Traffic-Sign Detection and Recognition for Autonomous Vehicles Applications

Report for the
Final Project of the Graduate Course
CS329: Machine Learning(H)

January 14, 2024

Prepared by:

Site Fan
12111624@mail.sustech.edu.cn
Department of Computer Science and Engineering
Southern University of Science and Technology
Shenzhen, Guangdong, China 518055

Jiachen Xiao
12112012@mail.sustech.edu.cn
Department of Computer Science and Engineering
Southern University of Science and Technology
Shenzhen, Guangdong, China 518055

Abstract

Traffic sign detection and recognition (TSDR) constitute essential components of autonomous driving and advanced driver-assistance systems. Despite the integration of most signs into digital maps, challenges including outdated information and temporary signs persist. Therefore, accurate real-time TSDR remain indispensable.

This project reviews current mainstream models and optimization methods for TSDR, proposes methodologies like attention modules and data preprocessing to enhance the performance of existing models over the TT-100K dataset

Introduction

Background

In recent times, economic growth has precipitated a rise in vehicular traffic, subsequently escalating traffic accidents. The advent of intelligent technologies, particularly Traffic Sign Detection and Recognition (TSDR) systems, is pivotal in addressing these challenges. TSDR, through precise detection and recognition of road signs, plays a crucial role in alerting drivers, promoting safe driving practices, and mitigating accidents. Investigating efficient and accurate TSDR methods is therefore imperative.

TSDR's effectiveness in real-world scenarios is significantly influenced by factors like lighting and obstructions. Additionally, the exigencies of autonomous driving necessitate real-time road condition analyses, making the development of proficient TSDR algorithms paramount. Traditional region-based computer vision methods, which involve extracting potential areas and classifying them, are hindered by inefficiency. To reconcile speed and accuracy in TSDR systems, models like YOLO and other one-step methods are preferable. Enhancing model applicability to real-world conditions involves dataset modifications, including erasure and occlusion, and training with the augmented dataset.

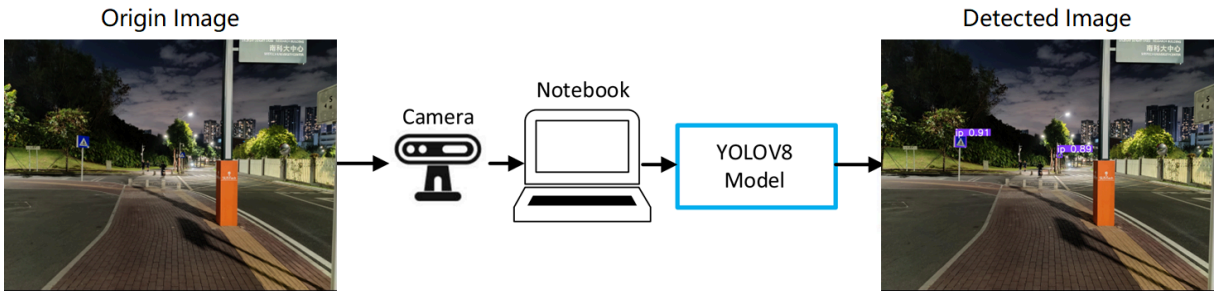

Figure 1: Block Diagram of The TSDR System

Motivation and Challenges

While traffic sign recognition tasks typically unfold in natural scenes, various weather conditions such as rain, snow, or fog can obscure the information conveyed by these signs. Moreover, issues like overexposure and low light conditions tend to diminish the visibility of traffic signs. Besides, the continuous exposure of traffic signs throughout the year may lead to fading, obscurity, or damage to their surfaces.

Moreover, there is still room for improvement, particularly in the area of real-time detection. Single-stage target detection algorithms, such as YOLOv8, have shown great potential in this regard, but further research and development are needed to optimize its performance.

This project is driven by the desire to contribute to this field of study, with the aim of developing a single-stage deep learning detection approach that can enhance the real-time and accurate detection of traffic signs.

This will not only contribute to the advancement of autonomous vehicle technology but also to the broader goal of improving road safety and efficiency.

Related Work

Tsinghua-Tencent 100K[1]

The dataset used for training, validating and testing is Tsinghua-Tencent 100K Annotations 2021. Dataset and preprocessing TT-100K(Tsinghua-Tencent 100K) is a dataset which contains 100000 Tencent Street View panoramas, including large variations in illuminance and weather conditions in China. All of the traffic-sign in the pictures is annotated in a json-format file with a special class label, which marks the center of a box containing traffic-signs and also the normalized width and height.

