**Detection of small and far Traffic signs using Deep Learning**

**Introduction/Problem Statement:**

The United States experiences more than 6 million car accidents annually, making it a prevalent issue. These road crashes are the primary cause of death in the country, claiming the lives of over 38,000 individuals each year. According to the Bureau of Labor Statistics, a car accident occurs approximately every 13 minutes in the U.S. Despite advancements in technology and regulations aimed at enhancing vehicle safety, car accidents remain a significant cause of fatalities. Failure to adhere to traffic rules and obey traffic signs is a major contributor to accidents on the roads. Disregarding speed limits, neglecting stop signs, and not yielding the right of way create dangerous situations that increase the likelihood of collisions. Traffic signs play a crucial role in regulating traffic flow, providing important warnings, and conveying vital information. One emerging trend that is expected to reduce car accidents worldwide is the development of autonomous and shared transportation options.

To comply with road regulations, autonomous vehicles need to be able to identify and comprehend traffic signs. The ability to detect traffic signs from a significant distance plays a vital role in addressing the issue of non-compliance with traffic rules and signs. It is common for drivers to overlook or miss small traffic signs, particularly when approaching them from afar. By detecting these signs from a distance, drivers have a better chance of noticing them, even if they are small or inconspicuous. Improved visibility ensures that important information displayed on the signs, such as speed limits, warnings, or regulations, is effectively communicated to drivers, reducing the likelihood of rule violations. Early detection allows drivers to have more time to react appropriately, adjust their driving behavior, and adhere to the traffic rules and instructions. For instance, if a small "No U-Turn" sign is detected early, drivers can plan alternative routes or avoid sudden and unsafe maneuvers, thereby decreasing the risk of accidents caused by abrupt actions. This encourages responsible driving habits and minimizes the occurrence of accidents resulting from hasty or uninformed decisions. To summarize, the detection of small traffic signs from a significant distance is of utmost importance in preventing accidents. This is achieved through increased visibility, early identification and response, improved decision-making, and advanced driver assistance systems.

**Challenges involved in small and far traffic sign detection:**

Previously, conventional computer vision techniques were used to detect and categorize traffic signs, but this approach involved extensive manual effort to carefully design significant features within images. However, by utilizing deep learning in this context, we can develop a model that accurately classifies traffic signs from a considerable distance. This model learns to automatically identify the most relevant features for this task, eliminating the need for laborious manual feature engineering. Detecting small traffic signs from a distance presents specific challenges that require attention due to their size and distance. Real-world scenarios involve significant scale variations in far and small traffic signs. Deep learning models must be capable of effectively detecting and recognizing these signs across a wide range of sizes. However, the limited resolution of small signs may result in the loss of fine details and features, making it challenging for deep learning models to capture essential information accurately. Furthermore, traffic signs are susceptible to occlusion by objects such as trees, poles, vehicles, and background clutter, which can obstruct visibility and hinder sign recognition, leading to detection errors. Counterfeit traffic signs with surface reflecting properties can also mislead detection models.

Moreover, detecting small objects within relatively large images is not the only challenge. Real-world conditions often feature complex backgrounds, including skies, buildings, roads, trees, pedestrians, vehicles, and streetscapes, rather than clean and monotonous backgrounds. Various environmental interferences, such as advertising symbols and other indications, often share similar color saturation and contrast with traffic signs, making them indistinguishable. Additionally, adverse environmental conditions such as low lighting, extreme weather, or poor image quality can further degrade the visibility and recognition of signs, resulting in detection failures. To address these challenges, deep learning models need to exhibit robustness against environmental variations, employing techniques like domain adaptation, adversarial training, or data augmentation with simulated environmental effects.

**Dataset:**

Datasets play a crucial role in developing and evaluating robust traffic sign detection models. While there are several datasets available for traffic sign detection such as the Belgian Traffic Sign Dataset (BEL-TSD), Traffic Sign Recognition Dataset (TSRD), Tsinghua-Tencent 100 K, German Traffic Sign Recognition Benchmark (GTSRB), Chinese Traffic Sign Detection Benchmark (CCTSDB), and BDD100K, each offering a diverse range of annotated images to train and evaluate models. One widely used dataset is the German Traffic Sign Recognition Benchmark (GTSRB), which contains over 50,000 images of 43 different traffic sign classes, including variations in lighting conditions, occlusions, and weather. However, in the GTSRB, the traffic signs occupy a large proportion of each traffic scene, which reduces the difficulty of classifying the traffic signs. Therefore, by using GTSRB dataset we cannot address the issue of small and far traffic sign detection.

