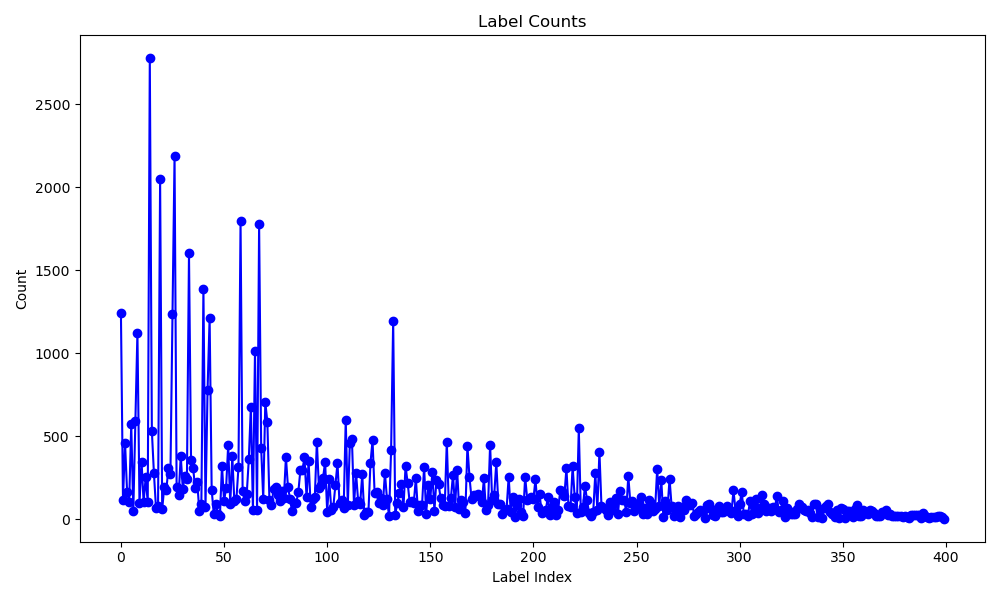
**Experimental Analysis of Modifications and Training Strategies for Small Traffic Sign Detection using YOLOv5s**

**1. Experimental Setup and Training:**

**1.1 Dataset that resembles realistic conditions and varying distances:** To simulate realistic conditions, MTSD dataset was used. It presents an extensive array of annotated images, with a special focus on traffic signs situated within diverse urban and highway scenarios. MTSD not only covers a broad spectrum of environmental conditions, lighting variations, and camera viewpoints but also offers high-resolution data crucial for effective small traffic sign detection. With meticulously crafted annotations encompassing bounding box coordinates and class labels, the dataset serves as an invaluable resource for training and evaluating deep learning models, specifically geared towards enhancing the accuracy and efficiency of algorithms for detecting small traffic signs.

During the data exploration process, I began with a dataset comprising a substantial 41,909 data points, each accompanied by annotations. Additionally, the dataset contained 10,544 test images, which lacked annotations. Within this diverse dataset, a comprehensive total of 374 unique labels were identified. The instances of these labels exhibited a wide spectrum, ranging from a minimum of 1 instance to a remarkable 2800 instances as shown in the below figure.



**1.2 Data Pre-processing:**

To streamline the dataset and focus on more prevalent and relevant labels, I initiated a filtering process. Labels with instance counts falling below the threshold of 600 were eliminated, as they could potentially introduce noise to subsequent analyses. For training a real-time image detection model such as YOLOv5 (You Only Look Once), annotations were modified to ensure optimal model performance. The dataset consisted of annotated images capturing a diverse range of real-world traffic sign scenarios. The annotations, originally provided in JSON format, were transformed to align with YOLO's input requirements. This involved converting the provided xmin, xmax, ymin, and ymax values into the format of x\_center, y\_center, width, and height of bounding boxes. Additionally, these coordinates were normalized to fit within the range of [0, 1] to facilitate model training.

