Real-Time Street Parking Sign Detection and Recognition [*]

Hieu Chau[†] , **Yin Jin**[†] , **Jiayu Li** , **Juhua Hu**[‡] and **Wei Cheng**
Tacoma School of Engineering and Technology, University of Washington, WA, USA
{hqchau, jinyin, jiayu7, juhuah, uwcheng}@uw.edu

Abstract

Street parking sign detection and recognition are beneficial to autonomous driving and for current drivers, though it is a more challenging task than traffic sign detection due to its diversity and complexity. In this paper, we address the problem of limited public street parking sign data by collecting street parking signs across cities in United States. We then study the feasibility of commonly adopted deep learning methods on two related tasks in the real-time application scenario. That is, street parking sign detection from street level images or videos, and parking sign symbol object detection for precise understanding of each parking sign. We find that YOLO is more applicable in terms of detection performance (i.e., 96.8% AP@0.5 for parking sign detection and 98.3% mAP@0.5 for symbol detection), detection speed in real-time (i.e., 163 and 88 frames per second), and model size (i.e., 14.4MB and 68.4MB) fitting in small devices like smartphones or dash cameras.

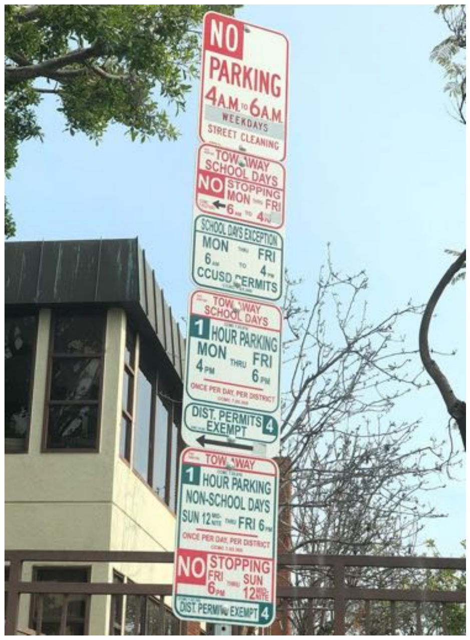


Figure 1: A stack of parking signs[1]

1 INTRODUCTION

Automatic detection and recognition of street parking signs are essential not only for autonomous driving in the future but also for drivers in the present looking for parking in the middle of complex traffic systems on a daily basis. While street parking signs tend to convey similar messages, their styles are vastly different across cities and states in the United States. Furthermore, some signs can get complicated (e.g., Fig. 1) through information stacking and distract even the most experienced drivers, potentially leading to dangerous situations or traffic congestion. Traffic sign detection was an extensively researched topic in recent decades [Sanyal *et al.*2020]. Though street parking sign detection and recognition is a similar task, it is a relatively unexplored problem. Street parking signs typically include multiple symbols with various texts as opposed to traffic signs that are designed to be fairly simple for quick user interpretation. Moreover, street parking signs are extremely diverse, making its recognition task more challenging.

To the best of our knowledge, there is no prior work which intends to cover all the common and meaningful parking symbols, so as to generate a precise interpretation for each street parking sign. For example, [Jiang2019, Li2020, Li *et al.*2021] studied the detection of only no parking symbols and direction arrows. [Faraji *et al.*2021], on the other hand, handled the street parking sign recognition task by assuming that each sign conveys one single parking rule of either No-Parking, Parking-Allowed, or No-Stopping. However, this assumption

[*]Yin, Jiayu, and Juhua are partially supported by KenSci Gift Funding. All opinions, findings, conclusions and recommendations in this paper are those of the author and do not necessarily reflect the views of the funding agencies.

[†]Equal contribution

[‡]Corresponding author

[1]https://www.reddit.com/r/CrappyDesign/comments/bvjz9u/the_parking_signs_in_la_are_next_level_crap/

