Object Detection of Traffic Signs using Faster Region-based Convolutional Neural Networks (Faster R-CNN)

Siddharth Thoviti [1], Naga Venkata Rama Sai Sri Harsha Bonala [2]

[1,2] Department of Computer Science & Information Management, Asian Institute of Technology, Bangkok, 12120, Thailand

[1] st121362@ait.ac.th

[2] st121327@ait.ac.th

Abstract **- Traffic Sign detection is the key-component involved in the fields of self-driving cars, autonomous vehicles. Although there are many deep learning techniques that are available for the object recognition such as RCNN, Faster-RCNN, YOLO, we apply the Faster Region-based Convolutional Neural Networks (Faster R-CNN) combined with ResNet50 feature extraction to the benchmark German Traffic Sign detection and provide the results obtained. We evaluate the model using the COCO metrics which are Average Precision (AP) and Average Recall (AR). The code is available in the github repository** **here**

Keywords - Object Detection, Deep Learning, Convolutional Neural Networks, RCNN, Faster-RCNN, Average Precision, Average Recall.

I. INTRODUCTION

Object detection which is one of the goals in the field of computer vision requires the understanding of the scenario present in the image. Traffic Sign Object Detection is an essential part in autonomous vehicles, traffic surveillance and self-driving cars which requires the identification of sign location in the frame and then classifying the sign accurately and the vehicle has to adjust according to the meaning behind the sign.

Traffic Signs are most important in terms of reducing accidents and hence they need to be identified and help in decision making in a much faster and accurate manner. They need to be identified by both pedestrians and the drivers inside the vehicle during both day and night.

But still, training models and making the model predict exactly what the sign indicates is still an expensive task considering the data size with all the images with different resolutions, sizes, scales, occlusions and light issues. Also, the speed in which a vehicle moves, the sensors and cameras need to identify the signs within moments to trust and avoid damage. Also, there can be more than one sign in the image which complicates the model.

Now, recently there have been many techniques that use the help of convolutional neural networks that are used in object detection. A few of which include Region-based Convolutional Neural Networks (R-CNN), Fast RCNN, Faster RCNN, Region-based Fully Convolutional Network (R-FCN), Single Shot Detector (SSD) and You Only Look Once (YOLO). Now among these, there might be a dilemma on which one to use and hence we evaluate the models using few metrics which include accuracy, precision, recall, average precision, average recall and also running and memory times of the model to predict.

We use the Faster R-CNN technique to address this issue and develop a model that can identify the traffic signs in the image and classify them with their corresponding classes. We use the German Traffic Sign detection data with Faster RCNN combined with ResNet50 feature extraction.

The rest of the paper is organized as follows. Section-II reviews the related work of traffic sign detection. Section-III includes the methodology and experiments conducted using the technique mentioned. Section-IV evaluates the developed model using the metrics. Section-V previews the results obtained from the model developed using the images predicted and with original images. Section-VI concludes and describes the future work required.

II. RELATED WORK

A. Traffic Sign Detection

Various approaches have been studied for traffic sign detection systems. Traditionally, techniques such as Histogram of Oriented Gradients (HOG), Scale invariant Feature Transform (SIFT) and Local Binary Patterns (LBP). Also, many machine learning techniques have also been implemented ranging from Support Vector Machines (SVM), Logistic Regression (LR) and Random Forests (RF). Recently, with the improvement of the Computer Vision field along with the Convolutional Neural Networks (CNNs), Traffic Sign Detection has also been tried with CNNs. Also, the CNN approach has been shown to outperform simple classifiers when tested on German Traffic dataset.

Here, let us discuss a few works that have been already developed in terms of traffic sign detection. Wang et al. proposed a method combining coarse filtering modules based on HOG along with Linear Discriminant Analysis and filtering modules which include HOG and SVM classifier on the German Traffic Dataset. Zang et al. combine a local binary pattern (LBP) feature detector with an AdaBoost classifier to extract Region Of Interests (ROI) for coarse selection followed by cascaded CNNs to reduce negative samples of ROI for traffic sign recognition. Zhu et al. develop a method based on a fully convolutional network. They extend the R-CNN by using an object proposal method, EdgeBox and achieve state-of-the-art results on Swedish Traffic Signs Dataset.

