Towards Real-time Traffic Sign and Traffic Light Detection on Embedded Systems

Oshada Jayasinghe, Sahan Hemachandra, Damith Anhettigama, Shenali Kariyawasam, Tharindu Wickremasinghe, Chalani Ekanayake, Ranga Rodrigo and Peshala Jayasekara

Abstract—Recent work done on traffic sign and traffic light detection focus on improving detection accuracy in complex scenarios, yet many fail to deliver real-time performance, specifically with limited computational resources. In this work, we propose a simple deep learning based end-to-end detection framework, which effectively tackles challenges inherent to traffic sign and traffic light detection such as small size, large number of classes and complex road scenarios. We optimize the detection models using TensorRT and integrate with Robot Operating System to deploy on an Nvidia Jetson AGX Xavier as our embedded device. The overall system achieves a high inference speed of 63 frames per second, demonstrating the capability of our system to perform in real-time. Furthermore, we introduce CeyRo, which is the first ever large-scale traffic sign and traffic light detection dataset for the Sri Lankan context. Our dataset consists of 7984 total images with 10176 traffic sign and traffic light instances covering 70 traffic sign and 5 traffic light classes. The images have a high resolution of 1920 x 1080 and capture a wide range of challenging road scenarios with different weather and lighting conditions. Our work is publicly available at [https://github.com/oshadajay/CeyRo](https://github.com/oshadajay/CeyRo)

I. INTRODUCTION

Traffic signs and traffic lights play a vital role in regulating the traffic by providing necessary information to drivers to safely maneuver on roads. Thus detection of these two elements becomes a fundamental perception task involved in the development of autonomous vehicles and advanced driver assistance systems (ADAS). Developing robust detection algorithms can be a challenging task due to several reasons. Traffic signs and traffic lights usually occupy a small area of a typical street view image. It can be hard to differentiate traffic signs from other similar objects such as billboards and advertisement boards. The algorithms should be robust to occlusions, illumination changes, varying weather conditions and the deterioration of traffic signs with time. While addressing these challenges, it is essential for a detection system to deliver real-time performance with limited computational resources.

Initial work done on traffic sign and traffic light detection [4], [6], [11], [17], [21], [24] mainly focus on traditional image processing based techniques and machine learning based algorithms. Recent deep learning based approaches [3], [5], [18], [19], [33], [35] have been able to outperform these classical approaches, especially in challenging and complex road scenarios. However, most of these implementations are carried out on high-end graphics processing units (GPUs) and less attention is given to delivering real-time performance on embedded systems, which is crucial for both autonomous vehicles and ADAS implementations.

In this work, we propose an end-to-end, deep learning based traffic sign and traffic light detection framework, which is robust to challenging road scenarios, and demonstrates real-time performance on an embedded system. We first create the CeyRo traffic sign and traffic light dataset consisting of 7984 total images belonging to 70 traffic sign and 5 traffic light classes. Our dataset comprises 10176 traffic sign and traffic light instances and covers a wide variety of challenging urban, sub-urban and rural road scenarios.

Our traffic sign and traffic light detection pipeline consists of two stages: 1. Detection or localization of the traffic sign or the traffic light in the original image considering the superclass. 2. Classification of each detection to its respective class. We train and evaluate the performance of two state-of-the-art object detectors, Faster R-CNN [28] and SSD [20] for the detection task and a separate ResNet-18 [13] classifier is trained for the classification task.

We then optimize both our detector and classifier models using TensorRT and integrate with Robot Operating System (ROS) [26] to deploy as a traffic sign and traffic light detection system on an Nvidia Jetson AGX Xavier device. The overall system is well capable of delivering real-time performance with a high inference speed of 63 frames per second (FPS). In summary, our contributions are three-fold:

- We introduce CeyRo: the first ever large-scale dataset for traffic sign and traffic light detection within the Sri Lankan context.
- We evaluate the approach of utilizing a two-staged convolutional neural network based model architecture for the traffic sign and traffic light detection task, and provide results in terms of speed and accuracy.
- We demonstrate the capability of our detection framework to perform in real-time, by deploying the trained and optimized models on an embedded system, integrated with the ROS ecosystem.

II. RELATED WORK

Traditional image processing based algorithms, as well as deep learning based approaches, have been used for...
both traffic sign detection and traffic light detection tasks. The availability of large-scale, high quality visual datasets has become a critical factor for the development of these algorithms, especially with deep learning. A summary of widely used publicly available traffic sign detection and traffic light detection datasets are presented in Table I and Table II respectively.

