#Traffic Sign Recognition

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

- The size of training set is ?

  the size of training set is 34799

- The size of the validation set is ?
- validation set is
- The size of test set is ?

  Test set is 12630

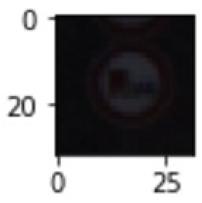
- The shape of a traffic sign image is ?

  Image data shape = (34799, 32, 32, 3)

- The number of unique classes/labels in the data set is ?

  The Number of classes = 43

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

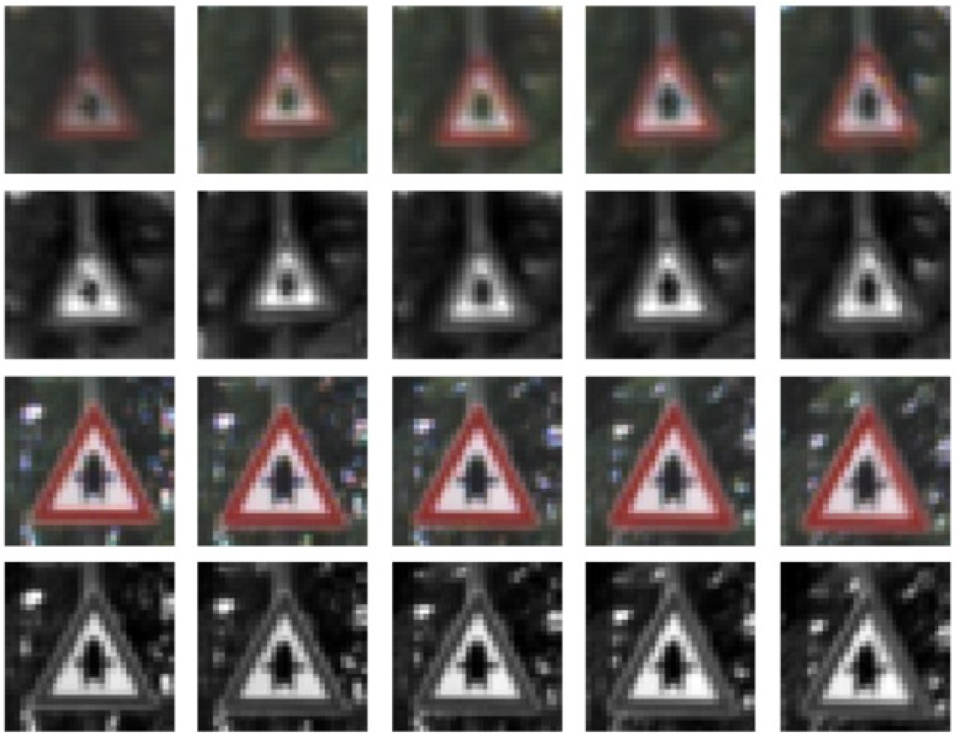


Here is an exploratory visualization of the data set. It is a bar chart showing how the shape of the data.

###Design and Test a Model Architecture

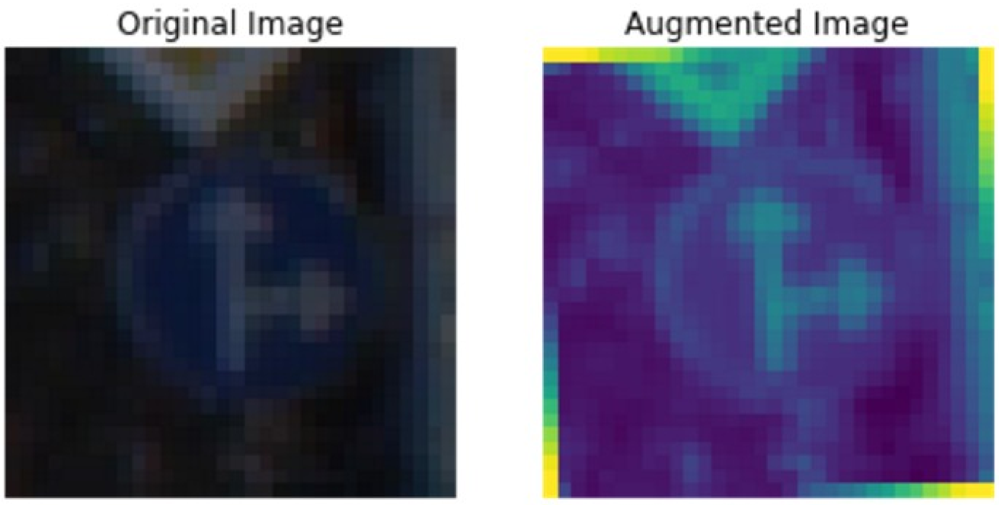
Question 1. Describe how you preprocessed the image data. What techniques were chosen and why did you choose these techniques? Consider including images showing the output of each preprocessing technique. Pre-processing refers to techniques such as converting to grayscale, normalization, etc. (OPTIONAL: 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 because it helps to reduce training time. The RGB shape is (32 , 32, 3), while grayscale shape is (32, 32, 1). Here is an example of a traffic sign image before and after grayscaling.



