Traffic Sign Classification for Self Driving Cars

Chandan Saha - cs1156 , Ramakanth Vemula - rv356 , Mayank - um45
{chandan.saha,ramakanth.vemula ,mayank.ru }@rutgers.edu

Abstract

Traffic signs are large collections of wide range signs classes in real-world environment .Road signs are designed and easily detected by human drivers in real-world scenario. However, for computers , traffic sign classification still seems to pose difficult problem because of different weather conditions, several viewing angles of same sign image, varying speed of the vehicle and other visibility blocks during traffic. In practice, there is a need for traffic signs to be classified quickly and with a high degree of accuracy, such that the algorithms should be viable for use in autonomous vehicles. We have implemented state of art machine learning algorithms to classify couple traffic sign datasets using GPU as well as CPU to benchmark with real-time processing time. In our research , Convolutional neural network (CNN) outperforms most other classification algorithms.

1. INTRODUCTION

The future of transportation is trending towards autonomous self driven vehicles. Advanced Driving assistant system(ADAS) have received more and more attention with innovations on self driven cars . Being a real time system, processing time is as important as the accuracy of detection and classification.

People can usually recognize the variety of existing road signs with a high degree of accuracy, but the rich context information, weather conditions and multiple views of the same traffic sign poses a very convoluted task for computers.

In this project, we try to explore the different features of traffic sign images and use state of the art machine learning algorithms to classify traffic signs for autonomous cars. In our analysis we also discussed impact of computer architecture (CPU vs GPU), algorithm, and training set size on the accuracy and speed of traffic sign classification will be explored. The Convolutional neural network was implemented on a GPU for traffic sign classification: Based on our experiment testing time for the CNN on a GPU was 3.5 ms/image, which was 5x as fast as running CNN on a CPU.

2. METHODS

Image classification is well reasearched area in computer vision and involves some commonly used features such as color histograms , histograms of oriented gradients (HOG) and Haar-like features. Popular classification methods explored for image classification are Neural Networks[7][8], Linear Discriminant Analysis (LDA)[1] and Support Vector Machines(SVM)[9].

2.1 Related Work

In literature , Hastie et al uses LDA linear discriminant analysis (LDA) to train traffic sign model and gives surprisingly good results in practice despite its simplicity .In German Traffic Sign Recognition Benchmark (GTSRB) held at IJCNN 2011, two groups using CNNs obtained classification accuracy of 98 % [1].Another interesting approach to solve image classification problem , where authors used transformed images into grey scale using SVM and then use convolutional neural networks with fixed and learnable layers for detection and recognition and achieves accuracy of 99.73 % in ”Danger” category on GTSRB dataset [10].

We have also looked into some of the successful case studies of Convolutional neural network . LeNet is one of the first and best known architecture of CNN . We have done some close study on the framework developed by Karen Simonyan and Andrew Zisserman which is known as VGGNet , which contributed in proving depth of network is a critical component for good performance.Our model is inspired from VGGNet where we are using depth of layers to train and test the model in LISA and GTSRB detaset [2].

<u>Base Evaluation Criteria</u>

Based on our literature survey , LISA dataset has processing time of 4ms / per image and accuracy of 98.75 % given 8 sign classes using convolutional neural network trained using GPU . On other hand , state of art machine learning algorithms gives 98.79 % based on results generated using 8 layer DNN during the German Traffic Sign Recognition Benchmark (GTSRB) held at IJCNN 2011.Team IDSIA: Committee of CNNs ran 25 epoch to train each DNN which takes about 2 hours .

ACM ISBN 978-1-4503-2138-9.
DOI: 10.1145/1235

2.2 Dataset,Preprocessing, Hardware

The LISA (Laboratory for Intelligent and Safe Automobiles) Traffic Sign Dataset is a collection of annotated images and videos containing traffic signs and videos containing traffic signs. It is comprised of over 6,000 frames that contain over 7,000 signs of 47 different types. image dimensions varies from 6*6 to 167*168 pixels.