A robust end-to-end convolutional neural network is proposed in [35], which outperforms state-of-the-art object detectors for their TT100K dataset, particularly with small traffic sign detection. The traffic sign detection and recognition problem becomes complex with the increased number of classes. In [33], this problem is addressed to detect and recognize around 200 traffic sign classes following the Mask R-CNN [12] architecture. A semantic segmentation based approach is followed in Seg-U-NET [18], where two state-of-the-art segmentation architectures are combined for detecting traffic signs and a separate classifier is used for the recognition part. A perceptual generative adversarial network (GAN), which consists of a generator network and a discriminator network, is used in [19] to detect small traffic signs with higher accuracy. Considering deep learning based approaches for traffic light detection, [5] has introduced a YOLO [27] based detection network and a small classification network to accurately detect and classify traffic lights. A Fast R-CNN [10] based network architecture has been used in [3] for traffic light detection with the DriveU [9] dataset.

High-end computational platforms have been used for almost all of the above mentioned algorithms, and implementation of traffic sign and traffic light detection systems on embedded systems is not a widely researched field. A colour based detection algorithm has been used in [30] to identify regions of interest and classify road signs into four categories using a Xilinx Spartan-3A DSP FPGA device. An Artix-7 FPGA device is used in [34] to implement a parallelism optimized AdaBoost based detection algorithm for real-time traffic light detection. Both of these approaches rely on classical techniques, which often have limited usage scenarios when compared with modern deep learning based approaches. Nvidia Jetson TX1 and TX2 devices are used in [23] for traffic light detection, which achieves an inference speed of 10 FPS. Their approach also uses a heuristic colour based candidate region selection module for identifying regions of interest, and a lightweight convolution neural network (CNN) has been used only for the classification part.

III. BENCHMARK DATASET

A. Data Collection

Most of the publicly available traffic sign and traffic light datasets are created using footage captured from cameras mounted on vehicles [5], [14] or images extracted from
TABLE III: Number of instances for each superclass in the CeyRo dataset.

| Superclass                  | Train | Test | Total |
|----------------------------|-------|------|-------|
| Danger Warning Signs (DWS)  | 2833  | 809  | 3642  |
| Mandatory Signs (MNS)       | 453   | 128  | 581   |
| Prohibitory Signs (PHS)     | 650   | 195  | 845   |
| Priority Signs (PRS)        | 115   | 26   | 141   |
| Speed Limit Signs (SLS)     | 735   | 237  | 972   |
| Other Signs Useful for Drivers (OSD) | 1619  | 498  | 2117  |
| Additional Regulatory Signs (APR) | 377   | 123  | 500   |
| Traffic Light Signs (TLS)   | 1075  | 303  | 1378  |
| **Total**                   | 7857  | 2319 | 10176 |

Fig. 2: Traffic sign and traffic light detection model architecture. Faster-RCNN-ResNet50 [13], [28] and SSD-MobileNet-v2 [20], [31] are evaluated as object detectors to first detect the traffic signs and traffic lights as bounding boxes under their superclasses. Then a ResNet-18 [13] classifier is used to classify the detections into their respective classes.

D. Evaluation Metric

We use $F_1$-score as the evaluation metric of our traffic sign and traffic light dataset. Each prediction with an intersection over union (IoU) higher than 0.3 with the ground truth is considered as a true positive. The precision, recall and $F_1$-score can be then calculated as follows where $TP$, $FP$, $FN$ denotes the total number of true positives, false positives and false negatives, respectively.

$$\text{precision} = \frac{TP}{TP + FP} \quad (1)$$

$$\text{recall} = \frac{TP}{TP + FN} \quad (2)$$

$$F_1\text{-score} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (3)$$

IV. METHODOLOGY

A. Model Architecture

The proposed traffic sign and traffic light detection model architecture is shown in Fig. 2. A state-of-the-art object detector model is used to first detect the traffic signs and traffic lights present in the input image as bounding boxes under their 8 superclasses. Then a separate classifier model is used to classify each detection into its respective class.

We evaluate the performance of two state-of-the-art object detectors for the traffic sign and traffic light detection task. Faster R-CNN [28] is used as a two stage object detector and SSD [20] is used as a single stage object detector. ResNet-50 [13] is used as the backbone of the Faster R-CNN model while MobileNet-v2 [15] is used as the backbone of the SSD model. The input resolution is set to $512 \times 512$ in both object detector models.

