# Traffic Sign Recognition

## Writeup

**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

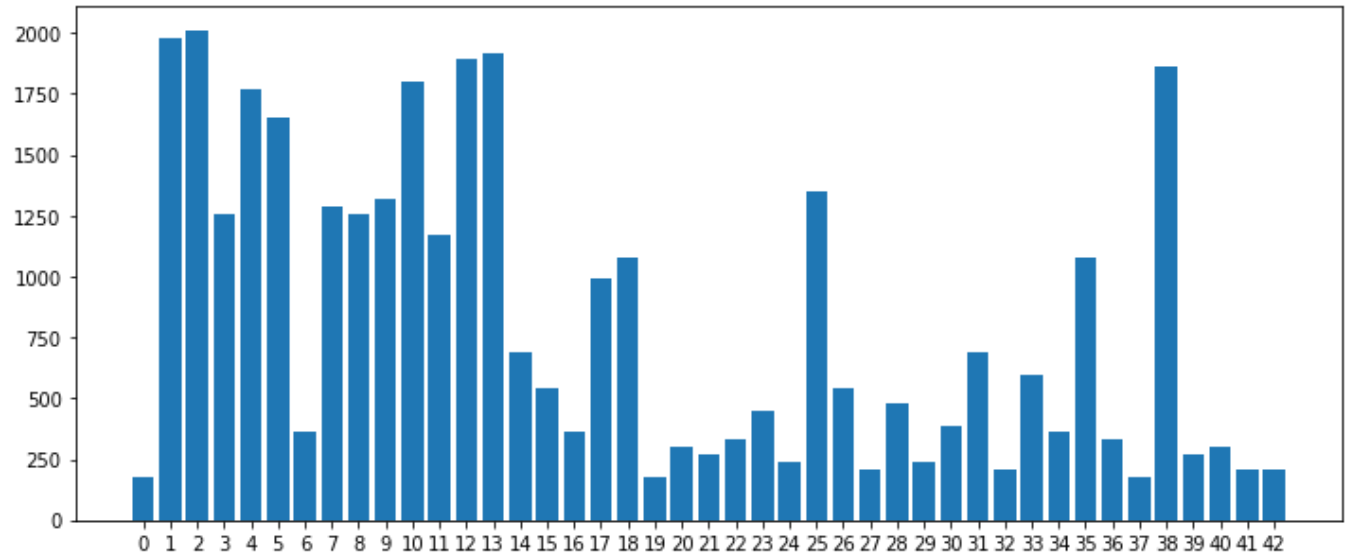
**1. Provide a basic summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.**

I used the the shape method to compute the stats of the data set:

- The size of training set is 34799
- The size of the validation set is 4410
- The size of test set is 12630
- The shape of a traffic sign image is 32x32
- The number of unique classes/labels in the data set is 43

**2. Include an exploratory visualization of the dataset.**

For a visual exploration of the dataset I used the Counter package. This way I was able to plot the exact distribution of labels in the training set.

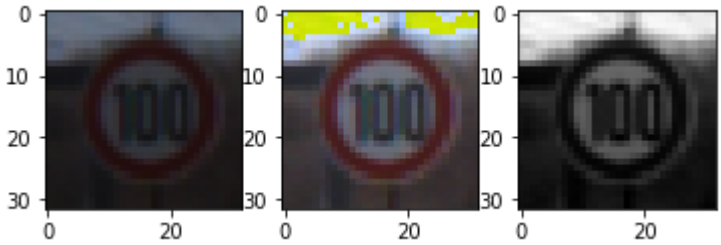



## Design and Test a Model Architecture

**1. Describe how 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. (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, and provide example images of the additional data. Then describe the characteristics of the augmented training set like number of images in the set, number of images for each class, etc.)**

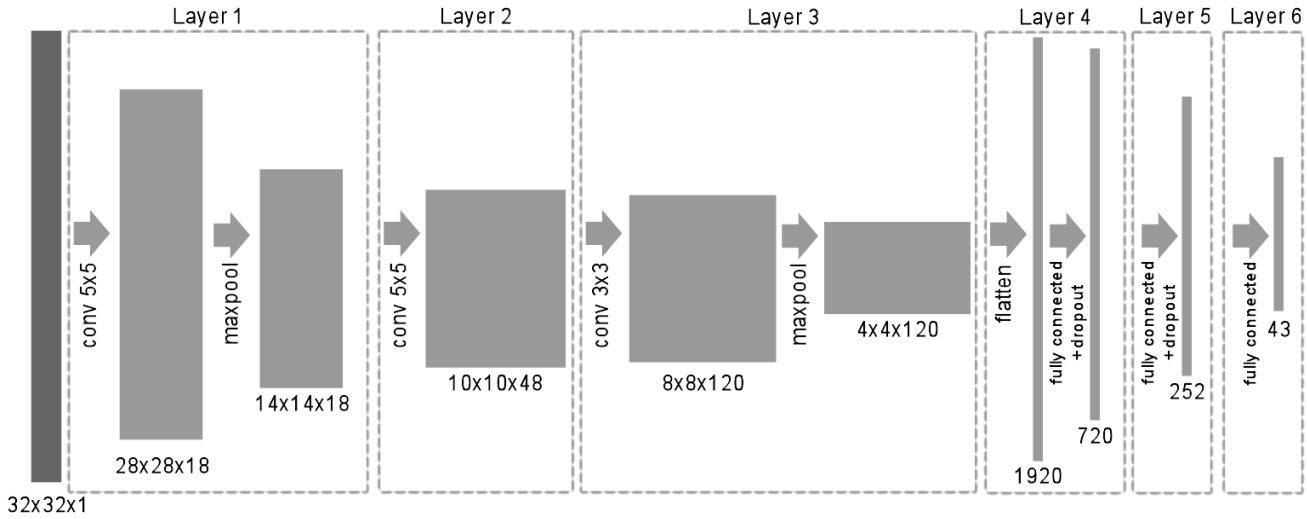
As a first step, I decided to convert the images to grayscale since the color in the traffic signs in only a redundant information channel. More importantly, by excluding the color information, we ignore changes in color due to environmental factors such as sunlight and weather which could confuse the network.