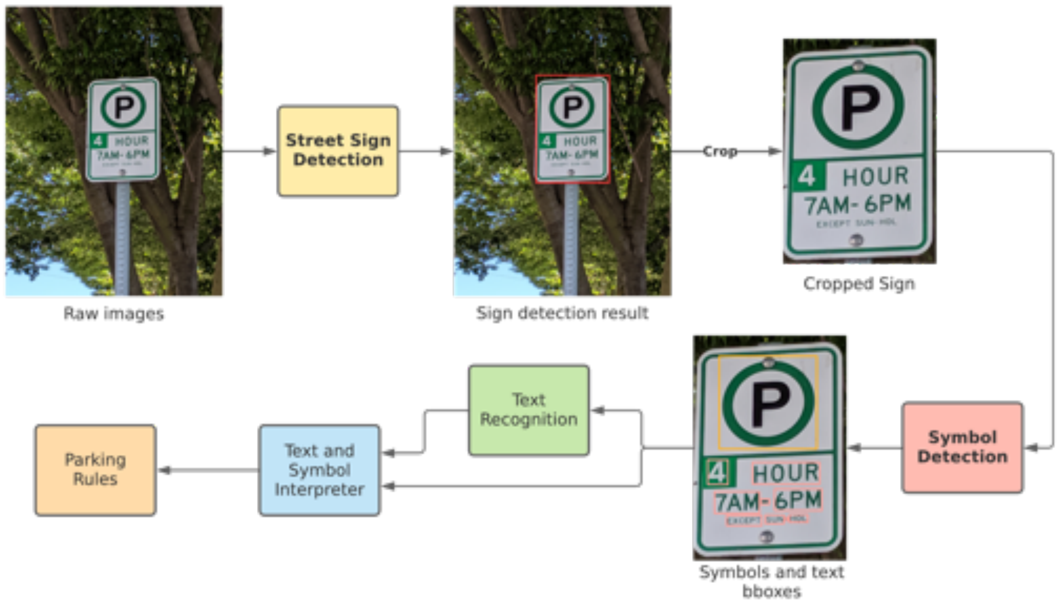


Figure 2: The parking sign interpretation pipeline. This paper evaluates the tasks of Sign Detection and Symbol Detection.

does not hold when a parking sign contains multiple symbols or is divided into sections (e.g., Fig. 1). Therefore, to fully understand a parking sign, in this work, we aim to extract as much information as possible with a scalable design for the real-time application.

Specifically, we address the lack of public data for street parking signs by collecting street parking signs across cities in the United States. Based on that, we study the feasibility of precise understanding of each parking sign in real-time using commonly adopted deep learning methods, such as RetinaNet [Lin *et al.*2018], YOLOv5 [Jocher2021], and Swin Transformer [Liu *et al.*2021]. Concretely, we propose a basic framework for street parking sign understanding as shown in Fig. 2, which is started by parking sign detection from street level images or videos. Then, for each detected parking sign, we aim to detect all symbols including texts painted on street parking signs. We find that for both street parking sign detection and parking symbol detection, YOLOv5 [Jocher2021] is more applicable in terms of detection performance (i.e., 96.8% AP@0.5 for parking sign detection and 98.3% mAP@0.5 for symbol detection), detection speed in real-time (i.e., 163 frames per second for parking sign detection and 88 frames per second for symbol detection), and model size (i.e., 14.4MB for parking sign detection model and 68.4MB for symbol detection model) fitting in small devices like smartphones or dash cameras.

2 RELATED WORK

2.1 Object Detection

Object detection is one of the most significant branches in computer vision, solving two key tasks: (1) detect the presence and location of specific objects in an input image and (2) recognize the category of objects. Due to the lack of effective image representation, traditional object detection methods [Viola and Jones2001, Dalal and Triggs2005, Felzenszwalb *et al.*2010] were built on handcrafted features. Since [Krizhevsky *et al.*2012], deep convolutional neural network (CNN) has largely improved the performance of object detection because of the capability of learning robust and high-level feature representations of an image. Generally, object detection is grouped into two genres: "two-stage detection" and "one-stage detection".

Two-stage detection frameworks divide object detection into two phases: (1) extract a set of object proposals as the regions of interest, and (2) feed proposals into CNNs to predict the presence and the category of objects. Regions with CNN (R-CNN) [Girshick *et al.*2014], as a typical two-stage detection architecture, employs selective search [Uijlings *et al.*2013] to generate up to approximately 2,000 candidate object boxes. Then each candidate box is resized to a fixed-size image and fed into a pre-trained CNN model to extract features. Finally, linear SVM classifiers are used to predict the presence of an object within each region and to recognize object categories. Even though R-CNN achieved staggering performance at the time, it suffered from extremely slow detection speed (14s per image with GPU) due to the redundant feature computations on a large amount of overlapped proposals. Nevertheless, R-CNN was a breakthrough in object detection and remained as the heart of many two-stage detectors such as Fast R-CNN [Girshick2015], Faster R-CNN [Ren *et al.*2015], and Feature Pyramid Networks (FPN) [Lin *et al.*2017].

One-stage detection frameworks, however, introduces a unified detection design, with a single CNN, to predict the bounding boxes and calculate the probability of the categories simultaneously by splitting the input images into grids as anchors. The most popular one for this genre is You Look Only Once (YOLO) [Redmon *et al.*2016], followed by Single Shot multi-box Detector (SSD) [Liu *et al.*2016] and RetinaNet [Lin *et al.*2018]. YOLO was the fastest architecture at the time by trading-off between the speed and accuracy, with which the standard version achieved 45 FPS, and the fastest version achieved 150 FPS. In 2020, YOLOv4 [Bochkovskiy *et al.*2020] was proposed to improve the accuracy by employing Mosaic data augmentation.

Besides the aforementioned genres, Transformer [Vaswani *et al.*2017] has gained lots of popularity due to its impressive performance. The architecture is based on self-attention mechanism, which learns the relationships between elements in a sequence, and thus is able to generalize long-range relationships. This perk is valuable since most state-of-the-art techniques incorporate CNN, which by design has localization inductive bias. However, the computational complexity for this mechanism is quadratic, and an image contains significantly more information than a sentence. Thus, Transformers still struggled to reach real-time detection speed. In this work, we study the feasibility of one state-of-the-art Transformer on real-time street parking sign understanding.

2.2 Parking Sign Understanding

The problem of traffic sign detection is of more popular interest in autonomous driving and only a few works have been done on the subject of street parking sign understanding as follows.

