# Street Sign Recognition Using a Mixture of Synthetic and Actual Data

1st Jaskirat Singh Bhatia
*Student*
*University Of Waterloo*
Waterloo, Canada
j6bhatia@uwaterloo.ca

***Abstract*—Due to the increase interest in autonomous diving, automatic traffic signs detection and recognition is a very important field of computer vision. In this project, I used convolutional Neural Network to classify the street signs. The Network consisted of 7 convolutional layers, some of them followed with stride. In order to prevent the model from overfitting, I took the average of of the feature maps and added a Global Average Pooling layer. I performed classification on the GTSRB dataset which contained 39209 training images and 12630 testing images. The Program gave me an accuracy of 92% (With 6 epochs; that's all what my PC could handle)**

***Index Terms*—Convolutional Neural Network, Computer Vision, object recognition, object detection**

## I. Introduction

Traffic sign recognition plays a humongous role in guiding traffic regulations, varying between warnings, road condition information, traffic regulation and destination information [2]. To recognize these traffic sings in an image, there are two main steps in almost all methods: Detection and Classification [1]. It was pretty difficult to perform these step earlier because there was no publicly available dataset until the release of GTSRB(German Traffic Sign Recognition Benchmark) [3] and the GTSDB(German Traffic Sign Detection Benchmark) [4] in 2011 and 2013 respectively. After the release of these datasets, the research on Traffic Signs took a big leap.

### A. Traffic Sign Classification

There are many methods which have been previously used to classify traffic signs such as LDA, SVM, ANN and many more. [5] use OCR systems and perform pictogram-based classification. Authors have also made use of LDA to distinguish between traffic signs. Another widely used approach is to classify using Multi-Layer Perceptron. SVM(Support Vector Machines) are also largely adopted to classify the inner parts of road signs. Ensemble techniques such as Random Forest are also used to classify Signs. Neural Networks is the most used of all. In this project, I use CNN's with an extra layer called the GAP(Global Average Pooling) layer which uses the average of all feature maps and helps reduce overfitting [6]

## II. Properties of Traffic Signs

Traffic signs have unique features that make them look different from other objects[2]

- They have a 2-D shape which includes squares, triangles, circles, rectangles etc
- The color used are generally simple primary colors(RGB), and also yellow as it is easily identifiable by the driver [7]
- Each traffic sign has a unique color and so do the writings on the traffic sign. [7]
- The signs could hold either a figure or a character; or even both [7]

## III. Data Set

The dataset was obtained from GTSRB's website.

- Contained 39209 Training Data Images
- Contained 12630 Testing Data Images
- There were 43 different type of images
- There was a meta image for each different type of image

### A. Discovering Dataset Balance

It is important that the proportion of each street sign is equal in both the training and testing data for maximum accuracy. A simple histogram of counts of each sign in both the datasets will tell us if it is balanced or not. Figure 1

As we can see (fig Discovering Dataset Balance), the dataset is balanced.

### B. Visualizing Target Class Images

It is not a sample in the dataset, it is just a picture of sign The meta images shows each Street Sign with its actual Sign(Total of 43 images). Figure 2

### C. Visualizing Testing Images

Visualizing these images give us a better understanding of what the dataset actually looks like. Figure 3

## IV. Synthetic Data Collection

In my system, the classification stage relies completely on the CNN model. Due to the complexity of CNN structure, it is required to have as much labeled data as possible to train a reliable CNN model. For this purpose, I used synthetic traffic signs. [1]

In the street view images, there are also a lot of symbolbased traffic signs. However, their distribution is not uniform. For

