Writeup / README

Data Set Summary & Exploration

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 numpy 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 (32, 32, 3)
- The number of unique classes/labels in the data set is 43

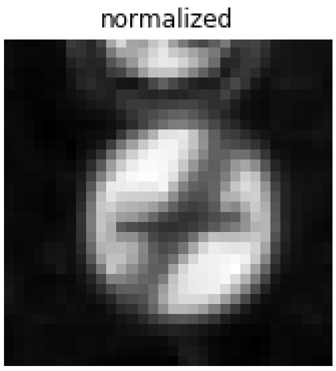
2. Include an exploratory visualization of the dataset and identify where the code is in your code file.
The code for this step is contained in the third code cell of the IPython notebook.

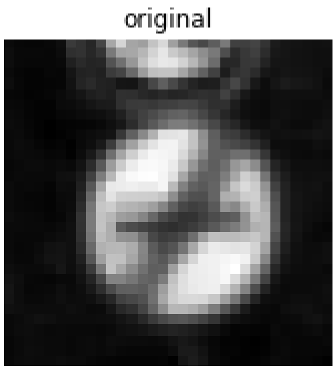
Design and Test a Model Architecture

1. Describe how, and identify where in your code, you preprocessed the image data. What tecniques 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.

As a first step, I decided to convert the images to grayscale because it helped in improving the accuracy.

Here is an example of a traffic sign image before and after grayscaling and normalizing





As a last step, I normalized the image data because it increases speed of training and performance.

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 following layers:

Layer	Description
Input	32x32x1 Grayscale image
Convolution 3x3	1x1 stride, same padding, outputs 28x28x6
RELU	
Input	28x28x6
Convolution 3x3	2x2 stride, same padding, outputs 14x14x10
RELU	
Input	14x14x10
Convolution 3x3	2x2 stride, same padding, outputs 8x8x16
RELU	
Max pooling	2x2 stride, kernel size 2x2
Output	16x16x64
Flatten	Input 4x4x16 Output 256
Fully connected	Input 256 Output 120

RELU	
Dropout	
Fully connected	Input 120 Output 100
RELU	
Fully connected	Input 100 Output 84
RELU	
Fully connected	Input 84 Output 43

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 used the following hyperparameters:
Epochs: 50
Batch size: 100
Learning rate: 0.0009

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 100
* validation set accuracy of 98
* test set accuracy of 94

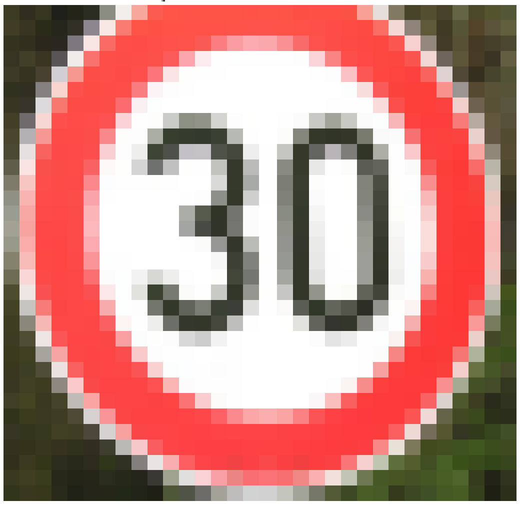
If an iterative approach was chosen:
* What was the first architecture that was tried and why was it chosen?
The first architecture that was tried was Lenet and was choosen because it had an accuracy of 90 percent which could be later improved upon for the given application.
* What were some problems with the initial architecture?
The initial architecture had a good accuracy of 90 percent but needed some minor improvements by adding more convolutional and fully connected layers so as to improve accuracy.
* 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.
Initially, I went for the Lenet architecture. Accuracy was around 90 percent. Later, the number of epochs was increased to 30 and learning rate was decreased to 0.0005 which gave an accuracy of 92 percent. After increasing the number of epcohs there was no significant change in the accuracy. Therefore, I had to make some changes to the model by adding convolutional and fully connected layers. At last with 50 epochs and 0.0009 learning rate and with a batch size of 100. 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.