[Mirsharif *et al.*2017] proposed a Linear SVM based framework to detect on-street parking signs. They made use of Google Street View API to collect street-level images in San Francisco. [Irshad *et al.*2018] adopted the same method to collect parking sign data but on a larger scale. They recognized the inefficiency of parsing Google Street View images for parking signs and developed an active learning method to solve this drawback. However, they only studied parking sign detection. [Faraji *et al.*2021] took a slightly different approach for street parking sign understanding. Instead of going through two phases of sign detection and then interpretation or recognition, the pipeline was combined into one single step by assuming each sign conveys one parking rule and there are only three classes of signs: no-stopping, no-parking, and parking-allowed. They used YOLOv4 [Bochkovskiy *et al.*2020] for street parking signs in Vancouver, Canada. The authors acknowledged the lack of text interpretation and their assumption is too strong. More than often, we encounter parking signs that contain more than one symbol or are divided into multiple sections, or both, which cannot be handled by their work.

[Jiang2019, Li2020, Li *et al.*2021] presented a framework for on-street parking sign detection and understanding in the perspective of mobile application development, where users expect parking rules from snapshots of parking signs taken from their phones. For parking sign recognition, they suggested that RetinaNet [Lin *et al.*2018] would yield the best performance in terms of accuracy and speed for this task. To understand the context of each parking sign, text locations are recognized by a convolutional recurrent neural network (CRNN) before being fed into a parking rule generation algorithm. They treated the detection of important objects on a sign as a binary classification task and used SqueezeNet [Iandola *et al.*2016] as the model for no parking and arrow symbol detection. This means each new type of symbol would require collecting data, labeling, and training a new model - the process which demands a substantial amount of manual work. Moreover, increasing the number of models in the pipeline involves more storage and processing time. Therefore, in this work, we aim to use only one model to cover all common and meaningful parking sign symbols for precise and efficient understanding.

3 Methodology

3.1 Building Street Parking Sign Dataset

To the best of our knowledge, there is no open-source annotated dataset for both street parking sign detection and parking symbol detection. Thus, we manually collected and annotated two sub-datasets, covering different cities in the United State. For the annotation software, we used Computer Vision Annotation Tool (CVAT) [Sekachev *et al.*2020].

Parking Sign Detection Data

The parking sign dataset contains 4,191 parking sign images, of which 2,097 are street-level images taken by hand, while the rest of them are street-level video frames from dash cameras. These videos cover four different cities: Tacoma, Boulder, Connecticut, and New York. Each image/frame is annotated with one or more bounding boxes for each parking sign inside. That means, for stacking parking signs, we annotated each parking sign in an independent bounding box. We collected parking signs with diverse shapes (e.g., Fig. 3) and diverse types (e.g., Fig. 4). We also involved variations of the parking signs under various situations, for example, light conditions (e.g., Fig. 5), shooting angle (e.g., Fig. 6), etc. We split our dataset into three splits: training, validation, and test, with ratio 81 : 9 : 10, respectively.

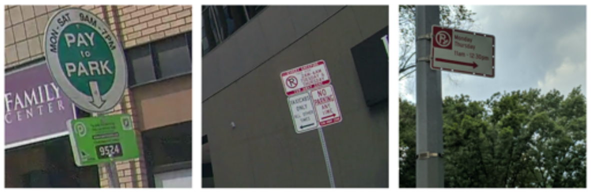


Figure 3: Parking signs with diverse shapes: circle, vertical, horizontal.

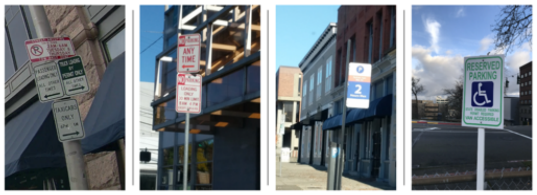
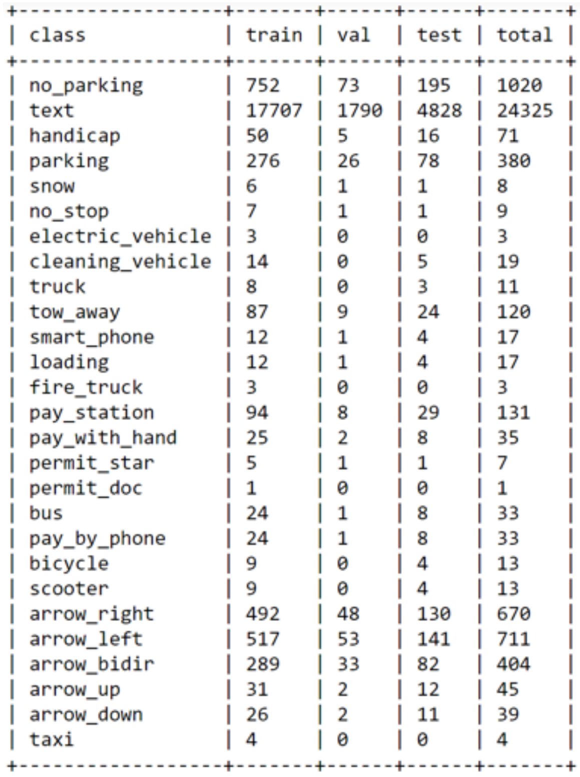


Figure 4: Parking signs with diverse types: loading only, truck only, taxicab, payment required, handicap reserved.