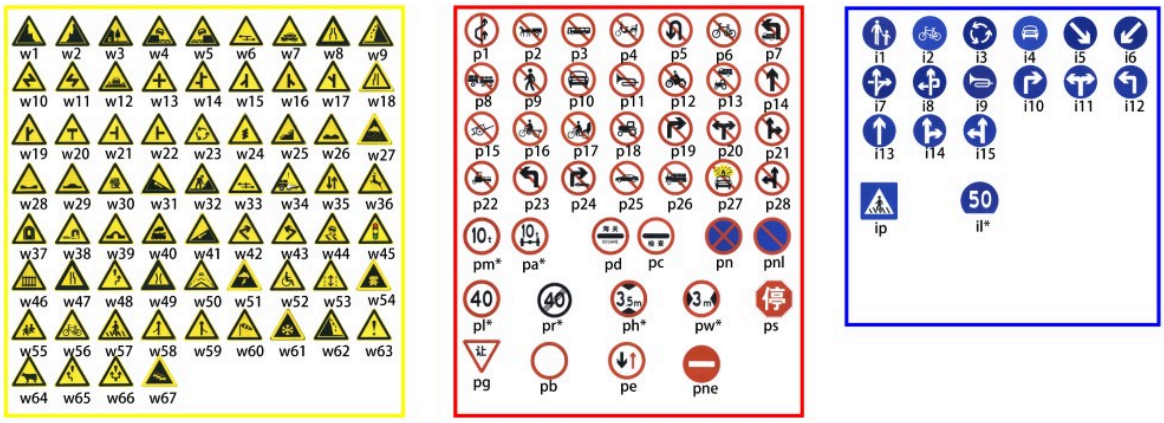

Figure 2: The Set of Traffic Signs in TT-100K

YOLOv8

Ultralytics' YOLOv8 significantly enhances Traffic Sign Detection and Recognition (TSDR) systems with its state-of-the-art architecture. Prioritizing speed and accuracy, YOLOv8 is adept at processing images rapidly, a critical requirement for real-time TSDR in dynamic driving conditions. Its improved recognition capabilities ensure accurate identification of traffic signs, crucial for safe navigation and autonomous driving.

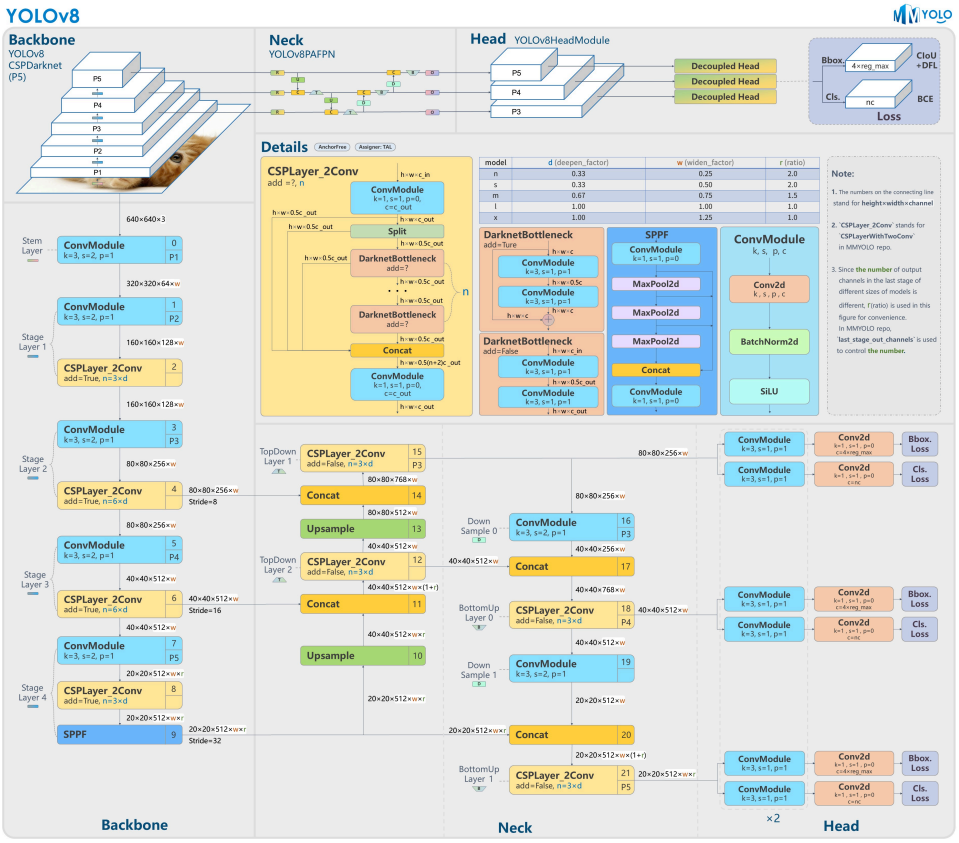

Figure 3: YOLOv8-P5 model structure

YOLOv8′s resilience to varied environmental factors such as lighting and occlusions makes it highly reliable for TSDR applications. The model's training on diverse datasets enables it to effectively recognize a wide range of traffic signs, even under challenging conditions. This versatility is essential for the complex and variable nature of road environments.

The model's streamlined integration and ease of use further enhance its applicability in TSDR systems. Its compatibility with various platforms allows for quick deployment in real-world scenarios, making it a valuable tool for advancing road safety and autonomous vehicle technologies. In essence, YOLOv8′s advancements present a substantial leap forward in the efficiency and reliability of Traffic Sign Detection and Recognition systems.