B. CNNs for Object Detection

Since 2013, Convolutional Neural Networks (CNNs) have become the standard for all the object detection tasks and for computer vision applications. In OverFeat, Sermanet et al. observed that convolutional networks are efficient when used in a sliding window fashion, as many computations can be reused in overlapping regions.

Another strategy for object detection using CNNs is to first calculate some generic object proposals and perform classification only on these candidates. This strategy was first used in R-CNN but was found slow and inefficient. Now, to improve the efficiency, the spatial pyramid pooling network (SPP-Net) calculates a convolutional feature map for the entire image and extracts feature vectors from the shared feature map for each proposal which increases the speed about 100 times.

Girshick et al. later proposed Fast R-CNN, which uses a softmax layer above the network instead of the SVM classifier which is actually used in R-CNN. Also, it takes around 0.3 seconds to process the image using Fast R-CNN.

Ren et al. next proposed the advanced network which is Faster R-CNN to overcome the bottleneck. This uses convolutional feature maps to generate object proposals which allows the object proposal generator to share a full-image convolutional features with the detection network. This achieved a frame rate of 5fps on GPU.

Next improvement is the proposal of another technique known as Mask R-CNN which is actually an extension of the Faster R-CNN. This consists of two modules out of which the first module is a Region Proposal Network (RPN) which is a deep fully convolutional network, that takes an input image and produces a set of rectangular object proposals, each with an objectness score. The second module is Fast R-CNN which is a region-based CNN, that classifies the proposed regions into a set of predefined categories.

III. METHODOLOGY

The process or method in the detection of the traffic signal in the image given is done in steps that are mentioned below.

Step-1 : Preparation of the data as per the requirements
Step-2 : Developing and training the model

Firstly, before getting to know the method of the execution, let us first understand the data that was used for the model development. The dataset considered is German Traffic Sign Detection Benchmark dataset from (GTSDB Data) which include 600 images and ground truth text file containing the coordinates of the bounding boxes of the traffic signs. Number of classes of traffic signs that are present in the dataset are 43. After downloading the dataset from the link mentioned above, the directory structure is like a data folder as the root folder and inside contains the gt.txt file which contains the ground truth and another folder named images which inside contains .ppm files as images.

In the step-1, we prepare the data as per our requirement, the following sub steps have been followed. Initially, we initialize a python dictionary with empty value and then store the image name as key along with the coordinates of the traffic sign and class value of the sign as values of that particular key. And in the next process, when checking the gt.txt file, we found out that not all images contain the sign coordinates and hence we only use the images with the sign coordinates and class value in the model training and so we separate them into another folder. Once the dataset is ready, we defined a custom pytorch dataset class to load the data and then a class for data augmentation using pytorch transforms. Once that everything is coded and ready, the dataset is loaded by calling the corresponding classes.

In the step-2, we develop the model that will use the data we input for the training dataset. For this, the following sub steps. First we split the dataset into training and validation datasets. And then we define the main model that is required to train the data using the Faster R-CNN technique with ResNet50 as backbone for feature extraction. Later in the model development, we define a few additional parameters that are mentioned below.

Parameter	***Value***
Optimiser	Stochastic Gradient Descent
Learning Rate Scheduler	CosineAnnealingWarmRestarts
Number of epochs	1000
Learning Rate	0.0005

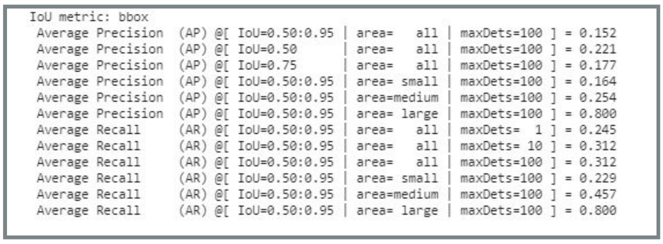
Table-1 : Parameters and values defined in the model

IV. EVALUATION

We have used COCO detection evaluation metrics to evaluate the developed model. The metrics that were used include Average Precision (AP), Average Recall (AR) which are defined as follows.

True Positive (TP): When the IoU over predicted bounding box and ground truth is greater than or equal to the threshold. False Positive (FP): When the IoU over predicted bounding box and ground truth is less than threshold. Average Precision (AP) is the number of true positives in the resulting bounding boxes. Average Recall (AR) is the proportion of true positives out of possible positives.

