An Algorithm for Street Parking Sign Rule Generation

Jiayu Li

A thesis submitted in partial fulfillment of the requirements for the degree of

Master of Science in Computer Science and Systems

University of Washington

2020

Committee:

Wei Cheng, Chair

Juhua Hu

Program Authorized to Offer Degree:

Computer Science and Systems

University of Washington

Abstract

An Algorithm for Street Parking Sign Rule Generation

Jiayu Li

Chair of the Supervisory Committee:
Dr. Wei Cheng
Computer Science and Systems

Autonomous driving and autonomous parking have been actively investigated for years. However, for autonomous street parking, the questions of how to determine where to park and how long can park have never been addressed. To solve these problems, we design a pipeline of detecting the valid street parking signs, recognizing the plain text on the signs, and understanding their semantic meanings. Specifically, in this thesis, we propose a street parking sign rule generation engine, which can figure out the allowed parking time and payment information for each vehicle category. To the best of our knowledge, this is the first work on designing the algorithm for street parking rule generation from street parking sign pictures. By utilizing the proposed work, our smart street parking system can detect and recognize the texts on street parking signs and extract street parking rules. The preliminary results illustrate that our system can successfully generate accurate street parking rules from different types of street parking signs.

Contents

List of Figures

List of Tables

GLOSSARY

OCR: Optical Character Recognition. Recognize the text in images.

STREET PARKING RULE: Collection of parking information, including the allowed parking time and payment information. Different vehicle categories may have different street parking rules.

OBJECT DETECTION: A task in Computer Vision. Detect the objects in images

RULE GENERATOR ENGINE: An engine which can specify the structure of parking signs and generate their street parking rules.

PSDRRG PIPELINE: Street Parking Sign Detection, Recognition, and Rule Generator Pipeline.

ACKNOWLEDGMENTS

Thanks for the help and suggestions from professors and friends. They give me the power to go far in research.

DEDICATION

to my parents and friends

1 Introduction

Today, there are autonomous vehicles running on roads. These smart vehicles with the robust control system [17] can automatically avoid pedestrians, change lanes, maintain the safe distance between vehicles, and handle emergencies via employing computer vision technologies (such as object detection, tracking, localization, and monitoring) to perceive the environment and make appropriate decisions and predictions. Although AI and IoT technologies have been widely used to assist smart vehicles to interact with the real world, there is still an open problem in smart street parking, which is how to automatically find an appropriate parking spot via correctly detecting, recognizing, analyzing, and understanding the street parking signs on roadsides.

In the United States, there are millions of street parking signs. Besides, street parking signs may significantly vary from state to state and sometimes city to city. Even for experienced drivers, they may be confused by street parking signs while driving in unfamiliar or even local areas. To help drivers, an efficient and reliable street parking sign interpreting tool is essential. It can help not only drivers select legal street parking spots but also smart cars implement autonomous street parking. .

For sign detection, intuitively, traffic sign detection is similar to street parking sign detection, and it has been deeply studied [12] [13] [14]. However, unlike the traffic signs (e.g. speed limit signs, warning signs, stop signs) that can be detected by color and/or shape, the street parking signs are much more complex. There are more information on them, and their layouts are diverse. Moreover, recognizing the text on street parking signs is paramount for the smart devices to understand and interpret street parking rules. To address these challenges, we introduce a novel and robust pipeline to detect and recognize street parking signs.

In this thesis, we build a street parking sign detection, recognition and rule generation(PSDRRG) pipeline. In PSDRRG, we first detect street parking signs from the pictures taken by cameras mounted on smartphones/smart vehicles. Next, we detect the text bounding boxes and crop them out. Then, OCR and post-processing are utilized to recognize the texts. Finally, we build a rule generation engine to understand street parking rules. Evaluations show that for different layouts of street parking signs, our rule generation algorithm can correctly calculate the allowed parking time and payment information for different types of vehicles.

Moreover, we build a back-end system and release considerable useful APIs to handle the street parking recognition requests generated from the front-ends (smartphones/smart vehicles). Users can upload and check street parking signs, plan the parking spot searching route based on our street parking sign data set, and get the nearby street parking information. To efficiently manage the user and street parking sign information, we design the schema of the database to support application development. In this thesis, we will also introduce the implementation details of each component.

1.1 Contributions

The contribution of this work can be summarized in the following:

- We propose a novel pipeline to detect, recognize, and understand street parking signs.
- We build the street parking sign data set and label street parking signs, text bounding boxes, and sign text.
- We present a hybrid solution to recognize the texts on street parking signs. Our solution outperforms existing OCR methods, such as Google OCR.

- We study a street parking rule generation algorithm to systematically analyze and understand the texts on the street parking signs. To the best of our knowledge, this is novel in the field.
- We design a database to integrate and manage the street parking sign information and user information.
- We implement the HTTP APIs to support application development.

2 Related Work

2.1 Object Detection and Street Parking Sign Detection

There are two categories of object detection methods: one-stage [1],[2],[3] and two-stage [4],[5],[6] approaches.

2.1.1 Two-stage Method

Two-stage methods firstly generate regions of interest as object detection candidates. In the second stage, candidates are sent to another neural network for object classification and bounding-box regression. These models achieve the relatively higher accuracy, but are typically slower.

2.1.2 One-stage Method

One-stage methods, such as YOLO (You Only Look Once) and SSD (Single Shot Multi-Box Detector) predict the class probabilities and the bounding box coordinates simultaneously from the input images. They are less time-consuming but their detection accuracy is lower than the two-stage ones.

2.1.3 Street Parking Sign Detection

In [7], a computer vision technology based application was built to detect and recognize the street-level imagery. However, the utilized SVM(Support Vector Machine) failed to provide reliable results. Moreover, due to the utilized sliding window method, the algorithm can only handle street parking signs in the fixed size.

In [8], the authors collected street-level images to build a street parking sign data set in

San Francisco. The images vary in shape, color, orientation, and scale. A framework was built to test two object detection methods: SSD and YoloV2. However, the collected street parking signs' resolutions are too small to recognize texts (Fig.1). We, therefore, need to build a street parking sign data set for street parking rule generation.

2.2 Text Recognition

After detecting street parking signs from a image, the next step is to understand the meaning of the texts on street parking signs. In [9], an effective approach was proposed to solve the problem of word detection and recognition from images. There are four stages: (1) detect the potential coordinates of characters by utilizing a sliding window and random fern; (2) recognize the characters by utilizing pictorial structures; (3) predict words by utilizing a trie-structured lexicon; and (4) re-evaluate the inferred words by utilizing the SVM classifier. [10] studied an end-to-end scene text recognition method via leveraging pixel-wise gradient magnitude projection. It proposed several expected extremal regions (ERs) and computes statistical and graphical features with *O(1)* complexity.