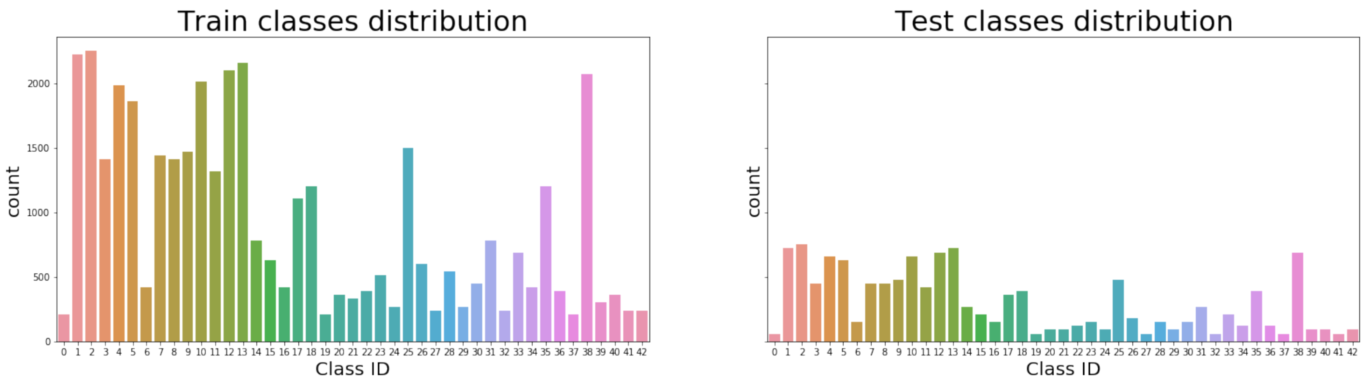

Fig. 1. Discovering Dataset Balance

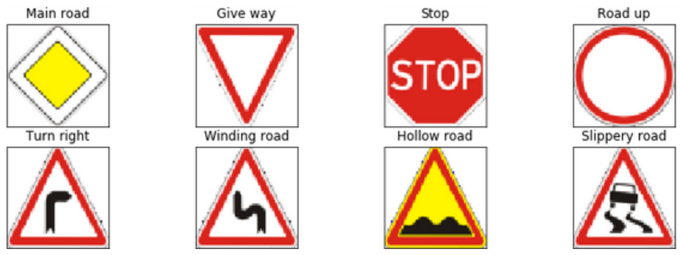

Fig. 2. Target Class Images

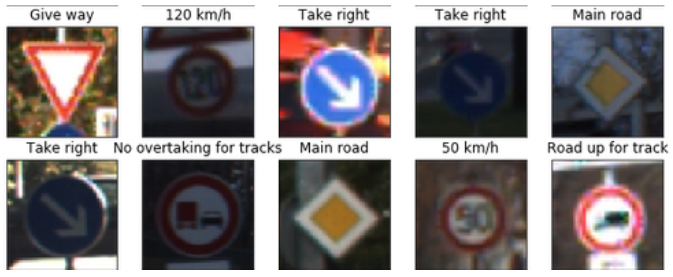

Fig. 3. Testing Images

example, speed limit signs are very common in real traffic scene, while the signs such as weight limit signs only exist in some special scenes. For the same reason, there are many subclass signs with different speed values in speed limit signs, and some signs with very low or high speed values are also not common in many scenes. To acquire more data with low cost and balance data quantity in each class at the same time, we also synthesize traffic sign images from standard sign templates. The pipeline of how to synthesize data is shown in Fig. 6. First, we get all standard signs that need to be classified. The standard signs are manually processed to add a mask channel which is important to synthesize the sign images with background. To simulate the viewpoint change in the real scene, random planar affine transformation is applied to standard signs. A planar affine transformation has the matrix representation of equation (3) with six parameters [142]

$$\begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} = \begin{bmatrix} a_{11} & a_{12} & d \\ a_{21} & a_{22} & d \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \tag{1}$$

or in the block form

$$x' = Hx = \begin{bmatrix} A & d \\ 0' & 1 \end{bmatrix} x \tag{2}$$

where A is a 2×2 non-singular matrix. However, the elements in A are the mixed results of geometry parameters such as rotation and scale, making it difficult to simulate. Fortunately, the affine matrix A can be decomposed as

$$A = \lambda R(\theta) R(-\phi) \begin{bmatrix} t & 0 \\ 0 & 1 \end{bmatrix} R(\phi) \tag{3}$$

where R(·) denotes a rotation matrix. The geometry meaning of each parameter can be found in [8].

From (1) and (3) we can control the six independent parameters, $dx, dy, \lambda, t, \theta, \phi$ to simulate geometry transformation. The bounding boxes of detected traffic signs in real scenarios include both the sign and background, especially for those non-rectangle signs. To model this property, we add background to the transformed sign generated in the previous stage. The background images are collected from real traffic scene without any signs. First, a patch of the same size as the sign is randomly cropped from the background image set. Then the sign image is added with the background image patch as following:

$$I_{out} = mask.I_{sign} + (1 - Mask).I_{background} \tag{4}$$

where $I_{out}, I_{sign}, I_{background}$ are output image, input sign image and background image patch respectively, and "." denotes element-wise multiplication. The mask is the mask channel matrix of the sign image, which has binary values. Given the variations in the time and weather conditions, we also simulate the lighting variation. To simplify this procedure, we only fuse the images with different lighting images as follow:

$$I_{out} = weightxI_{sign} + (1 - weight).I_{light} \quad (5)$$

where $I_{light}$ denotes the lighting image, and weight balance two images. Similar to background image patch, the lighting image is also randomly cropped from lighting image set which contains images with varying luminance. Finally, the synthetic images are blurred with Gaussian kernel of random size. Some synthetic images are shown in Fig 4. It can be seen that they have different appearances, which are crucial to train a network to recognize traffic signs with many variations.

