Robust and Efficient Detector for Traffic Lights

Qi Yu
qiyu6@illinois.edu

Junda Shen
jundas2@illinois.edu

Zijian Zhang
zhang417@illinois.edu

Hanwen Zhang
hanwen6@illinois.edu

Abstract

Traffic lights are common in everyday life. It's crucial to recognize them and then act as they signal. For some modern technologies like autonomous driving, accurately detecting and recognizing traffic lights in real-time is a key factor to determine whether that technology is successful or not. In our project, we used deep learning-based approaches to implement real-time traffic light detection with high efficiency and robustness under various scenarios. We also compared our approach with some previous methods to show the feasibility of our approach.

1. Introduction

Traffic lights are crucial for maintaining traffic order and safety, it is important for pedestrians and vehicles to act according to the traffic lights. Traffic light detection technology can automatically recognize and classify traffic lights, and can give drivers and autonomous driving systems signals to act in the proper way. Therefore, research on traffic light detection in real-time with high accuracy and confidence can have a significant impact. The minimum goal of our project is to detect the traffic lights accurately under normal scenarios, which can be a baseline for our project.

After achieving the minimum goal, we also enhanced our traffic light detector to make it more applicable in Autonomous Vehicles. Due to the variety of traffic lights and the interference from the environment, traffic light detection can still be challenging under certain scenarios. The first challenge we encountered is detecting traffic lights with poor lighting conditions, e.g. in the evening or having a backlight. Then we need to achieve detection in real-time without losing too much accuracy. We tried to solve those problems in different ways, including resampling the dataset to make it more balance across classes, selecting the proper deep learning models to improve performance, and modifying code structure to realize real-time detection.

2. Related Work

Before the large-scale application of deep learning, there have already been some traffic light detection algorithms aimed at helping recognize the signals. There were some algorithms that look for specific traffic light features like rule-based edge detection and curve circle detection, as proposed by Walad [22]. Omachi *et al.* [15] developed an algorithm to normalize the RGB color space so that a series of traffic light candidates can be captured, then they used Hough Transformation to detect the traffic lights from the previous candidates. Some researchers [7, 20] used different color spaces to represent the images, which potentially offered more opportunities to detect the traffic lights. Other than making too many efforts on traffic light colors, Lee *et al.* [9] used Haar-like features and SVM classifiers to show it is possible to detect and recognize traffic lights without the color information. As mobile computing continuously grows, the demand for mobile navigation applications increased drastically and information about traffic lights can be a wonderful add-on. To easily obtain traffic light information while filtering out false traffic light candidates, researchers [4, 11] found a priori knowledge useful in detecting traffic lights, since most of the traffic lights have similar features.

After the blowout development of deep learning, researchers noticed the potential of deep learning in detection problems. Most of the outstanding detection algorithms now use deep neural networks (DNN) as their core architectures, and use DNN to extract features, locate and classify images and videos. From the overall perspective, object detection can be divided into two- and single-stage algorithms. The most representative of the two-stage algorithms is Faster R-CNN [5], and the most representative of the single-stage algorithms is YOLO [16] and RetinaNet [10]. It was commonly believed that the two-stage algorithm has a relative advantage in detection accuracy, while the single-stage algorithm has a relative advantage in detection speed, so these two kinds of algorithms were usually used in different scenarios to achieve better performance on specific

metrics.

For detection of traffic lights with DNN, Janahiraman *et al.* [6] used Faster R-CNN to deliver accurate detection results while Tran *et al.* [21] used YOLOv4 [1] to achieve 30 fps real-time detection. Their work shows the different properties of the two networks. Naimi *et al.* [13] optimized the feature maps of SSD [12] to have a better trade-off between detection precision and efficiency. In our project, we show that modern single-stage detection algorithms can outperform two-stage algorithms from both efficiency and accuracy perspectives.

3. Methods

In this section, we first briefly introduce the baseline algorithm used for our traffic light detection tasks. Then, we analyse the drawbacks of the baseline methods, including data imbalance, poor detection performance at night. For the following subsections, we propose several solutions to the aforementioned deficiencies. In addition, we modified our baseline YOLOv5 algorithm to enable real-time detection of traffic lights. The results are shown in Section 4.