Figure 5: Parking signs under different light conditions.

Figure 6: Parking signs with different shooting angle.



```
+-----------------+-------+-----+------+-------+
| class           | train | val | test | total |
+-----------------+-------+-----+------+-------+
| no_parking      | 752   | 73  | 195  | 1020  |
| text            | 17707 | 1790| 4828 | 24325 |
| handicap        | 50    | 5   | 16   | 71    |
| parking         | 276   | 26  | 78   | 380   |
| snow            | 6     | 1   | 1    | 8     |
| no_stop         | 7     | 1   | 1    | 9     |
| electric_vehicle| 3     | 0   | 0    | 3     |
| cleaning_vehicle| 14    | 0   | 5    | 19    |
| truck           | 8     | 0   | 3    | 11    |
| tow_away        | 87    | 9   | 24   | 120   |
| smart_phone     | 12    | 1   | 4    | 17    |
| loading         | 12    | 1   | 4    | 17    |
| fire_truck      | 3     | 0   | 0    | 3     |
| pay_station     | 94    | 8   | 29   | 131   |
| pay_with_hand   | 25    | 2   | 8    | 35    |
| permit_star     | 5     | 1   | 1    | 7     |
| permit_doc      | 1     | 0   | 0    | 1     |
| bus             | 24    | 1   | 8    | 33    |
| pay_by_phone    | 24    | 1   | 8    | 33    |
| bicycle         | 9     | 0   | 4    | 13    |
| scooter         | 9     | 0   | 4    | 13    |
| arrow_right     | 492   | 48  | 130  | 670   |
| arrow_left      | 517   | 53  | 141  | 711   |
| arrow_bidir     | 289   | 33  | 82   | 404   |
| arrow_up        | 31    | 2   | 12   | 45    |
| arrow_down      | 26    | 2   | 11   | 39    |
| taxi            | 4     | 0   | 0    | 4     |
+-----------------+-------+-----+------+-------+
```

Figure 7: Distribution of all classes in parking symbol dataset.

Parking Sign Symbol Detection Data

Our symbol detection data consists of 3,369 street parking sign images from diverse cities around the United States such as San Francisco, New York, Chicago, Seattle, etc. We cropped the parking signs from the raw images/frames by our parking sign detection model and then annotated the symbols on them. Images are either taken directly from mobile devices, downloaded from public internet archives or extracted from our dash-cam videos, so their qualities are varied.

From the collected data, we figured out 27 types of symbols that are useful for interpreting and understanding the parking rules, most notably: text, no-parking symbol, parking-allowed symbol, handicap symbol, tow-away symbol, and arrows. Arrows are divided into five classes, each indicating a different arrow direction: up, down, left, right, and bidirectional. There are a total of 28,142 labels (86.43% are texts) across all images in our collected parking symbol dataset. We divided the data with a 72 : 8 : 20 label ratio for train, validation and test. Labels and data are preferentially placed in the training set if there are not enough samples, such as taxi symbol and permit with documentation-like symbol. Refer to Fig. 7 for the detailed distribution of classes.

We also created an extra test dataset, comprising 821 images extracted from our dash-cam videos recorded in the city of Boulder Nevada, and the state of Connecticut. Both locations are not presented in the training data, and we refer to this dataset as the "Boulder-CT" set. Similar to other samples obtained from dash-cam videos, the parking sign image resolutions are extremely low because the size of parking signs is considerately small compared to the whole size of frames. The notable characteristic of the Boulder-CT set is that it contains multiple instances of the rounded sign with a down arrow (e.g., Fig. 8 - left). It also includes extremely small handicap symbols (e.g., Fig. 8 - right), though its type and placement on the sign are not unique.

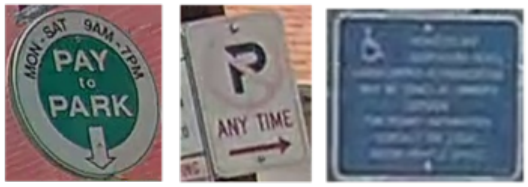


Figure 8: Examples in the Boulder-CT set. Their image resolutions are 168×189, 83×144, and 74×65, respectively.

3.2 Parking Sign Understanding Pipeline

To precisely understand each parking sign, we use the parking sign understanding pipeline proposed in our previous works [Jiang2019, Li2020, Li *et al.*2021], which is summarized in Fig. 2. Concretely, the input is a street-level image taken by the user's phone or extracted from a video, which may contain one or more parking signs.

This image is then passed through the street parking sign detection model where the sections are located and cropped out, where each section contains only one sign with the minimal background. Each cropped sign is afterwards fed to our symbol detection model to classify all relevant objects on the sign, and the results are eventually passed through the text and symbol interpreter to generate the final parking rules.

In this study, we focus on the first two tasks, that is, parking sign detection from street level images/frames and parking symbol detection in each cropped parking sign. We aim to evaluate the feasibility of commonly adopted object detection models (i.e., RetinaNet [Lin *et al.*2018], YOLOv5 [Jocher2021], and Swin Transformer [Liu *et al.*2021]) in real-time that is crucial for autonomous driving and current drivers. It should be noted that all two-stage detection models are excluded considering the real-time requirement. In the following, we briefly describe each model included in the evaluation.