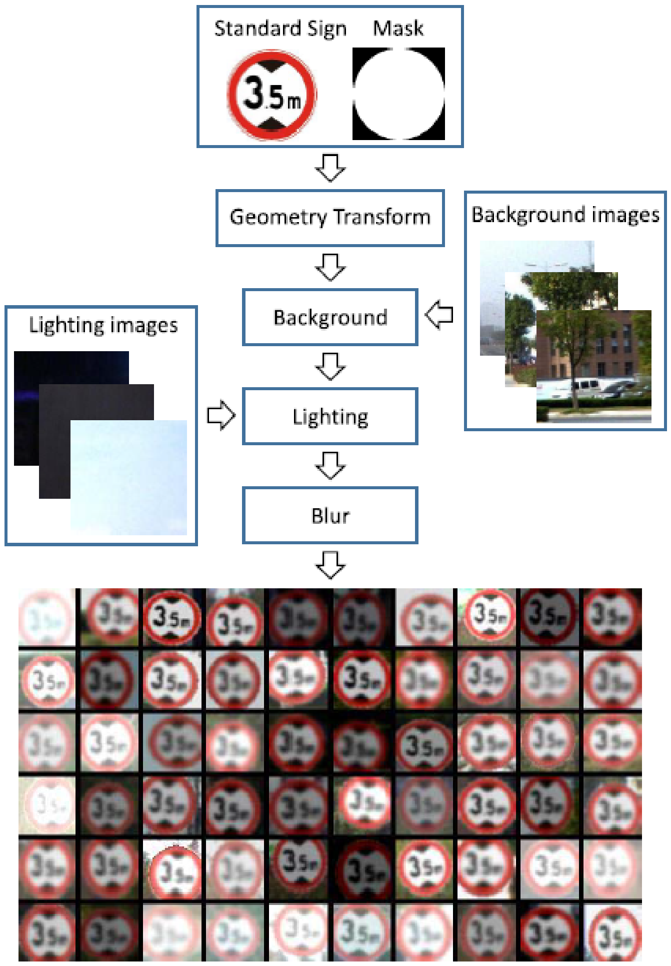

Fig. 4. Pipeline of synthesizing traffic signs

## V. Methodology

In this section, i will be explaining my approach to the problem.

### A. Convolutionary Neural Networks(CNN)

- It is a class of deep learning methods which has become dominant in various computer vision tasks and is attracting interest across a variety of domains [9]
- Inspired by the organization of animal vision cortex, it is used for data which has a grid pattern such as images [9]
- Composed of multiple building blocks, such as convolutional layers, pooling layers, and fully connected layers, and is designed to automatically adaptively learn spatial hierarchies of features through a backpropagation algorithm [9]

A basic CNN is shown in Image 5

### B. Implementation

The input to the Convolutionary Network will be a three dimensional object with dimensions:- height, width and color where color is R,G or B. Originally, the images were of varying sizes with their heights and widths varying from 60-90 All the images were reshaped to 60 X 60 so that the input was consistent; and this input was fed to the convolutional layer.

The Convolutional Layer extracts features maps by linear convolutional filters from input image which is then followed by some nonlinear activation functions (Sigmoid, Relu, tanh etc.). Then, extracted feature maps are passed through another layer, out of which two are followed by stride 2. In CNN, stride is used with Convolutional Layer for downsampling. A brand new set of feature map is then created by passing the filters over the output from first down-sampling. The amount by which the filter shifts is the stride. We are using stride 2, which means 2 positions to be skipped. Another benefit of using Strides is that it controls the filter which convolves around the input volume. When the kernel is sliding the input, stride is extensively used to calculate how many positions are to be skipped(2 in my case). Linear Rectifying Units (Relu) is one of the most vastly used activation function for hidden layer in deep learning. Relu has output 0 if the input is negative and if the input is greater than zero it gives us a value which is equal to the input. Hence, it does not reduce the size of network. The result of using Relu activation function is that it is way faster to train large network and also increases its nonlinear property. Global Average Pooling (GAP) used to minimize the number of parameters and to protect model from overfitting. The idea is to reduce filter size by simply taking average of whole feature map first introduced by [10]. Although, GAP is similar to max pooling layer but it perform more extreme type of dimension reduction. Where dimension h × w × d is reduced into the dimension size 1 × 1 × d. GAP reduce each feature map by simply taking the average of whole feature maps. As the last layer, also called classification layer we used softmax layer. It can be used for predicting hundreds and thousands of classes. The output of softmax is equal to the total number of fruit classes. As shown in figure 5, the network contains a total of seven convolutional layers, in which the second and third is followed by stride 2 and the final layer is followed by a GAP layer. Relu activation function is applied at the output of

