Street Parking Sign Detection, Recognition and Trust System

Zhongyu Jiang

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

Master of Science in Computer Science and Systems

University of Washington

2019

Committee:

Wei Cheng, Chair

Juhua Hu

Program Authorized to Offer Degree:
Computer Science and Systems

University of Washington

Abstract

Street Parking Sign Detection, Recognition and Trust System

Zhongyu Jiang

Chair of the Supervisory Committee:
Assistant Professor Wei Cheng
Computer Science and Systems

Parking is one of the major problems in autonomous driving. Although cars can park in a parking spot automatically now, they can't find where they can park. In this thesis, we propose a novel street parking sign detection and recognition pipeline. This pipeline detects and recognizes street parking signs in images based on deep neural networks. After testing several models, we adopt RetinaNet [17] as our street parking sign detection model, and CTPN [34]-EAST [37]-CRNN [30] as our street parking sign recognition pipeline. To train the neural network, we build a street parking sign dataset containing street parking sign images, bounding boxes, and text. To the best of our knowledge, this is the first work on street parking sign detection and recognition.

In addition, we also build a street parking sign detection and recognition system, which includes a server and an app. Users can use the app for uploading and check any street parking signs around. For improving the data credibility, we purpose a trust system for evaluating the confidence scores and reputation level of parking sign uploading and users. Based on this trust system, our system will get much more reliable data and provide correct information for users.

TABLE OF CONTENTS

Page

List of Figures . . . iii

Glossary . . . iv

Chapter 1: Introduction . . . 1
- 1.1 Contributions . . . 2

Chapter 2: Relative Work . . . 3
- 2.1 Object Classification . . . 3
- 2.2 Object Detection and Street Parking Sign Detection . . . 3
- 2.3 Text Recognition . . . 4
- 2.4 Trust System . . . 7

Chapter 3: System Architecture . . . 8

Chapter 4: Method . . . 10
- 4.1 Street Parking Sign Detection . . . 10
- 4.2 Content Recognition . . . 10
- 4.3 Trust System . . . 16

Chapter 5: Street Parking Sign Detection and Recognition Dataset . . . 20

Chapter 6: Experiment . . . 23
- 6.1 Experiment Environment . . . 23
- 6.2 Street Parking Sign Detection . . . 23
- 6.3 Text Recognition . . . 23
- 6.4 Arrow and No Parking Classification . . . 23
- 6.5 Trust System . . . 23

6.6 Parking Sign Detction and Recognition System 27
6.7 Conclusion 27

Bibliography 29

LIST OF FIGURES

Figure Number Page

2.1 Some examples for San Francisco Parking Sign Dataset [13] 5

3.1 PSDR Pipeline 9

3.2 Trust system Pipeline 9

4.1 Figure 4.1a: example for block in invert color indicating the length of allowed parking time (2 hours) and bidirectional arrow. Figure 4.1b: example for no parking symbol. Figure 4.1c: example for allowed parking symbol. Figure 4.1d: example of other types of symbols (indicating that violators will be towed away). 12

4.2 Trust system Pipeline 13

4.3 Example for multiple blocks in invert color, and neither of them locates in the left upper corner. 14

4.4 $R_b(u, x)$ with $\mu_{up} = 4.9$, $\sigma_{up} = 1.4$ and $(u) = 1$ 19

5.1 Some examples in our street parking sign detection dataset 21

5.2 Some examples in our parking sign text recognition dataset 21

5.3 Some examples in our arrow and No Parking symbol classification dataset. Figure 5.3a and Figure 5.3b are two signs with right and bidirectional arrows. Figure 5.3c and Figure 5.3d are two signs with and without No Parking symbol. 22

6.1 Adversary simulation with different GPS errors 25

6.2 Adversary simulation with different account ages 26

6.3 Some screen shots of our app 28

GLOSSARY

ARTIFICIAL NEURAL NETWORK: A computation system, which is inspired by biological neural network and performs well in machine learning.