3.1. Baseline YOLOv5

We chose YOLOv5 as the baseline algorithm for our traffic detection tasks.

YOLO (You Only Look Once) [16] is a single-stage object detection algorithm that proposes the use of an end-to-end neural network that performs predictions of bounding boxes and class probabilities all at one stage. YOLO algorithm works by first dividing the image into grids having an equal dimension of S x S each, as shown in Figure 1. Each of these grids is responsible for the detection and localization of the object it contains. Correspondingly, these grids predict B bounding box coordinates relative to their cell coordinates, along with the object label and probability of the object being present in the cell. Because of such mechanism, YOLO naturally brings forth a lot of duplicate predictions in each cell and it makes use of Non-Maximal Suppression to remove such duplications. The original YOLO algorithm came with the limitation of struggling to detect and segregate small objects in images that appear in groups. YOLOv2 [17] was proposed to fix such issues by allowing the prediction of multiple bounding boxes from a single cell, as opposed to one prediction per cell in the original YOLO algorithm. It also added batch normalization layers into the original YOLO network and it successfully improved the mean Average Precision (mAP). YOLO9000 was proposed along with YOLOv2 to be trained simultaneously on the COCO detection dataset and the ImageNet classification dataset, to enable detection and predictions for object classes that don't have labeled detection data. YOLOv3 [18] further improve the YOLO family by introducing modern convolutional neural networks such as residual networks and skip connections. YOLOv4 [1] proposed the addition of several techniques compared to YOLOv3, including Weighted Residual Connections (WRC), Cross Stage Partial connections (CSP), Cross mini-Batch Normalization (CmBN), Self-Adversarial Training (SAT), and Mish activation. YOLOv5 is an open-source project [8] developed by Ultralytics that consists of a family of object detection models based on the YOLO model pre-trained on the COCO dataset.

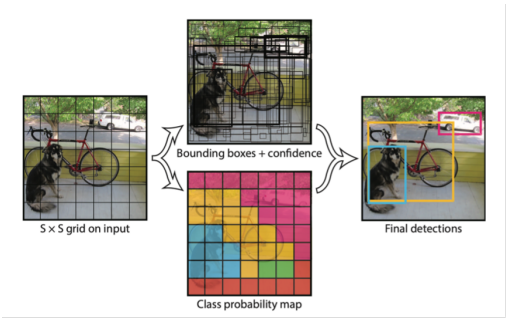

Figure 1. Illustrations of the YOLO algorithm [16]

For our project, we chose YOLOv5 as the baseline algorithm for our traffic lights detection task. Specifically, we selected the second-smallest model YOLOv5s to balance between effectiveness and efficiency. We established our training and test datasets based on S^2TLD [24] and used them to train our detection models.

3.2. Data Imbalance

Data imbalance has been a fundamental problem in Object Detection tasks. Oksuz *et al.* [14] grouped the imbalance problem in Object Detection into four main categories: class imbalance, scale imbalance, spatial imbalance, and objective imbalance. The three most common solutions for the foreground-foreground class imbalance problem, which is what we face in this project, are hard sampling, soft sampling, and using generative models. For hard sampling, it direct drops bounding boxes for certain classes to balance the number of instances among different classes. For soft sampling, it adjusts the contribution of different classes by assigning different weights to different classes in the loss function. For the generative-model method, it generates artificial instances using generative model such as GAN to balance the original dataset.

For the traffic light detection and classification problem, datasets collected are usually imbalanced because yellow and off lights simply appear less frequently than red and green lights, thus making it hard to collect the former two kinds of data. In our project, the original S^2TLD dataset [24] contains 6473 instances of red light, 4528 instances of green light, 190 instances of yellow light, and 503 instances of off (no) light. The numbers of instances of red

and green light are significantly higher than that of yellow and off light. Therefore, models trained on the original S^2TLD dataset are likely to be biased toward red and green lights and yielding poor performance in detecting yellow and off lights.