A ResNet-18 [13] classifier is trained for the traffic sign and traffic light classification task. The input image resolution is set to $100 \times 100$ and the number of output classes is set to 75, which includes the 70 traffic sign classes and the 5 traffic light classes. The traffic sign and traffic light instances present in the train set of our dataset are cropped and extracted out to create the train set for the classifier.
B. Model Training

We use TensorFlow Object Detection API [16] to train the two object detector models. For training the SSD-MobileNet-v2 [20], [31] model, RMSProp [29] optimization is used with an initial learning rate of 0.004 and a momentum of 0.9, and the batch size is set to 24. For training the Faster-RCNN-ResNet50 [13], [28] model, SGD with momentum [32] optimization is used with an initial learning rate of 0.0003 and a momentum of 0.9, and the batch size is set to 8.

The ResNet-18 [13] classifier is trained for 30 epochs using PyTorch [25]. The cross entropy loss is used as the loss function and SGD algorithm with a learning rate of 0.01 and a momentum of 0.9 is used as the optimization function. The batch size is set to 512. We use a computational platform with an Intel Core i9-9900K CPU and an Nvidia RTX-2080 Ti GPU to train our models.

C. Data Augmentation

To reduce the effect of the class imbalance problem, we use data augmentation techniques to increase the number of instances of less frequent traffic signs and traffic lights. Random horizontal flip is used as an augmentation technique when training both detector and classifier models to mirror the traffic signs and traffic lights and create new instances where applicable. Furthermore, color jitter augmentation technique is used when training the classifier to randomly change the brightness, contrast, saturation and hue of the input images.

D. Embedded System Implementation

We use an Nvidia Jetson AGX Xavier as the embedded device to deploy our traffic sign and traffic light detection system. The detector and classifier models have comparatively low speeds when directly inferenced on the embedded device due to its resource constrained nature. Thus, we optimize the trained models using TensorRT optimization with half-precision floating-point (FP16) quantization to effectively utilize the CUDA and Tensor cores present in the device. We use the SSD-MobileNet-v2 [20], [31] model as the detector for the embedded system implementation, since it is much faster than the Faster-RCNN-ResNet50 [13], [28] model.

The traffic sign and traffic light detection model trained using the TensorFlow Object Detection API [16] can be optimized using TensorRT as shown in Fig. 3. First, the frozen inference graph and the configuration parameters of the model are used to generate an intermediate file in the UFF format using GraphSurgeon and UFF libraries. Second, the intermediate file is quantized into a FP16 TensorRT engine using the UFF Parser in the TensorRT Python API. The classifier model which was trained using PyTorch [25] can be optimized using torch2trt [2] as shown in Fig. 4. A direct conversion of the trained PyTorch model in .pth file format to a FP16 quantized TensorRT engine is facilitated by torch2trt which utilizes the TensorRT Python API.

We implement our traffic sign and traffic light detection system in the Robot Operating System (ROS) [26] ecosystem as shown in Fig. 5. The image_feeder node retrieves each frame from a given video file and publishes them to the input_frame topic. The traffic_sign_and_traffic_light_detector node detects traffic signs and traffic lights in the current frame using the generated TensorRT engines. The detections are then published to the traffic_sign_detections and traffic_light_detections topics respectively. The visualizer node marks the detected traffic signs and traffic lights in the current frame and the resultant image is published to the

![Fig. 3: TensorRT conversion and quantization process of the detector model trained using TensorFlow Object Detection API.](image)

![Fig. 4: TensorRT conversion and quantization process of the classifier model trained using PyTorch.](image)

![Fig. 5: RQT graph for the implementation of the traffic sign and traffic light detection system in the ROS ecosystem.](image)
output frame topic. The RViz visualization tool can be used to visualize the traffic sign and traffic light detections in real-time.

V. RESULTS

The traffic sign and traffic light detection results of the two trained models are tabulated in Table IV including the $F_1$-scores for the 59 classes in the test set, overall precision, overall recall, overall $F_1$-score and the inference speed on the workstation with the Nvidia RTX-2080 Ti GPU. The inference speed is calculated as the average FPS for the 1841 test images.

It can be observed that the SSD-MobileNet-v2 [20], [31] model detects traffic signs and traffic lights more accurately than the Faster-RCNN-ResNet50 [13], [28] model. This is contrary to the general belief that two stage object detectors perform better than single stage object detectors. The SSD-MobileNet-v2 [20], [31] model also achieves a higher inference speed of 83 FPS than the Faster-RCNN-ResNet50 [13], [28] model. $F_1$-score values for some traffic sign and traffic light classes are comparatively low, which could be mainly due to the lower number of instances of those classes in the train set.

The results of the TensorRT optimization process are shown in Table V. Each row indicates whether the detector model is optimized, whether the classifier model is optimized and the resulting $F_1$-score and the inference speed on the Nvidia Jetson AGX Xavier device. It can be observed that with a slight drop in accuracy, the inference speed can be increased significantly by optimizing and quantizing both detector and classifier models through TensorRT.