There are text recognition applications built on the convolutional neural network image processing technology. For example, Wang *et al.* [11] introduced a framework driven by lexicon on scene text for end-to-end character recognition by utilizing CNN. Instead of relying on carefully hand-engineered features or large amounts of prior knowledge, this system integrates large multilayer representational neural networks. According to the experiment results, the CNN-based method outperformed the previous methods such as [9] by 20% and [10] by 30%.

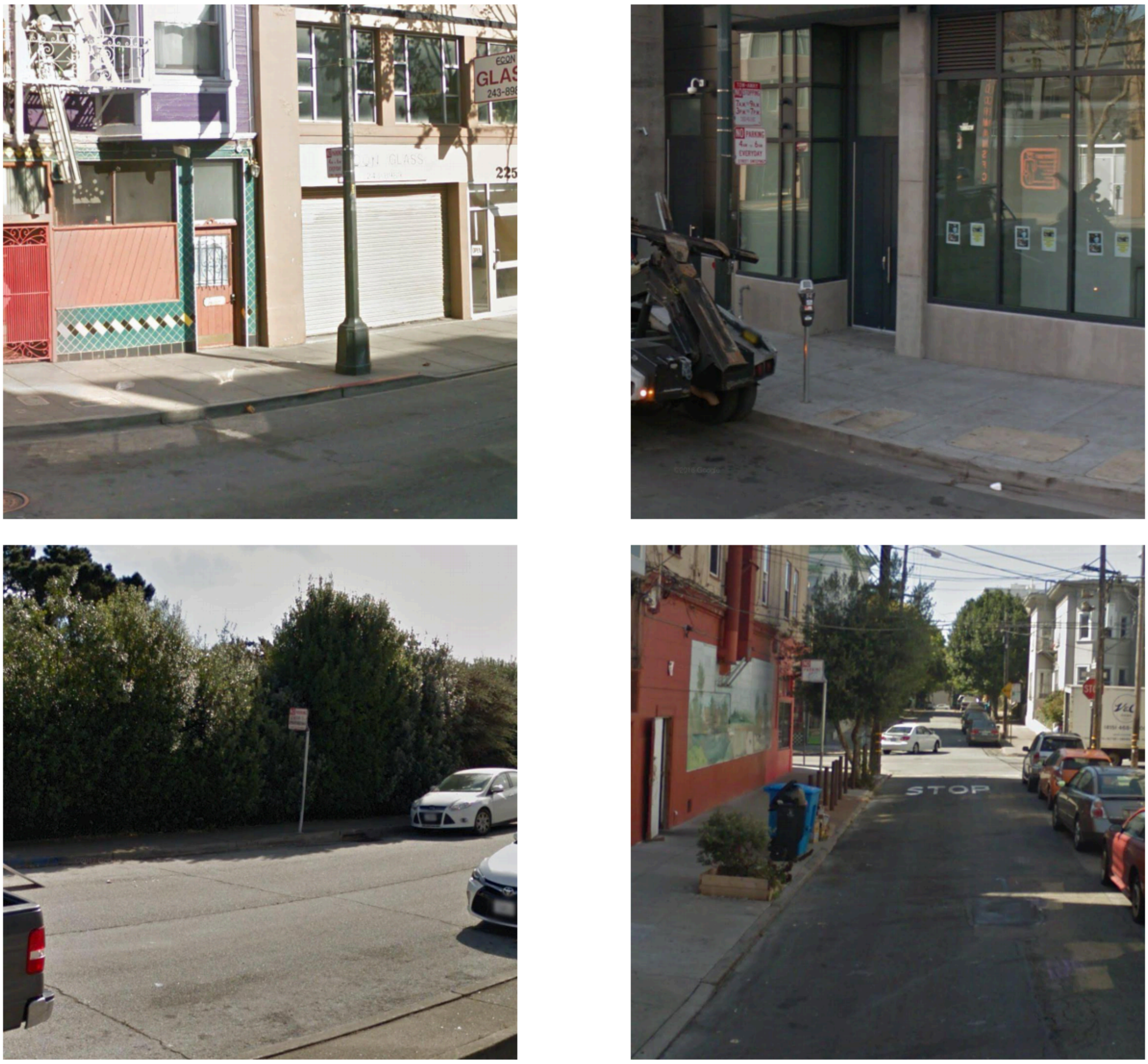


Figure 1: Some examples in San Francisco Street Parking Sign Data Set [8].The size of street parking signs are really small compared with the whole size of images.

2.3 Traffic Sign Recognition

To the best of our knowledge, there is no previous study on recognizing and understanding street parking signs. There are works on recognizing traffic signs. Some traffic signs can be recognized by the appearance features, such as shape and color. In [12], it described an integrated approach for color and shape modeling to recognize the speed limit signs and instruction signs. It normalizes the images and sends the first 25 most significant features, which are generated by linear discriminant analysis, to a maximum likelihood classifier. In [13], it presented a multistage CNN architecture to recognize the traffic signs. The subsampling results of the 1st stage and 2nd stage were concatenated and fed into a fully connected classifier. [14] compared the performance of several traffic sign recognition methods. It turned out that CNN-based methods outperformed the ones using linear discriminant analysis.

Intuitively, understanding street parking rules is much more complex than sign recognition, which is a classification problem. In addition, the classification fails to reproduce the text that is subtle and varies among different signs. Moreover, the number of classification categories is limited, and it cannot provide the information of how long a driver can park. Therefore, the traditional classification method is insufficient due to the lacks of text recognition and rule generation.

There are research on recognizing the text on traffic signs by utilizing Optical Character Recognition (OCR) methods. For example, [15] extracted textual information from simple traffic signs by utilizing MLP (Multilayer Perceptron) Neural Network. [16] presented a system combining two widely used technologies, OCR and RFID, to recognize the numbers on vehicle plate, which determined whether the vehicle is authorized to park in a specific parking lot. However, these solutions are too simple to apply to street parking sign text recognition.

3 System Architecture

Street Parking Sign Detection, Recognition, and Rule Generation(PSDRRG) pipeline (Fig. 2) contains five components: street parking sign detection, parking text detection, text recognition, text processing, and rule generation.

The images containing street parking signs are taken by users or smart vehicles through their cameras. After processing the image through a street parking sign detection network, we will get the bounding box coordinates of each street parking sign. The cropped sign images will be fed into a street parking sign text detection model. We will crop out the single word images based on the text bounding boxes and send them to a text recognition model, which will generate the recognized words with their bounding box coordinates. Next, an reorder function will be utilized to determine the sequence of sign text and generate the ordered sign texts. The text processing module will conduct the preprocessing for the rule generation engine. Then, the ordered sign texts will be sent to the rule generation engine to produce street parking rules. Finally, we will store the street parking rules into database. The detailed design of the pipeline will be described in the following section.