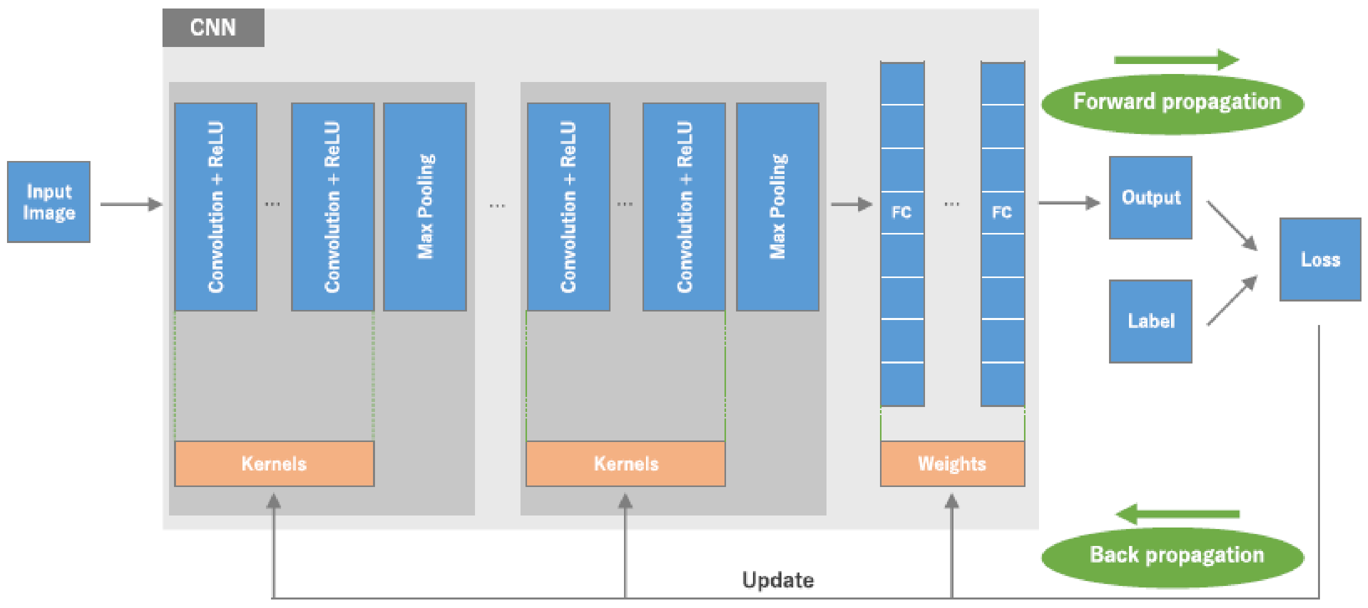

Fig. 5. Basic CNN

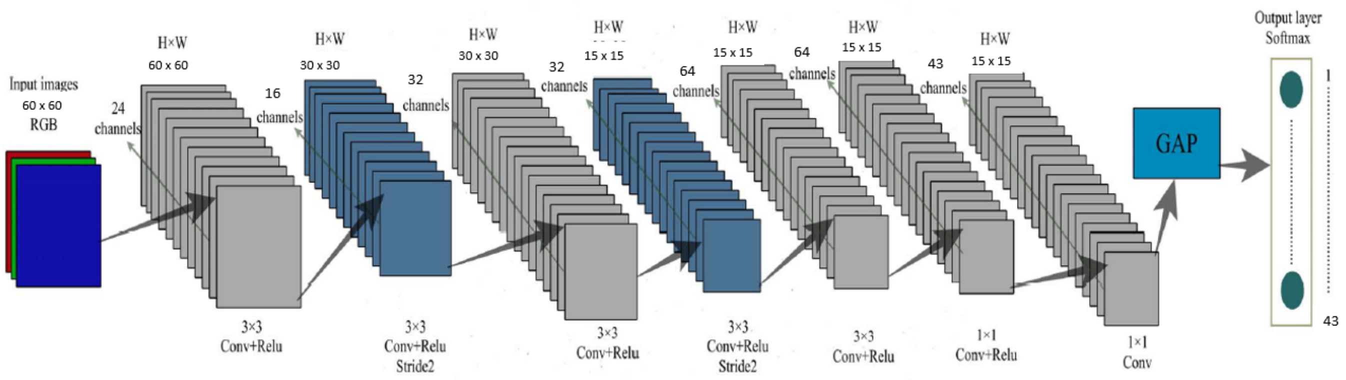

Fig. 6. Architecture Used

each convolutional layer. For faster processing, I batched the input with 128 images in each batch. The size of each image was a 60 × 60 grid. This input was then passed through the convolutional layer of 16 filters; each filter had a size 3 x 3 and then for dimension reduction, it was followed by a Stride 2 layer in the second convolutional layer. Then it was passed through the same series of layers a second time. But the layers now contained 32 kernels of size 3×3 as shown in layer 3 and 4. The output from these layers was then passed into one more convolutional layer which contained 64 kernels of size 3 × 3. To simplify the network further, the output from this layer was fed into a basic 1 × 1 convolutional layer from layer 6, and then it underwent another Relu activation unit. The idea here is to reduce the size of filters from 3×3 to 1×1 by taking the mean of all the feature maps first investigated by [10]. In the final layer, the resulting output is passed through GAP layer, then it is directly fed into softmax. The weights of the model are updated after every iteration by back propagation to reduce the loss. To achieve good performance, the model was trained on 6 epochs (I trained overnight and thats all my PC could do) and a total of 76,657 training parameter were used. When I was testing , a forward pass was applied on an input image and the highest score is predicted to be the sign category.

The Architecture is shown in Image 6

## *C. Language Used*

- I used Python to implement the project
- The library TensorFlow was used to implement the Neural Network
- Matplotlib was used to display the graphs, figures
- pandas was used to read the dataset
- code available here

| PCNN + GAP layer | | CNN + FC layers | | CNN + FC layers + Dropout | |
|---|---|---|---|---|---|
| Layer Type | Dimensions | Layer Type | Dimensions | Layer Type | Dimensions |
| Convolutional Layer | 16X3X3 | Convolutional Layer | 16X3X3 | Convolutional Layer | 16X3X3 |
| | | Convolutional Layer | 16X3X3 | Convolutional Layer | 16X3X3 |
| Convolutional Layer Stride 2 | 16X3X3 | Max-Pooling Layer (2X2) | | Max-Pooling Layer (2X2) | |
| | | | | Dropout 0.10 | |
| Convolutional Layer | 32X3X3 | Convolutional Layer | 32X3X3 | Convolutional Layer | 32X3X3 |
