**Traffic Sign Recognition**

**Writeup**

**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](https://review.udacity.com/" \l "!/rubrics/481/view) 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](https://github.com/udacity/CarND-Traffic-Sign-Classifier-Project/blob/master/Traffic_Sign_Classifier.ipynb)

**Data Set Summary & Exploration**

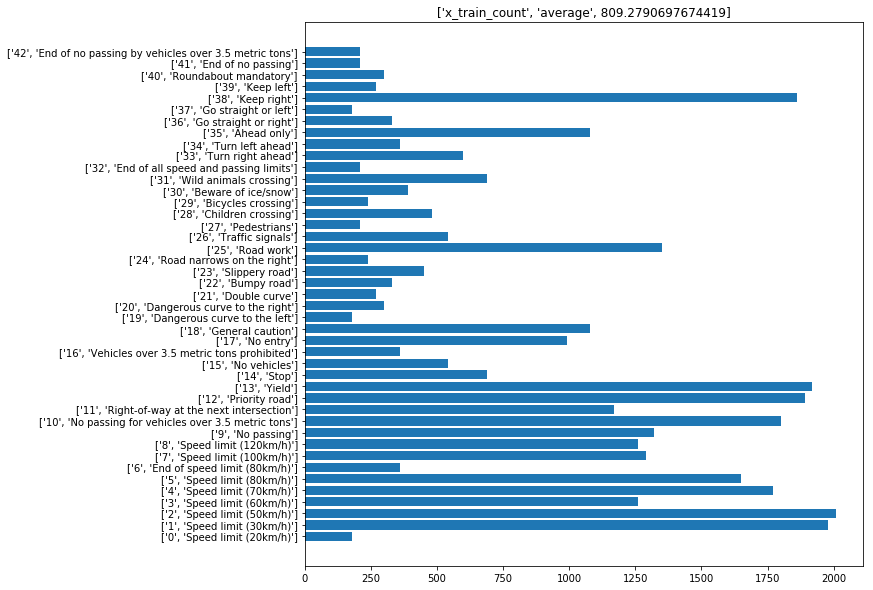
**1. Provide a basic summary of the data set. In the code, the analysis should be done using python, numpy and/or pandas methods rather than hardcoding results manually.**

私は、交通標識データセットの要約統計量を計算するためにPythonを使用しました：

* The size of training set is ?  
  初期状態で、training set は34799でした。（後ほど3倍に拡張します）
* The size of the validation set is ?  
  初期状態で、validation set は 4410でした。
* The size of test set is ?  
  初期状態で、test set は 12630でした。
* The shape of a traffic sign image is ?  
  幅32pixel　高さ32pixel　3チャンネルRGBでした。
* The number of unique classes/labels in the data set is ?  
  ラベル総数は43でした。

**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 data …



**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.)**

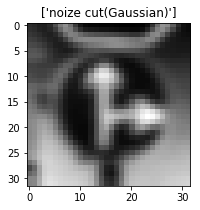
最初のステップとして、私は training set　 validation set　 test set画像をグレースケールに変換することに決めました。なぜなら道路標識は「色に基づく情報」よりも「形状に基づく情報」のほうが識別しやすいと判断したからです。また「色に基づく情報」を取り除くことでRGB24bitからグレースケール8bitとデータ量を削減し、パソコンの処理速度を向上できます。

例：グレースケール変更前　　　　　　　　　　　例：グレースケール変更後



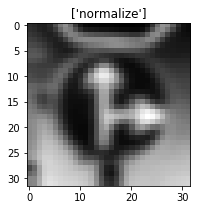
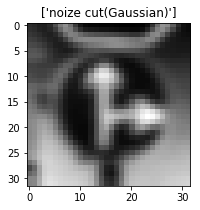
次にtraining set　 validation set　 test set別に、画像データに含まれるノイズを取り除くべくガウシアンフィルタ処理を行いました。

ノイズ除去前画像 ノイズ除去後画像

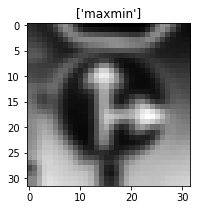
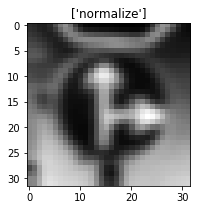


データセット画像の「明るさの基準」を揃えるため、画素平均値をとって画素標準偏差で割る正規化処理を加えました。

例：正規化処理前　　　　　　　　　　　　　　　　例：正規化処理後



最後にtraining set　 validation set　 test set別に、データセット画像のコントラストを整えるために、各画像内の画素の最低輝度が0、最大輝度が255となるよう階調補正処理を加えました。