The below image represents the sample evaluation of the model that was obtained during one of the iterations or epochs training on the data.



```
IoU metric: bbox
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.152
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.221
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.177
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.164
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.254
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.800
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.245
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.312
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.312
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.229
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.457
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.800
```

Fig-1: Sample of the evaluation output

The above picture is explained here. The AP @ IoU=0.5:0.95 for area = large is 0.800 which means that when the model detects an object with large area, 80% of the time it matches the ground truth objects. The AR @IoU=0.5:0.95 for area = large is 0.800 which means that the model detects 80% of objects with large area, correctly.

The other type of evaluation used is by plotting the loss curves once the data is completely trained. Now when it comes to the evaluation, we can say that the lower the loss value, the better the model performs. Below image represents the loss curve that was obtained after successful training of the data on the model for 1000 epochs.

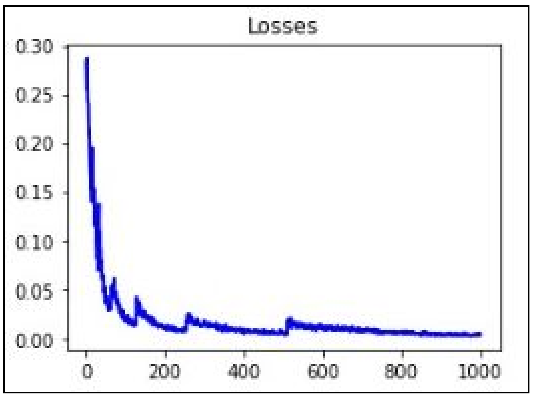
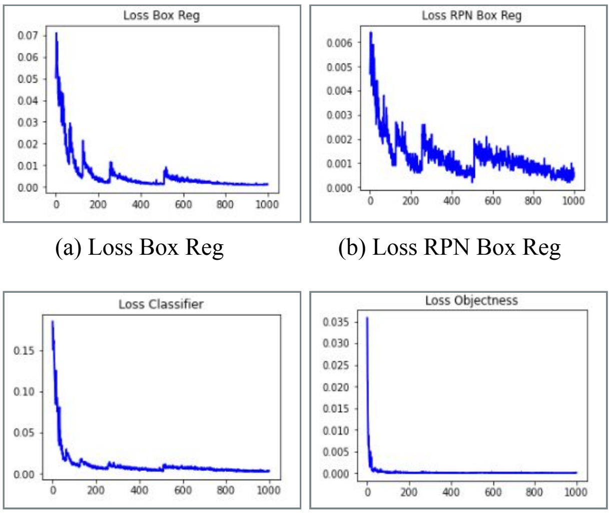


Fig-2: Loss Curve after training for 1000 epochs

From the figure, we can see that the model works fine as the loss is almost near to zero. The other plots that were also used in the evaluation process includes various types of loss curves which are Loss Box Reg, Loss RPN Box Reg, Loss Classifier and Loss Objectness. The meaning of these terms are mentioned below.

Loss Box Reg is the measure of how tightly the model predicted the bounding box around the true object. Loss RPN Box Reg measures the performance of the network for retrieving the region proposals. Loss Classifier measures the performance of the object classification for detected bounding boxes. Loss Objectness measures the performance of a network for retrieving bounding boxes which contain an object. The respective plots are shown below after the data is trained on the model for 1000 epochs.



(a) Loss Box Reg

(b) Loss RPN Box Reg

(c) Loss Classifier

(d) Loss Objectness

Fig-3: Various Loss Curves used in evaluation after training for 1000 epochs

V. RESULTS

Once the model is completed training and evaluated, the model is tested on the unseen image set. A few images from the test dataset are chosen in random and are loaded into the model for the prediction process. From the trained model, the following results were obtained.

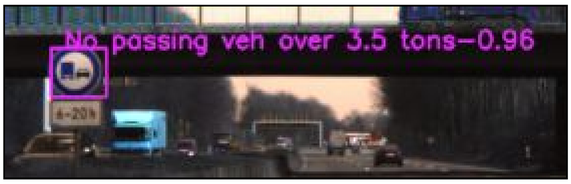


Fig-4: Model predicting correctly the sign along with the probability

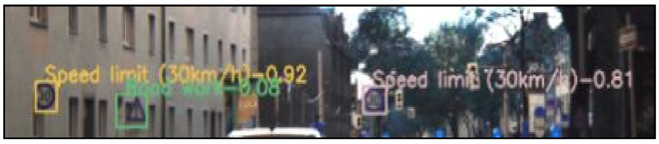


Fig-5: Model predicting correctly multiple signs in single image with probabilities.