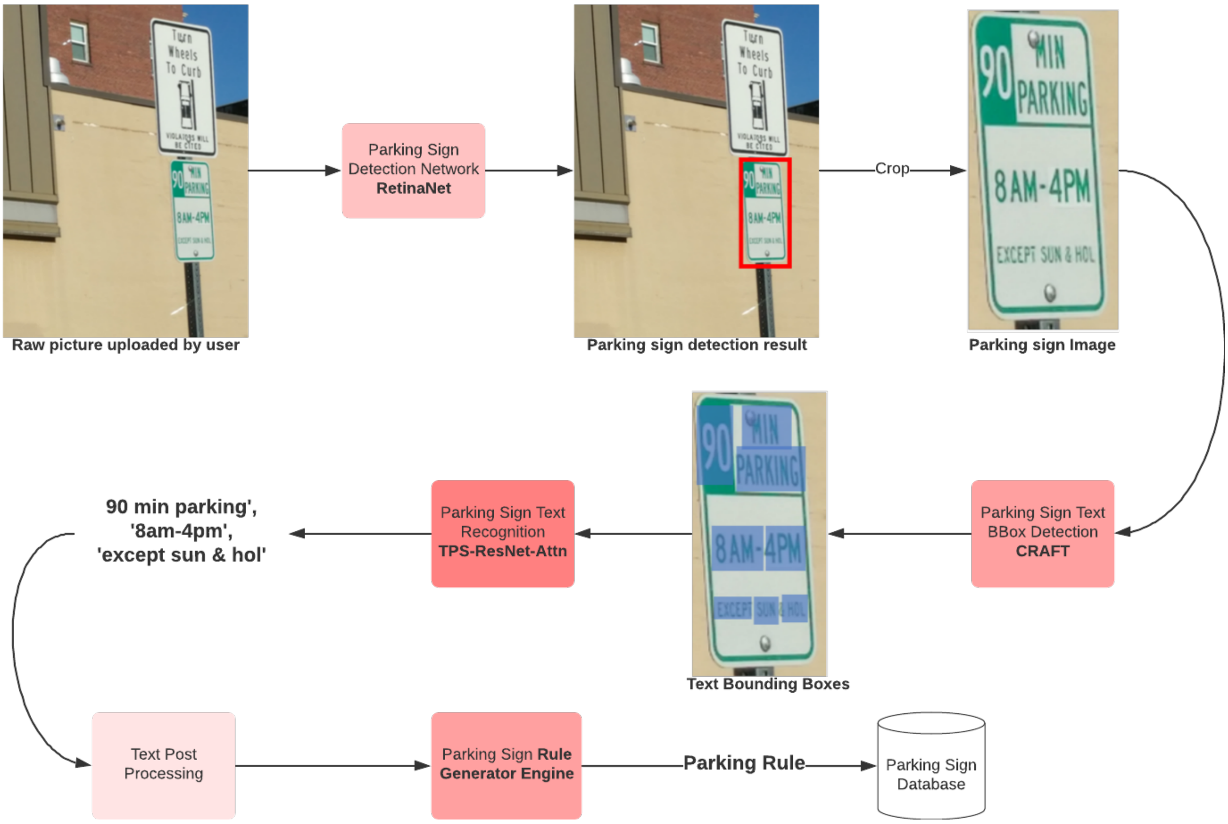


Figure 2: RSDRRG pipeline

4 Proposed Solution

4.1 Street Parking Sign Detection

There are some popular models for object detection, such as SSD, Yolov3, RetinaNet, and Faster_RCNN. We test the accuracy and speed of different models for the street parking sign detection. We eventually chose RetinaNet, which is a one-stage object detection method. It uses a FPN backbone on the top of the feedforward ResNet Architecture to generate rich and multiscale convolutional features. The network is intentionally simple and faster than two-stage methods and has reliable enough performance. We fine-tuned our street parking sign detection model with the pretrained weight on ImageNet.

4.2 No Parking Symbol Detection

Besides the text on street parking signs, there are also symbols that need to be recognized to understand the content of the signs. The most popular symbol is the one for 'No Parking' as shown in Fig. 3.

We treat the no parking symbol detection problem as a classification problem. The signs are divided into two categories: with the symbol and without the symbol. We adopt SqueezeNet as our network as its network architecture has fewer parameters to learn and it is able to achieve reliable enough performance. We fine-tuned the SqueezeNet on our no parking symbol data set. The classification results will be sent to the Rule Generation Engine and be merged as the beginning of street parking sign text.

Figure 3: No Parking symbol examples

4.3 Street Parking Sign Text Recognition

To obtain the recognized text from the sign images, we first crop out each word images based on the text bounding box coordinates, which are generated by the text detection model. We adopt the connectionist text proposal network (CTPN) [24], which can accurately localize text lines in image by a vertical anchor mechanism that predicts the location and text score of each fixed-width proposal simultaneously. Besides, CTPN are connected by a neural network, which enables it to capture the rich contextual information. However, CTPN can only detect the text line, not the single word. For a text line which includes more than one word, the detected result contains more than one word. Therefore, we need to feed the CTPN results to another network to detect the space between words.

A method in [25] builds a multichannel text detector by utilizing a convolutional neural network named EAST model. It performs better than CTPN on a smaller scale. In addition, EAST can detect the spaces between the words and helps us to segment the words. However, EAST does not perform as well as CTPN on detecting text lines. Therefore, our sign text detection solution is to feed the images of the street parking sign into CTPN first to get the bounding boxes of text lines. Then, we crop out the region of each detected text line and use EAST to split the text line into bounding boxes of single words.

[9], [10], [11] demonstrated that, for text recognition,CNN-based methods usually outperform the ones without CNN. We, therefore, utilize the convolutional recurrent neural network (CRNN) [26] to recognize the text from word images. CRNN uses convolutional layers to extract features and classifies through a recurrent network with LSTM. CRNN model takes an image of a single word or text line as input. The output provides the predicted text. To understand the street parking sign, not only the text itself matters, but also the order of the text matters. Thus, we order the output of CRNN by the bounding boxes provided by the EAST model and the no parking symbol detection model. Reorder function is based on the IOU metric.

4.4 Rule Generation Engine

To build a Rule Generation Engine that can produce street parking rules, we design the architecture of the rule generation engine as shown in Fig. 4. The input of the Rule Generation Engine is a list of ordered sign texts, such as ['no parking', '8am-8pm', 'mon-fri']. Some phrases are not directly related to the parking time information such as 'street sweeping', and 'tow away zone'. We ignore those phrases in Rule Generation Engine for now. The goal of Rule Generation Engine is to transfer the ordered street parking sign text into the structured data, to allocate it to the matching function, to apply the calculation function to get the parking information such as the allowed parking time, and to store the information into the data set. The engine consists of four components.