To tackle the imbalance issue, we have experimented with two possible solutions, which are hard sampling and soft sampling.

3.2.1 Hard Sampling

For the first and most straightforward solution, we use the hard sampling method and tried to balance the dataset by simply reducing the instances of red, green, and off lights. We balanced the dataset by first selecting all the images that contain instances of yellow light (least of the four labels). Then, we iteratively added images from the original dataset until our balanced dataset contain a similar number of instances for the four classes.

The resulting balanced dataset contains 308 instances of red light, 204 instances of green light, 190 instances of yellow light, and 199 instances of off (no) light.

3.2.2 Soft Sampling

For the second solution, we made use of *Focal Loss* proposed by Lin *et al*. [10]. The focal loss is designed to address the data imbalance issues in the one-stage Object Detection scenario. The focal loss is defined over the cross entropy loss as:

$$\mathrm{FL}(p_t) = -(1-p_t)^{\gamma}\log(p_t) \tag{1}$$

where

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise} \end{cases} \tag{2}$$

where $y \in \{\pm 1\}$ specifies the ground-truth label and $p \in [0, 1]$ specifies the estimated probability for the class labeled $y = 1$. In practice, the α-variant of the focal loss is usually adopted, which is

$$\mathrm{FL}(p_t) = -\alpha_t(1-p_t)^{\gamma}\log(p_t) \tag{3}$$

The above notations are based on binary classification and we can easily extend them to multi-class classification tasks.

For our project, we enabled focal loss in our baseline YOLOv5 model and trained it on the same imbalanced dataset as where the baseline method is trained. Specifically, We set $\gamma = 1.5$ as the YOLOv5 model suggested.

3.3. Real Time Detection

For real-time detection, high performance and accuracy are required. Compare to YOLOv5, YOLOv7 is 127 FPS faster and 10.7% more accurate on AP [2]. We chose to use YOLOv7 [23] object detection model for the real-time system. This model is state-of-the-art in real-time object detection.

Although it can detect traffic lights, it cannot consider signals from traffic lights. Therefore, we need to make it able to understand which signals are coming from traffic lights. We use a rule-based approach to make the system understand the signals. Figure 2 shows some examples of traffic signal lights. The system understands the signal of the traffic light through the following process [19].

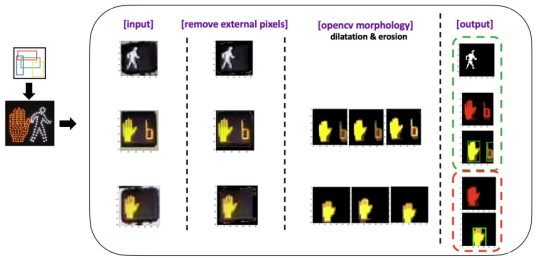

Figure 2. Image processing to detect the signals from the traffic light

First, by detecting traffic lights with the YOLOv7 model, we can find the input image with different colors and shapes of the traffic signal lights. Secondly, the external pixels are removed, which means it makes the edge pixels darker. Then, the signal is made clearer by repeating the expansion and erosion using the OpenCV morphology library so that the system clearly identifies the traffic signal light. Finally, recognize it in red, yellow and green.

3.4. Poor Detection Performance at Night

During our initial training of YOLOv5, we found that the model has significant poor performance on detecting traffic light at night. The model generally has low confidence level or even cannot detect traffic light in night images.

Figure 3 shows an example of the model failing to detect traffic light in night images. However, since we only use 550 images in the initial training which we think is too small to train the model completely, we then decide to increase the training set and propose the baseline yolov5 by randomly choosing 2000 images. However, although increasing the training set size improves the performance of the model by allowing yolov5 be able to detect traffic light at night in most scenarios, it still generally has low confidence level. After manually checking the data set, we found that it is because our data set has most images in day time

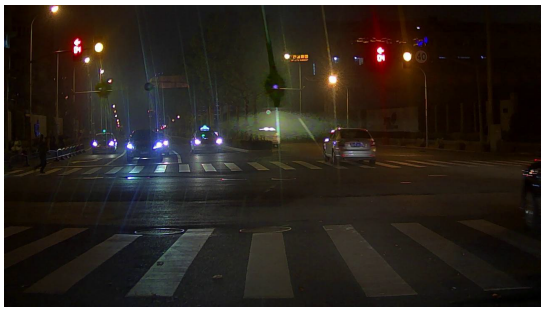

Figure 3. Example of missing Detection on traffic light in night image

while only a small portion of it is in night time. Therefore, by hard sampling the training set, we make sure that there are roughly same amount of night time images and day time images while still randomly choose enough images from data set to train the model.