トレーニングを行う前に私に追加データを生成しtraining set　に加えることに決めました。

私が採用した「形状に基づく情報」を優先した識別方法に対して、「ノイズを取り除くためのガウシアンフィルタ」は画像のエッジなどの「形状に基づく情報」の質を低下させる効果があるためです。

training setに追加するデータは、「ガウシアンフィルタを適用しない素のグレースケール画像」に正規化処理、階調補正処理を加えた画像です。

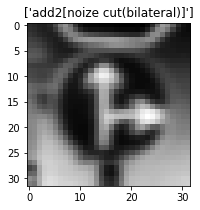
training setへの追加画像の例



training setにはさらに、「ガウシアンフィルタの代わりにバイラテラルフィルタを適用したグレースケール画像」に正規化処理、階調補正処理を加えた画像を追加しました。

バイラテラルフィルタの持つ、「ノイズカット」と「エッジ情報の保存効果」に期待しました。

training setへの追加画像の例



両方の追加画像を加えた結果、**training set** のデータセットは**104397**（**34799**の**3**倍）個に拡張されました。

**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 gray image |
| Convolution 5x5 | 1x1 stride, valid padding, outputs 32x32x1→28x28x6 |
| RELU | Dropout 100→90％ |
| Max pooling | 2x2 stride, outputs 14x14x6 |
| Convolution 5x5 | 1x1 stride, valid padding, outputs 14x14x6→10x10x6 |
| RELU | Dropout 100→80％ |
| Max pooling | 2x2 stride, outputs 5x5x16 |
| Flatten | 5x5x16→400 |
| Fully connected | 400→120 |
| RELU | Dropout 100→50％ |
| Fully connected | 120→84 |
| RELU | Dropout 100→50％ |
| Fully connected | 84→43 |
| Softmax | 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.**

より制度の高いモデルを訓練するために、先の述べたよう「**training set** のデータセットを**104397**（**34799**の**3**倍）個に拡張しました。

次にまずエポック回数を10回から20回に増やしました。

加えて学習率を定数値0.001から、検証結果がある一定値を向上する度に「学習率を小さく変動」させるよう、可変型に変更しました。

最後に各RELU処理の後にドロップアウト処理を加えました。ドロップアウトはConvolution層では50%維持Fully connected層では80%以上維持する設定としました。

**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 ?  
  training Accuracy は 1.000でした
* validation set accuracy of ?  
  validation Accuracy は 0.949でした
* test set accuracy of ?  
  test Accuracy は 0.920でした

If an iterative approach was chosen:

If a well known architecture was chosen:

私はLeNet Lab Solutionを参考に今回の階層を構築しました。

文字の識別は、私が採用した「形状に基づく情報」を優先した識別方法が基本となりますので、標識の識別と親和性が高いと考えたからです。

また過学習を回避する効果を期待してドロップアウトを組み込みました。

最後に標識モデルの収束性を考慮し、学習率を可変式としました。

**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.**

私はウェブ上で６つのドイツ交通標識を探して今回のプロジェクトに利用しました。

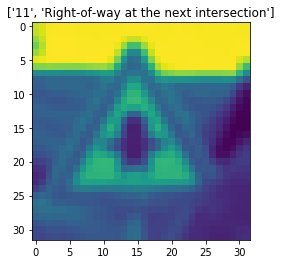
test set の画像と比較して、それらに最適なラベルを割りつけました。

各画像は、validation set　 test set同じく加工（RGB→グレースケール→正規化→階調補正）を施して標識識別器に入力しました。

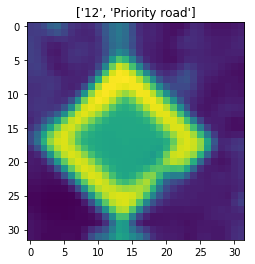
各6枚の画像を次頁に示します。

加工済み画像グレースケールに変換されていますが、目で見て形状の変化・構成などの内容を確認しやすいようカラーマップで表示しております。

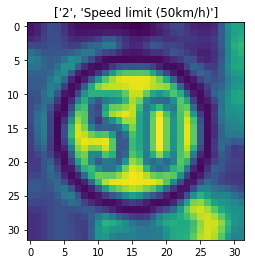
WEB取得画像① 加工済み画像①



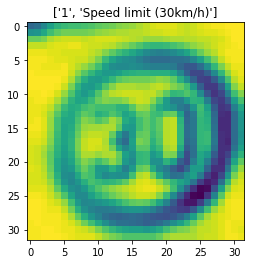
WEB取得画像② 加工済み画像②



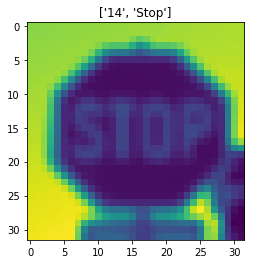
WEB取得画像③ 加工済み画像③



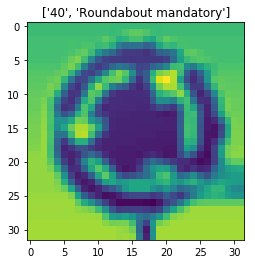
WEB取得画像④ 加工済み画像④



WEB取得画像⑤ 加工済み画像⑤



WEB取得画像⑥ 　　　　　　　　加工済み画像⑥



WEB取得画像④はかすれていて形状が曖昧で識別が難しいかもしれません。

WEB取得画像⑥は写真ではなく絵（またはCG?）であり右下に文字が描かれています。識別が難しいかもしれません。

**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).**

今回のモデルは、6つの交通標識全てを正しく推測することができ、少なくとも 9.999％以上の精度を得ることができました。

予測結果は次のとおりです。

| 私の画像 | **Image** | **Prediction** |
| --- | --- | --- |
| WEB取得画像① | 'Right-of-way at the next intersection | 'Right-of-way at the next intersection |
| WEB取得画像② | Priority road | Priority road |
| WEB取得画像③ | Speed limit (50km/h) | Speed limit (50km/h) |
| WEB取得画像④ | Speed limit (30km/h) | Speed limit (30km/h) |
| WEB取得画像⑤ | Stop | Stop |
| WEB取得画像⑥ | Roundabout mandatory | Roundabout mandatory |

**6枚のWEB取得画像全体の識別結果Analyze Performance** 0.920**でした**

**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.**

今回の標識モデルを用いて6つの交通標識に対する識別結果は、Ipythonノートブックのコメント「 Function (20) Output predictions 」におけるコードを実行することで得られます。 Function (20) Output predictions関数で得たSOFTMAX**Probability**値を以下に示します。

WEB取得画像①

TopKV2(values=array([[ 1.00000000e+00, 1.72792530e-10, 1.51868145e-12,

7.44638793e-14, 7.07576155e-14]], dtype=float32), indices=array([[11, 30, 20, 23, 42]], dtype=int32))

[array([ True], dtype=bool), 11, 11]

WEB取得画像②

TopKV2(values=array([[ 1.00000000e+00, 1.74434028e-17, 2.63003606e-18,

6.14423878e-20, 7.21043646e-21]], dtype=float32), indices=array([[12, 29, 25, 41, 35]], dtype=int32))

[array([ True], dtype=bool), 12, 12]

WEB取得画像③

TopKV2(values=array([[ 9.99999642e-01, 4.10662437e-07, 7.41619469e-29,

8.99546478e-36, 2.42810738e-36]], dtype=float32), indices=array([[2, 1, 5, 3, 4]], dtype=int32))

[array([ True], dtype=bool), 2, 2]

WEB取得画像④

TopKV2(values=array([[ 1.00000000e+00, 1.43758866e-10, 1.78828899e-14,

2.51846770e-19, 3.71558287e-20]], dtype=float32), indices=array([[1, 5, 6, 2, 3]], dtype=int32))

[array([ True], dtype=bool), 1, 1]

WEB取得画像⑤

TopKV2(values=array([[ 1.00000000e+00, 4.56010270e-08, 1.75836956e-10,

7.53677242e-12, 8.56492665e-13]], dtype=float32), indices=array([[14, 17, 33, 1, 3]], dtype=int32))

[array([ True], dtype=bool), 14, 14]

WEB取得画像⑥

TopKV2(values=array([[ 1.00000000e+00, 1.73694992e-17, 5.10418373e-18,

7.65182650e-19, 1.05310645e-19]], dtype=float32), indices=array([[40, 7, 11, 12, 16]], dtype=int32))

[array([ True], dtype=bool), 40, 40]

| 私の画像 | **Image** | **Prediction** | **SOFTMAX** |
| --- | --- | --- | --- |
| WEB取得画像① | 'Right-of-way at the next intersection | 'Right-of-way at the next intersection | 1.00000000e+00 |
| WEB取得画像② | Priority road | Priority road | 1.00000000e+00 |
| WEB取得画像③ | Speed limit (50km/h) | Speed limit (50km/h) | 9.99999642e-01 |
| WEB取得画像④ | Speed limit (30km/h) | Speed limit (30km/h) | 1.00000000e+00 |
| WEB取得画像⑤ | Stop | Stop | 1.00000000e+00 |
| WEB取得画像⑥ | Roundabout mandatory | Roundabout mandatory | 1.00000000e+00 |

**(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?**

**申し訳ないが、別件で時間を確保できなかった。今回は Optional なので実施を見送ります。**