RetinaNet [Lin *et al.*2018] marked the first time a one-stage method surpassed the accuracy of two-stage ones while still retaining the benefit of fast detection. The authors proposed Focal Loss which shifts the model's attention to hard misclassified samples and achieves better accuracy for dense object detection tasks. RetinaNet was proven to perform well for the sign detection task in our previous works [Jiang2019, Li2020, Li *et al.*2021].

YOLO was introduced in [Redmon *et al.*2016] as a Unified Detection, where it overlays a grid of size $S \times S$ on top of the input image. Each cell in this grid is responsible for predicting two parameters: 1) B number of bounding boxes and confidence scores for these boxes and 2) the C class probabilities.

The confidence score reflects how confident the model thinks a cell is containing an object, regardless of its class. It is calculated by the product $P(\text{Object}) * \text{IOU}_{\text{pred}}^{\text{truth}}$ where $P(\text{Object})$ is either 0 or 1 indicating the presence of the object and $\text{IOU}_{\text{pred}}^{\text{truth}}$ is the intersection over union value between the predicted box and the ground truth. Besides that, each bounding box prediction also consists of 4 other values: x, y, w, h. The coordinates (x, y) is the center point of the bounding box relative to the bounds of each grid cells, while the width-height pair (w, h) is relative to the whole image.

Each class probability among C predictions, denoted $P(\text{Class}_i|\text{Object})$, is conditioned on the grid cell containing an object. This probability is determined only once for every grid cell, regardless of the number of B bounding boxes it is predicting, so all boxes in the same grid will share the same predicted class. Finally, YOLO uses Non-Maximum Suppression (NMS) to filter off bounding boxes that do not contain any valid objects and keep only the strongest predictions if two or more boxes overlap over a certain IOU threshold. To arrive at the class confidence score for each bounding box, the class probability is multiplied with the bounding box confidence score as

$$P(\text{Class}_i|\text{Object}) * P(\text{Object}) * IOU_{pred}^{truth}$$

The first version of YOLO was able to provide good performance while achieving 155 frames per second (FPS). Over the next five years, incremental improvements were made and YOLOv5 [Jocher2021] is the latest adaptation. YOLOv5 incorporates cross-stage partial network (CSPNet) [Wang *et al.*2020]. CSPNet solves the problem of duplicate gradient information, effectively reducing the number of model parameters and computational bottleneck, resulting in smaller model size, faster training and converging, and better inference speed. This is important for our purpose of embedding a real-time street parking sign interpreting system into mobile devices or autonomous driving cars. Secondly, as part of YOLOv5's neck, path aggregation network (PANet) [Wang *et al.*2019], which is an advanced version of FPN, is used to boost the propagation of low-level features, increasing the localization accuracy of prediction bounding boxes. Finally, YOLOv5 utilizes the detection head mechanism of YOLOv3 [Redmon and Farhadi2018] which predicts bounding boxes at 3 different scales of feature maps to achieve multi-scale prediction, allowing the model to identify small, medium, or large objects. These perks give YOLOv5 the ability to accurately recognize small objects in cases of texts or low resolution images from video feeds.

Swin Transformer [Liu *et al.*2021] is a variant of Vision Transformer (ViT) [Dosovitskiy *et al.*2020]. ViT achieved a top performance on some computer vision tasks, but with the problem of quadratic computational complexity. Swin Transformer constructs a hierarchical representation of the image by starting with small patches and gradually merging them in the process of feature extraction. To ensure the global self-attention, it also implements a shifting window approach. Swin is able to bring the train and test complexity to almost linear while still producing strong detection results.

4 Experiments

We trained and evaluated our models on a Lambda machine with Intel(R) Core(TM) i9-9820X @ 3.30GHz 20-core CPU, 32GB of RAM, and an GeForce RTX 2080 graphics card with 10 GB memory.

4.1 Setup

YOLOv5 [Jocher2021] provides various kinds of architectures with a trade-off between speed, model size, and accuracy, that is YOLOv5s, YOLOv5m, and YOLOv5l. We evaluated the performance of different architectures. During the training, we used Adam optimizer for parking sign detection and stochastic gradient descent (SGD) for parking symbol detection. We set initial learning rate to 0.001 for parking sign detection and 0.01 for parking symbol detection, momentum to 0.95 for parking sign detection and 0.937 for parking symbol detection, weight decay of 0.0005 for parking symbol detection, the input image size to 640x640, and the batch size of 16. We trained our parking sign detection model over 200 epochs and parking symbol detection model over 250 epochs.

For Swin Transformer [Liu *et al.*2021], we adopted it only for the harder parking symbol detection task and used the Swin-T (i.e., tiny version) backbone and Cascade Mask R-CNN [Cai and Vasconcelos2017] as the detection method without mask heads, since instance segmentation is not involved in our task. Weights are initialized with the pre-trained model from [Liu *et al.*2021]. We also adopted the similar training settings: multi-scale training of resizing the input that the shorter side is between 480 and 800, while the longer side is at most 1333. We adopts Adam optimizer with weight decay of 0.05.

