

Traffic Sign Recognition

Writeup Template

You can use this file as a template for your writeup if you want to submit it as a markdown file, but feel free to use some other method and submit a pdf if you prefer.

Build a Traffic Sign Recognition Project

The goals / steps of this project are the following:

- Load the data set (see below for links to the project data set)
- Explore, summarize and visualize the data set
- Design, train and test a model architecture
- Use the model to make predictions on new images
- Analyze the softmax probabilities of the new images
- Summarize the results with a written report

Rubric Points

Here I will consider the [rubric points](#) individually and describe how I addressed each point in my implementation.

Writeup / README

1. Provide a Writeup / README that includes all the rubric points and how you addressed each one. You can submit your writeup as markdown or pdf. You can use this template as a guide for writing the report. The submission includes the project code.

You're reading it! and here is a link to my [project code](#)

Data Set Summary & Exploration

1. Provide a basic summary of the data set and identify where in your code the summary was done. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.

The code for this step is contained in the second code cell of the IPython notebook.

I used the pandas library to calculate summary statistics of the traffic signs data set:








- The size of training set is 34799
- The size of test set is 12630
- The shape of a traffic sign image is 32323
- The number of unique classes/labels in the data set is 43

2. Include an exploratory visualization of the dataset and identify where the code is in your code file.

The code for this step is contained in the 7 code cell of the IPython notebook.

Here is an exploratory visualization of the data set. It is a bar chart showing how the data ...



	36: Go straight or right	330	<div></div>
	37: Go straight or left	180	<div></div>
	38: Keep right	1860	<div></div>
	39: Keep left	270	<div></div>
	40: Roundabout mandatory	300	<div></div>
	41: End of no passing	210	<div></div>
	42: End of no passing by vehicles over 3.5 metric tons	210	<div></div>

Design and Test a Model Architecture

1. Describe how, and identify where in your code, you preprocessed the image data. What techniques were chosen and why did you choose these techniques? Consider including images showing the output of each preprocessing technique. Pre-processing refers to techniques such as converting to grayscale, normalization, etc.

The code for this step is contained in the 10 code cell of the IPython notebook.

As a first step, I decided to convert the images to grayscale because color does not effect to much, edge and shape is dominat.

Here is an example of a traffic sign image before and after grayscaleing.



As a last step, I normalized the image data because

After normalize grayscale image, the value is distributed in range $-0.5 \sim 0.5$, mean = 0

Each image is in the same range and centered, this won't result in one weight is too large and to another is too small.

2. Describe how, and identify where in your code, you set up training, validation and testing data. How much data was in each set? Explain what techniques were used to split the data into these sets. (OPTIONAL: As described in the "Stand Out Suggestions" part of the rubric, if you generated additional data for training, describe why you decided to generate additional data, how you generated the data, identify where in your code, and provide example images of the additional data)

The code for splitting the data into training and validation sets is contained in the fifth code cell of the IPython notebook.

To cross validate my model, I randomly split the training data into a training set and validation set. I did this by ...

My final training set had X number of images. My validation set and test set had Y and Z number of images.

i do not generate additional data

3. Describe, and identify where in your code, what your final model architecture looks like including model type, layers, layer sizes, connectivity, etc.) Consider including a diagram and/or table describing the final model.

The code for my final model is located in the nineteen cell of the ipython notebook.

My final model consisted of the following layers:

Layer	Description
Input	32x32x3 RGB image
Convolution 5x5	1x1 stride, same padding, outputs 28x28x64
RELU	
Max pooling	2x2 stride, outputs 14x14x64
Convolution 5x5	1x1 stride, same padding, outputs 10x10x64
RELU	
Max pooling	2x2 stride, outputs 5x5x64
Flatten	5x5x64
Fully connected	384
DropOut.	0.5
RELU	
Fully connected	192
DropOut.	0.5
RELU	
Fully connected	43
softmax	

4. Describe how, and identify where in your code, you trained your model. The discussion can include the type of optimizer, the batch size, number of epochs and any hyperparameters such as learning rate.

The code for training the model is located in the 22, 24, 25 cell of the ipython notebook.

To train the model, I used

Epoch = 20

Batch size = 128

Learning rate = 0.001

5. Describe the approach taken for finding a solution. Include in the discussion the results on the training, validation and test sets and where in the code these were calculated. Your

approach may have been an iterative process, in which case, outline the steps you took to get to the final solution and why you chose those steps. Perhaps your solution involved an already well known implementation or architecture. In this case, discuss why you think the architecture is suitable for the current problem.