Some of the qualitative results of the traffic sign and traffic light detection task obtained by the SSD-MobileNet-v2 [20], [31] model are visualized in Fig. 6 including urban, rural, expressway, dazzle light and occlusion conditions. Examples for false detections and undetected instances have also been included.

VI. CONCLUSION

In this work, we proposed a simple, end-to-end, deep learning based two-staged detection pipeline for real-time traffic sign and traffic light detection in an embedded system. Furthermore, we introduced the CeyRo traffic sign and traffic light dataset covering a wide range of challenging road scenarios in Sri Lanka. Our benchmark contains 7984 total images and 10176 traffic sign and traffic light instances belonging to 70 traffic sign classes and 5 traffic light classes. The effectiveness of the proposed framework is justified using both qualitative and quantitative results. We further demonstrated the capability of our system to deliver real-time performance in an embedded system using an Nvidia Jetson AGX Xavier device. The detection models were optimized using TensorRT and integrated with Robot Operating System to deploy as a traffic sign and traffic light detection system which achieves a high inference speed of 63 FPS. We believe this is a promising step towards real-time traffic sign and traffic light detection in challenging road scenarios with limited computational resources.

---

### Table IV: Traffic sign and traffic light detection results.

| Label         | SSD-512 | FRCNN-512 |
|---------------|---------|-----------|
| DWS-01        | 0.9583  | 0.9963    |
| DWS-02        | 0.9774  | 0.9978    |
| DWS-03        | 0.9737  | 1.0000    |
| DWS-04        | 1.0000  | 0.9989    |
| DWS-10        | 0.9680  | 0.9434    |
| DWS-11        | 1.0000  | 0.9941    |
| DWS-12        | 1.0000  | 0.9444    |
| DWS-13        | 1.0000  | 0.9412    |
| DWS-14        | 1.0000  | 1.0000    |
| DWS-17        | 0.9765  | 0.9827    |
| DWS-18        | 0.9412  | 0.9412    |
| DWS-19        | 0.9412  | 0.9346    |
| DWS-21        | 0.9545  | 0.9718    |
| DWS-25        | 0.9474  | 0.8421    |
| DWS-26        | 0.9825  | 0.9483    |
| DWS-32        | 0.9673  | 0.9579    |
| DWS-33        | 0.9573  | 0.9836    |
| DWS-35        | 0.9677  | 0.8837    |
| DWS-40        | 1.0000  | 0.9744    |
| DWS-41        | 0.9630  | 1.0000    |
| MNS-01        | 0.8571  | 0.8000    |

### Table V: TensorRT optimization results.

| Optimization | Detector | Classifier | $F_1$-score | FPS  |
|--------------|----------|------------|-------------|------|
| ✔️           | ✔️       | ✔️         | 0.9214      | 13   |
| ✔️           | ✔️       |            | 0.9214      | 16   |
| ✔️           | ✔️       | ✔️         | 0.9193      | 38   |
| ✔️           | ✔️       | ✔️         | 0.9193      | 63   |
Fig. 6: Visualization of traffic sign and traffic light detection results. The first ten images show accurate detections in different road scenarios while the last two images show failure cases including false detections and undetected instances.

REFERENCES

[1] Labelling. https://github.com/tzutalin/labelling

[2] torch2im. https://github.com/NVIDIA-AI-IOT/ torch2im

[3] M. Bach, D. Stumper, and K. Dietmayer. Deep convolutional traffic light recognition for automated driving. In 21st International Conference on Intelligent Transportation Systems, pages 851–858, 2018.

[4] D. Barnes, W. Maddern, and I. Posner. Exploiting 3d semantic scene priors for online traffic light interpretation. In IEEE Intelligent Vehicles Symposium (IV), pages 573–578, 2015.

[5] K. Behrendt, L. Novak, and R. Botros. A deep learning approach to traffic lights: Detection, tracking, and classification. In IEEE International Conference on Robotics and Automation (ICRA), pages 1370–1377, 2017.

[6] A. Ellahyani, M. E. Ansari, I. E. Jaafari, and S. Charfi. Traffic sign detection and recognition using features combination and random forests. International Journal of Advanced Computer Science and Applications, 7, 2016.

[7] C. Ertler, J. Majek, T. Ollmann, L. Porzi, G. Neuhold, and Y. Kuang. The maipillary traffic sign dataset for detection and classification on a global scale. In European Conference on Computer Vision (ECCV), pages 68–84, 2020.

