Traffic Sign Recognition

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

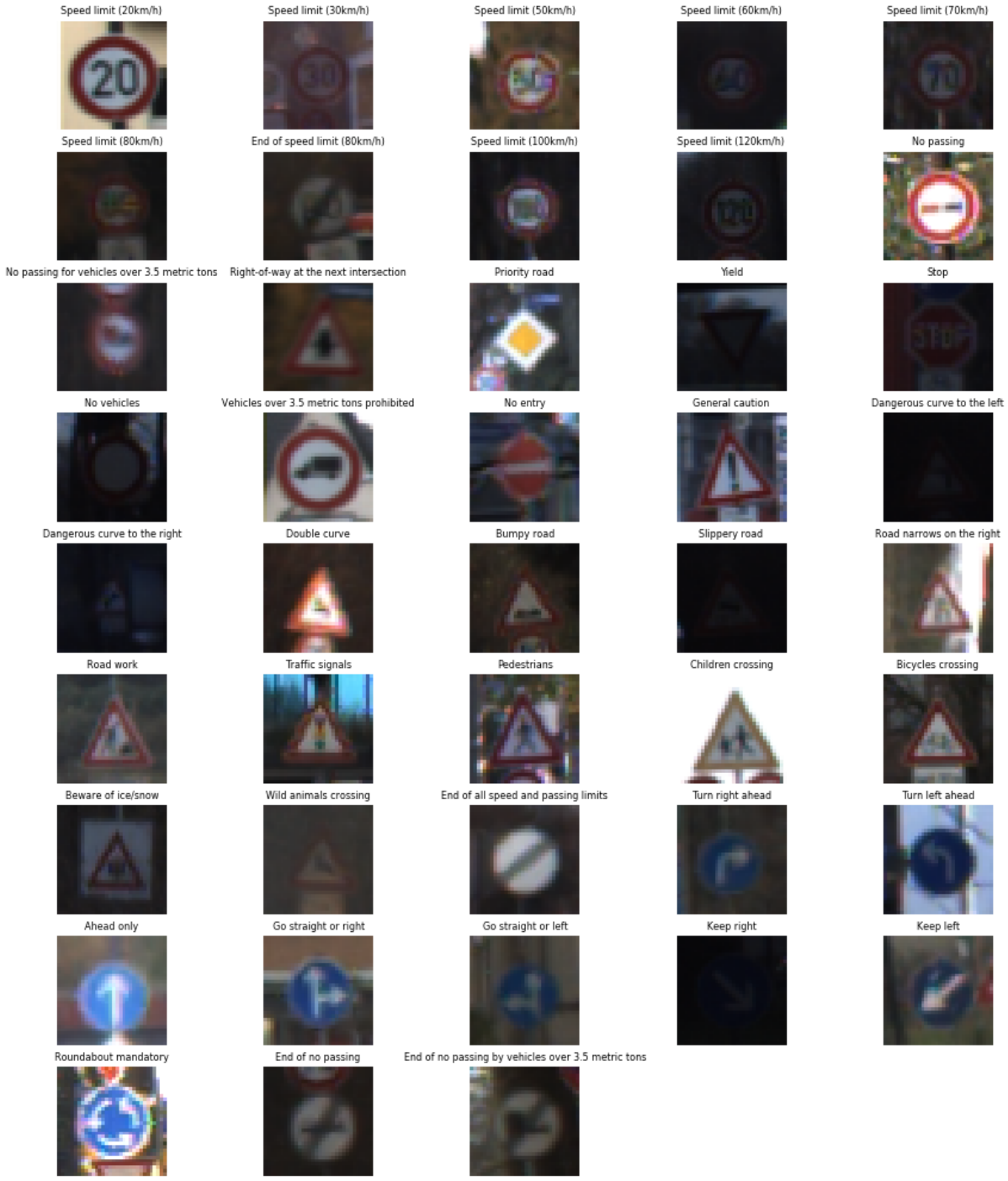
Data Set Summary & Exploration

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.