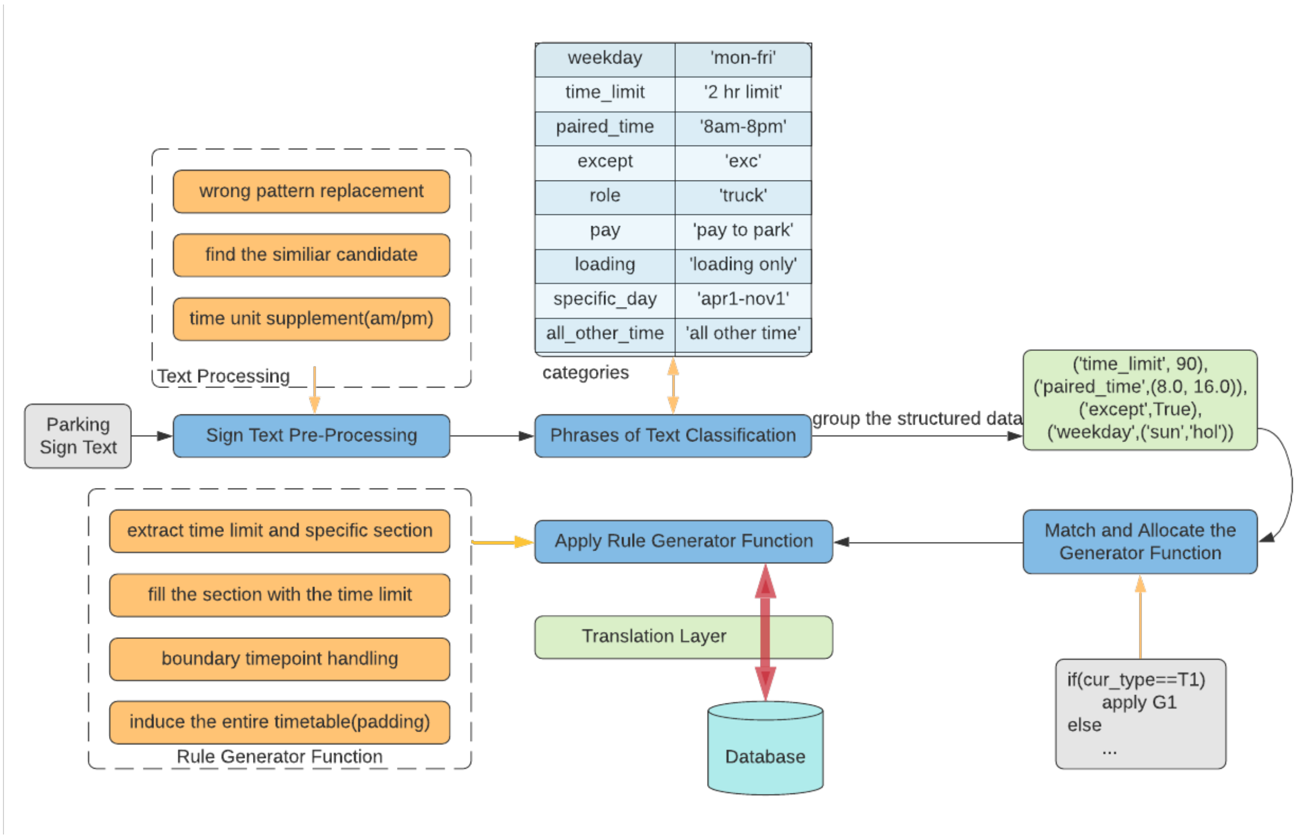


Figure 4: Architecture of rule generation engine

Wrong Pattern	Correct Text
Bam	8am
Tam	7am
Ist	1st
mm	min

Table 1: Examples of the wrong pattern replacement

4.4.1 Text Pre-Processing

This subsection focuses on correcting the wrong recognized text and completing some common values such as the time unit.

A. *Wrong Pattern Replacement*

Due to the limitations of our OCR model, which cannot tell the difference between the letter 'o' and the number '0'. As a result, the model is likely to recognize '30min' as '3omin'. We collected the wrong recognition cases to summary their common patterns, which are listed in Table 1.

B. *Find Similar Candidate*

This part maintains a street parking sign text dictionary, which contains 423 words from more than 3000 street parking signs. If there is a recognized word not in the street parking sign dictionary, such as 'loaning', we will find the most similar candidate in the dictionary and replace it by the Levenshtein distance defined as the following:

$$lev(a,b) = \begin{cases} |a|, & |b| = 0 \\ |b|, & |a| = 0 \\ lev(tail(a), tail(b)), & a[0] = b[0] \\ 1 + min(lev(tail(a), b), lev(a, tail(b)), lev(tail(a), tail(b)), & otherwise \end{cases} \tag{1}$$

C. *Supplement Time Unit*

In some street parking signs, the time unit for the first time number is missing, such as '7-9am'. In this part, the time unit will be supplemented based on the time unit for the second number to classify and group the structured data.

4.4.2 Tokens Classification

We define the structured data of the street parking sign as a combination of categories and values (Table 2). Token Classification, splits the ordered sign text into several tokens and categorizes the tokens into different types to group the structured data as following.

- Number. Cache the number and wait for a unit (min, hour, am or pm).
- Hour/Minute. Group with the last cached number as the value of Time Limit category and pop back the number.
- To/Thru. Set thru flag as True. Thru flag helps to form the Weekday or Paired Time and is cancelled by the start and end weekday or time.
- AM/PM. Group with the last cached number as the value of Time and pop back the number. The system checks if there is any cached start time. If so, group it with start time into value of Paired Time category and set. If not, cache it as start time and wait for another grouped time.
- No. Cache it and wait for the word 'parking'.
- Parking. The system checks if there is cached 'no'. If so, group it with 'no', and add True value to a No Parking category. If no, cache it.
- Except. The system adds a new category Except. The value is True.

Category	Value Example
weekday	mon,tues
time_limit	90 min
paired_time	8am-8pm
except	exc
role	truck
pay	metered
loading	loading
specific_day	apr1-nov1
all_other_time	all other time

Table 2: Categories of street parking sign text

- Weekday. The system checks if the thru flag is true in this text line. If so, cache the day from start day (the day already cached before) to it as the value of Weekday category. Then, clear the Day Cache and set the thru flag as False. Else, cache it in Day Cache. If the next category is not weekday nor thru flag, cache the weekdays in Day Cache as the value of Weekday category and clear the Day Cache.

- Role. Add it as the value of Role category.

After categorization, the ordered sign text is converted into the combination of categories and values, which are the structured data of street parking sign. Next, we send them to the allocator.

4.4.3 Match and Allocation

So far, we summarized 91 different structures of street parking signs. We merge the forest branches into code branches as shown in Fig.5. Street parking signs with similar structures share the same rule generation function. There are 26 different code branches in 5 different rule generation functions for covering the 91 street parking sign forest branches.