4. Experiment Settings & Results

4.1. Baseline YOLOv5

For the our baseline YOLOv5 algorithm, we first randomly selected 2,000 images from the original S^2TLD as our training and validation set, and 200 more images as our test set. The statistics of the training and test set are shown in Table 1. We implemented the baseline YOLOv5 algorithm based on the open-source Ultralytics YOLOv5 project [8], and have deployed it on our baseline dataset. We used Matpool https://matpool.com as our cloud sever, and selected NVIDIA RTX A4000 GPU for the trainings our models.

Traffic lights	Training set	Test set
Red	2806	272
Yellow	81	13
Green	1981	188
Off	201	18

Table 1. Statistics of the baseline training and test set

To demonstrate the effectiveness of our baseline YOLOv5s model, we also implement Faster-RCNN and RetinaNet model and trained them using the baseline training set. The resulting performance of these three models are shown in Table 2.

We can see from the results that our baseline YOLOv5 model outperforms Faster-RCNN and RetinaNet significantly in term of mAP. We also found that the performance of our baseline model exceed the YOLOv3 model in [24] in term of mAP50-95 by 12%, demonstrating the effectiveness of our baseline methods. The training curves of the baseline YOLOv5 model are shown in Figure 4. An example of detection results are shown in Figure 5.

Method	mAR50	mAP50	mAP50-95
YOLOv5s	0.956	0.98	0.691
Faster-RCNN	0.641	0.888	0.529
RetinaNet	0.683	0.880	0.561

Table 2. Detection results

4.2. Data Imbalance

For the data imbalance issues encountered in our project, we experimented with two possible solutions motioned in Section 3.2, which are hard and soft sampling methods. For the soft sampling methods, the statistics of the balanced dataset is shown in Table 3. Note that to compare the performances of YOLOv5 models trained on different settings and datasets, we chose the same imbalanced test set for these three models.

Traffic lights	Training set	Test set
Red	283	272
Yellow	177	13
Green	194	188
Off	189	18

Table 3. Statistics of the balanced training and test set

The resulting performance of the baseline and the proposed two solutions are shown in Table 4. We can see from Table 4 that training on manually balanced dataset did improve the recall of the underrepresented labels such as the off traffic lights, from 0.889 to 1. This demonstrates that the YOLOv5 model trained on balanced dataset are more capable of detecting underrepresented classes than the baseline model. However, we did notice that the baseline model outperforms the model trained on the balanced dataset in term of precision of each class (as well as mAP). This is probably because of the differences in size between the baseline and balanced dataset. By comparing statistics between Table 1 and Table 3, we can see that there are significantly more instances of red, green and off lights in the baseline dataset. Therefore, the baseline model are more sufficiently trained than the model trained on the balanced dataset, making it outperforms in term of mAP.

For the soft sampling methods, we introduced the focal loss into the baseline YOLOv5 model and set the hyperparameter $\gamma = 1.5$. We trained the corresponding model on the same imbalanced dataset as the baseline. We can see from Table 4 that introducing focal loss does not improve the overall performance of our implemented