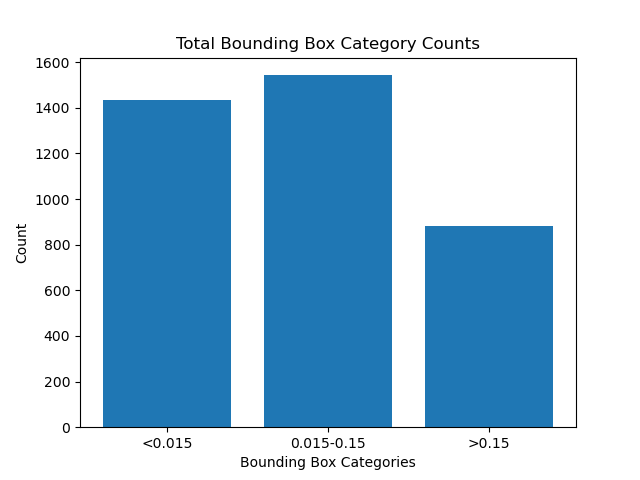
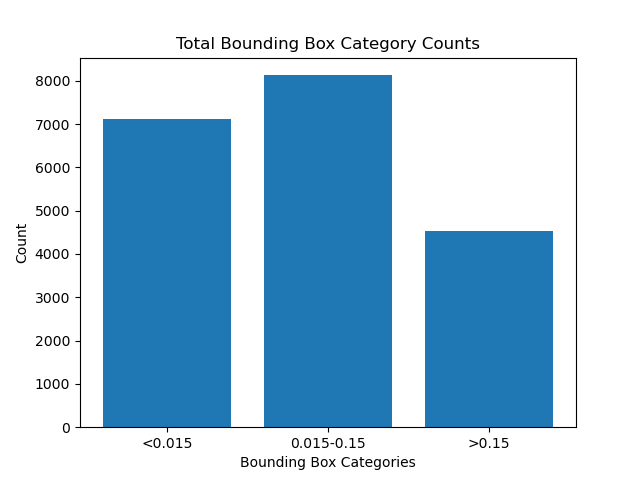
xc = xmin/img\_width + b\_width/2

yc = ymin/img\_height + b\_height/2

b\_width = xmax/img\_width - xmin/img\_width

b\_height = ymax/img\_height - ymin/img\_height

To ensure the model's effectiveness across different scenarios, the dataset was curated to include a variety of traffic sign instances with varying distances from the camera. This enabled the model to learn the visual characteristics of signs at different scales and distances. I opted to remove labels that didn't meet a specific criterion related to the number of small traffic sign instances. By conducting this meticulous filtering process, I aimed to create a refined dataset that would serve as a more accurate foundation for subsequent analyses and model development.



Number of label instances per each category in training data Number of label instances per each category in validation data

**1.3 Experimental environment and parameter settings:**

The Linux operating system, with an GPU was employed for the experiments. Pytorch, a deep learning framework, and Python were used for the programming. Stochastic gradient descent (SGD), an optimization algorithm, was employed for model training. Initial learning was occurring at a rate of 0.01, momentum was 0.937, and weight decay was 0.0005. The batch size of was set at 16, and each method of the model was trained for 40 epochs overall.

**2 Implementation/Methods:**

With several benefits like high detection accuracy, quick operation, and simple deployment, the YOLOv5 model from the YOLO series is widely employed in various industrial applications. However, the network must be enhanced to increase its performance in small item detection due to its low detection performance on small objects. In this project, a small traffic sign detection network called M-YOLO was built for complicated scenarios because of the high demands of traffic sign identification for model accuracy and speed. YOLOv5 was chosen as the baseline network for further improvement. Instead of 32 times of downsampling, 16 times were used in the feature extraction phase to prevent the loss of small object information during feature propagation. To compensate for the reduction in the receptive field brought on by the adjustment of the sub-sampling process, the C3 module was replaced with a feature extraction module called C4 that has a stronger characterization capability.

The NWD was introduced to calculate the localization loss to balance the sensitivity of the IoU to the location deviation of small objects based on CIoU loss. To increase the matching degree of tiny pieces, the K-means++ method was employed to replace the K-means algorithm in the original network. K-means++ is an optimization algorithm built on top of the K-means algorithm. Its main goals are to enhance the initial point selection, increase the efficientcy of training dataset's anchor box selection, and boost the model's ability to recognize small objects with more accuracy.