Squeeze-and-Excitation Attention[2]

The Squeeze-and-Excitation (SE) Networks represent a significant advancement in the field of CNNs. The central building block of CNNs is the convolution operator, which enables networks to construct informative features by fusing both spatial and channel-wise information within local receptive fields at each layer.

The SE Networks introduce a novel architectural unit, termed the "Squeeze-and-Excitation" (SE) block, that adaptively recalibrates channel-wise feature responses by explicitly modelling interdependencies between channels. This focus on the channel relationship, as opposed to the spatial component, sets SE Networks apart from other CNNs.

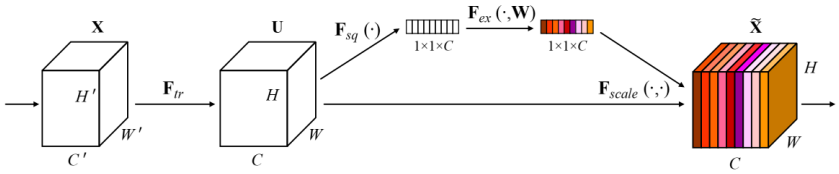

Figure 4: Squeeze-and-Excitation Block

These SE blocks can be stacked together to form SENet architectures that generalize extremely effectively across different datasets. Importantly, SE blocks bring significant improvements in performance for existing state-of-the-art CNNs at a slight additional computational cost.

Efficient Channel Attention[3]

The Efficient Channel Attention (ECA) model, proposed in the paper "ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks", represents a significant advancement in the field of deep convolutional neural networks (CNNs).

The ECA model introduces an lightweight attention module that boosts the performance of deep CNNs. This model is designed to overcome the paradox of performance and complexity trade-off often seen in CNNs. The ECA module involves only a handful of parameters but brings clear performance gain.

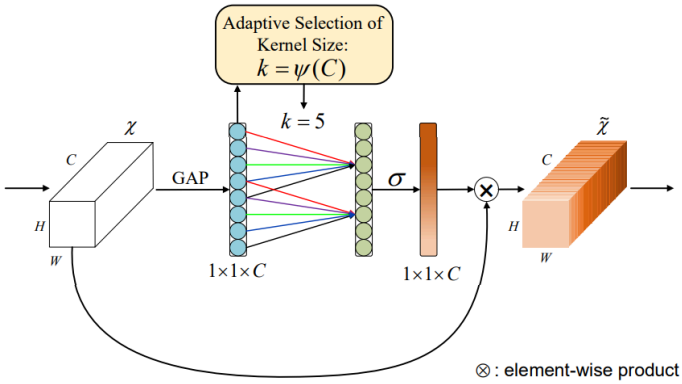

Figure 5: Squeeze-and-Excitation Block

By dissecting the channel attention module in Squeeze-and-Excitation Networks (SENet), the authors empirically show that avoiding dimensionality reduction is important for learning channel attention. They propose a local cross-channel interaction strategy without dimensionality reduction, which can be efficiently implemented via 1D convolution.

Novelties of This Work

In this project, two attention modules are introduced to enhance the performance of YOLOv8x over the TT-100K dataset[1] and a customize dataset, and data preprocessing methods for data augmentation and training set balancing.

From the aspect of innovating the structure of networks, although attention modules are proposed earlier than YOLOv8, there are still very few researches combining this mechanism with the state-of-the-art YOLOv8 model. In this project, we have modified the model architecture of YOLOv8 by adding two popular attention modules: Squeeze-and-Excitation and Efficient Channel Attention.

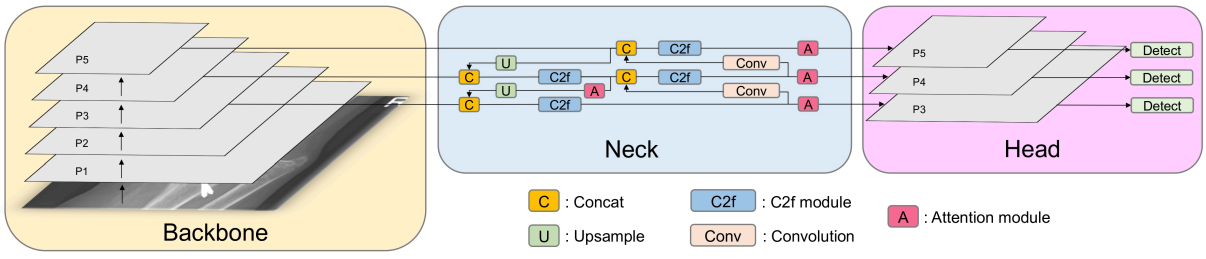

Figure 6: Improved YOLOv8x Structure

Even though the behavior of YOLOv8x is fast and accurate enough for TSDR tasks in daylight condition, the performance of the model decays greatly for testing set of night condition. Therefore this project proposes some data processing methods to improve the accuracy and capability for dim light condition.