| | | Convolutional Layer | 32X3X3 | Convolutional Layer | 32X3X3 |
| Convolutional Layer Stride 2 | 32X3X3 | Max-Pooling Layer (2X2) | | Max-Pooling Layer (2X2) | |
| | | | | Dropout 0.15 | |
| Convolutional Layer | 64X3X3 | Convolutional Layer | 64X3X3 | Convolutional Layer | 64X3X3 |
| | | Convolutional Layer | 64X3X3 | Convolutional Layer | 64X3X3 |
| Convolutional Layer | 64X3X3 | Max-Pooling Layer (2X2) | | Max-Pooling Layer (2X2) | |
| | | | | Dropout 0.20 | |
| Convolutional Layer | 43X1X1 | Flatten Layer | | Flatten Layer | |
| GAP Layer | | FC Layer | 256 | FC Layer | 256 |
| Softmax(Activation) | 43 | Softmax | 43 | Softmax | 43 |

TABLE I
MODEL DESCRIPTION OF THREE DIFFRENT CLASSIFIERS; PURE CONVOLUTIONAL NEURAL NETWORK (PCNN) WITH GLOBAL AVERAGE POOLING (GAP) LAYER, CNN ALONG WITH FULLY CONNECTED (FC) LAYER AND CNN WITH FC LAYER AND DROPOUT.

## VI. RESULTS

### A. Analyzing results from each method

The experiment shows that CNN model which has FC layers lacks performance. One possible reason for this could be that the neurons in the FC layers have connections with all activation in the previous layers. Adding FC layers may give us better accuracy for image classification but the complexity and model parameters increased significantly. Network with FC layer's accuracy is 89% and total number of wrongly predicted images are 108. Further,to avoid overfitting, dropout used in third network to drop connection of some layers during training. Although, dropout avoid overfitting and improve accuracy but the number of parameters remains same as CNN without dropout. Average accuracy slightly increased to 90% but the wrongly predicted images decreased to 101. As presented , PCNN with GAP not only achieved highest accuracy but also the number of wrongly predicted images are less than other two models.

The Loss of PCNN with GAP is shown in Figure 8

The Accuracy of PCNN with GAP is shown in Figure 7

### B. Performance On GTSRB

### C. Confusion Matrix

A confusion matrix is also known as the error matrix. [11] It is a table which compares the predicted values to the actual values, and is generally used to describe the performance of a classification model. It can also be used to visualize these

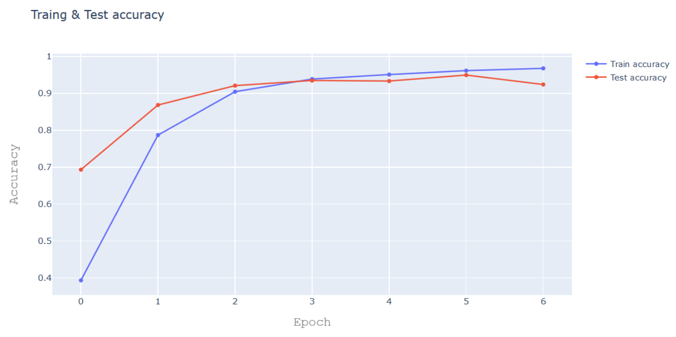

Fig. 7. Accuracy

values using an ROC. [12] To construct a Confusion matrix, the correct and the incorrect values in each class are counted and summarized. Figure 9