**2.1 Evaluation metrics:**

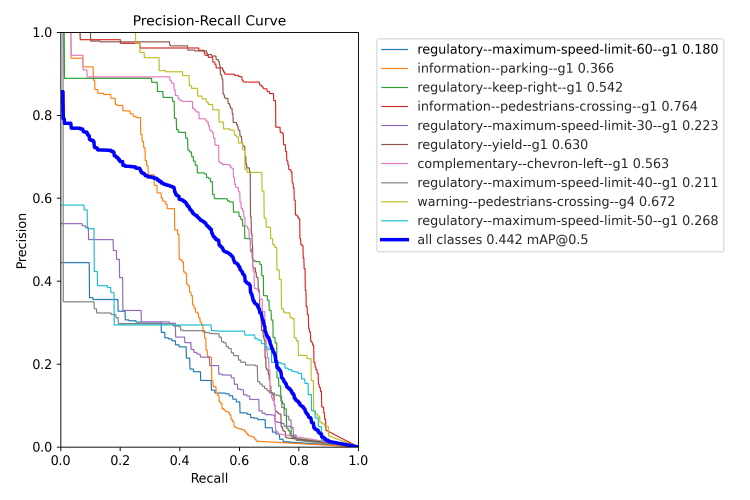
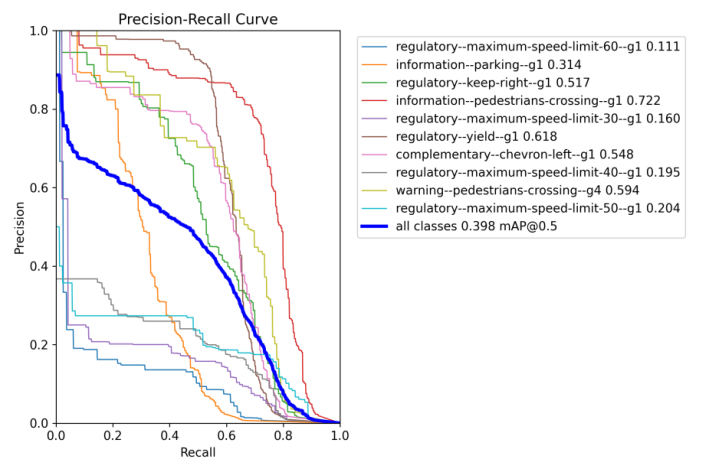
Precision (P) mainly measures the degree of model error detection, recall (R) measures the degree of model missed detection, average precision (AP) is the area under the P-R curve, and mAP is the average value of AP of all the classes. GFLOPS (Giga Floating-Point Operations Per Second) is a metric used to measure the performance of a computer system, particularly its ability to perform floating-point arithmetic operations. The model complexity is calculated using the number of parameters, which is calculated using the below formula:

Params = C0 x (kw x kh x C1 +1)

**3 Results/Inference:**

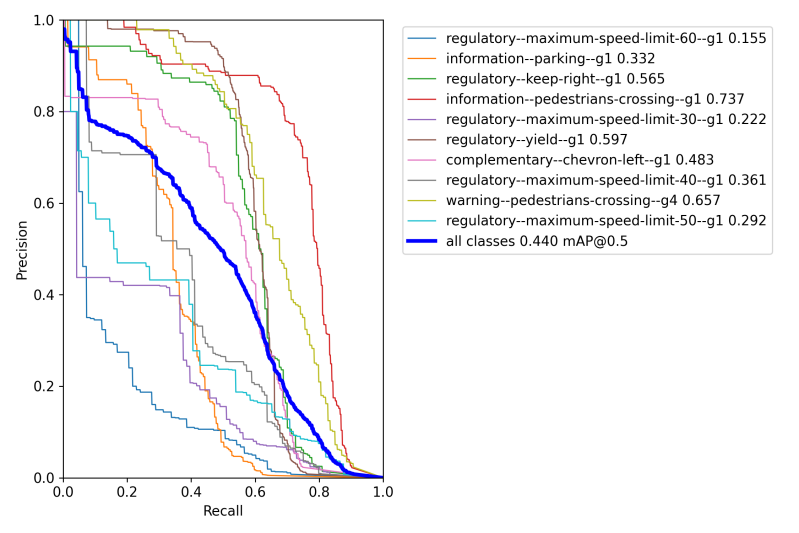
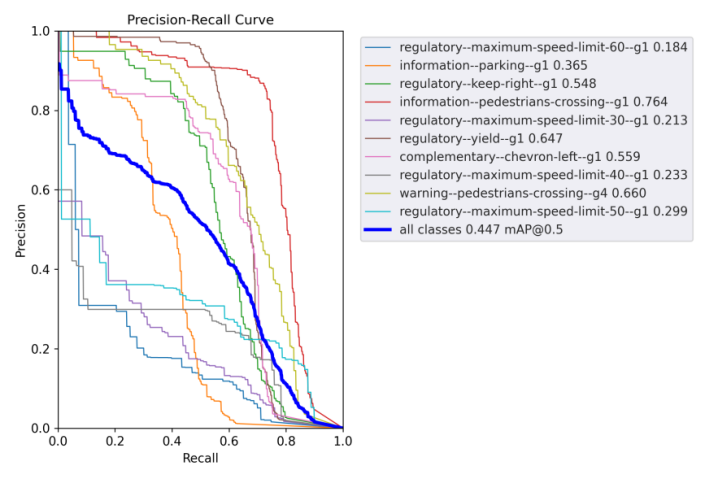
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Baseline** | **K-Means++** | **NWD+CIoU** | **C4** | **mAP (%)** | **Params (M)** | **GFLOPS** |
| Checkmark with solid fill |  |  |  | 39.8 | 7.04 | 16 |
| Checkmark with solid fill | Checkmark with solid fill |  |  | 44.2 | 7.04 | 16 |
| Checkmark with solid fill |  | Checkmark with solid fill |  | 44.0 | 7.04 | 16 |
| Checkmark with solid fill |  |  | Checkmark with solid fill | 42.4 | 7.19 | 18.9 |
| Checkmark with solid fill | Checkmark with solid fill |  | Checkmark with solid fill | 44.7 | 7.19 | 18.9 |
| Checkmark with solid fill | Checkmark with solid fill | Checkmark with solid fill | Checkmark with solid fill | 46.0 | 7.19 | 18.9 |