In the [Tsinghua-Tencent 100K](https://cg.cs.tsinghua.edu.cn/traffic-sign/) dataset, most of the traffic signs are quite small, measuring 50 × 50 pixels or less, and they are scattered within a larger 2048 × 2048-pixel image. Each individual sign occupies less than 0.1% of the total image area, even the larger signs with a size of 400 × 400 pixels only cover 3.8% of the image. This contrasts with other commonly used datasets where traffic signs may account for around 20% of each image. The dataset captures images from real-world urban environments, offering a realistic representation of the challenges involved in detecting small traffic signs. It encompasses diverse scenes with varying lighting conditions, occlusions, and background clutter, which are typical factors affecting the visibility and recognition of small signs. Some of the images are shown below. This dataset proves valuable for evaluating algorithms that can effectively handle the complexities of real-world scenarios. With a substantial collection of 100,000 high-resolution images, the Tsinghua-Tencent 100K dataset provides ample data for training and testing models focused on small traffic sign detection. Hence, for the current problem statement, Tsinghua-Tencent 100K dataset would be most suitable for better evaluation of the deep learning model.

A road with fog and clouds

Description automatically generated A road with a blue car on it

Description automatically generated A car driving on a road

Description automatically generated

Fog effect Normal Image Snow effect

**Analysis of various deep learning models:**

The RCNN framework was accurate but slow and required a lot of storage space because it had to generate region proposals from the original image. Fast RCNN improved upon RCNN in two ways. First, it generated regional proposals from the feature map, sharing the computation of deep CNNs and improving efficiency. It also directly fed features into the classifier, reducing storage requirements. Second, it added an RoI-pooling layer to obtain fixed-size features from the region proposal. However, its region proposals were generated by a selective search, which couldn't be accelerated by the GPU.

Faster RCNN introduced the region proposal network (RPN). It generated nine anchor boxes for each pixel using different scales and aspect ratios. Each anchor box corresponded to a region proposal based on translation invariance. However, the original Faster RCNN had difficulty robustly detecting small traffic signs. This was mainly because it often used VGG-16 as the backbone to extract image features. The feature map dimensions were only 1/16 of the input image, resulting in a large receptive field for each pixel. Additionally, the high-level feature map had a lower resolution, making it unable to effectively represent small objects and leading to poor localization performance.

YOLOv1: YOLOv1 introduced the concept of object detection using a single neural network, but it suffered from lower accuracy compared to later versions. This was mainly due to the absence of finer-grained feature maps and the use of larger grid cells, which limited its ability to detect small objects like small traffic signs effectively. As the model architecture includes a lot of convolution layers and anchor boxes, is has high processing time, more parameters, and increased memory usage.

YOLOv2: YOLOv2 improved accuracy by introducing several modifications, including the addition of skip connections and multi-scale training. Optimizations such as network pruning and the use of anchor boxes of various sizes, reduced the computational complexity and improved processing speed compared to YOLOv1. However, it was still not the fastest compared to other versions.

YOLOv3: YOLOv3 achieved higher accuracy compared to YOLOv2 through the introduction of additional layers and feature maps of different resolutions. It also incorporated more advanced techniques like feature pyramid networks (FPN) and Darknet-53 as the backbone architecture, enabling better feature extraction and improved object detection. However, the large model size led to a greater number of parameters that led to increased memory usage and longer training/inference times.

YOLOv4: YOLOv4 achieved state-of-the-art accuracy by incorporating numerous advancements such as CSPDarknet53 as the backbone, PANet for feature fusion, and various optimization techniques like Mish activation and CIOU loss. These improvements collectively enhanced the model's ability to detect objects accurately, including small traffic signs. However, the increase in model complexity resulted in a larger model size, demanding more computational resources during training and inference compared to YOLOv5.

YOLOv5: YOLOv5 achieved competitive accuracy by leveraging advancements such as the implementation of a more efficient architecture, the use of advanced augmentation techniques, and the introduction of the PANet feature fusion module. These improvements, combined with the focus on model optimization, helped YOLOv5 achieve very high accuracy by significantly reducing the number of model parameters compared to previous versions, leading to a lightweight model. This reduction in complexity enables faster training and inference, lower memory requirements, and efficient deployment on resource-constrained devices.

**Model Architecture:**

Analyzing various advantages and disadvantages of different models, YOLOv5 (reference architecture was shown below) was chosen as it provides better tradeoff between model accuracy and the model complexity in terms of processing time and number of parameters. However, considering the specific application of small and far traffic sign detection there are certain modifications that were proven to improve the performance of YOLOv5. Based on the literature review, the following modifications were selected to get better results.