We also analyze the TT-100K dataset and find that the dataset is unbalanced in terms of amounts of signs of different types, therefore we introduce data augmentation methods and collect and mark a customize dataset as extension to the TT-100K dataset.

System Setup

The training process was performed on 4 * RTX2080Ti with 10G vRAM each. The dataset consists of 232 classes, totaling approximately 6000 images. The training parameters include a batch size of 32, spanning a total of 200 epochs. Each epoch took around 70 seconds. Our training environment was built on a Python platform, leveraging packages such as ultralystics, PyTorch, OpenCV (cv2), and timm.

For more details, you may visit the repository for our experiments and follow the setup guide[4].

Proposed Method

Model Selection

At the first phase of our project, we review several popular models for object detection and recognition, including R-CNN[5], faster R-CNN[6], ResNet[7] and TSR-YOLO[8]. Hassam Tahir et al. proposed that in terms of precision, recall and accuracy, faster R-CNN is better than mask R-CNN, and the latter is better than ResNet50[9]. Also, a research comparing faster R-CNN, SSD, YOLOv4, YOLOv5 and YOLOv8 suggests that YOLOv8 outperforms all of the other models.[10]

Method	Backbone	Model Size	Accuracy for mAP 0.5	FPS	Average Inference Time	Batch Size	Input Resolution
Faster R-CNN	ResNet101	72.8 MB	100%	19	0.0528 seconds	8	600x600
SSD	Mobilenet-v2	9.2 MB	86%	433	0.00023 seconds	8	320x320
YOLOv4	CSPDarknet53	162.2 MB	96%	1083	0.0009 seconds	64	416x416
YOLOv4-Tiny	CSPDarknet53-Tiny	22.5 MB	97%	1015	0.0009 seconds	64	416x416
YOLOv5 small	yolov5s	14.1 MB	99%	1002	0.001 seconds	16	416x416
YOLOv8 nano	EfficientNet	6.2 MB	100%	1100	0.0009 seconds	16	640x640

Figure 7: Comparison between Mainstreaam Models

After reviewing the results above, Our choice for the baseline is YOLOv8. Not only it is open-source and easy to modify, but also it demonstrated proficiency in detection and recognition.

Model Preparation

Building upon the YOLOv8 baseline, our approach includes incorporating attention modules into the network architecture to further boost performance. Inspired by the findings in a wrist fracture detection[11] study, we explored two different attention mechanisms: Squeeze-and-Excitation (SE) and Efficient Channel Attention (ECA).

The integration of these attention modules resulted in two modified YOLO models, enhancing their capabilities in detection and recognition tasks.

As shown in Figure 8, we insert 4 attention modules into the neck network of YOLOv8, each after a C2f module to utilize the attention mechanism to enphasize or reduce the weights of diffrent parts of the images.

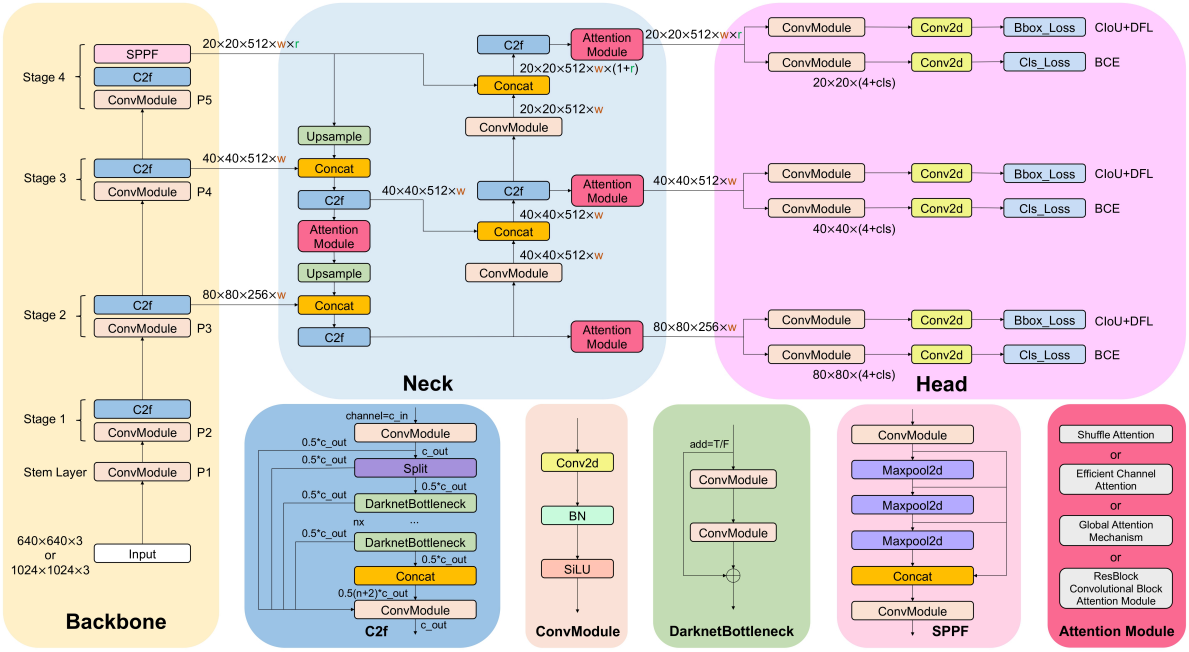

Figure 8: YOLOv8 Refined with Attention Modules

Dataset Preprocessing

Our training, testing, and validation dataset is based on the TT-100K[1] dataset, complemented by additional data collected from our campus. Since the original TT-100K dataset is not in YOLO format, we converted it into YOLO's YAML format. During the preprocessing phase, we observed that some labels had insufficient images. To solve this, we performed data augmentation techniques, including rotation, brightness adjustment, and zooming.