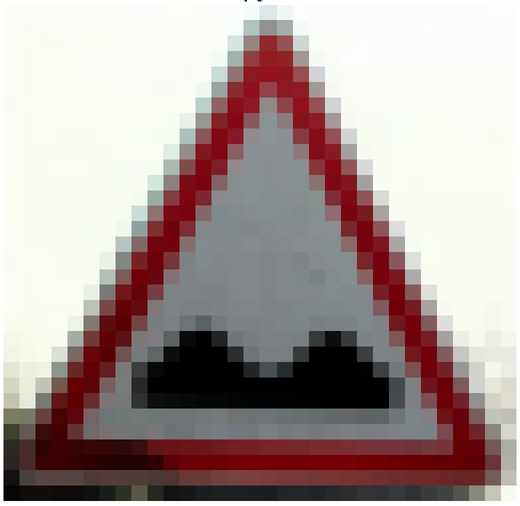
* Which parameters were tuned? How were they adjusted and why?
The learning rate was gradually decreased from 0.001 to 0.0005 and then back to 0.0009. the number of epochs was increased from 25 to 30 and eventually 50.
* What are some of the important design choices and why were they chosen? For example, why might a convolution layer work well with this problem? How might a dropout layer help with creating a successful model?
It is important to have the ideal number of convolutional and fully connected layers which can only be possible after a lot of hit and trial methods. For creating a successful model, the learning rate should be quite low for the model to get to a stable accuracy.
If a well known architecture was chosen:
* What architecture was chosen?
A lenet architecture was chosen on which convolutional and fully connected layers were added.
* Why did you believe it would be relevant to the traffic sign application?
It is relevant to the trafffic sign application because it has 43 classes and one needs to create a model which has sufficient number of convolutional and fully connected layers and also ensure that the model is not very sophisticated in order to prevent overfitting.
* How does the final model's accuracy on the training, validation and test set provide evidence that the model is working well?
The validation accuracy of th model is 98 percent and the test accuracy is 94 percent. Since, the model has never seen the test set before it tells us that the model is working well in classifying the traffic signs.

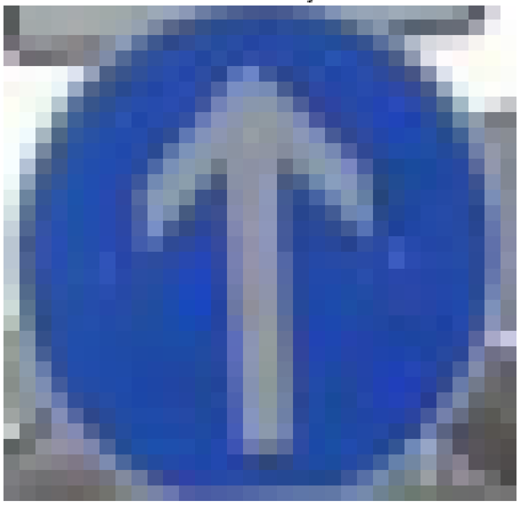
Test a Model on New Images

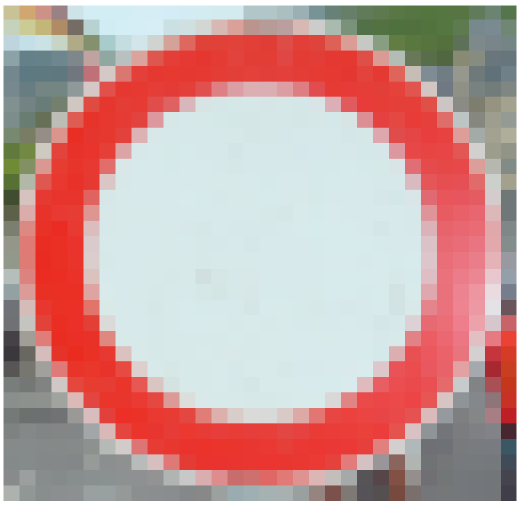
1. Choose five German traffic signs found on the web and provide them in the report. For each image, discuss what quality or qualities might be difficult to classify.

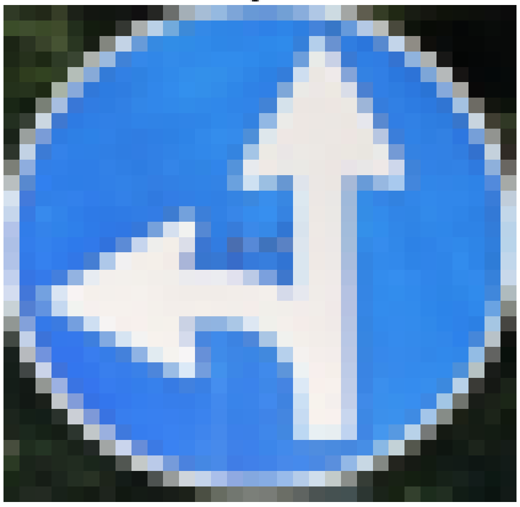
Here are six German traffic signs that I found on the web:

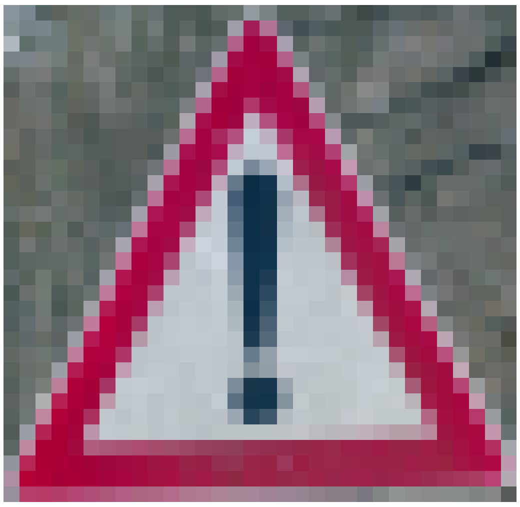

Speed limit (30km/h)


Bumpy road


Ahead only


No vehicles


Go straight or left


General caution

After converting them to grayscale and normalizing them,