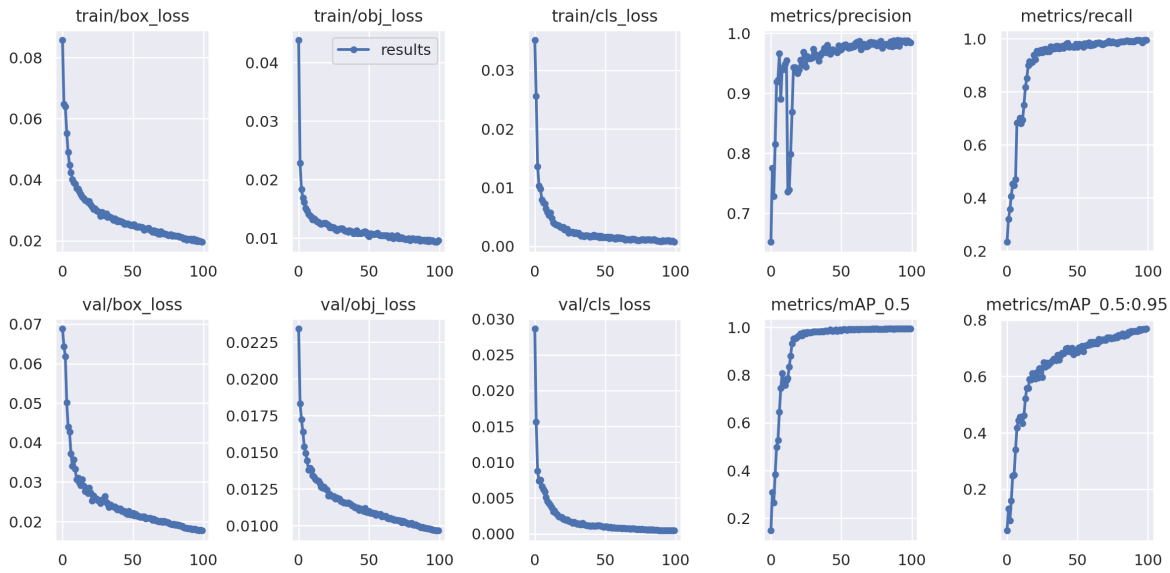

Figure 4. Training curves for baseline YOLOv5 model.

Figure 5. Baseline detection examples

YOLOv5 model. This is probably because that hyperparameter evolution is usually required along with the introduction of focal loss to boost the model's performance, whereas the computational resources of our project are too scarce to do hyperparameter tuning. It's also worth mentioning that the mAR for yellow lights are 0 in test set for focal loss model due to the imbalance nature of our test set. We can see from Table 1 that there are only 13 instances of yellow lights and 18 instances of off lights, which can lead to extreme test results. There are also some related works challenging the necessity of soft sampling methods such as focal loss and GHM. Chen *et al.* [3] reveal that the performance degradation cause by data imbalance comes from an unreasonable classification gradient magnitude, rather than a lack of re-sampling / re-weighting. They proposed a Sampling-Free mechanism to achieve a reasonable classification gradient magnitude by initialization and loss scaling. This could be a potiential solution to the data imbalance issue encounter in our traffic light detection task.

4.3. Output Result Of Real Time Detection

We use 24 frames per second video, and the goal criterion to satisfy the real time system condition is that the average latency per frame is less than 40 ms. As seen as figure 6, the maximum of the total operation time per frame is 29.5ms. 29.5ms is a number that meets our criteria. Furthermore, the pedestrian movement intention is predicted once every 16 frames, the actual operating time would be

YOLOv5s	Red		Yellow		Green		Off		All	
	mAR50	mAP50	mAR50	mAP50	mAR50	mAP50	mAR50	mAP50	mAR50	mAP50
Baseline	0.977	0.990	1.000	0.995	0.959	0.987	0.889	0.947	0.956	0.980
Hard sampling	0.849	0.954	1.000	0.948	0.854	0.936	1.000	0.921	926	0.940
Soft sampling	0.978	0.983	0.000	0.619	0.952	0.984	0.848	0.920	0.695	0.877

Table 4. Performance of different models aimed at resolving data imbalance issues

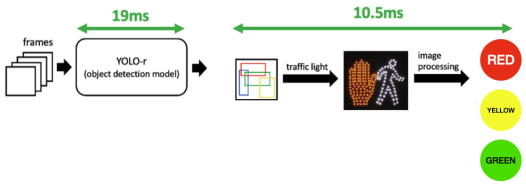

Figure 6. Image processing to detect the signals from the traffic light

Figure 7. System out put example

less than 29.5ms.