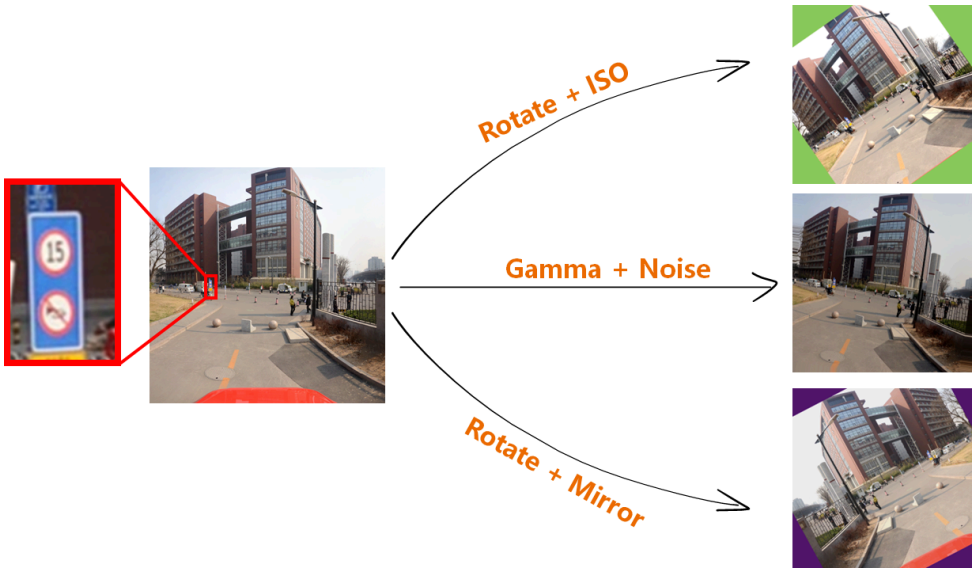

Figure 9: Example of Image Augmentation

For processing a night video, we can adjust the brightness, contrast to change the light in night close to daylight. We can also utilize CV methods like sharpening and HSV to emphasize the regions where the signs most likely to appear, leading to higher confidence and accuracy.

Origin Preprocessed

Figure 10: Enhance TSDR Performance Night by Image Preprocessing

Model Training

To develop our object detection model, we take the TT-100K dataset as the main source of the training, validating and testing dataset, and we also use tools to annotate bounding boxes and types of traffic signs for custom dataset then use it as extension of TT-100K.

Following this annotation, we divided the data randomly into training and validation sets, allocating 5587 images for training, 619 images for validation. The training set was instrumental in model training, whereas the validation set served to assess the model's efficacy.

Subsequently, we crafted configuration files to outline the model's architecture, hyperparameters, and training guidelines. The training phase commenced with the model being fed

image batches from the training set, enabling iterative weight adjustments. We vigilantly monitored the model's performance on the validation set during this phase, fine-tuning as needed. Our strategy also involved comparing various models to identify the most effective one. An essential prerequisite for training YOLOv8 with a custom dataset was the installation of the ultralytics package.

An epoch represents a complete iteration through the entire image dataset in YOLO. During each epoch, the YOLO model traverses all the bounding boxes in the dataset and updates its model parameters based on the loss function and optimization algorithm mentioned above. The higher the number of epochs, the more times the YOLO model will go through the entire dataset. However, the choice of the epoch number must be made carefully. A low epoch number can lead to underftting, where the model fails to capture important image features. Conversely, a high epoch number can result in overftting, where the model becomes overly biased towards the training data and performs poorly on new data. Therefore, selecting an appropriate epoch number often involves a trial-and-error process.

Results

Performance Comparison

To use trained model, you also needs to setup the same environment as training. Detect and recognition performance on a video is as follow:

on single RTX 2080 Ti: Speed: 2.1ms preprocess, 16.5ms inference, 0.9ms postprocess per image at shape (1, 3, 384, 640)

We splited the original dataset into train set and test set. So can conviniencely run model in test set to get its performance. So the final training result are as follows:

First, we trained two baseline models: YOLOv8 nano(YOLOv8n) and YOLO v8 Extra Large(YOLOv8x), with different model size. We set the number of epoches as 400 to let the models converges. The results show that the two models converge at around 280 and 230 epoches, respectively. The size of model files are 6MB vs. 130MB, corresponding to the difference in parameter numbers.

