Report: Red Round Sign Detection

Part 2: Training:

In this section, I will outline the CNN based architecture that I implemented for the given task. The original task was reformulated in a simple binary classification format — to identify whether the traffic signs are red round or not. Before that, I separated the training data again, randomly into train and validation partitions, where now we have 776 training images and 194 validation images, each of size (48 x 48 x 3). The validation dataset will be used to monitor the training progress, and will help us to understand the region where overfitting starts to occur.

Training hyper-parameters:

- learning rate = 0.0001
- number of steps = 1500
- batch size = 12
- drop out = 0.5

The CNN architecture (layer by layer):

1. First convolutional layer consisting of 16 kernels / filters of size 3 x 3
2. Non linear activation ReLU
3. Batch Normalization
4. Max Pooling with stride = 2
5. Second convolutional layer consisting of 32 kernels / filters of size 3 x 3
6. Non linear activation ReLU
7. Batch Normalization
8. Max Pooling with stride = 2
9. Fully Connected (FC) layer consisting of 256 neurons
10. Non linear activation ReLU
11. Apply dropout
12. Softmax output

The above model is basically a shallow convolution based neural network, which is suitable for this kind of object detection / classification tasks. Due to time constraint, I was unable to design other neural network structure and experiment on them.

Additionally, I was unable to experiment with varying batch sizes, drop out probabilities and learning rate for the same reason.

Pros:

- Convolution based approach helps us learn filters that can help us to extract low level features such as edge maps, bright spots etc in the early layers, followed by high level features such as shapes or circular patterns in the higher layers.
- The number of trainable parameters in convolutional layers are much less compared to fully connected layers, resulting in faster training and inference.

Cons:

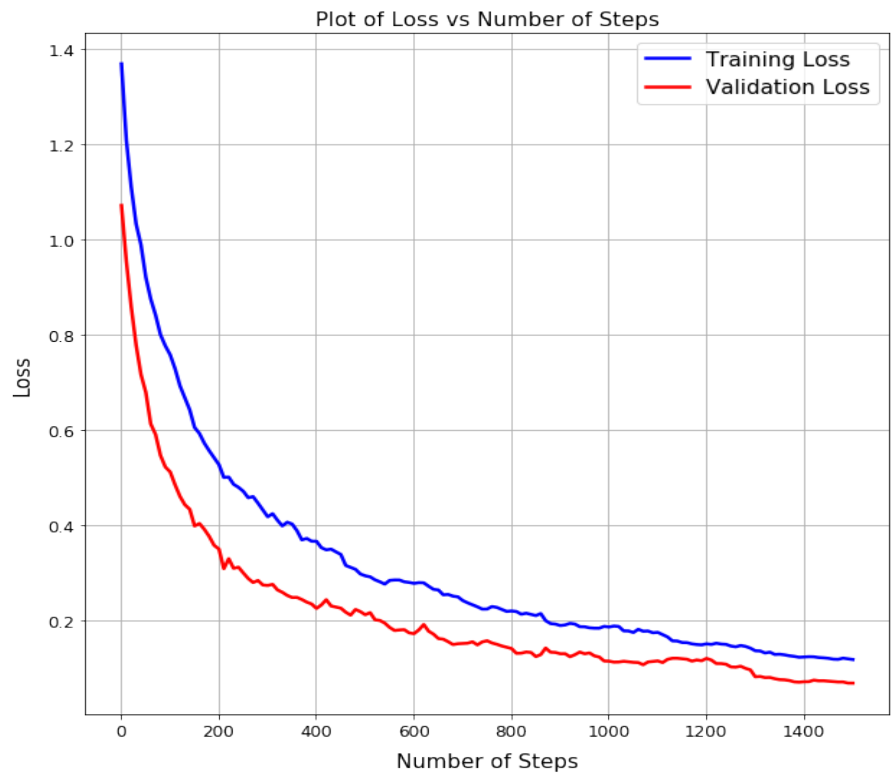
- Introduction of a FC layer drastically increases number of trainable parameters, which is problematic since we do not have enough training data.
- The model is shallow. A deeper model with sufficient amount of data will result in better performance.
- The model is trained with simple SGD, which doesn't have an adaptive learning rate. So the performance of the model heavily depends on the learning rate initialization.

