Speed limit (80km/h)	6.938812e-10
3	Speed limit (60km/h)	3.641608e-11
4	Double curve	2.737111e-11

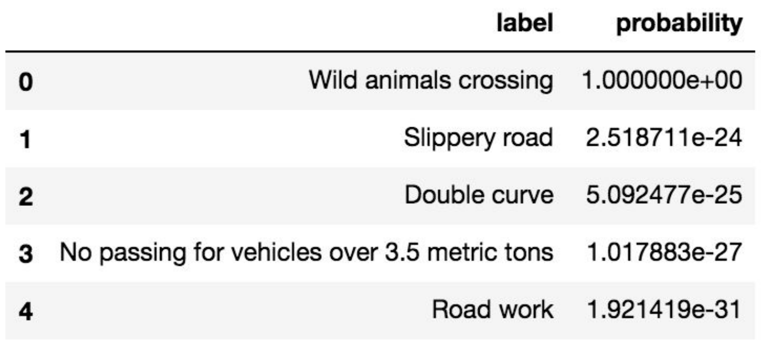
The other classes predicted with high probabilities are expectedly the other speed limit signs.

Fourth image (stop sign):



	label	probability
0	Speed limit (50km/h)	1.000000e+00
1	Speed limit (30km/h)	4.718102e-08
2	Speed limit (80km/h)	6.938812e-10
3	Speed limit (60km/h)	3.641608e-11
4	Double curve	2.737111e-11

Fifth image (wild animal crossing):



	label	probability
0	Wild animals crossing	1.000000e+00
1	Slippery road	2.518711e-24
2	Double curve	5.092477e-25
3	No passing for vehicles over 3.5 metric tons	1.017883e-27
4	Road work	1.921419e-31