OCR: Optical Character Recognition. Recognize text in images.

OBJECT CLASSIFICATION: A task in computer vision. Classifying objects in images.

OBJECT DETECTION: A task in computer vision. Detecting objects in images.

TRUST SYSTEM: A system which can detect and eliminate invalid data in a data collection system.

PSDR PIPELINE: Street Parking Sign Detection and Recognition Pipeline

ACKNOWLEDGMENTS

Thanks to the help from my advisors and friends. They give me the power to go far in research.

DEDICATION

to my parents and friends

Chapter 1

INTRODUCTION

With the development of Artificial Intelligence, Network, and related technologies, autonomous driving no longer becomes a word only appearing in science fiction nowadays. By combining object detection, object tracking, and depth estimation, computer vision helps cars to recover their trajectories and locate other surrounded objects. Besides, the Internet of Things(IoT) connects the city surveillance system, cars, and all the other sensors, which makes cars obtain information from a much broader environment.

While lots of experiment self-driving cars already run on the roads now. To make a car fully "self-driving," how to find a parking spot is still a problem. To find a parking spot, cars should find and recognize parking signs on the roadsides. However, around the United States, there are millions of parking signs, and in different states(even cities), parking signs may be significantly diverse. So, sometimes, even humans cannot recognize the parking rule from several parking signs in an unfamiliar environment.

Traffic sign detection [1, 4, 29, 32] used to be a popular topic in the first ten years of the 21st century and is similar to street parking sign detection. However, there are much more information and diversity on park signs, and the text on parking signs make the recognition problem of parking signs become significantly more difficult than the problem of traffic signs. So, we can not directly adopt traffic sign detection methods for our problem.

For solving this problem, firstly, we need to build a street parking sign detection and recognition(PSDR) pipeline. Although street parking signs are similar to traffic signs, and there is much work about traffic sign detection, there are much more information and diversity on park signs. So, we introduce a new, more robust method.

Object detection already becomes quite mature these years, but the lack of data is the primary difficulty for us to build a robust street parking sign detection model. In this thesis, we introduce a new street parking sign detection dataset and use yolov3 as a baseline for detection. This dataset contains signs from more than 10 states and 20 cities across the United States.

We build an app for collection, sharing, and analyzing the data. Users can upload and check signs on the map since the users' uploading data is not reliable. To get more reliable data, we also purpose a trust system to evaluate the uploading data. The trust system will generate a confidence score for uploading based on the uploading user, the uploading location, and the sensor's error. Then, the app will only show those uploading whose confidence score is higher enough.

1.1 Contributions

We propose a novel pipeline to detect, recognize, and understand street parking signs and a reliable trust system. We improve the previous models to detect parking signs. We present a new hybrid pipeline to recognize the text on the parking signs, which outperforms the state-of-the-art open-source OCR methods. We design an algorithm for systematically analyzing and understanding the content of the parking signs and inducting parking rules based on the content, which is a brand new work to the best of our knowledge. We present our dataset of street parking signs with the labels of the bounding boxes and text on each parking sign and images taken from several different angles of view. By applying our trust system, we can detect and eliminate the error or hostile data from users, which improves the reliability of the whole system.

Chapter 2

RELATIVE WORK

2.1 Object Classification

Object classification is a popular topic in the early 2010s. Deep learning methods first extract features from original images by the convolutional network, then use fully connected layers for classifying those features. Since the publication of ImageNet, more and more neural networks appears, such as AlexNet [16](8 layers), VGG [31](19 layers), GoogLeNet [33](22 layers), and ResNet [9](152 layers). However, deeper networks require more time for computing. So, researchers start working on model optimization.

Howard *et al.* [11] modify the residual layer, which reduces the number of parameters in the network and speeds up the inference. Iandola *et al.* [12] design a new module, called Fire Module, which contains several convolutional layers and activation layers and make a new network, SqueezeNet. By using the Fire Module, SqueezeNet can significantly decrease the size of the model and achieve excellent performance.