- Number of training examples = 34799
- Number of Validation examples = 4410
- Number of testing examples = 12630
- Image data shape = (32, 32, 3)
- Number of classes = 43

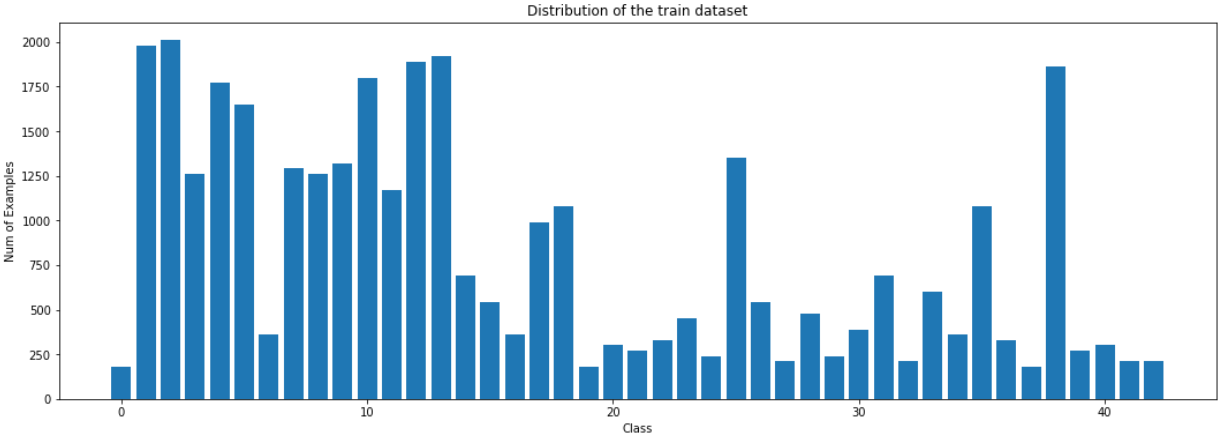
2. Include an exploratory visualization of the dataset.

Here is an exploratory visualization of the data
set.



Here is the distribution of the data set. It is a bar chart showing how many dataset in each classes and clearly indicates that dataset is

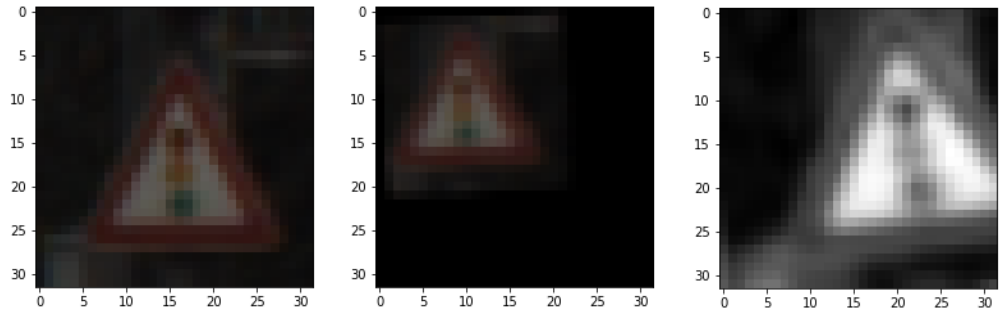
unbalanced.



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.)

- Intital step was to upsample the dataset as for few classes images are very less as compared to other. Hence model will be biased towards class which having more dataset. To avoid that we need to upsample classes with low values.
- Possible way was to upsample random images and make various duplicates of it in each class. But again that technique create various duplicates. To avoid that random samples are chosen from each class and they are transformed by varierty of parameters like angular rotation,translation,crop,zoom. Transform_image function was used to implement this.
- Then all images are converted into grayscale because we are expecting network to detect traffic sign by geometry and shape and not by color.
- At last step, I all images are normalized with zero mean and equal variance as it helps optimizers to find optimum solution. Here is an example of the whole process.i.e.image data comes in 0 to 255 pixle value, to normalize it I have used (X-Xmean)/std.



**Existing data

- Number of training examples = 34799
- Number of Validation examples = 4410
- Number of testing examples = 12630

**Data after Augmentation

- Number of training examples after image augument = 86430
- Number of testing examples after image augument = 12630
- Number of training labels after image augument = 86430
- Number of testing labels after image augument = 12630

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:

Input 32x32x1 Grayscale image

Convolutional #1 outputs 28x28x6

- 5x5 Filter and 1x1 stride
- Activation any activation function, we will relu
- Max pooling 2x2 stride, The output shape should be 14x14x6.
- Dropout Keep probability = 0.7
- Input Depth of 1 and output Depth of 6

Convolutional #2 outputs 10x10x16.