The code for calculating the accuracy of the model is located in the 25th cell of the lpython notebook.

My final model results were:

training set accuracy of 1.00

validation set accuracy of 0.971

test set accuracy of ? 0.948

If an iterative approach was chosen: * What was the first architecture that was tried and why was it chosen?

i try to use LeNet because it is simple to build

- What were some problems with the initial architecture?
the depth of convolution layer is to small, do not apply dropout
- How was the architecture adjusted and why was it adjusted? Typical adjustments could include choosing a different model architecture, adding or taking away layers (pooling, dropout, convolution, etc), using an activation function or changing the activation function. One common justification for adjusting an architecture would be due to over fitting or under fitting. A high accuracy on the training set but low accuracy on the validation set indicates over fitting; a low accuracy on both sets indicates under fitting.
at the first training accuracy is too low (0.89 ~ 0.90) so i increase the number of filter of convolution layer and validation accuracy is not hight enough(0.93), i apply dropout on fully connection layer
- Which parameters were tuned? How were they adjusted and why?
increase the number of filter of convolution layer fromr 16 to 64
- What are some of the important design choices and why were they chosen? For example, why might a convolution layer work well with this problem? How might a dropout layer help with creating a successful model? ANS: more filter of convolution and lower learning rate can increase accuracy, apply dropout can mitigate overfitting

If a well known architecture was chosen: *What architecture was chosen? i use LeNet* Why did you believe it would be relevant to the traffic sign application? so far i just know LeNet and i will try to modify it to make it better * How does the final model's accuracy on the training, validation and test set provide evidence that the model is working well? both validation and test accuracy up to 0.94, i think it is not bad.

Test a Model on New Images

1. Choose five German traffic signs found on the web and provide them in the report. For each image, discuss what quality or qualities might be difficult to classify.

Here are five German traffic signs that I found on the web:





The first and fourth image might be difficult to classify because ...

the first image(Children Crossing) is similar with fifth image(Road Work) only the bottom-right shape is different, this would lead to wrong prediction
but i am not sure why the fourth image(Speed Limit, 30km/h) is predicted as "Keep Right", it shape is not similar

2. Discuss the model's predictions on these new traffic signs and compare the results to predicting on the test set. Identify where in your code predictions were made. At a minimum, discuss what the predictions were, the accuracy on these new predictions, and compare the accuracy to the accuracy on the test set (OPTIONAL: Discuss the results in more detail as described in the "Stand Out Suggestions" part of the rubric).

The code for making predictions on my final model is located in the 54 cell of the lpython notebook.

Here are the results of the prediction:

Image	Prediction
children crossing	Road work

Go straight or right	Go straight or right
No entry	No entry
Speed limit (30km/h)	Keep right
Road work	Road work

The model was able to correctly guess 3 of the 5 traffic signs, which gives an accuracy of 60%. This compares favorably to the accuracy on the test set of ...

the result is lower than test accuracy, i think the accuracy of this CNN will drop when i test new image of traffic sign, maybe this CNN architecture is not good enough maybe we need more training image to train.

3. Describe how certain the model is when predicting on each of the five new images by looking at the softmax probabilities for each prediction and identify where in your code softmax probabilities were outputted. Provide the top 5 softmax probabilities for each image along with the sign type of each probability. (OPTIONAL: as described in the "Stand Out Suggestions" part of the rubric, visualizations can also be provided such as bar charts)

The code for making predictions on my final model is located in the 79 cell of the lpython notebook.

For the first image, the model is relatively sure that this is a stop sign (probability of 0.6), and the image does contain a stop sign. The top five softmax probabilities were

Probability	Prediction
0	0.9408318400382996
1	0.029495202004909515
2	0.025709396228194237
3	0.0035343191120773554
4	0.00018940673908218741

For the second image

Probability	Prediction
1	1.0
2	2.4997980574426037e-08
3	6.561291787665624e-11
4	1.0298305611056158e-11
5	5.138845620734911e-13

For the third image

Probability	Prediction
1	0.9989874958992004
2	0.0009114257409237325
3	3.193494558217935e-05
4	1.5488320059375837e-05

5	1.2587855962920003e-05
---	------------------------

For the fourth image

Probability	Prediction
1	0.430292010307312
2	0.2660292387008667
3	0.21420225501060486
4	0.014428967610001564
5	0.011855375953018665

For the fifth image

Probability	Prediction
1	1.0
2	6.534788884332305e-15
3	5.198017580601566e-15
4	1.715955211099122e-16
5	6.932715195603702e-17