The rule generation functions are determined by the category of the first word in the sign text. To the best of our knowledge, the street parking signs start with one of the 5 types: no parking, role, loading, time limit, and pay.

Besides the first category, the code branches are determined by the the number of the specific category. In other words, the existence of a specific category, such as ’role’ or ’loading’, determines the different code branches to generate different street parking rules. For example, if a role is mentioned in street parking signs, two sets of street parking rules should be generated: one for the mentioned role and the other for all the other roles. If ‘loading only’ appears on street parking signs, vehicles can not park.

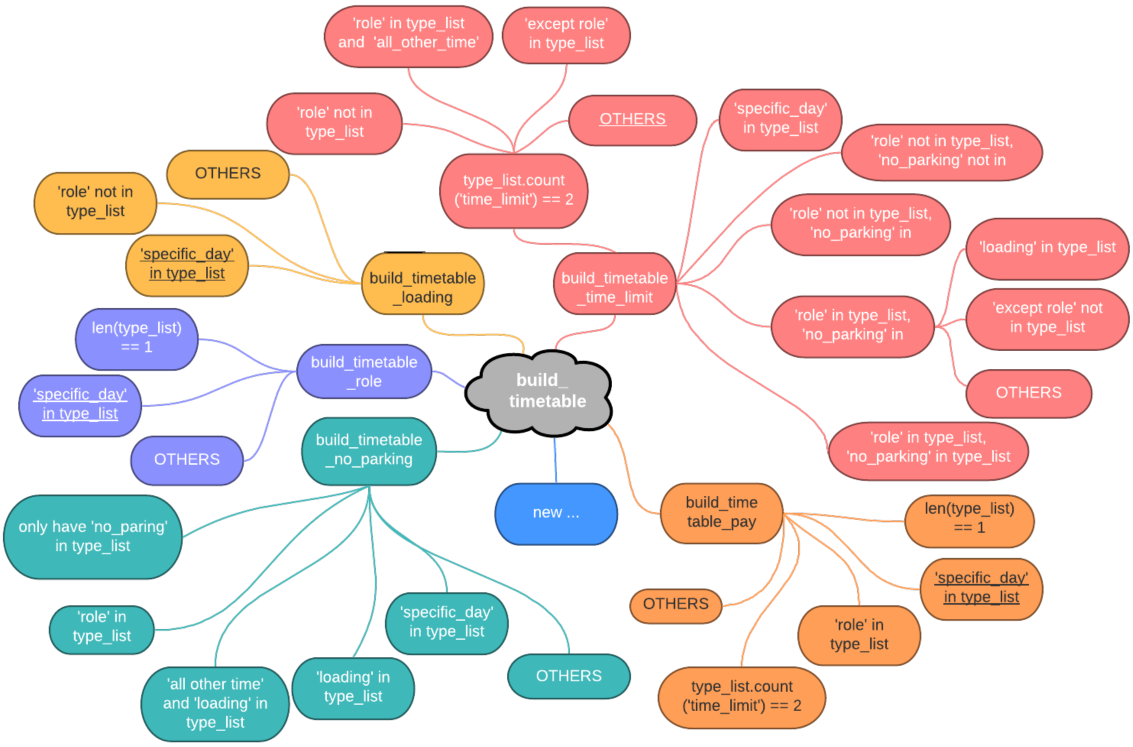


Figure 5: Code branches of rule generation engine

4.4.4 Apply Rule Generation Function

The part aims to build our allowed parking time tables. The number of timetables is determined by whether there is 'role' mentioned in the street parking sign.

A. *Extract time limit and section*

In this subsection, we build the section that applies to each time limit. Every section contains at most one Weekday type or one Paired Time type depending on the text structure as there are W-P signs and P-W signs. For each sign, if P appears ahead of W, then it is P-W sign, and vice versa. If there is a time limit without any section, manually build a section W = {everyday}.

B. *Fill the section with time limit*

For the allowed parking time table, we find all time points within the section and assign the value to the corresponding time limit value. The default time unit is minute.

C. *Handle boundary time points*

Before calculating the entire timetable, we label the boundary time points for more processing. We define that if the time point is within the section and the time point plus the allowed parking time equals to or is greater than the end of the section, the time point is treated as the boundary time point. For example, in Fig 2, on Monday, 3pm is the boundary time point as 90 minutes after it is 4:30 pm, which is greater than the end of the section.

D. *Induce the entire timetable*

There are two modes to induce the entire timetable based on the code branches. The first one is No Parking mode, which assigns all the values of time points beyond the section to zero.

The other one is Parking mode, which assigns all the values of time points beyond the section based on the latter time point. The start time point for the first section will be found and induce other time points one by one. If the time point within the section is not the boundary time point, we keep the value and skip. Otherwise, the allowed parking time will be recalculated.

5 Street Parking Sign Data Set

To the best of our knowledge, there is no open-source and well-organized street parking sign data set with the ground truth label information. Therefore, we collect our own data set and make the annotations manually.

Figure 6: Examples in the street parking sign detection

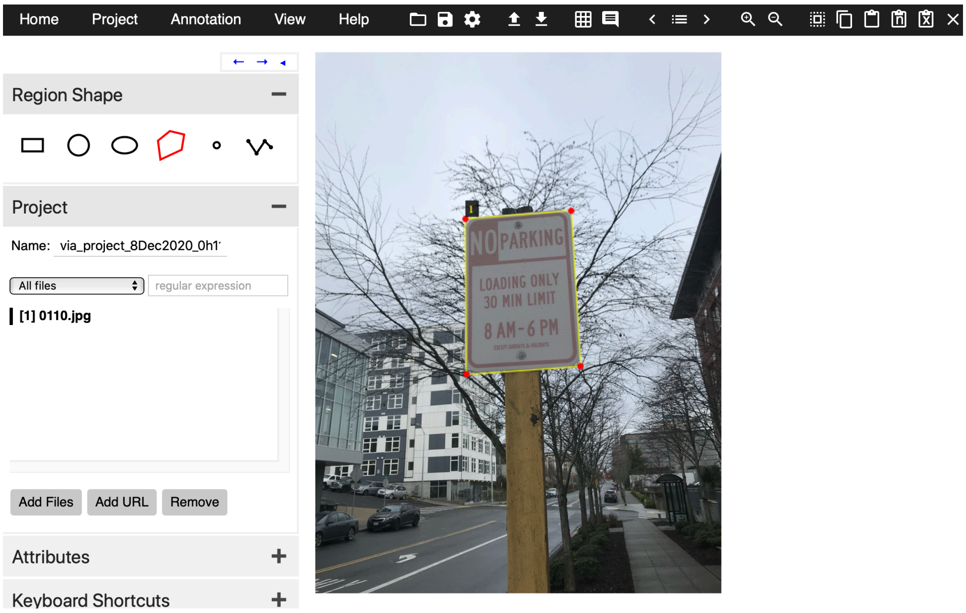


Figure 7: Screenshot of the sign detection annotation.