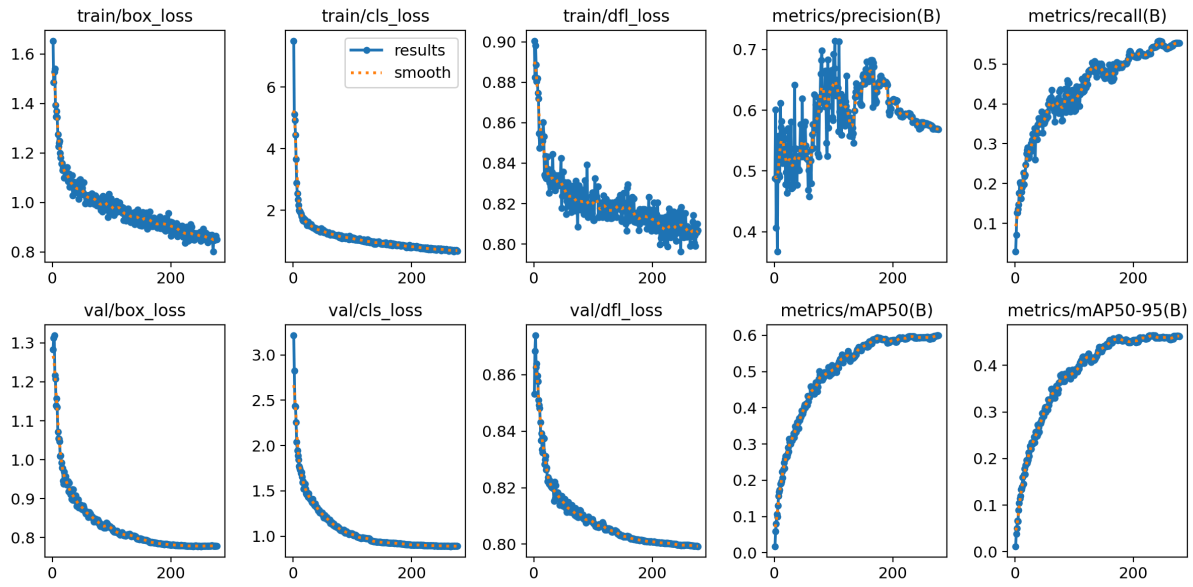

Figure 11: YOLO v8n Result

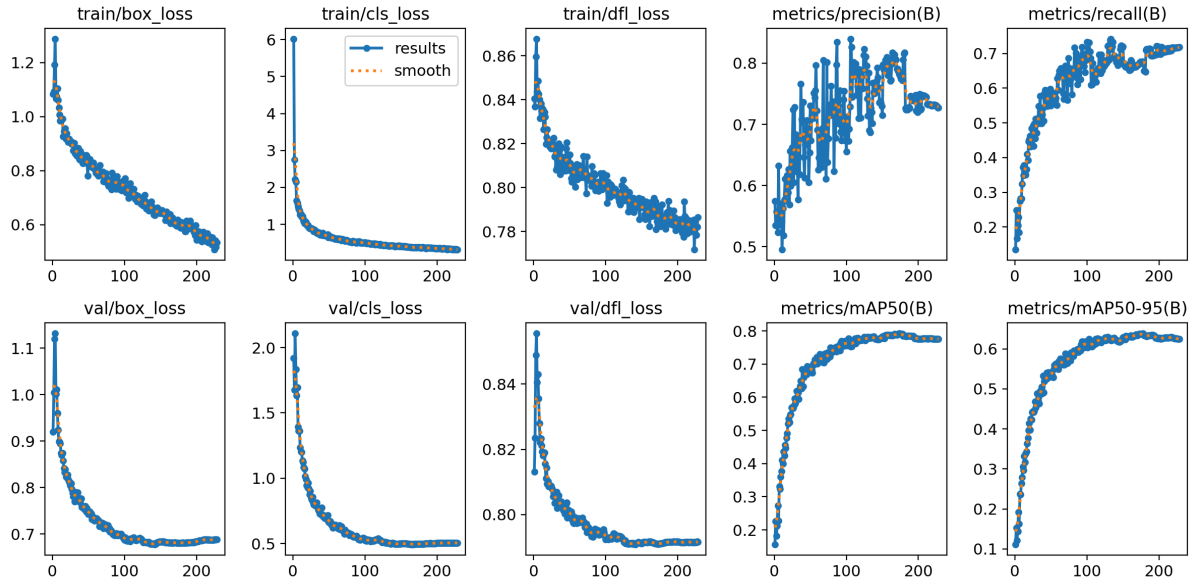

Figure 12: YOLO v8x Result

Based on the base model YOLO v8 Extra Large, we introduce SE attention modules and ECA modules to the neck network. Since the base model converges at less than 300 epoches, we set the limit of number of epoches at 200. The results are shown in Figure 13 and Figure 14. From the aspect of training efficiency, introducing the ECA modules does not affect the training time, while the SE net is about 15% slower. In terms of model size, the result models are both 130MB, which is the same as the base model YOLOv8x, while the runtime model size of YOLOv8x + SE is 4 times larger, taking up 522MB. The numbers of parameters list in order are: YOLOv8x + SE > YOLOv8x + ECA > YOLOv8x >> YOLOv8n.