Additionally, for parking symbol detection, we augmented our data to avoid overfitting by adjusting brightness and saturation for both YOLOv5 and Swin. For YOLOv5, we also used perspective manipulation and random copy-paste data augmentation [Ghiasi *et al.*2021]. During our experiments, we observed that random horizontal flip augmentation worsened the performance of both models, specifically for only two classes arrow-left and arrow-right, because of the fact that they are the mirror-image of each other. Therefore, random horizontal flip augmentation were removed from our training pipeline.

4.2 Evaluation Metric

We employed mean Average Precision(mAP) to evaluate the performance of our object detection models. For an object, if there is no predicted bounding box overlapping with the ground truth bounding box, it is called False Negative(FN). If there is an overlapping predicted box, we calculate IoU as

$$\mathrm{IoU} = \frac{\text{Area of Overlap}(\cap)}{\text{Area of Union}(\cup)}. \tag{1}$$

If IoU value is greater than a pre-defined threshold, it is classified as True Positive(TP), and False Positive(FP) otherwise. Precision metric measures among all the predictions considering how many are predicted correctly (Precision $= \frac{TP}{TP+FP}$). Recall metric measures among all the objects considering how many are predicted as positive (Recall $= \frac{TP}{TP+FN}$). The precision-recall(PR) curve is a plot of precision as a function of recall, which shows the trade-off between these two metrics for varying IoU threshold values. AP@α is the area under the precision-recall curve(AUC-PR). AP is defined mathematically as,

$$\mathrm{AP@}\alpha = \int_0^1 p(r)dr \tag{2}$$

Notice, AP@α means Average Precision(AP) as the IoU threshold of α. Therefore, AP@0.50 and AP@0.75 mean AP at IoU threshold of 50% and 75% respectively. A high AUC-PR implies high precision and high recall. With more than one classes for the objects, the average AP (i.e., mAP) over all classes is used.

4.3 Evaluation

Considering the real-time application requirements, we evaluate these models with the following metrics: mean average precision (mAP) at the intersection over union (IoU) threshold of 0.5 and 0.5-0.95 with the stride 0.05, recall, inference speed, and model size. The most significant practical constraint is inference speed and we consider the requirement is at least 30 FPS.

Parking Sign Detection

For parking sign detection, we first evaluated the performance of different architectures of YOLOv5 on our parking sign detection data. As shown in Table 1, their performance on $\mathrm{AP}_{0.5}$ were quite close. However, for the model size and inference speed, YOLOv5l held 6.5 times larger model size and 2.5 times slower detection time compared to the YOLOv5s. Therefore, we used YOLOv5s as the beginning weights to train the parking sign detection model.

Table 1: Performance comparison of parking sign detection on different YOLOv5 architectures.

Model	AP@0.5	Speed	Model Size
YOLOv5s	0.831	5.4ms	14.4MB
YOLOv5m	0.853	9.7ms	42.4MB
YOLOv5l	0.858	13.7ms	93.7MB

We eventually obtained a model of size 14.4MB with an inference speed of 163 FPS and an AP@.5 of 0.968. Compared to the RetinaNet [Lin *et al.*2018], YOLOv5s had a 9% increase on AP@.5 with only 5% of model size as shown in Table 2. Moreover, RetinaNet can only predict 5 frames per second, while it is 163 FPS for YOLOv5s. We also illustrate some exemplar detection results on complex parking sign images (e.g., Fig. 9) and frames of a street view video (e.g., Fig. 10), which further demonstrates the feasibility of YOLOv5s for street parking sign detection in real-time with our training data.

Table 2: Performance comparison of parking sign detection.

Model	AP@0.5	Speed	Model Size
RetinaNet	0.886	4.9FPS	290MB
YOLOv5s	0.968	163FPS	14.4MB

Parking Symbol Detection

For parking symbol detection, we carried out two evaluation experiments. First, we used the train split to train the models and evaluated their performance on test samples from the same pool of locations. Secondly, we used the Boulder-CT set to assess the trained models on signs in a brand new location.

The detection performance on the test split is shown in Table 3. It should be noted that considering the symbol detection a harder task than sign detection, we adopted YOLOv5m and YOLOv5l in this task. We can observe that Swin Transformer generated better results

Table 3: Performance comparison of street parking symbol detection.

Model	Recall	mAP$_{0.5}$	mAP$_{.5:.95}$	FPS	Size(MB)
YOLOv5m	0.980	0.983	0.859	**88.5**	**68.4**
YOLOv5l	0.976	0.992	0.892	56.8	147.0
Swin-T	**0.997**	**0.996**	**0.894**	5.0	293.0

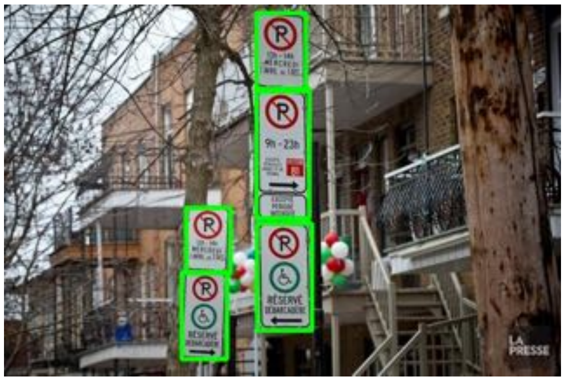


Figure 9: Sign detection on complicated street view.