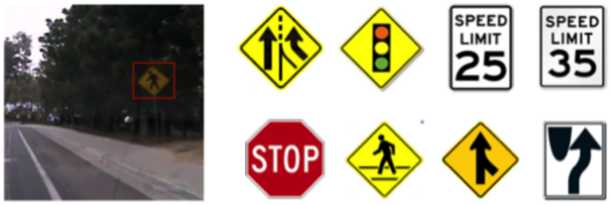


Figure 1: Image from the LISA dataset (left) and sample of 8 signs out of 16 used in our experiment (right).

Wide range of dimensions make LISA dataset ideal for real time scenario . We have focused on traffic sign classification with atleast 250 sample images for each sign type(For e.g. ”Prohibitory”, ”Mandatory” and ”Danger” etc) so that convolutional neural network has reasonable data to get trained. In total ,we have trained our model on 16 different classes from LISA dataset .Dataset is randomly divided into training and test sets using 80:20 rule . For purpose of classification , we have converted all images to 32*32 pixels and to grey scale .

We also compare our classification model with state-of-the-art machine learning algorithms on larger available dataset. These classification results were in context with German Traffic Sign Recognition Benchmark (GTSRB) held at IJCNN 2011. German Traffic dataset contains 39000 sign images of 43 different classes .

The relative class frequencies of the classes are shown in Fig. 2.

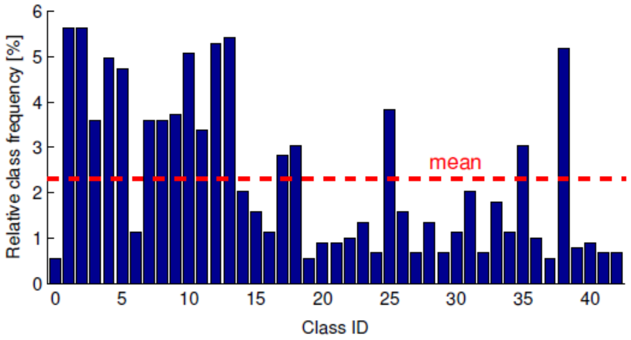


Figure 2: Relative class frequencies in GTSRB dataset.

Some of sample difficult images are given below ,

Figure 3 :- Difficult Sample Images

We have used CUDA parallel computing platform to train CNN model . An Intel i7-3770K processor and Nvidia GeForce 940 were used to run all of the algorithms.

3. CONVOLUTIONAL NEURAL NETWORK

Convolutional Neural Networks can achieve a very high degree of accuracy for a problem like Traffic Sign Classification. This maybe so because the different images used during training and testing of the model tend the contain similar features - orientations, aspect ratios and color composition to name a few.

The effectiveness of a Convolutional Neural Network depends on the architecture of the CNN. Competitions such as the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) are held worldwide, where people showcase the architectures they have implemented to achieve good results. A few of the popular architectures which have resulted are - AlexNet, LeNet, VGGNet, etc.

VGGNet is a fairly simple architecture which manages to produce good results. We chose to implement a variation of VGGNet to our Traffic Sign Dataset. A model of architecture we constructed is given below.

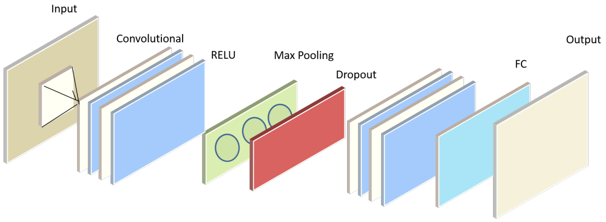


Figure 4 :- Architecture of Classification Model

Some of the key aspects of our model are :

- Before we could proceed with CNN, we had to do a bit of image preprocessing. The Traffic Sign dataset consists of images which were to be cropped out of still frames of a video. The images as a result were of different sizes and were rescaled to 32 x 32. Also, the dataset was a mixture of greyscale and color images. During preleminary experiments, we got bad results with the original dataset and decided to change all the images to greyscale. Performance marginally increased so we decided to use grey scale images for all further experiments.

- Filtering is done during the convolution phase of CNN. We use 32 filters during each convolution. Given the small size of images, we decided to use a filter size of 3x3 with strides of 1 pixel.Using a stride of 1 allows us to leave all spatial down-sampling to the POOL layers.