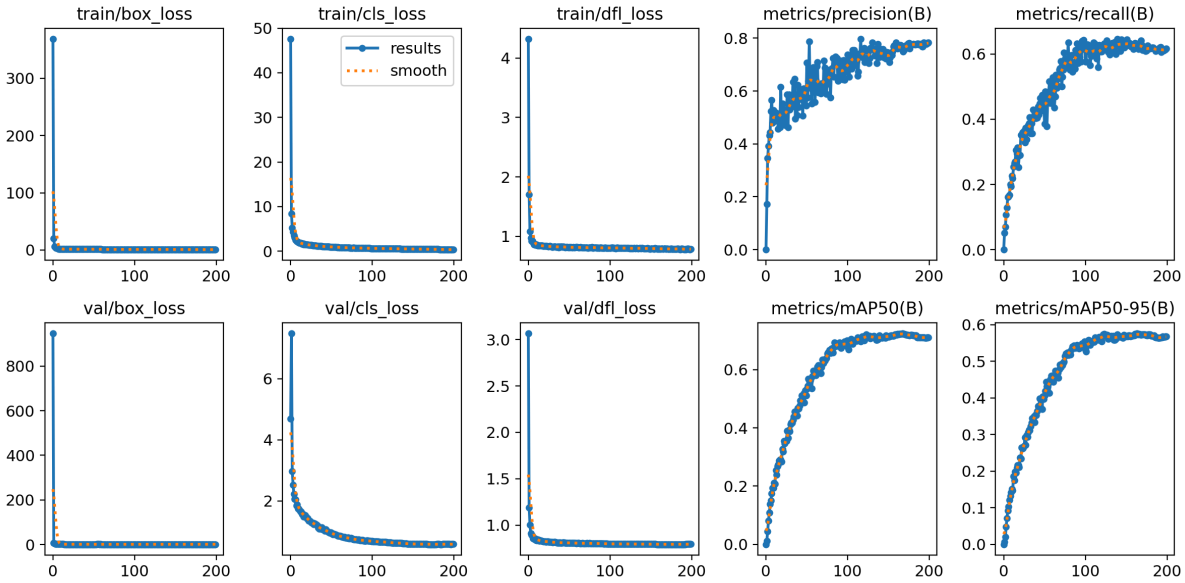

Figure 13: YOLO v8x+SE Result

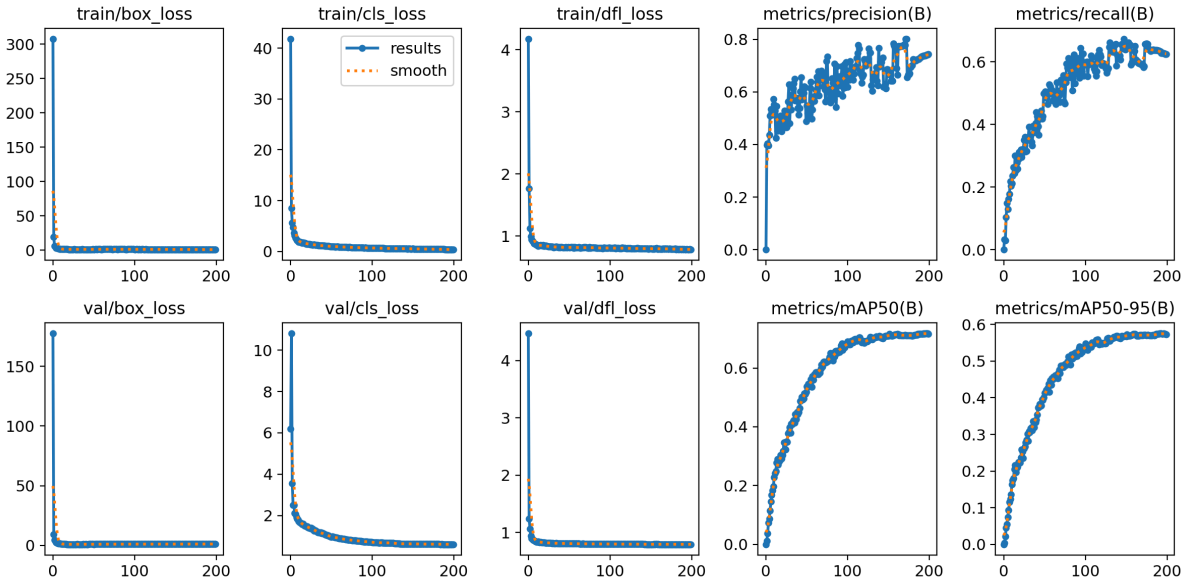

Figure 14: YOLO v8x+ECA Result

From the results shown in Table 1, even though the training epoches are cut in half, the YOLOv8x model refined with SE increases in terms of testing set precisions, and the performances of other metrics are close to those of YOLOv8x. Also, from the results of our custom testing set, the confidence and accuracy for realtime TSDR, the refined models are more suitable for TSDR tasks than YOLOv8x, which is good over test sets while can be confused by low confidence and accuracy in real cases.