2.2 Object Detection and Street Parking Sign Detection

Object detection methods can be divided into two categories: one-stage [17, 18, 26] and two-stage methods [7, 8, 27].

2.2.1 Two-stage Method

Two-stage detection methods will firstly estimate region proposals as detection candidates. Then running another neural network or regression model, they will predict several more accurate bounding boxes and classification results.

2.2.2 One-stage Method

One-stage detection methods estimate the bounding boxes and classes of different objects simultaneously. So, one-stage methods usually are faster than two-stage methods, but have low detection accuracy.

2.2.3 Street Parking Sign Detection

Street parking sign detection is a new field, and only a few papers are discussing this problem. Mirsharif *et al.* [22] use SVM(Support Vector Machine) to detect the parking sign in the picture. However, this traditional machine learning method can't get us reliable results. Also, because of the sliding windows method, their algorithm can not find parking signs in different sizes.

Irshad *et al.* [13] build their own street parking sign detection dataset. They collect parking sign images in San Francisco and test two detection methods: SSD [18] and Yolov2 [25]. However, signs in their dataset are too small to recognize text(Fig. 2.1). So, we introduce a new street parking sign detection dataset in this thesis.

2.3 Text Recognition

The second step of our street parking sign detection and recognition system is to understand the content of the parking signs, and most of the information is provided by the text on the signs, in which character recognition plays an important role. Wang *et al.* [15] proposed an effective pipeline to detect and recognize characters. The pipeline includes four steps: to detect the potential location of characters using a sliding window and random ferns [2], to recognize the characters by pictorial structures [6], to infer words by a trie-structured lexicon, and to re-score the predicted words by SVM classifier. Another kind of character recognition method proposed by Neumann and Matas [23] leverages pixel-wise gradient to propose several expected extremal regions (ER) and computes statistical and graphical features [21] for the AdaBoost classifier [28].