- The dataset used by the developers of VGGNet consist of images with a good resolution. Also, the images contain a multitude of objects with different features. Constrastingly the traffic sign images are not as complex.

  We know that each Convolution Layer helps in extracting some features of an images. The initial convolution layer extracts simple features and the complexity of the features increases for the subsequent layers in the model. As mentioned above, since the traffic images are not composed of a very many features, good accuracy can be achieved by using fewer convolution layers.

- Output of the convolutional layer is given as input of Max-pooling which is a form of non-linear down-sampling. In max pooling layer we take small rectangle block from previous convolutional layer's output and subsamples the it to get a single maximum value . The max pooling filter used by us is of side 2 x 2 and proceeds in strides of 2.

  Max-pooling is useful for two reasons. Firstly, it reduces computation for upper layers by eliminating non-maximal values,. Secondly, it provides a form of translation invariance.

- Activation Function : For the first two convolutional layers , we are using RELU activation function applied on feature maps from input layer .

  $$h = max(a)$$

  Advantage of using RELU is sparsity in the data and a reduced likelihood of vanishing gradient . In case of traffic sign images , images are not very complex like face detection and sparsity arises when $a \leq 0$ compared to sigmoid activation function which is likely to generate some non-zero value resulting in dense representation.
  In last convolution layer , we are using *SoftMax* activation function because it measures certainty at output layer by converting raw feature values into posterior probability .

- Loss Function : Evaluation of our model are done using categorical cross entropy . In this training model is being optimized based on two distribution probability of events. Mathematically the function computes ,

  $$H(p,q) = -\sum_{x} p(x)log(q(x))$$

  where p(x) is true distribution q(x) is coding distribution .The above loss function is applied with adaptive learning rate method of gradient descent called ADADELTA.

- Fully Connected Layer : Finally , after multiple convolutional and pooling layer input is given to a Fully Connected layer where every single neurons(features) are connected with each other. In case LISA dataset , output layer contains 16*1 vector , one for each sign classes.

The architecture of our CNN is as follows :

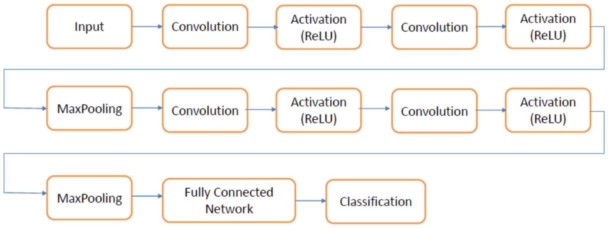
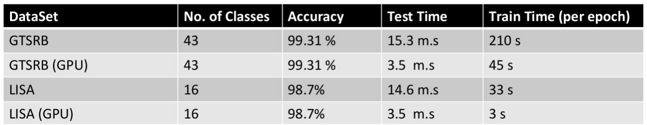


Figure 5: The Flow of our Architecture

Our model as shown above consists of two Convolution Layers initially each of which is followed by RelU Activation. This is followed by a Max Pool layer to reduce computation for further layers. The previous pattern is repeated once more - 2 convolution layers (RelU Activation) and a Max Pool layer. This is followed by a fully connected network and thereafter the Classification phase.

4. RESULTS AND DISCUSSION

The following Table 1 gives the accuracy, training time and testing time of the CNN model.



DataSet	No. of Classes	Accuracy	Test Time	Train Time (per epoch)
GTSRB	43	99.31 %	15.3 m.s	210 s
GTSRB (GPU)	43	99.31 %	3.5 m.s	45 s
LISA	16	98.7%	14.6 m.s	33 s
LISA (GPU)	16	98.7%	3.5 m.s	3 s

Table 1 : CNN results

We see a considerable difference between the training and testing time of CPU and GPU Convolutional Neural Network. With the German Traffic Sign dataset we got an accuracy of 99.31% and with LISA dataset 98.7%. These differences are due to the imaging conditions like weather, speed of the vehicle, occlusions and light conditions.