5.1 Street Parking Sign Detection Data Set

In our street parking sign detection data set (Fig. 6), we discarded the images with low resolution or the street parking sign is relative small in the images. There are 2064 street parking sign images in total. 1832 street parking sign images were used to fine-tune the detection model, and 232 street parking sign images were used for testing.

Those images are taken from 10 states around the US, such as WA, CA, DC, and NY. Manual annotation was performed using *VGG Image Annotator(VIA)* [18]. VIA is a simple and standalone manual image annotation software as shown in Fig.7.

5.2 Street Parking Sign Text Bounding Box Detection Data set

We first got the cropped street parking sign images from the street parking sign detection model. Then, we discarded the street parking sign images, whose text can not be recognized by human eyes. Finally, we manually annotated 1566 street parking sign images and generate 9157 text bounding boxes. 80% of them were used for training, and 20% of them were used for testing.(Fig. 8)

Figure 8: Screenshot of the text bounding box detection annotation.

5.3 Street Parking Sign Text Recognition Data set

We cropped all word images in the text bounding box labeled street parking signs (Fig. 9) based on the CRNN model results. In order to facilitate the manual annotation, we first fed the word images into Google OCR API to get the preliminary results [19]. Then, we manually filtered the blurry word images and modified the wrong recognition text result to improve the label accuracy. After processing, there are 9157 word images in total in the street parking sign text recognition data set.

To improve the performance of the text recognition model, we introduced two open-source data sets. (1) MJSynth Data Set(MJ) [20] consists of the synthetically generated English words. The synthetic data is generated by font rendering, border/shadow rendering, base coloring, projective distortion, natural data blending, and noise. (2) SynthText (ST) [21] is a synthetically generated data set, where word instances are placed in natural scene images,

Figure 9: Examples in our street parking sign text recognition data set

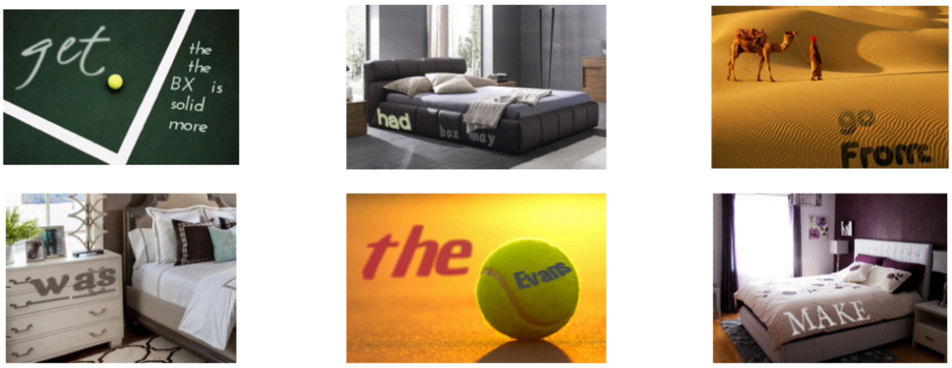


Figure 10: Examples in ST data set

while taking into account the scene layout (Fig. 10). The batch ratio of MJ, ST, and our labeled word images is 0.4:0.4:0.2.

5.4 No Parking Symbol Classification Data Set

In the no parking symbol classification data set, street parking signs are divided into two categories: with and without the no parking symbol. There are 1822 sign images for training and 284 sign images for testing.

5.5 Rule Generation Engine Data Set

We adopted the sign plain text from the Seattle DoT website to develop the rule generation engine. There are 1866 entries. We build a street parking sign forest as shown in Fig. 11 to specify the structure of street parking signs. Each node indicates one category of the street

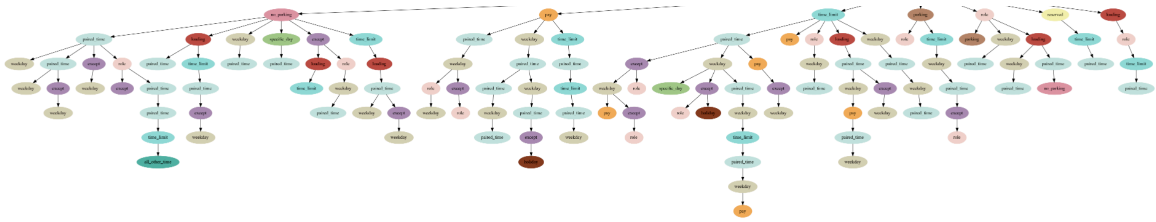


Figure 11: Street Parking Sign Forest

parking sign. The structure of an entire street parking sign can be indicated as a path from a root node to a leaf node. In the forest, there are 91 branches with 7 root nodes.

6 Evaluation

6.1 Experiment Environment

We conduct the tests on a desktop PC with Intel(R) Core(TM) i7-8700 CPU, 16GB of RAM, and a GTX-1080Ti graphics card with 11GB of memory.

6.2 Street Parking Sign Detection

We use two metrics to evaluate our system in street parking sign detection: mAP(mean Average Precision) to evaluate the performance, FPS(Frame Pre Second) to evaluate the speed. We test several different models on our street parking sign detection data set. As shown in table 3, with different IOU thresholds, RetinaNet with ResNet50 [22] can achieve the second highest accuracy and the second highest speed. Taking both the accuracy and speed into account, we choose the RetinaNet to detect the street parking sign from the images.

6.3 Text Recognition

The CRNN consists of four different stages [23]: (1) Normalize the input text image by transformation, (2) Map the image to the representation, which concentrates on the relative attributes by feature extraction, (3) Capture the context within a sequence of characters by utilizing sequence modeling, and (4) Estimate the output character sequence by prediction. Table 4 shows the accuracy of different models. AS we can observe, TPS-ResNet-BiLSTM-Attn has the best accuracy. Therefore, we use it for the street parking sign text recognition. After fine-tuning the CRNN with TPS-ResNet-BiLSTM-Attn, the final accuracy is 97.818%

Model	mAP	$AP_{0.5}$	$AP_{0.75}$	FPS
SSD-300	0.803	0.978	0.968	16.10
Faster_RCNN(ResNet50)	0.827	0.979	0.970	14.20
Faster_RCNN(ResNet101)	0.834	0.976	0.975	11.60
Yolov3	0.825	0.979	0.975	18.79
RetinaNet(ResNet50)	0.849	0.984	0.981	17.00
RetinaNet(ResNet101)	0.852	0.982	0.981	12.70