- 5x5 Filter and 1x1 stride
- Activation any activation function, we will relu

- Max pooling 2x2 stride, The output shape should be 5x5x16
- Dropout Keep probability = 0.7
- Input Depth of 6 and output Depth of 16

Flatten Flatten the output shape of the final pooling layer

- Fully Connected #1 outputs 120

Activation any activation function, we will relu

- Fully Connected #2 outputs 84

Activation any activation function, we will relu

- Fully Connected (Logits) #3 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, As discussed in the tutorial i used LeNet Model for implementing this image classification project. On using model in its raw form not giving a good accuracy. In order to increase accuracy Dropout probability and adam optimizer is used.Tried with SGD optimzer also but Adam gives better result. EPOCHS = 70 and BATCH_SIZE = 100 is used for the training purpose. Best results observed from mu = 0 and sigma = 0.1

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.

To acehive better validation accuracy I worked out on various hyperparameter like learning rate, epoch's, dropout keep probability and its position in network.

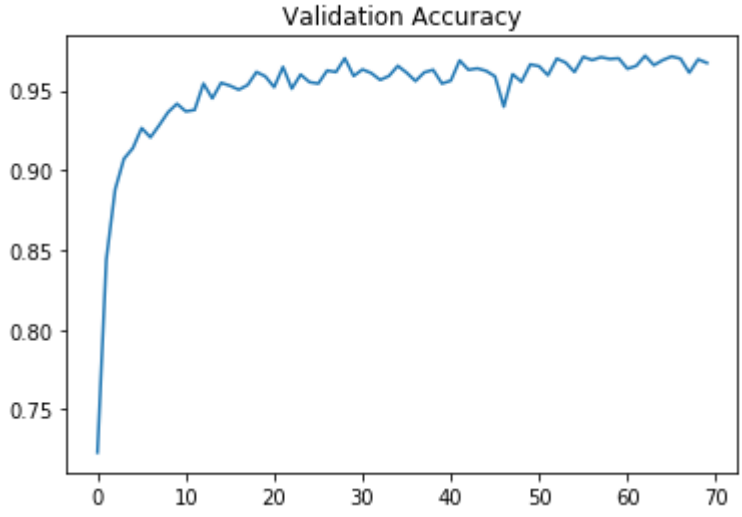
Following parameter were used to yield best result for my network:

- Learning Rate = 0.0009
- Epoch = 70

- batch Size = 100
- Dropout at FC 0 layer = 0.6
- Dropout at FC 2 layer = 0.6
- Dropout at Conv 1 layer = 0.7

My final model results were:

- Train Accuracy = 0.991
- Valid Accuracy = 0.971
- Test Accuracy = 0.948
  Below graph shows validation accuracy over number of Epoch's:



If an iterative approach was chosen:

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

I have used LeNet architecture as suggested in class tutorial without doing much changes.

What were some problems with the initial architecture?

While working on LeNet architecture in its stock format I have observed that accuracy of model not increasing as image data was not pre-processed also the model overfitting due to unbalanced datasets and network without dropouts.

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.

In order to avoid overfitting data we have introduced dropout at various convolution layer. Dropout probability of 30% for first convolutional layer and Dropout probability of 40% at each fully connected layer were introduced to avoid overfitting.

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

- Learning Rate:- Higher Learning rate train model faster but stagnant earlier than acheving its full potential, whereas for lower learning rate model train slower but it achieves lowest possible loss for that model.in my model learning rate of 0.0009 yields better results.
- Batch Size:- Since we can not train whole model at once due to computation power limitation, so we split model in batches, calculate all parameter for each batches and cascade it to top level for complete model, ats again on model size we can deside batch size, in our model Batch size is 100.
- Keep Probability:- To avoid overfitting in model we have to introduced dropout at different layer, In our model I have added dropout at Conv1 layer, at first fully connected layer, and at last fully connected layer but with different keep probability, like at earlier layer it is higher 0.7 in our case and at later layer it is 0.6.

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
Speed limit (30km/h)	Speed Limit(30km/h)
Keep left	Keep left
Speed limit (80km/h)	Keep right
Bicycles crossing	Bicycles crossing
Right-of-way at the next intersection	Right-of-way at the next intersection
Stop	Speed limit (120km/h)
Priority road	Priority road
Turn right ahead	Turn right ahead

Figure showing Top 3 Probabilities for detected images.

The model was not able to correctly guess Speed limit (80km/h) as there are text also available in the image and those text were not available in training image which gives an accuracy of 87.5%.

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.

For the third and forth image, the model is not sure about exact class category. Though it identifies correctly for Bicycles crossing but not with 100% accuracy. And for third image model is totally wrong giving only .45% accuracy.

Probability	Prediction
1	Speed Limit(30km/h)
1	Keep left
.45	Speed limit (80km/h)
.83	Bicycles crossing
1	Right-of-way at the next intersection
1	Speed limit (120km/h)
1	Priority road
1	Turn right ahead

(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?