In the case of using GPUs we get a significant advantage over using CPUs because we are able to accelerate computations with float32 data-type. The computations such as Matrix multiplication, convolution, and large element-wise operations can be accelerated to upto 5-50x on GPU than on CPU. GPU performance is a trade-off between the advantages listed above and disadvantages like, summation of tensors on rows/columns might be a little slow on GPU than on a CPU and also copying large data can be a little slower.[6] Now we show the accuracy graph depending on the number of iterations over training data.

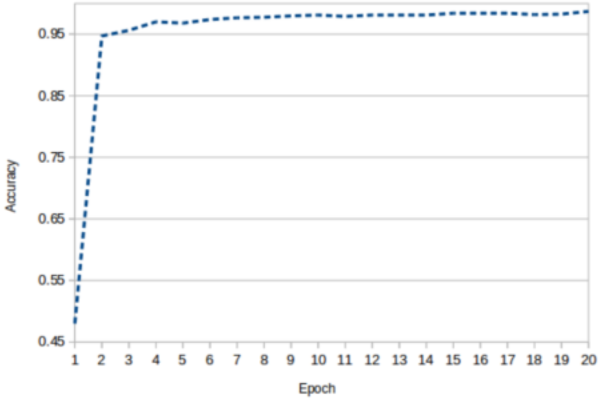
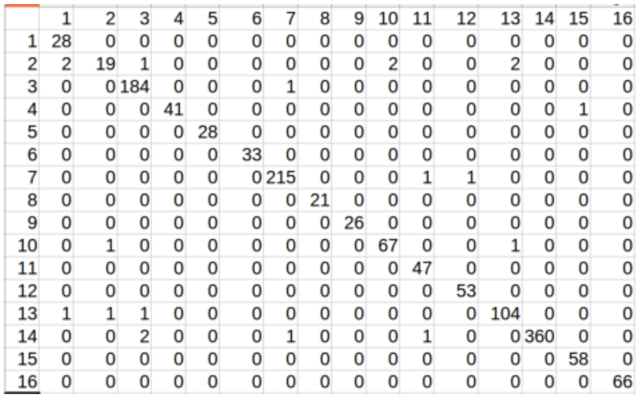


Figure 6: Accuracy vs Epoch

We got the maximum accuracy after 20 epoch, these may depend on the training data size and varies with the number of classes and number of training images. Next we got the following confusion matrix.



	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	2	19	1	0	0	0	0	0	0	2	0	0	2	0	0	0
3	0	0	184	0	0	0	1	0	0	0	0	0	0	0	0	0
4	0	0	0	41	0	0	0	0	0	0	0	0	0	0	1	0
5	0	0	0	0	28	0	0	0	0	0	0	0	0	0	0	0
6	0	0	0	0	0	33	0	0	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	215	0	0	0	1	1	0	0	0	0
8	0	0	0	0	0	0	0	21	0	0	0	0	0	0	0	0
9	0	0	0	0	0	0	0	0	26	0	0	0	0	0	0	0
10	0	1	0	0	0	0	0	0	0	67	0	0	1	0	0	0
11	0	0	0	0	0	0	0	0	0	0	47	0	0	0	0	0
12	0	0	0	0	0	0	0	0	0	0	0	53	0	0	0	0
13	1	1	1	0	0	0	0	0	0	0	0	0	104	0	0	0
14	0	0	2	0	0	0	1	0	0	0	1	0	0	360	0	0
15	0	0	0	0	0	0	0	0	0	0	0	0	0	0	58	0
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	66

Matrix 1 : Confusion matrix of Lisa Dataset

Here we have the following classes:
Label name: speedLimit45 label: 1
Label name: speedLimitUrdbl label: 2
Label name: signalAhead label: 3
Label name: laneEnds label: 4
Label name: speedLimit30 label: 5
Label name: stopAhead label: 6
Label name: pedestrianCrossing label: 7
Label name: schoolSpeedLimit25 label: 8
Label name: school label: 9
Label name: speedLimit25 label: 10
Label name: yield label: 11
Label name: merge label: 12
Label name: speedLimit35 label: 13
Label name: stop label: 14
Label name: addedLane label: 15
Label name: keepRight label: 16

We observed that our model has confusion on similar traffic signs such as speedLimit45, speedLimitUrdbl and speedLimit35. Just like stated before, these errors are due to bad imaging conditions. To get rid of these errors we take a look at the sequence of images that we will get while driving a vehicle and select the best possible image. We got a similar confusion matrix for GTSRB dataset but as we took 43 classes of the dataset we are unable to produce the matrix here. We can see the following Images that got misclassified.