We test our model by using a traffic signal light video. Figure 7 show the result output of our system. The system can detect the traffic light signal well. If the traffic light turns red, a red box will appear around the traffic signal light. If the traffic light turns yellow, a yellow box will appear around the traffic signal light. If the traffic light turns green, a green box will appear around the traffic signal light. And a sample result video has been uploaded to this link or go to the following url: https://drive.google.com/file/d/10y7MB6bGoZ-IQW1czZ_sVGUcGWcKAQw8/view?usp=sharing. As shown in the output video, our system can detect the traffic light signals very well.

4.4. Result of Improve Detection Performance in Night Images

We use the same data set settings and model settings as discussed before in section 4.1. The only difference is that we manually balance the amount of night images and day time images in the training set. Under this premise, we then randomly choose images from data set to train the model. The result shows that by manually balance the amount of night images and day time images which increases the portion of night images in training set compared to the baseline model, the model can recognize and detect traffic light in night images in most scenarios. Moreover, the confidence level of the detection also significantly increases. Figure 8 and Figure 10 show the result of baseline model while Figure 9 and Figure 11 show the result of improved model correspondingly

Figure 8. Baseline Model

Figure 9. Improved Model

Figure 10. Baseline Model

Figure 11. Improved Model

Method	mAR50	mAP50	mAP50-95
Baseline	0.725	0.744	0.528
Improved	0.641	0.888	0.529

Table 5. Baseline and Improved Model Result

According to the Table 5, it clearly shows that the improved model has better overall performance than the baseline model. However, the baseline model has strangely low performance which we think is due to the size of the training set. Therefore, we should test both models again with larger data set but the table indeed shows that improved model is better than baseline model.

5. Conclusion & Discussion

This project shows the feasibility of detecting traffic lights accurately and efficiently with a single-stage detector. We used different sampling techniques to alleviate the data imbalance issue so that the models have lower bias during training. We also tried to improve the detection confidence in some scenarios and we presented the comparison results for images that were taken in the evening. We also realized real-time detection of traffic lights, which can provide robust signals to autonomous driving applications.

For our future work, we can further resolve our data imbalance issue by collecting more training data for our deep learning model. In an engineering perspective, this might be a time-consuming yet the simplest and most effective approach to tackle data imbalance issue. We could also try hyperparameter tunning if computational resources are sufficient. In addition, the Sampling-Free mechanism proposed by Chen *et al*. [3] could be another potiential solutions to our data imbalance issue. Considering it's intrinsically hard to collect as many yellow lights instances as red and green light instances, such soft sampling technique and modifications in an algorithmic level could be effective solution to the data imbalance issue encountered in the traffic light detection tasks. We can also try to solve the issue that some traffic lights are blocked by other objects, this can be achieved by interpolating previously detected traffic light coordinates then use math methods to predict the blocked traffic light. By interpolating the image frames, we can further provide more robust predictions.

6. Individual Contribution

Qi Yu was responsible for the training and testing of the baseline YOLOv5 model as well as solving data imbalance issues. He was also responsible for writing section 3, 4 and 5 of the paper that corresponds to the baseline YOLOv5 model and data imbalance issue.

Junda Shen was responsible for the training and testing of Faster-RCNN and RetinaNet model. He was also responsible for writing the Abstract, Related Work, as well as Conclusion sections in our paper.

Zijian Zhang was responsible for solving issues about poor detection at night and training (testing) the corresponding models. He was also responsible for writing section 3, 4 and 5 of the paper that corresponds to the poor detection at night issues.

Hanwen Zhang was responsible for extending our model to handle real time detection. He was also responsible for writing section 3, 4 and 5 of the paper that corresponds to the real time detection.

The names metioned above are NOT listed in order of contribution to this project.