Then I normalizing the data to the range of (-1, 1). by equation (pixel - 128)/ 128, the training set is normalized. From the lessons, I know a wider distributio in the data would make it more difficult to train.

Last , I decided to generate additional data because the larger number of the set, the more accuracy of the data. To add more data to the the data set, I augment the dataset by functions such as random_scale, random_rotate. random rotation around image center (random value between -15 and 15 deg) and random vertical stretching (as the simplest way to simulate different viewing angle) by random value up to 40 %. It could consider the situation, such like camera is shaking or the traffic sign has some angle in real. Here is an example of an original image and an augmented image:



The difference between the original data set and the augmented data set is the following ...

Number of training examples = 382789 Number of testing examples = 12630 Image data shape = (34799, 32, 32, 3) Number of classes = 43

####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.

To get a better performance, the layer is the deeper, the better. parameter efficiency get much more performance with pure parameters by going deeper rather than wider.Secondly, I tend to have a hierarchical structure.

My final model consisted of the following layers:

Layer	Description
Input	32x32x1 Y channel image
Convolution 5x5	1x1 stride, valid padding, outputs 28x28x6
RELU	
Max pooling	2x2 stride, outputs 14x14x6
Convolution 5x5	1x1 stride, valid padding, outputs 10x10x16
RELU	
Max pooling	2x2 stride, outputs 5x5x16
Flatten	Input 5x5x16 output 400
Fully connect	Input = 400. Output = 120
RELU	
Dropout	
Fully Connected	Input = 120. Output = 84
RELU	
Dropout	
Fully Connected	Input = 84. Output = 10

####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.

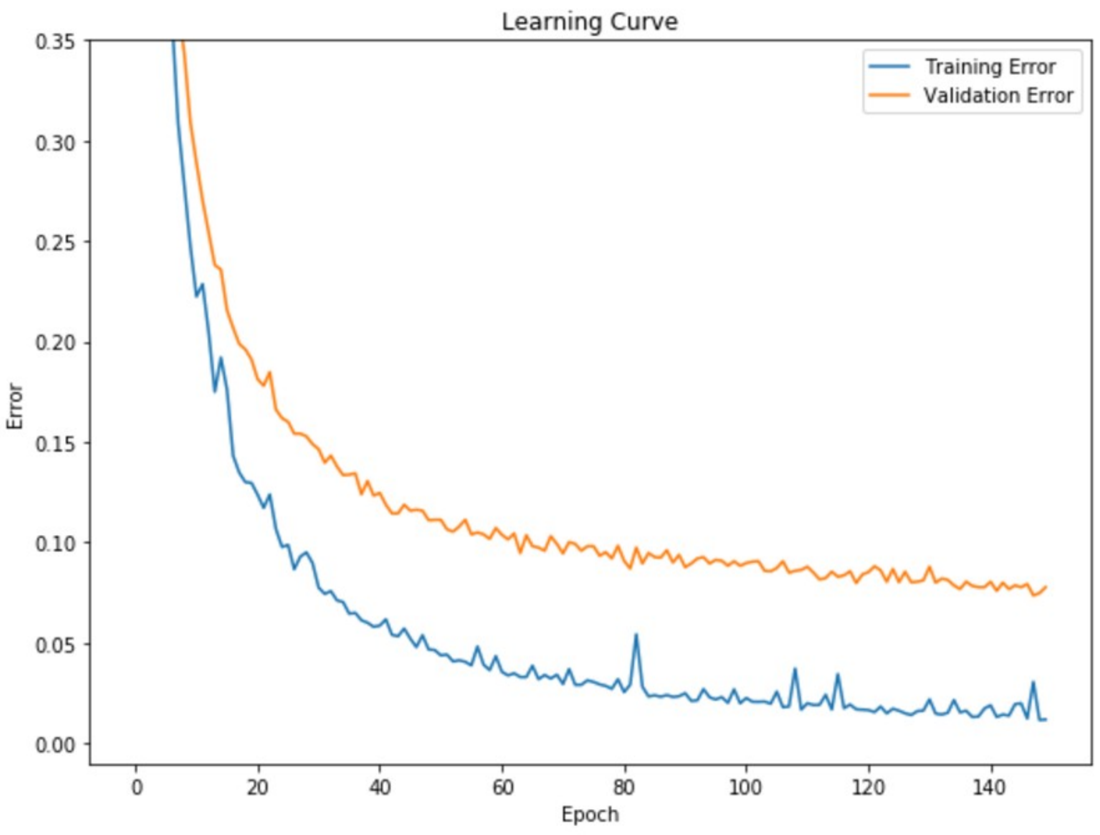
Model	Description
the type of optimizer	Adam optimizer
batch size	128
epochs	150
learning rate	0.0001
dropout	0.3