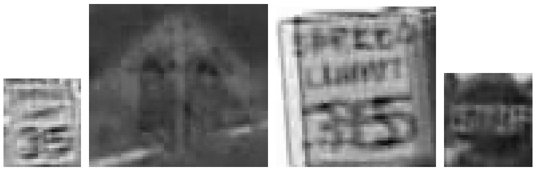


Figure 7 :Misclassified images

As mentioned above, these misclassified images are due to poor lighting conditions, occlusions, small size of image and blurring due to speed of vehicle. These can be classified correctly by selecting the image from the sequence of image we get while driving towards the traffic sign.
Additionally we wanted to test our data on new signs which will applicable in real world when there is a new traffic sign introduced in the city. We got impressive results in terms of training time and accuracy. We were able to achieve accuracy in between 98.6%-98.7%. Also the training time per epoch just increased by 0.2 seconds per epoch. So in real world this CNN should work almost perfectly given a good camera with high fps.

5. CONCLUSION

In general, this project gives insight to the problem of traffic sign classification in real world. Our algorithm would be fairly accurate when applied to an autonomous car.
Even though we had very few images our classifier was able to get a very high accuracy of 98.7% with 16 types of traffic signs. Given more training images our classifier will work well even with more types of traffic signs. We also concluded that using a GPU enhances our classifier's training and testing time, depends on the computation power of the GPU. This demonstrated the power of parallel computing for classification. The testing time is 3.5 ms/image and can be easily applied to a live video feed.

6. FUTURE WORK

In this project we deal only with classification of the given image of the traffic sign and although we achieved a good accuracy, the real world implementation of traffic sign recognition would deal with detection of the traffic sign and then classifying the image detected. So, in order to do this we need a detection algorithm which is as good as our classifier because our classifier would be of no use if the traffic sign bounding box is not detected by the algorithm.
Also in order to make our classifier more robust we need a larger training dataset consisting of images in every type of imaging condition which would be a better representative of our testing images.

7. REFERENCES

[1] Stallkamp, J., et al. ”Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition.” Neural networks 32 (2012): 323-332.

[2] Simonyan, Karen, and Andrew Zisserman. ”Very deep convolutional networks for large-scale image recognition.” arXiv preprint arXiv:1409.1556 (2014).

[3] P Sermanet , Y LeCun . Traffic Sign Recognition with Multi-Scale Convolutional Networks .

[4] Ivan Filkovic.Traffic Sign Localization and Classification Methods: An Overview.

[5] Lisa Lab. ”LISA Traffic Sign Dataset” http://cvrr.ucsd.edu/LISA/lisa-traffic-sign-dataset.html

[6] Deeplearning using Theano - deeplearning.net/software/theano/tutorial/using_GPU

[7] Torresen, J, Bakke, J, and Sekanina, L. Efficient recognition of speed limit signs. In Intelligent Transportation Systems, 2004. Proceedings. The 7th International IEEE Conference on, pages 652 âĂŞ 656, 2004.

[8] Nguwi, Y.-Y and Kouzani, A. Detection and classification of road signs in natural environments. Neural Computing and Applications, 17:265âĂŞ289, 2008. 10.1007/s00521-007-0120-z.

[9] Lafuente-Arroyo, S, Gil-Jimenez, P, Maldonado-Bascon, R, Lopez- Ferreras, F, and Maldonado-Bascon, S. Traffic sign shape classification evaluation i: Svm using distance to borders. In

Intelligent Vehicles Symposium, 2005. Proceedings. IEEE, pages 557 âĂŞ 562, 2005.

[10] Y Wu , Y Liu, j Li ... . Traffic Sign Detection based on Convolutional Neural Networks . Proceedings of International Joint Conference on Neural Networks, Dallas, Texas, USA, August 4-9, 2013 .