Figure 10: Sign Detection on street-level video frame from dash-cam.

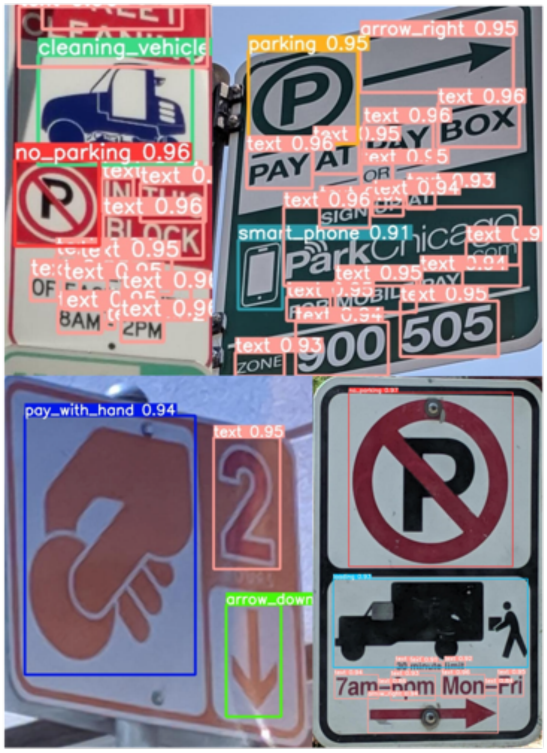


Figure 11: Symbol detection by YOLOv5m for diverse categories of symbols.

than YOLOv5 in both precision and recall, but its detection speed is nowhere close to the real-time requirement. The performance of YOLOv5m is approximately comparable to YOLOv5l while offering half the model size and the extra 30 FPS for inference speed. Therefore, YOLOv5m is suitable for real-time parking symbol detection in small devices. Fig. 11 demonstrates some detection results from YOLOv5m for various types of symbols, which further confirms its feasibility. While Swin Transformer model did not meet the practical requirement for real-time applications, its competitive detection mAP is beneficial for future labeling of new data, which is a decently labor-intensive task.

Then, we investigate the generalization ability of the model to inference new types or poor quality symbols that are common for signs extracted from video feeds, by Boulder-CT test set, which is summarized in Table 4. It can be observed that when a new and low-quality data from other cities was introduced, our model performance decreased as expected. However, the performance is still acceptable, which demonstrates the generalization ability of parking symbol detection using our training data.

Table 4: Performance of street parking symbol detection on Boulder-CT dataset.

Model	Recall	AP@0.5	AP@0.5:0.95
YOLOv5m	0.746	0.804	**0.669**
YOLOv5l	0.747	**0.805**	0.616
Swin-T	**0.784**	0.775	0.580

Finally, besides the overall performance, we examined the detection performance for each different classes as shown in Table 5. Note that any class not included was not involved in Boulder-CT data and the 'Counts' shows the number of appearances for each class. While the performance of most classes is acceptable for both models, they struggled to identify new variants in handicap and arrow_down classes. It is because that even though these symbols and the symbols in the training data have a few characteristics in common, they are of different shapes and outlooks in general. Therefore, for brand new types of symbols, even for existing classes, we need to gradually increase the coverage of the training data.

Table 5: Performance per class on Boulder-CT dataset.

Class	Counts	Recall		AP$_{0.5}$	
		YOLOv5m	Swin-T	YOLOv5m	Swin-T
no_parking	92	0.957	0.967	0.963	0.961
text	5303	0.880	0.904	0.923	0.853
handicap	**100**	**0.469**	**0.690**	**0.727**	**0.690**
tow_away	45	0.933	1.000	0.966	0.997
arrow_right	181	0.950	0.978	0.956	0.975
arrow_left	167	0.891	0.881	0.912	0.872
arrow_bidir	100	0.861	0.850	0.950	0.848
arrow_down	**117**	**0.034**	**0.000**	**0.041**	**0.000**

5 Conclusion

Street parking sign understanding is an important task for both autonomous driving and current drivers. However, this task is relatively unexplored and there is no public annotated data. In this work, we introduced a street parking sign dataset that includes 4,191 street-level images with annotated parking signs and 3,369 cropped parking signs with annotated symbols. Utilizing this data, we applied our street parking sign understanding pipeline, focused on the evaluation of parking sign detection and symbol detection using commonly adopted object detection models, and evaluated their feasibility in the real-time application scenario.

We find that for the parking sign detection, YOLOv5s can provide a model with AP@.5 of 96.8% and a processing speed of 163 FPS. Moreover, the size of sign detection model is only 14.4MB, making it lightweight to deploy in mobile devices or self-driving automobiles. For the parking symbol detection, we find that YOLOv5m is able to produce a lightweight (i.e., 68.4MB) and fast model (i.e., 88 FPS), while still achieving 98.3% mAP@0.5. This demonstrates the feasibility of YOLOv5 using our training data for the task of real-time street parking sign understanding. Moreover, we investigated the generalization ability of the trained parking symbol detection model on a brand new test data collected from other cities compared to the training data. The acceptable performance also demonstrates the applicability of YOLOv5 trained using our data.