Each module adopted in this work improved the detection accuracy of the network to some extent. Compared with the YOLOv5s, the updated anchor boxes using Kmeans++ algorithm made significant improvements (mAP from 39.8% to 44.2%), with the mAP being improved by 4.4%. This shows that generating candidate boxes that are more suitable for small object detection can better cover small objects in the dataset and effectively solve the problem of the low detection rate of candidate boxes in small object datasets.



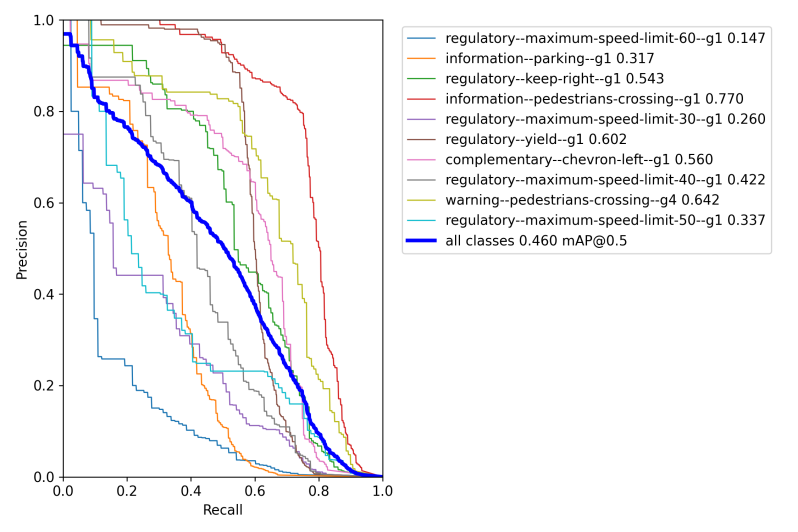
P-R curve for Base model P-R curve for Baseline & K-Means++

When the C4 module proposed in this paper was applied to the YOLOv5s model, mAP was increased by 2.6%. This shows that the C4 module can effectively extract features with a better discriminating ability for small object detection and, at the same time, expand the receptive field to ensure the accuracy of medium and large-size objects. When the combination of NWD and CIoU was used as the loss function, mAP was increased by 4.2%, This shows that the introduction of NWD into the regression loss function is helpful in improving the sensitivity of the IoU-based metrics to small object position deviations, thereby improving the detection accuracy of small objects.



PR curve for Baseline & Kmeans++ & C4 PR curve for Baseline with NWD + CIOU

The P–R curve with each improved module added to the YOLOv5s network are given below. It can be clearly seen that the curve with the C4 module, NWD, and K-means ++ modules added at the same time covers the curve with a single module added. Intuitively, it is concluded that each improved module provides a certain performance improvement to the network.



P-R curve when all the modules included (Kmeans++ & NWD & C4)



Model output on few of the test images

**4 Conclusion**

This research offers concepts for small traffic sign detection and can be applied to autonomous vehicles' perception of their environment. However, there are other challenges to seeing traffic signs at night, including interference from reflected light and street lighting. The M-YOLO algorithm's resistance to dark conditions has not been tested. Datasets could one day be gathered with the goal of detecting traffic signs in scenery at night. The performance and stability of the model should be further optimized by using a high-resolution feature map for the model input, reducing the learning rate, and training the model for more epochs. Dataset can be divided further by splitting each label into subclasses such as small-informatory, medium-informatory, large-informatory etc. to evaluate the model on each scale of detection.