References

[1] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. 2

[2] Gaudenz Boesch. Yolov7: The most powerful object detection algorithm (2022 guide), 2022. 3

[3] Joya Chen, Dong Liu, Tong Xu, Shiwei Wu, Yifei Cheng, and Enhong Chen. Is heuristic sampling necessary in training deep object detectors? *IEEE Transactions on Image Processing*, 30:8454–8467, 2021. 5, 7

[4] Nathaniel Fairfield and Chris Urmson. Traffic light mapping and detection. In *2011 IEEE International Conference on Robotics and Automation*, pages 5421–5426. IEEE, 2011. 1

[5] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 1

[6] Tiagrajah V Janahiraman and Mohamed Shahrul Mohamed Subuhan. Traffic light detection using tensorflow object detection framework. In *2019 IEEE 9th International Conference on System Engineering and Technology (ICSET)*, pages 108–113. IEEE, 2019. 2

[7] Chulhoon Jang, Chansoo Kim, Dongchul Kim, Minchae Lee, and Myoungho Sunwoo. Multiple exposure images based traffic light recognition. In *2014 IEEE intelligent vehicles symposium proceedings*, pages 1313–1318. IEEE, 2014. 1

[8] Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, imyhxy, Lorna, Zeng Yifu, Colin Wong, Abhiram V, Diego Montes, Zhiqiang Wang, Cristi Fati, Jebastin Nadar, Laughing, UnglvKitDe, Victor Sonck, tkianai, yxNONG, Piotr Skalski, Adam Hogan, Dhruv Nair, Max Strobel, and Mrinal Jain. ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation, Nov. 2022. 2, 4

[9] Sang-Hyuk Lee, Jung-Hawn Kim, Yong-Jin Lim, and Joonhong Lim. Traffic light detection and recognition based on haar-like features. In *2018 International Conference on Electronics, Information, and Communication (ICEIC)*, pages 1–4. IEEE, 2018. 1

[10] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pages 2980–2988, 2017. 1, 3

[11] Frank Lindner, Ulrich Kressel, and Stephan Kaelberer. Robust recognition of traffic signals. In *IEEE Intelligent Vehicles Symposium, 2004*, pages 49–53. IEEE, 2004. 1

[12] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016. 2

[13] Humaira Naimi, Thangarajah Akilan, and Mohammad AS Khalid. Fast traffic sign and light detection using deep learning for automotive applications. In *2021 IEEE Western New York Image and Signal Processing Workshop (WNYISPW)*, pages 1–5. IEEE, 2021. 2

[14] Kemal Oksuz, Baris Can Cam, Sinan Kalkan, and Emre Akbas. Imbalance problems in object detection: A review. *IEEE transactions on pattern analysis and machine intelligence*, 43(10):3388–3415, 2020. 2

[15] Masako Omachi and Shinichiro Omachi. Traffic light detection with color and edge information. In *2009 2nd IEEE International Conference on Computer Science and Information Technology*, pages 284–287. IEEE, 2009. 1

[16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. 1, 2

[17] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. 2

[18] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 2

[19] Kreimir Romic, Irena Galic, Hrvoje Leventic, and Kreimir Nenadic. Real-time multiresolution crosswalk detection with walk light recognition for the blind. *Advances in Electrical and Computer Engineering*, 18(1):11–20, 2018. 3

[20] Hwang Tae-Hyun, Joo In-Hak, and Cho Seong-Ik. Detection of traffic lights for vision-based car navigation system. In *Pacific-Rim Symposium on Image and Video Technology*, pages 682–691. Springer, 2006. 1

[21] Tai Huu-Phuong Tran and Jae Wook Jeon. Accurate real-time traffic light detection using yolov4. In *2020 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia)*, pages 1–4. IEEE, 2020. 2

[22] Kavya P Walad and Jyothi Shetty. Traffic light control system using image processing. *International Journal of Innovative Research in Computer and Communication Engineering*, 2(5):289, 2014. 1

[23] Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*, 2022. 3

[24] Xue Yang, Junchi Yan, Wenlong Liao, Xiaokang Yang, Jin Tang, and Tao He. Scrdet++: Detecting small, cluttered and rotated objects via instance-level feature denoising and rotation loss smoothing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 2, 4