[8] M. Everingham, S. M. A. Eslami, L. V. Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. International Journal of Computer Vision, 111(1):98–136, 2015.

[9] A. Fregin, J. Muller, U. Krebel, and K. Dietmayer. The driveu traffic light dataset: Introduction and comparison with existing datasets. In IEEE International Conference on Robotics and Automation (ICRA), pages 3376–3383, 2018.

[10] R. Girshick. Fast r-cnn. In IEEE International Conference on Computer Vision (ICCV), pages 1440–1448, 2015.

[11] A. E. Gomez, F. A. R. Alencar, P. V. Prado, F. S. Osório, and D. F. Wolf. Traffic lights detection and state estimation using hidden markov models. In IEEE Intelligent Vehicles Symposium (IV), pages 750–755, 2014.

[12] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 2980–2988, 2017.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, 2016.

[14] S. Houben, J. Stallkamp, J. Salmen, M. Schlipping, and C. Igel. Detection of traffic signs in real-world images: The German Traffic Sign Detection Benchmark. In International Joint Conference on Neural Networks, 2013.

[15] A. G. Howard, M. Zhu, B. Chen, D. Kalenichenko, W. Wang, T. Weyand, M. Andreetto, and H. Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861, 2017.

[16] J. Huang, V. Rathod, C. Sun, M. Zhu, A. Korattikara, A. Fathi, L. Fischer, Z. Wojna, Y. Song, S. Guadarrama, and K. Murphy. Speed/accuracy trade-offs for modern convolutional object detectors. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 7310–7311, 2017.

[17] M. Jensen, M. Philipsen, Chris C. H. Balslev, A. Mogelmose, T. Moeslund, and M. Trivedi. Traffic light detection at night: Comparison of a learning-based detector and three model-based detectors. In International Symposium on Visual Computing, pages 774–783, 2015.

[18] U. Kamal, T. I. Tonmoy, S. Das, and M. K. Hasan. Automatic traffic sign detection and recognition using segu-net and a modified tversky loss function with 11-constraint. IEEE Transactions on Intelligent Transportation Systems, 21(4):1467–1479, 2020.

[19] J. Li, X. Liang, Y. Wei, T. Xu, J. Feng, and S. Yan. Perceptual generative adversarial networks for small object detection. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 1951–1959, 2017.

[20] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In European Conference on Computer Vision (ECCV), pages 21–37, 2016.

[21] S. Maldonado-Bascón, S. Lafuente-Arroyo, P. Gil-Jimenez, H. Gomez-Moreno, and F. Lopez-Ferreras. Road-sign detection and recognition based on support vector machines. IEEE Transactions on Intelligent Transportation Systems, 8(2):264–278, 2007.

[22] A. Mogelmose, M. M. Trivedi, and T. B. Moeslund. Vision-based traffic sign detection and analysis for intelligent driver assistance systems: Perspectives and survey. IEEE Transactions on Intelligent Transportation Systems, 13(4):1484–1497, 2012.

[23] Z. Ouyang, J. Niu, Y. Liu, and M. Guizani. Deep cnn-based real-time traffic light detector for self-driving vehicles. IEEE Transactions on Mobile Computing, 19(2):300–313, 2020.

[24] G. Overett and L. Petersson. Large scale sign detection using hog feature variants. In IEEE Intelligent Vehicles Symposium (IV), pages 326–331, 2011.

[25] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. In Advances in Neural Information Processing Systems, pages 8024–8035, 2019.

[26] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: An open-source robot operating system. In ICRA Workshop on Open Source Software, 2009.

[27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 779–788, 2016.

[28] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(6):1137–1149, 2017.

[29] S. Ruder. An overview of gradient descent optimization algorithms. arXiv preprint arXiv:1609.04747, 2016.

[30] S. S. M. Sallah, F. A. Hassis, and M. Z. Yusoff. Road sign detection and recognition system for real-time embedded applications. In International Conference on Electrical, Control and Computer Engineering (InECCCE), pages 213–218, 2011.

[31] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 4510–4520, 2018.

[32] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. On the importance of initialization and momentum in deep learning. In 30th International Conference on Machine Learning, pages 1139–1147, 2013.

[33] D. Tabernik and D. Sko ˇcaj. Deep learning for large-scale traffic sign detection and recognition. IEEE Transactions on Intelligent Transportation Systems, 21(4):1427–1440, 2020.

[34] X.-H. Wu, R. Hu, and Y.-Q. Bao. Parallelism optimized architecture on fpga for real-time traffic light detection. IEEE Access, 7:178167–178176, 2019.

[35] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu. Traffic sign detection and classification in the wild. In IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 2110–2118, 2016.