A screenshot of a computer

Description automatically generated

YOLOv5s architecture diagram (source: [yolov5](https://github.com/ultralytics/yolov5/issues/1333))

Feature extraction module: Replace the 32 times down sampling with the 16 times down sampling to reduce the loss of small object information during feature propagation.

Four convolutional modules and the Swin Transformer Block (C4STB): Instead of a down-sampling 3x3 convolution operation, 1x1 convolution operation will be implemented to ensure the transmission of more detailed information.

Loss function: Yolo v5 uses CIoU as the regression loss function. However, IoU based metrics are excessively sensitive to location deviation of tiny objects, therefore, applying anchor-based detectors results in a drastic decrease of model performance. NWD is not sensitive to the scale of the objects, meaning it is more suitable for measuring the similarity between small objects. Hence to improve this, we can combine both CIoU and NWD to calculate the localization loss.

Clustering algorithm: The original YOLOv5s model uses the K-means algorithm to cluster the dataset. However, the K-means algorithm is affected by the random selection of the initial cluster center, which may cause the initial cluster center to be far away from the optimal cluster center location; this not only affects the convergence speed of the model, but also leads to poor detection results. To solve this, we can use the K-means++ algorithm (optimization algorithm based on K-Means) to re-cluster all labeled object frames in the training dataset.

**Evaluation of model:**

There are several evaluation metrics used to assess the performance of a deep learning model. While the most common ones were accuracy, precision, and recall, metrics such as mAP - mean average precision (measures the average precision across different object classes by calculating the area under the precision-recall curve), and intersection over union (IoU) are commonly used for object detection and segmentation tasks. In addition, metrics such as working area size (represents the dimensions or resolution of the input images that the model can effectively handle) and detection time (time taken by the model to process an input image or a video frame and perform object detection or recognition tasks), and BFLOPS - Billion Floating-Point Operations per Second (measure the computational complexity or computational efficiency of a deep learning model. It represents the number of floating-point operations like additions, multiplications, etc. the model can perform in one second, in terms of billions) are helpful to measure the efficiency and speed of the model especially in real-time or near real-time applications.

**References:**

1. Álvaro Arcos-García, et al. Evaluation of deep neural networks for traffic sign detection systems. Neurocomputing, Volume 316 (2018) <https://doi.org/10.1016/j.neucom.2018.08.009>.
2. Zhou, Weijun, Zhang, Baojun, et al. Detecting Small Chinese Traffic Signs via Improved YOLOv3 Method. Mathematical Problems in Engineering <https://doi.org/10.1155/2021/8826593>
3. Wan, J., Ding, W., Zhu, H. et al. An Efficient Small Traffic Sign Detection Method Based on YOLOv3. J Sign Process Syst 93, 899–911 (2021). <https://doi.org/10.1007/s11265-020-01614-2>
4. Liu, Z., Li, D., Ge, S.S. et al. Small traffic sign detection from large image. Appl Intell 50, 1–13 (2020). <https://doi.org/10.1007/s10489-019-01511-7>
5. Han, C., Gao, G. & Zhang, Y. Real-time small traffic sign detection with revised faster-RCNN. Multimed Tools Appl 78, 13263–13278 (2019). <https://doi.org/10.1007/s11042-018-6428-0>
6. Shijin Song, Zhiqiang Que, et al. An efficient convolutional neural network for small traffic sign detection, Journal of Systems Architecture, Volume 97 (2019) <https://doi.org/10.1016/j.sysarc.2019.01.012>.

[7] Lai, H.; Chen, L.; Liu, W.; Yan, Z.; Ye, S. STC-YOLO: Small Object Detection Network for Traffic Signs in Complex Environments. Sensors 2023, 23, 5307. <https://doi.org/10.3390/s23115307>

[8] R. Hu, H. Li, D. Huang, X. Xu and K. He, "Traffic Sign Detection Based on Driving Sight Distance in Haze Environment," in IEEE Access, vol. 10, pp. 101124-101136, 2022, doi: 10.1109/ACCESS.2022.3208108.

[9] [Bureau of Transportation statistics](https://www.bts.gov/content/motor-vehicle-safety-data)

[10] [CAR ACCIDENT STATISTICS YOU NEED TO KNOW IN 2023](https://www.ddlawtampa.com/resources/car-accident-statistics-you-need-to-know-in-2021/#:~:text=Total%20number%20of%20car%20accidents%20per%20year%20in%20the%20U.S.&text=On%20average%2C%20there%20are%20over,losing%20their%20lives%20each%20year.)

[11] [ChatGPT](https://chat.openai.com/)