## VII. EXPERIMENTS

### A. Wrong Predictions

To explore the model further, I took a sample of images which were classified wrong. Figure 10

### B. Grad - CAM

It stands for Gradient-weighted Class Activation Mapping, [14] It uses the gradients from any target concept( for example

|  | Testing Accuracy | Training Accuracy | Testing Loss | Training Loss |
|---|---|---|---|---|
| Epoch 1 | 69.3% | 39.35% | 0.93 | 2.04 |
| Epoch 2 | 86.6% | 78.7% | 0.48 | 0.67 |
| Epoch 3 | 92.1% | 90.4% | 0.33 | 0.31 |
| Epoch 4 | 93.4% | 93.8% | 0.25 | 0.20 |
| Epoch 5 | 94.9% | 95.3% | 0.18 | 0.12 |
| Epoch 6 | 92.4% | 96.7% | 0.43 | 0.11 |

TABLE II
ACCURACY WITH EPOCHS

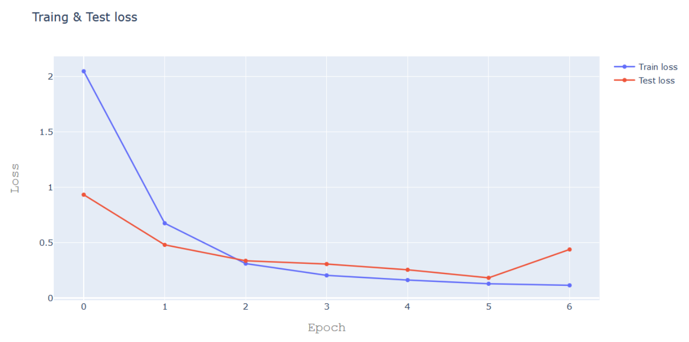

Fig. 8. Loss

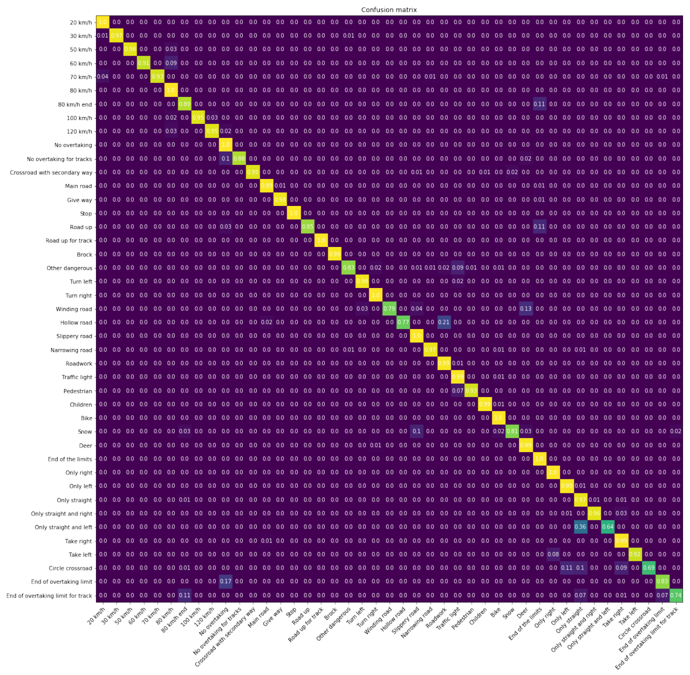

Fig. 9. Confusion Matrix

a cat in a classification network) flowing into the final convolutional layer to produce a coarse localization map highlighting the important regions in the image for predicting the concept

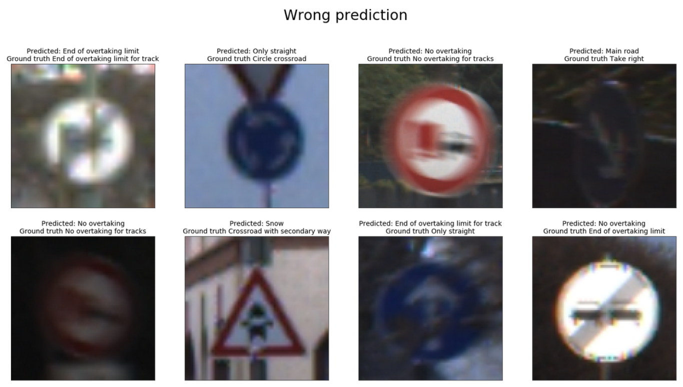
Wrong prediction
Predicted: End of overtaking limit
Ground truth End of overtaking limit for track
Predicted: Only straight
Ground truth Circle crossroad
Predicted: No overtaking
Ground truth No overtaking for tracks
Predicted: Main road
Ground truth Take right
Predicted: No overtaking
Ground truth No overtaking for tracks
Predicted: Snow
Ground truth Crossroad with secondary way
Predicted: End of overtaking limit for track
Ground truth Only straight
Predicted: No overtaking
Ground truth End of overtaking limit