Table 3: mAP and FPS of different object detection models

Model	Accuracy
Google OCR API	87.180
None-ResNet-None-CTC.pth	86.646
None-VGG-BiLSTM-CTC.pth	85.127
None-VGG-None-CTC.pth	76.392
TPS-ResNet-BiLSTM-Attn.pth	**90.823**
TPS-ResNet-BiLSTM-CTC.pth	89.81

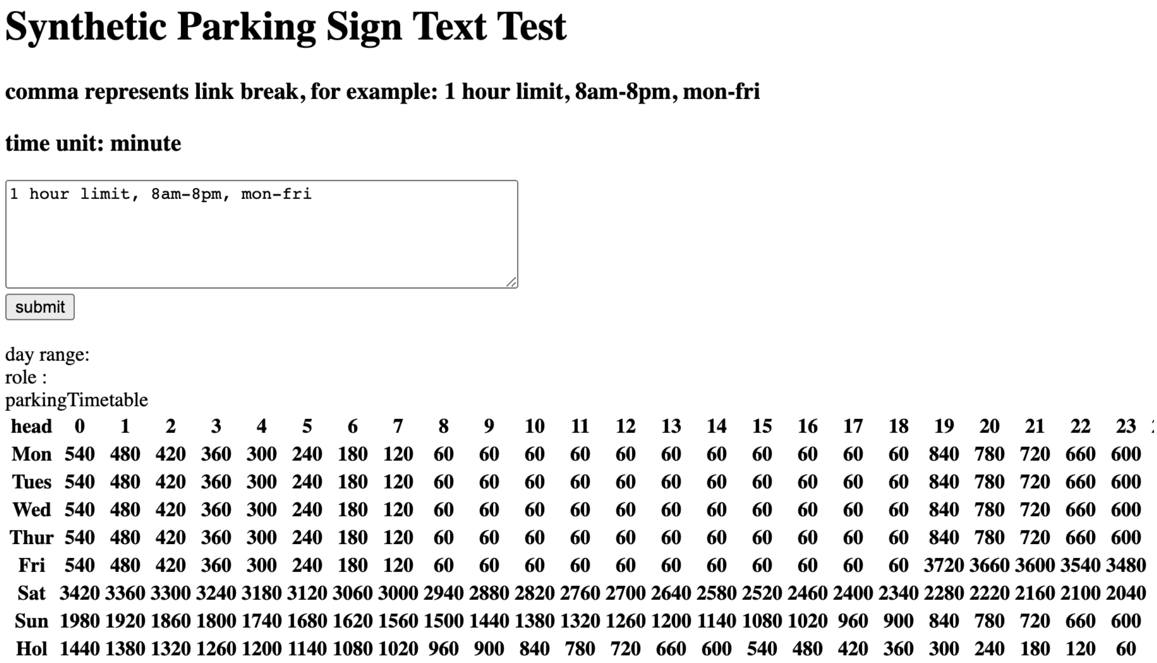
Table 4: Accuracy of different architectures

6.4 No Parking Symbol Classification

We fine-tuned SqueezeNet for no parking symbol detection. The accuracy of detection is 99%.

6.5 Rule Generation Engine

To cover the 91 branches in the street parking sign forest, we develop a collection of similar rule generation functions. However, until now, there is no ground truth for street parking rules to conduct the tests automatically. The output must be evaluated one by one manually. For each code branch as shown in Fig. 5, we test at least 3 different street parking signs to ensure it can output reliable and accurate results. We developed a website to test the Rule Generation Engine as shown in Fig. 12. Users can input the street parking sign text on the website for testing. The website will print out the timetable for the allowed parking time at each time point on every weekday for different kinds of vehicles.



Synthetic Parking Sign Text Test

comma represents link break, for example: 1 hour limit, 8am-8pm, mon-fri

time unit: minute

```
1 hour limit, 8am-8pm, mon-fri
```

submit

day range:
role :
parkingTimetable

head	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23 :
Mon	540	480	420	360	300	240	180	120	60	60	60	60	60	60	60	60	60	60	60	840	780	720	660	600
Tues	540	480	420	360	300	240	180	120	60	60	60	60	60	60	60	60	60	60	60	840	780	720	660	600
Wed	540	480	420	360	300	240	180	120	60	60	60	60	60	60	60	60	60	60	60	840	780	720	660	600
Thur	540	480	420	360	300	240	180	120	60	60	60	60	60	60	60	60	60	60	60	840	780	720	660	600
Fri	540	480	420	360	300	240	180	120	60	60	60	60	60	60	60	60	60	60	60	3720	3660	3600	3540	3480
Sat	3420	3360	3300	3240	3180	3120	3060	3000	2940	2880	2820	2760	2700	2640	2580	2520	2460	2400	2340	2280	2220	2160	2100	2040
Sun	1980	1920	1860	1800	1740	1680	1620	1560	1500	1440	1380	1320	1260	1200	1140	1080	1020	960	900	840	780	720	660	600
Hol	1440	1380	1320	1260	1200	1140	1080	1020	960	900	840	780	720	660	600	540	480	420	360	300	240	180	120	60

Figure 12: Screenshot of the Rule Generation Engine Test website

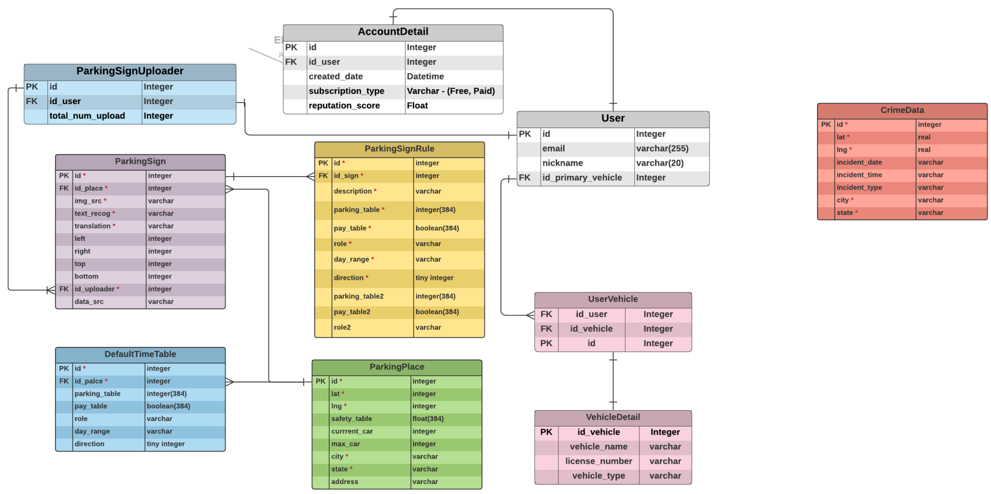


Figure 13: ER diagram of the system database

6.6 Street Parking Sign Detection and Recognition System

We deploy our PSDRRG pipeline on a server in Django to process the street parking sign images and store them in our database(Fig. 13). The database also includes information related to street parking rules, users, and vehicles.