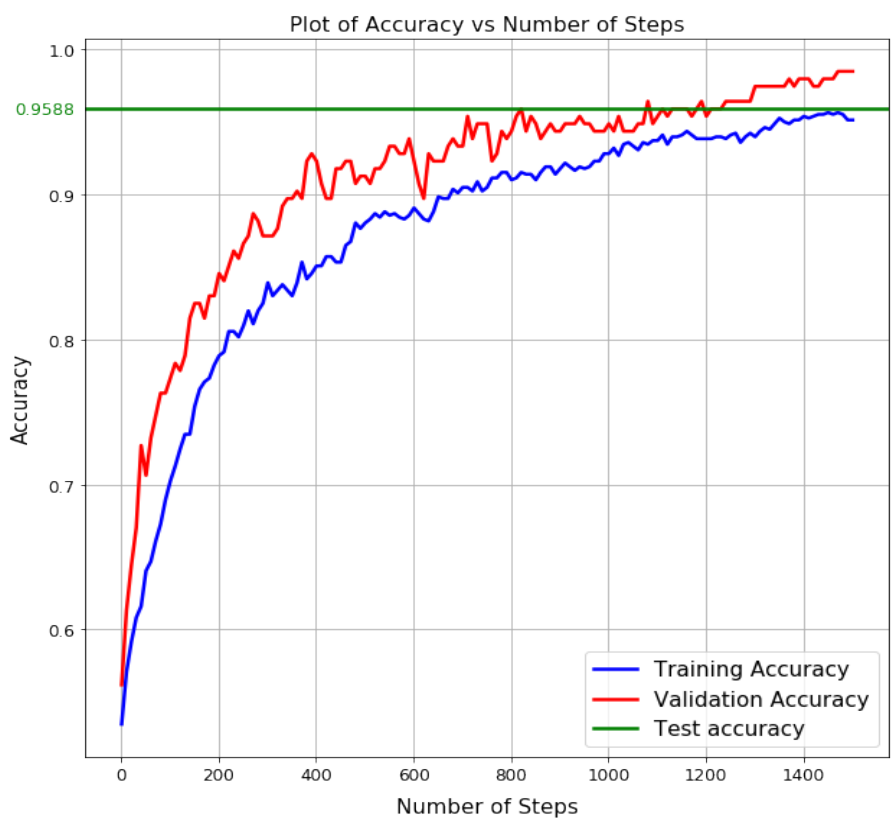
The model is trained by formulating the loss function as the binary softmax cross entropy loss with the predictions and ground truth labels. I used Stochastic Gradient Descent optimizer to minimize the loss function, with learning rate = 0.0001. All the weights and biases are initialized using Xavier initializer. To evaluate the performance, I computed the metric Accuracy using the predicted and ground truth labels (one hot encoded). During training, I also used three Image Augmentation techniques (mentioned in previous report) to introduce variance and tackle inadequacy of the training data.

I trained for a total of 1500 steps (or 23 epochs) with batch size of 12. After every 10th step, I evaluated the current model performance by accumulating the loss and accuracy using both entire train and validation dataset. Additionally, Early stopping criteria was also implemented, that monitors validation loss for identifying the region when overfitting occurs.

After training, we achieved a testing accuracy of **0.9588**, which shows that the model is reasonably able to distinguish between red round traffic signs and the negatives. Over the course of training, we didn't see the model to overfit at any point. On the contrary, the validation loss and accuracy curve was decreasing and increasing respectively.

The following plots shown the learning curve of the model and the testing accuracy.





If I had more time, I would consider experimenting with the hyper-parameters such as learning rate and batch size, and monitor the performance of the model. Experimenting with different optimizers such as Adam, Adagrad etc can also be explored. Since, I didn't have time to design other model architectures, I would try to outline other structures and inspect their performance on the training data. Finally, I would try to solve the original problem rather than the reformulated classification one, where I would predict bounding box regressors for the red round traffic signs, using Intersection over Union (IoU) metric. Therefore, in this case, I need to go back to Part 1 and modify the train and test datasets, to make it suitable for this purpose.