As an additional preprocessing step, the images have also been normalized. The image below shows an image after normalization and grayscaling:




**2. Describe 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.**

My final model consisted of the 6 layers as illustrated in the diagram below:




All activation functions are relus.

**3. Describe how 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.**

To train the model I built a training pipeline using a softmax with cross entropy loss function and an AdamOptimizer. The loss function has been extended with an L2 regularizer to avoid overfitting.

**4. Describe the approach taken for finding a solution and getting the validation set accuracy to be at least 0.93. 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.**

My final model results were:

- training set accuracy of **0.994** (computed in code cell "Train the model", line 25)
- validation set accuracy of **0.967** (computed in code cell "Train the model", line 28)
- test set accuracy of **0.950** (computed in code cell "Evaluation with test data", line 5)

I first started using the LeNet architecture which I adapted to the new input format. This yielded an initial validation accuracy of **0.88**.

I then added some regularization techniques (dropout, L2-regularization) to make sure I was not overfitting. This yielded a small accuracy boost of roughly 1-2%.

Next, I noticed the accuracy was increasing up until the last epoch. To make sure, the training would not stop until achieving full accuracy, I reduced the batch size to 64. This meant more batches and yielded another 2-3% accuracy on the validation set.

I managed to again improve the accuracy by increasing the number of filters (i.e. the "depth") by 3x. This placed by at **0.94**. While I also experimented with some 1x1 convolutions, these did not improve the result.

Finally I decided to alter the architecture of the network in my quest for a better result. One thing that bothered by at the LeNet architecture, was the multiple pooling layers which repeatedly threw out information. Whereas replacing the pooling with an increased stride for the convolutions negatively impacted the results, removing one of the pooling actually helped. Instead of the pooling, I opted for a third convolution layer (this time, a 3x3) to help with the dimensionality reduction. This placed me well above **0.95**.

Interestingly, there was no difference in accuracy between grayscale and color input.

## 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:


70


STOP


All images are cropped from actual natural scenes, making them accurate representations of traffic signs in the real world. The third image might be difficult to recognize due to a slight sun glare and picture 4 is slightly covered by vegetation.

**2. Discuss the model's predictions on these new traffic signs and compare the results to predicting on the test set. 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).**

Here are the results of the prediction:

| Image | ID | Prediction | ID |
|---|---|---|---|
| Priority Road | 12 | Priority Road | 12 |
| Stop | 14 | Priority Road | 12 |
| Speed limit (70km/h) | 4 | Speed limit (30km/h) | 1 |
| Children crossing | 28 | Children crossing | 28 |
| Keep right | 38 | Keep right | 38 |

The model was able to correctly guess 3 of the 5 traffic signs, which gives an accuracy of 60%. This is an unexpected result since the test set accuracy was well above 95%.

Particularly interesting si the misclassification of the stop sign, which should be very easy to recognize. However, the lack of color information could explain this. In fact, removing the grayscale conversion does lead to a better recognition of the stop sign (although in that case the children crossing sign gets misclassified).

**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. 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)**

For the first image (Priority Road), the model is very sure that this is a priority road sign (probability of 0.99), and the image does contain a priority road sign. The top five soft max probabilities were:

| Probability | Prediction |
|---|---|
| .99 | **Priority Road** |
| <.01 | Roundabout mandatory |
| <.01 | Speed limit (100km/h) |
| <.01 | End of no passing |
| <.01 | Ahead only |

For the second image (Stop), the model is confused about the sign. In fact, the best scoring class is the wrong one with a probability of 0.29. The correct class receives the second highest probability of 0.28. As mentioned above, this might be because of the lack of color information or the weird angle in which the photo was taken. The top five soft max probabilities were:

| Probability | Prediction |
|---|---|
| .29 | Priority Road |
| .28 | **Stop** |
| .07 | Speed limit (100km/h) |
| .05 | End of no passing |
| .03 | Ahead only |

For the third image (70km/h), the model is relatively sure that this is a 30 km/h sign (probability of 0.74), however this is wrong. Even worse, the correct class is not even in the top 5. It is difficult to say why this is the case. One possibility is that the model rather learned the background than the sign itself. The top five soft max probabilities were:

| Probability | Prediction |
|---|---|
| .74 | Speed limit (30km/h) |
| .06 | General caution |
| .03 | Wild animals crossing |
| .03 | Keep right |
| .02 | Right-of-way at the next intersection |

For the fourth image (Children crossing), the model correctly recognized the children crossing sign (probability of 0.87). The top five soft max probabilities were:

| Probability | Prediction |
|---|---|
| .87 | **Children crossing** |
| .10 | General caution |
| <.01 | Dangerous curve to the right |
| <.01 | Keep right |
| <.01 | Speed limit (60km/h) |

For the final image (keep right), the model correctly recognized the sign (probability of 1.0). The top five soft max probabilities were:

| Probability | Prediction |
|---|---|
| 1.0 | **Keep right** |
| <.01 | Speed limit (50km/h) |
| <.01 | Yield |
| <.01 | Turn left ahead |
| <.01 | Bumpy road |

## (Optional) Visualizing the Neural Network (See Step 4 of the Ipython notebook for more details)

**1. Discuss the visual output of your trained network's feature maps. What characteristics did the neural network use to make classifications?**