Metrics	YOLOv8n (400 Epoches)	YOLOv8x (400 Epoches)	YOLOv8x_SE (200 Epoches)	YOLOv8x_ECA (200 Epoches)
metrics/precision(B)	0.575	0.727	0.783	0.742
metrics/recall(B)	0.553	0.717	0.617	0.624
metrics/ mAP50(B)	0.600	0.776	0.711	0.717
metrics/ mAP50-95(B)	0.462	0.625	0.568	0.574
val/box_loss	0.778	0.688	1.14	1.15
val/cls_loss	0.900	0.505	0.581	0.615
val/dfl_loss	0.800	0.792	0.793	0.792

Table 1: Performance Comparison between Models Trained in This Project

Experiments over Custom Dataset

YOLOv8x YOLOv8x_ECA

Figure 15: higher confidence and accuracy of ECA example1

YOLOv8x

YOLOv8x_ECA

Figure 16: higher confidence and accuracy of ECA example2

As shown in Figure 15 and Figure 16, the model refined with ECA modules outperforms the base model in terms of confidence for our custom test sets.

Discussion

1. Due to the time limit, changed models only trained half time as original model. Consider they having a higher metrics precision, having a longer training time may improve performance on other indicators.
2. From the result above, we can see that from YOLOv8 nano to YOLOv8 extra large, bigger model performance better.
3. Add attention module can improve performance in some conditions.
4. The introduction of attention modules to the neck part does not decrease the efficiency of inference.
5. Robustness: When testing a night video, all models fail to recognize signs due to lack of night dataset. From the aspect of improving performances over night images and videos, we come up with 2 solutions: add night datasets and datasets under different light conditions to the training set; preprocess the input image during inference using CV methods like HSV and sharpening.

Conclusion

In this project, we enhance the YOLOv8 framework for traffic sign detection and recognition. Our improvements included the integration of an ECA (Efficient Channel Attention) Attention Network and the implementation of data augmentation. These enhancements have increased accuracy, particularly in scenarios involving small objects or limited datasets.

However, our system still faces challenges in night conditions, where variable lighting significantly impacts performance. To address this, our future work will focus on two main areas: First, we aim to enrich our dataset with a wider range of lighting conditions, especially those representative of night-time scenarios. Second, we plan to develop and apply sophisticated color augmentation methods to better recognize and interpret faded traffic signs.

By addressing these challenges, we aim to push the boundaries of traffic sign detection and recognition, making it robust across all lighting conditions.

Acknowledgement

We express our deepest gratitude to Professor Hao, Teaching Assistant Daxing Wang and student assistants for their invaluable guidance, insightful feedback, and unwavering support throughout the Machine Learning(H) course. Their expertise and mentorship have been pivotal in shaping the direction and success of our learning and research.

This project stands as a testament to the collaborative effort and collective wisdom of all involved, and we are profoundly appreciative of the opportunity to work under such distinguished guidance and alongside such talented peers.

References

[1] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, “Traffic-Sign Detection and Classification in the Wild”, in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] J. Hu, L. Shen, S. Albanie, G. Sun, and E. Wu, “Squeeze-and-Excitation Networks”. 2019.

[3] Q. Wang, B. Wu, P. Zhu, P. Li, W. Zuo, and Q. Hu, “ECA-Net: Efficient Channel Attention for Deep Convolutional Neural Networks”. 2020.

[4] J. Xiao and S. Fan, “Experiments: Refining YOLOv8 with SE and ECA for TSDR”. [Online]. Available: https://github.com/Jayfeather233/ML-Project

[5] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation”, *CoRR*, 2013, [Online]. Available: http://arxiv.org/abs/1311.2524

[6] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks”. 2016.

[7] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition”. 2015.

[8] W. Song and S. A. Suandi, “TSR-YOLO: A Chinese Traffic Sign Recognition Algorithm for Intelligent Vehicles in Complex Scenes”, *Sensors*, vol. 23, no. 2, 2023, doi: 10.3390/s23020749.

[9] H. Tahir, M. Shahbaz Khan, and M. Owais Tariq, “Performance Analysis and Comparison of Faster R-CNN, Mask R-CNN and ResNet50 for the Detection and Counting of Vehicles”, in *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, 2021, pp. 587–594. doi: 10.1109/ICCCIS51004.2021.9397079.

[10] E. Soylu and T. Soylu, “A performance comparison of YOLOv8 models for traffic sign detection in the Robotaxi-full scale autonomous vehicle competition”, *Multimedia Tools and Applications*, 2023, doi: 10.1007/s11042-023-16451-1.

[11] R.-Y. Ju and W. Cai, “Fracture Detection in Pediatric Wrist Trauma X-ray Images Using YOLOv8 Algorithm”. 2023.