To train the model, I used an Adam optimizer , learning rate of 1e-4 , dropout rate of 0.3 and batch size of 128.

####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 99%
- •validation set accuracy of 92%
- •test set accuracy of 91%



If an iterative approach was chosen:

•What was the first architecture that was tried and why was it chosen?

The first architecture is the classical LeNet neural network, with 2 convolution layers and 3 fully connected layers. I also droped the layer 5 from my LeNet.

•What were some problems with the initial architecture?

The error is initial so big, and the training time is much long. So I changed the LeNet structure through adding dropout layers. Dropout makes my test accuracy into the 90's by preventing some of that overfitting.

•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 overfitting or underfitting. 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.

•Which parameters were tuned? How were they adjusted and why?

I tuned the learning rate from 0.0009 to 0.00001. when the rate is higher than 0.00005, the validation error is always above 0.95. It seems that the error could not decrease.

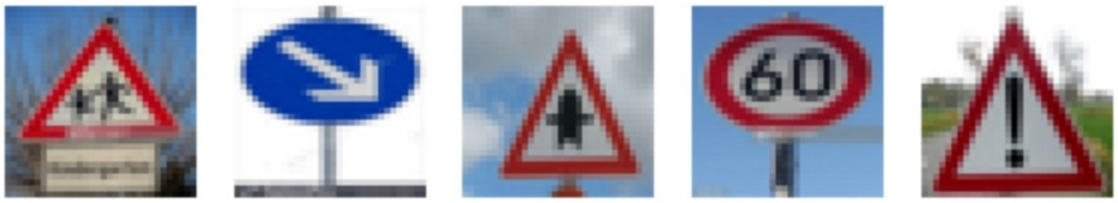
Secondly, I tuned the final output shape from 43 to 10.

the droprate from 0.3 to 0.7

###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 image might be difficult to classify because ...

####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	Prediction
Children crossing	Priority road
Right-of-way at the next intersection	Right-of-way at the next intersection
Roundabout mandatory	Roundabout mandatory
60 km/h	Priority road
General caution	General caution

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 ...

####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)

The code for making predictions on my final model is located in the 11th cell of the Ipython notebook.

```
Top 5 Labels for image 'children_crossing':

  - 'Priority road' with prob = 0.99
  - 'No entry' with prob = 0.00
  - 'End of no passing' with prob = 0.00
  - 'Roundabout mandatory' with prob = 0.00
  - 'End of no passing by vehicles over 3.5 metric tons' with prob = 0.00

Top 5 Labels for image 'Right-of-way at the next intersection.png':
  - 'Right-of-way at the next intersection' with prob = 0.93
  - 'Beware of ice/snow' with prob = 0.07
  - 'General caution' with prob = 0.00
  - 'Double curve' with prob = 0.00
  - 'Pedestrians' with prob = 0.00

Top 5 Labels for image 'Roundabout mandatory':
  - 'Roundabout mandatory' with prob = 0.58
  - 'No entry' with prob = 0.16
  - 'Priority road' with prob = 0.13
  - 'Turn left ahead' with prob = 0.05
  - 'Slippery road' with prob = 0.03


Top 5 Labels for image 'Speed limit (60km/h)':
  - 'Priority road' with prob = 0.40
  - 'Keep right' with prob = 0.35
  - 'No entry' with prob = 0.15
  - 'Stop' with prob = 0.04
  - 'Yield' with prob = 0.02

Top 5 Labels for image 'General caution'':
  - 'General caution' with prob = 1.00
  - 'Traffic signals' with prob = 0.00
  - 'Pedestrians' with prob = 0.00
  - 'Wild animals crossing' with prob = 0.00
  - 'Road narrows on the right' with prob = 0.00
```