Fig. 10. Wrong Predictions

Grad - CAM is applicable to

- CNN's with fully connected layers [13]
- CNN's used for structured output [13]
- CNN's used in tasks with multimodal inputs [13]

After performing Grad - CAM the output was : Figure 11

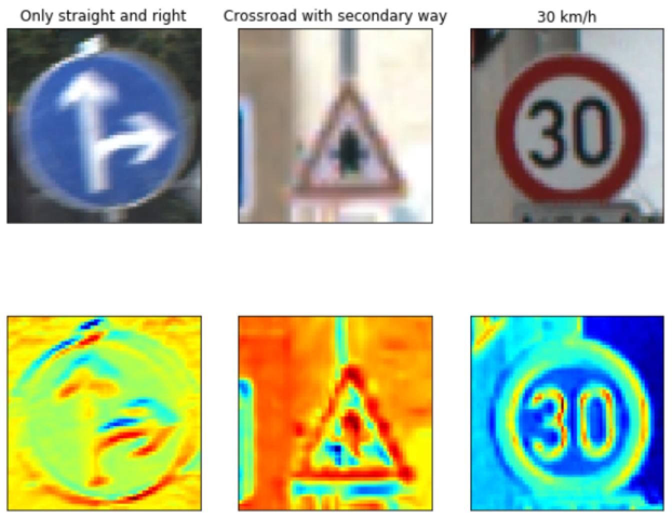

Fig. 11. Grad - CAM

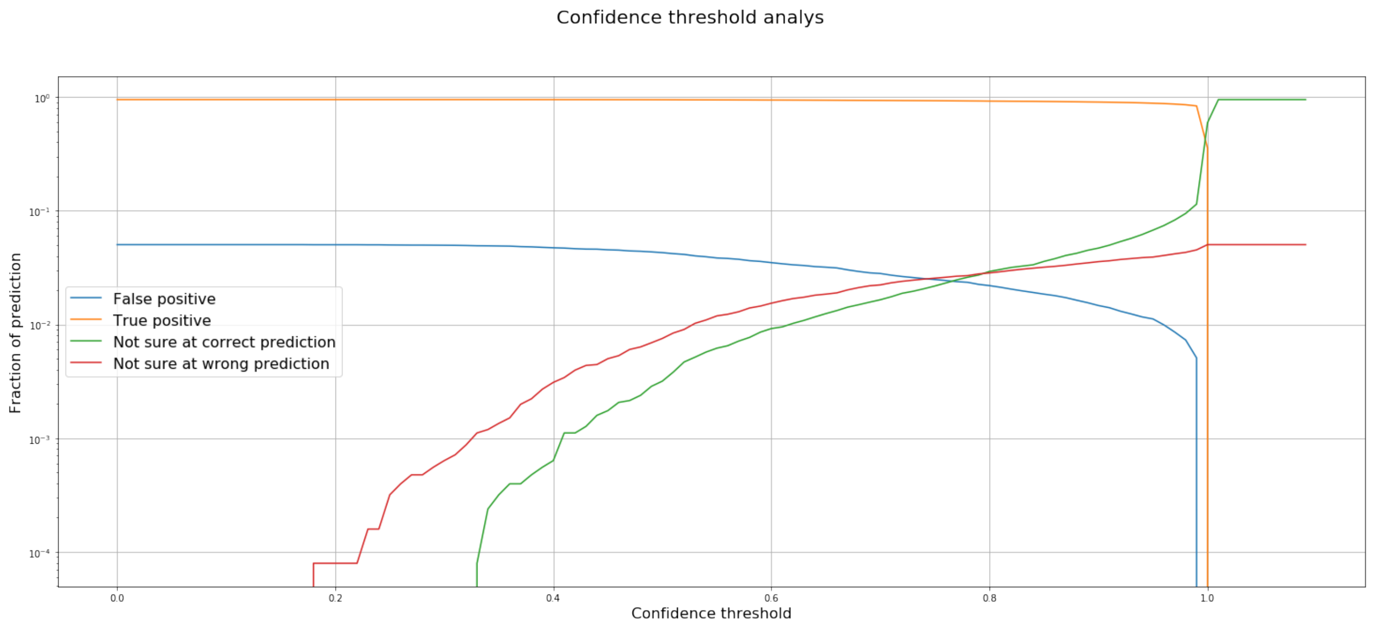

Fig. 12. Confidence Interval

## C. *Confidence Interval*

A confidence interval is calculated from the observed data and is a type of estimate. [15] It gives us an interval where some value might lie with a high probability (95%). In more formal terms, "they allow interpretation of a given result to be extended to the world of possible replications of the experiment, given the random nature of the data" [12]