The server is deployed on a machine with Intel(R) Core(TM) i7-8700 CPU, 16GB of RAM, and a GTX-1080Ti graphics card with 11GB of memory. We release several web APIs for interacting with saved street parking signs in our database. For example, users can upload the street parking sign image and check the nearby street parking information. When users input a destination address, the system can provide the nearby street parking information.

7 Conclusion and Future Work

We present a novel pipeline of detecting, recognizing, and understanding street parking signs. Our system can guide both human drivers and smart vehicles for street parking. In addition, we build and annotate data sets to fine-tune OCR models for street parking sign text recognition. The evaluations have shown that our system can achieve a high accuracy on detecting street parking signs and generating street parking rules.

Due to the lack of data, we still did not get the fine-tuned perfect models for the street parking sign text detection and recognition. Therefore, we plan to collect and annotate more street parking signs to improve the system's performance. Moreover, even our street parking signs were from 10 states, we can not ensure that we have already covered all types of street parking signs. As a result, we need to seek for more street parking signs and add more reliable rule generation functions to the engine as needed.

References

[1] T. Lin, P. Goyal, R. Girshick, K. He and P. Dollár, "Focal Loss for Dense Object Detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 2, pp. 318-327, 1 Feb. 2020, doi: 10.1109/TPAMI.2018.2858826.

[2] Redmon, J. & Farhadi, A. (2018), 'YOLOv3: An Incremental Improvement' , cite arxiv:1804.02767Comment: Tech Report.

[3] Liu W. et al. (2016) SSD: Single Shot MultiBox Detector. In: Leibe B., Matas J., Sebe N., Welling M. (eds) Computer Vision – ECCV 2016. ECCV 2016. Lecture Notes in Computer Science, vol 9905. Springer, Cham. https://doi.org/10.1007/978-3-319-46448-0_2

[4] R. Girshick, J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation," 2014 IEEE Conference on Computer Vision and Pattern Recognition, Columbus, OH, 2014, pp. 580-587, doi: 10.1109/CVPR.2014.81.

[5] R. Girshick, "Fast R-CNN," 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, 2015, pp. 1440-1448, doi: 10.1109/ICCV.2015.169.

[6] S. Ren, K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 6, pp. 1137-1149, 1 June 2017, doi: 10.1109/TPAMI.2016.2577031.

[7] Mirsharif, Q., Théophile Dalens, M. Sqalli and V. Balali. "Automated Recognition and Localization of Parking Signs Using Street-Level Imagery." (2017).

[8] Irshad, Humayun & Mirsharif, Qazaleh & Prendki, Jennifer. (2018). Crowd Sourcing based Active Learning Approach for Parking Sign Recognition.

[9] Kai Wang, B. Babenko and S. Belongie, "End-to-end scene text recognition," 2011 International Conference on Computer Vision, Barcelona, 2011, pp. 1457-1464, doi: 10.1109/ICCV.2011.6126402.

[10] L. Neumann and J. Matas, "Real-time scene text localization and recognition," 2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, 2012, pp. 3538-3545, doi: 10.1109/CVPR.2012.6248097.

[11] T. Wang, D. J. Wu, A. Coates and A. Y. Ng, "End-to-end text recognition with convolutional neural networks," Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012), Tsukuba, 2012, pp. 3304-3308.

[12] Bahlmann, C. et al. "A system for traffic sign detection, tracking, and recognition using color, shape, and motion information." IEEE Proceedings. Intelligent Vehicles Symposium, 2005. (2005): 255-260.

[13] P. Sermanet and Y. LeCun, "Traffic sign recognition with multi-scale Convolutional Networks," The 2011 International Joint Conference on Neural Networks, San Jose, CA, 2011, pp. 2809-2813, doi: 10.1109/IJCNN.2011.6033589.

[14] J. Stallkamp, M. Schlipsing, J. Salmen and C. Igel, ”The German Traffic Sign Recognition Benchmark: A multi-class classification competition,” The 2011 International Joint Conference on Neural Networks, San Jose, CA, 2011, pp. 1453-1460, doi: 10.1109/IJCNN.2011.6033395.

[15] M. O. Rahman, F. A. Mousumi, E. Scavino, A. Hussain and H. Basri, ”Real time road sign recognition system using artificial neural networks for bengali textual information box,” 2008 International Symposium on Information Technology, Kuala Lumpur, 2008, pp. 1-8, doi: 10.1109/ITSIM.2008.4631688.

[16] Y. Joshi, P. Gharate, C. Ahire, N. Alai and S. Sonavane, ”Smart parking management system using RFID and OCR,” 2015 International Conference on Energy Systems and Applications, Pune, 2015, pp. 729-734, doi: 10.1109/ICESA.2015.7503445.

[17] Mariusz Bojarski, Davide Del Testa, Daniel Dworakowski, Bernhard Firner, Beat Flepp, Prasoon Goyal, Lawrence D. Jackel, Mathew Monfort, Urs Muller, Jiakai Zhang, Xin Zhang, Jake Zhao and Karol Zieba.”End to End Learning for Self-Driving Cars”, 2016;arXiv:1604.07316.

[18] https://www.robots.ox.ac.uk/ vgg/software/via/

[19] Google LLC. Google cloud vision ai, 2019.

[20] Jaderberg, Max et al. ”Reading Text in the Wild with Convolutional Neural Networks.” International Journal of Computer Vision 116 (2015): 1-20.

[21] A. Gupta, A. Vedaldi and A. Zisserman, ”Synthetic Data for Text Localisation in Natural Images,” 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, 2016, pp. 2315-2324, doi: 10.1109/CVPR.2016.254.

[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 770-778, 2016.

[23] Baek, Jeonghun et al. ”What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis.” 2019 IEEE/CVF International Conference on Computer Vision (ICCV) (2019): 4714-4722.

[24] Zhi Tian, Weilin Huang, Tong He, Pan He, and Yu Qiao. Detecting text in natural image with connectionist text proposal network. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, Computer Vision - ECCV 2016, pages 56-72, Cham, 2016. Springer International Publishing.

[25] Xinyu Zhou, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. East: An efficient and accurate scene text detector. In The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), July 2017.

[26] B. Shi, X. Bai, and C. Yao. An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(11):2298-2304, Nov 2017.

[27] Jiang, Zhongyu. "Street Parking Sign Detection, Recognition and Trust System." 2019.

VITA

Jiayu Li received the B.E. degree in Network Engineering from the Beijing University of Posts and Telecommunications, Beijing, China, in 2019. She is a Computer Science and Systems M.Sc. student at the University of Washington, Tacoma.