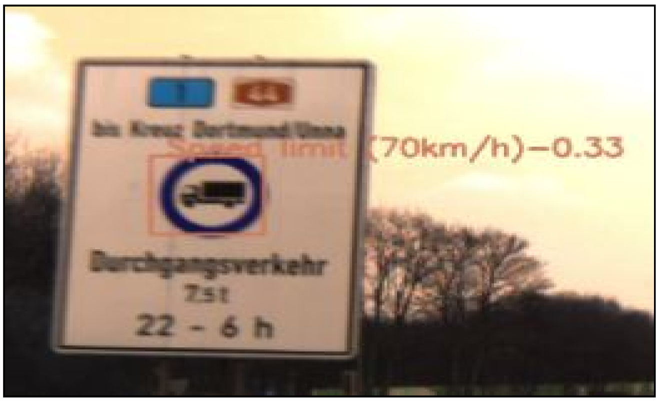


Fig-6: Model predicting wrongly the sign along with probability

VI. CONCLUSION & FUTURE WORK

In this paper, an experiment of detecting the traffic signs using Faster Region-based Convolutional Neural Networks (Faster R-CNN) has been applied on the benchmark dataset of German traffic signs and with ResNet50 as a feature extractor.

And then, the model is evaluated using the COCO detection evaluation techniques such as average precision, average recall and also along with the various versions of the loss curves. From the results, we conclude the below.

a. For area = medium and small, the model does not do well and maybe this was probably caused because of the small size of the dataset.
b. Using the Loss Box Reg curve, it can be observed that the model works well to fit the bbox tightly to the object.
c. Using the Loss RPN Box Reg, the plot shows that further training may be required to decrease the loss. This may require more data to improve the results significantly.
d. Using the Loss Classifier, the plot shows that the model performs well in classifying the objects in the detected bounding boxes.
e. Using the Loss Objectness curve, we can infer that the model is detecting the object very well.

Now, for the future work, we plan to train the data and detect the traffic signs on the more advanced technique called the Mask R-CNN; an improved version of the Faster RCNN; the model we used for the detection. Also, we would like to train the data for a few more epochs which is quite costly in terms of time and hence we would like to use more powerful GPUs for the execution for more epochs and try to improve the model. Also, our model has given quite a few wrong predictions and hence would like to improve in that issue as well. Also, we would like to test our model in real-time by implementing an interface and connecting the camera module and trying whether the model predicts or not when run on roads.

REFERENCES

[1] G. Wang, G. Ren, Z. Wu, Y. Zhao, L. Jiang, A robust, coarse-to-fine traffic sign detection method, in: Proceedings of the 2013 International Joint Conference on Neural Networks, IJCNN, 2013, pp. 1–5, doi:10.1109/IJCNN.2013.6706812.

[2] D. Zang, J. Zhang, D. Zhang, M. Bao, J. Cheng, K. Tang, Traffic sign detection based on cascaded convolutional neural networks, in: Proceedings of 2016 17th IEEE/ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing, SNPD, 2016, pp. 201– 206, doi:10.1109/SNPD.2016.7515901.

[3] Y. Zhu, C. Zhang, D. Zhou, X. Wang, X. Bai, W. Liu, Traffic sign detection and recognition using fully convolutional network guided proposals, Neurocomputing 214 (2016) 758–766.

[4] German Traffic Sign Benchmark Dataset : http://benchmark.ini.rub.de/?section=gtsdb&subsection=dataset

[5] Domen Tabernik and Danijel Skocaj, Deep Learning for Large-Scale Traffic-Sign Detection and Recognition, https://arxiv.org/pdf/1904.00649v1

[6] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun, Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks, https://arxiv.org/pdf/1506.01497

[7] Zhe Zhu, Dun Liang, Songhai Zhang, Xiaolei Huang, Baoli Li , Shimin Hu, Traffic-Sign Detection and Classification in the Wild, Traffic-Sign Detection and Classification in the Wild (thecvf.com)