The confidence interval for this experiment is in Figure 12

## VIII. Conclusion

I presented a novel approach to improve street sign classification using Pure Convolutional Neural Network (PCNN) with Global Average Pooling (GAP). We have found that using GAP layer, rather than Fully Connected (FC) layer, provides superior performance and overcome overfitting problem. To train a reliable network,street view data and synthetic images are used to generate larger number of data with low cost. My system gets the state-of-the-art result on a challenging new data set. PCNN can be successfully trained to classify various types of street sign images. This makes it possible to use same approach for both object recognition and multi-class image classification.

## References

[1] Hengliang Luo, Yi Yang, Bei Tong, Fuchao Wu, and Bin Fan "Traffic Sign Recognition Using a Multi-Task Convolutional Neural Network" IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, VOL. 19, NO. 4, APRIL 2018, pp1100–1111

[2] Joshua Paolo N. Acilo, Allyana Grace S. Dela Cruz, Michael Kevin L. Kaw, Maynald D. Mabanta, Vemeni Grace G. Pineda, Edison A. Roxas "Traffic Sign Integrity Analysis Using Deep Learning", 2018 IEEE 14th International Colloquium on Signal Processing and its Applications (CSPA 2018), 9 -10 March 2018, Penang, Malaysia, pp 107–112

[3] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "The German traffic sign recognition benchmark: A multi-class classification competition," in Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN), Jul. 2011, pp. 1453—1460.

[4] S. Houben, J. Stallkamp, J. Salmen, M. Schlipsing, and C. Igel, "Detection of traffic signs in real-world images: The German traffic sign detection benchmark," in Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN), Aug. 2013, pp. 1—8.

[5] Long Chen, Qingquan Li, Ming Li and Qingzhou Mao, "Traffic Sign Detection and Recognition for Intelligent Vehicle", 2011 IEEE Intelligent Vehicles Symposium (IV) Baden-Baden, Germany, June 5-9, 2011, pp 908–913

[6] Asia Kausar, Mohsin Sharif, JinHyuck Park and Dong Ryeol Shin, "Pure-CNN: A Framework for Fruit Images Classification", 2018 International Conference on Computational Science and Computational Intelligence (CSCI), pp. 404–408

[7] A. de la Escalera and M. A. Salichs, "Road Traffic Sign Detection and Classification," IEEE Transactions on Industrial Electronics, vol. 44, issue 6, Dec, 1997, pp. 848 – 859.

[8] R. Hartley and A. Zisserman, Multiple View Geometry in Computer Vision. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[9] Rikiya Yamashita and Mizuho Nishio and Richard Kinh Gian Do and Kaori Togashi, "Convolutional neural networks: an overview and application in radiology", Insights into Imaging (2018) 9: pp. 611-–629

[10] Jifeng Dai, Yi Li, Kaiming He, and Jian Sun. R-fcn: "Object detection via region-based fully convolutional networks.", Advances in neural information processing systems, 2016, pp 379-–387.

[11] Ariza-López F.J., Rodríguez-Avi J., Alba-Fernández M.V., "Complete Control of an Observed Confusion Matrix", IGARSS 2018 - 2018 IEEE International Geoscience and Remote Sensing Symposium, 22-27 July 2018, pp. 1222-1225

[12] John Kerekes, "Receiver Operating Characteristic Curve Confidence Intervals and Regions", IEEE GEOSCIENCE AND REMOTE SENSING LETTERS, VOL. 5, NO. 2, APRIL 2008, pp. 251–255

[13] Ramprasaath R. Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, Dhruv Batra, "Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization" 2017 IEEE International Conference on Computer Vision (ICCV), 22-29 Oct. 2017

[14] Pietro Morbidelli, Diego Carreray, Beatrice Rossiy, Pasqualina Fragnetoy, Giacomo Boracchi, "Augmented Grad-CAM, Heat-Maps Super

Resolution Through Augmentation“, ICASSP 2020, Jan 27, pp. 4067–4071

[15] Brahim Hamadicharef, “Frequentist Versus Bayesian Approaches for AUC Confidence Interval Bounds“, 10th International Conference on Information Science, Signal Processing and their Applications (ISSPA 2010), pp. 341–344