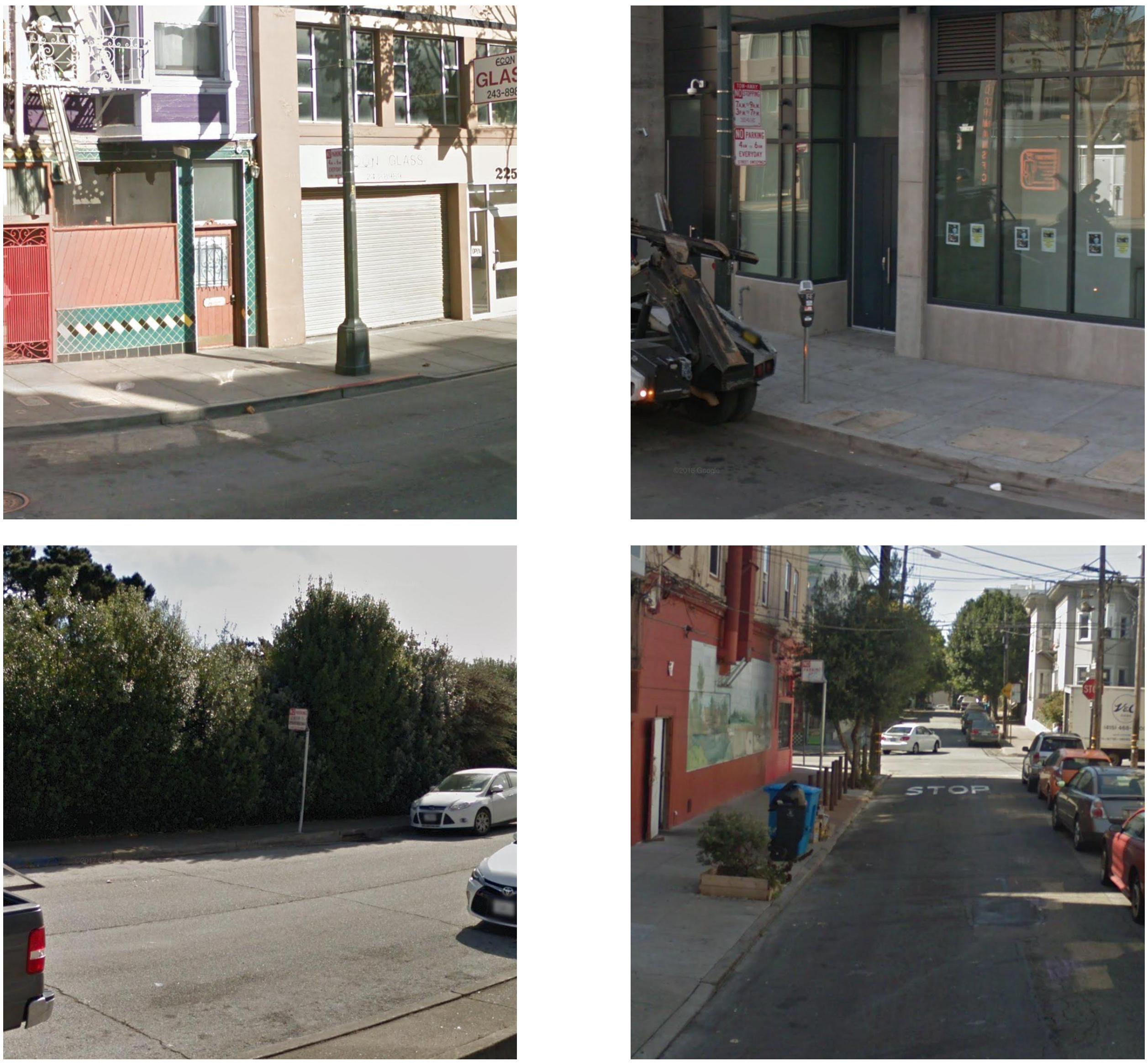

Figure 2.1: Some examples for San Francisco Parking Sign Dataset [13]

As the emergence of the application of a convolutional neural network to image processing and computer vision, Wang *et al.* [35] presented a novel method to perform end-to-end character recognition by CNN. Instead of using pixel-wise features, this network performs convolution several times, to extract feature maps of different depth, and classifies those feature maps to locate text lines of the corresponding size. According to the result claimed in [35], the CNN method outperformed the previous methods without CNN, 20% better than [15], and 30% than [23].

In our research, we figure out that there are no previous studies on recognizing and understanding the specific category of parking signs. Instead, there are some relevant works in recognizing generic traffic signs. Traffic sign recognition methods mainly depend on recognition by classification, which means for different kinds of signs, they are different categories, for example, speed limit signs with different speed limits. Bahlmann *et al.* [1] proposed a pipeline to detect, track and recognize traffic signs. The method is based on graphical features like color and shape. The features are extracted after normalization of the images, and the first 25 most significant features by linear discriminant analysis are fed into a maximum likelihood classifier. Sermanet and LeCun [29] proposed a method to recognize traffic signs leveraging convolutional neural networks. Convolution is performed in two stages to extract multi-scale features, and the output features of those two stages are concatenated and fed into a full-connection classifier of 30 categories. Stallkamp *et al.* [32] compared several traffic sign recognition methods. The result illustrated that CNN methods outperformed the methods using linear discriminant analysis.

However, parking sign recognition is much more difficult. The most significant drawback of sign recognition by classification is that the methods are not able to reproduce the text on the signs, and the number of categories is limited. The text on the parking signs is subtle and varies among different signs, for which the classification method without text recognition is not sufficient.

Some studies are conducted to apply Optical Character Recognition(OCR) methods for traffic sign recognition. In the study of Rahman *et al.* [24], the text recognition of simple text constructed from one or two single words is applied to recognize the traffic signs. Joshi *et al.* [14] presented a system that leverages OCR to recognize plate numbers of vehicles and authorize whether the vehicles are allowed to park in specific parking lots.