However, we also find that even for existing symbol classes in the training data, it is hard to recognize symbols of totally different styles (e.g., different shapes). Therefore, we need to gradually enlarge the training data. Besides, for the sign detection model, given street-level dash-cam videos, the model sometimes fails to detect one parking sign in every present frame. One potential solution is to gather additional information from continuous frames to improve the parking sign detection. Ultimately, we will evaluate the performance of text detection and recognition more extensively in future work, since text is vital for understanding the parking signs.

References

[Bochkovskiy *et al.*, 2020] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection, 2020. 3

[Cai and Vasconcelos, 2017] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection, 2017. 6

[Dalal and Triggs, 2005] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005. 2

[Dosovitskiy *et al.*, 2020] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 5

[Faraji *et al.*, 2021] Parnia Haji Faraji, Hamid Reza Tohidypour, Yixiao Wang, Panos Nasiopoulos, Simon Ren, Arash Rizvi, Cloris Feng, Mahsa T Pourazad, and Victor CM Leung. Deep learning based street parking sign detection and classification for smart cities. In *Proceedings of the Conference on Information Technology for Social Good*, pages 254–258, 2021. 1, 3

[Felzenszwalb *et al.*, 2010] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, 2010. 2

[Ghiasi *et al.*, 2021] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D. Cubuk, Quoc V. Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation, 2021. 6

[Girshick *et al.*, 2014] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. 2

[Girshick, 2015] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. 2

[Iandola *et al.*, 2016] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet: Alexnet-level accuracy with 50x fewer parameters and¡ 0.5 mb model size. *arXiv preprint arXiv:1602.07360*, 2016. 3

[Irshad *et al.*, 2018] Humayun Irshad, Qazaleh Mirsharif, and Jennifer Prendki. Crowd sourcing based active learning approach for parking sign recognition. *arXiv preprint arXiv:1812.01081*, 2018. 3

[Jiang, 2019] Zhongyu Jiang. Street parking sign detection, recognition and trust system. 2019. 1, 3, 4, 5

[Jocher, 2021] Glenn Jocher. Github - ultralytics/yolov5: Yolov5 in pytorch >onnx >coreml >tflite, 2021. 2, 5

[Krizhevsky *et al.*, 2012] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012. 2

[Li *et al.*, 2021] Jiayu Li, Putthida Samrith, Nicole Guobadia, Juhua Hu, and Wei Cheng. Automatic street parking sign reading. *IEEE IOT-AHSN TC Newsletter*, 1(14):3–4, Jun 2021. 1, 3, 4, 5

[Li, 2020] Jiayu Li. An algorithm for street parking sign rule generation. 2020. 1, 3, 4, 5

[Lin *et al.*, 2017] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017. 2

[Lin *et al.*, 2018] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018. 2, 3, 5, 6

[Liu *et al.*, 2016] 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. 3

[Liu *et al.*, 2021] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 2, 5, 6

[Mirsharif *et al.*, 2017] Qazaleh Mirsharif, Théophile Dalens, Mehdi Sqalli, and Vahid Balali. Automated recognition and localization of parking signs using street-level imagery. In *Computing in Civil Engineering 2017*, pages 307–315. 2017. 3

[Redmon and Farhadi, 2018] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 5

[Redmon *et al.*, 2016] 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. 3, 5

[Ren *et al.*, 2015] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015. 2

[Sanyal *et al.*, 2020] Banhi Sanyal, Ramesh Kumar Mohapatra, and Ratnakar Dash. Traffic sign recognition: A survey. In *2020 International Conference on Artificial Intelligence and Signal Processing (AISP)*, pages 1–6. IEEE, 2020. 1

[Sekachev *et al.*, 2020] Boris Sekachev, Nikita Manovich, Maxim Zhiltsov, Andrey Zhavoronkov, Dmitry Kalinin, Ben Hoff, TOsmanov, Dmitry Kruchinin, Artyom Zankevich, DmitriySidnev, Maksim Markelov, Johannes222, Mathis Chenuet, a andre, telenachos, Aleksandr Melnikov, Jijoong Kim, Liron Ilouz, Nikita Glazov, Priya4607, Rush Tehrani, Seungwon Jeong, Vladimir Skubriev, Sebastian Yonekura, vugia truong, zliang7, lizhming, and Tritin Truong. opencv/cvat: v1.1.0, August 2020. 3

[Uijlings *et al.*, 2013] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. *International journal of computer vision*, 104(2):154–171, 2013. 2

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017. 3

[Viola and Jones, 2001] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, 2001. 2

[Wang *et al.*, 2019] Kaixin Wang, Jun Hao Liew, Yingtian Zou, Daquan Zhou, and Jiashi Feng. Panet: Few-shot image semantic segmentation with prototype alignment. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9197–9206, 2019. 5

[Wang *et al.*, 2020] Chien-Yao Wang, Hong-Yuan Mark Liao, Yueh-Hua Wu, Ping-Yang Chen, Jun-Wei Hsieh, and I-Hau Yeh. Cspnet: A new backbone that can enhance learning capability of cnn. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 390–391, 2020. 5