2.4 Trust System

The aim of trust system is to improve the reliability of a data collection system. Wang *et al.* [36] build a trust system on a anonymous data collection system. Based on information collected by some nearby sensors, data centers can know the ”reputation” of each sensor and can detect the invalid or inaccurate data. Fan *et al.* [5] focus building a encryption system to make the data more secure.

Chapter 3
SYSTEM ARCHITECTURE

Our pipeline(Fig. 3.1) contains three components: street parking sign detection, parking sign text recognition.

The input images can be taken by users or self-driving cars, and after passing through a street parking sign detection network, we will crop the signs out based on detected bounding boxes. A text recognition module will process each of the cropped signs and generate recognized words and their bounding boxes. After concatenating those words and their bounding boxes from different signs in the same image together, a text processing module will take them as input and produce the final parking rules based on all the signs.

After getting the detection results, the trust system will calculate the confidence score of this uploading, which is based on the GPS error σ_g, the user claimed location L_u, GPS location L_g and user's reputation level R_u. We will introduce the detailed information in the following section.

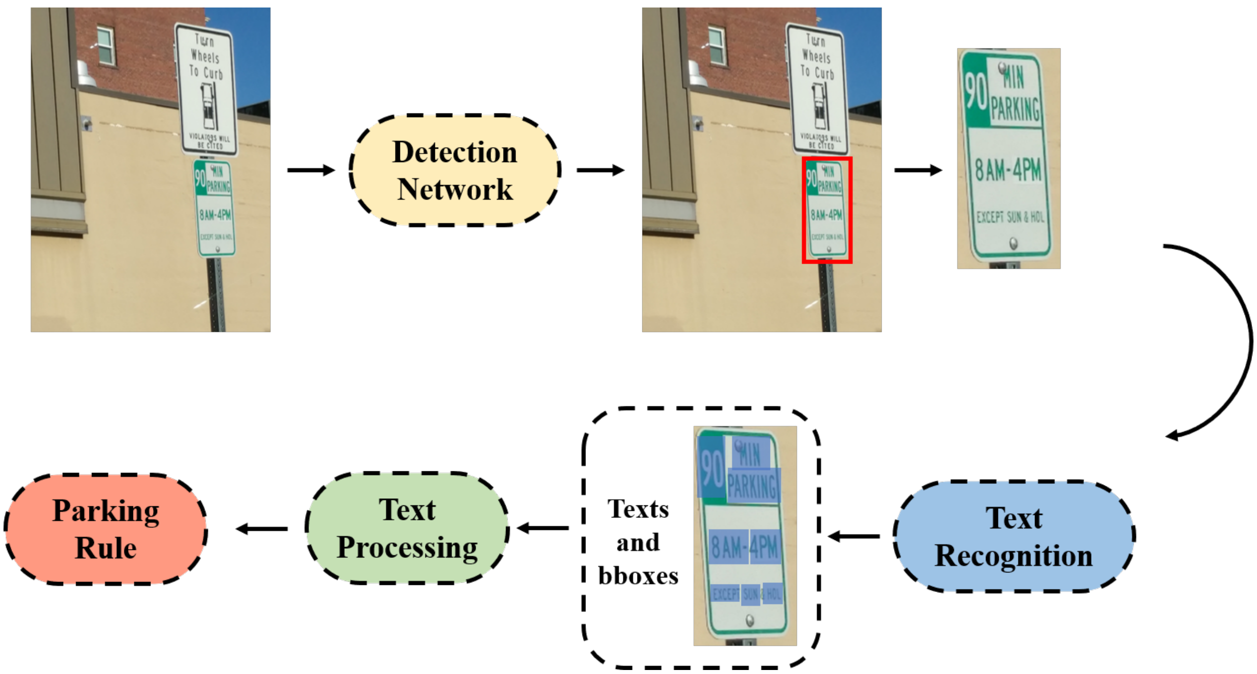

Figure 3.1: PSDR Pipeline

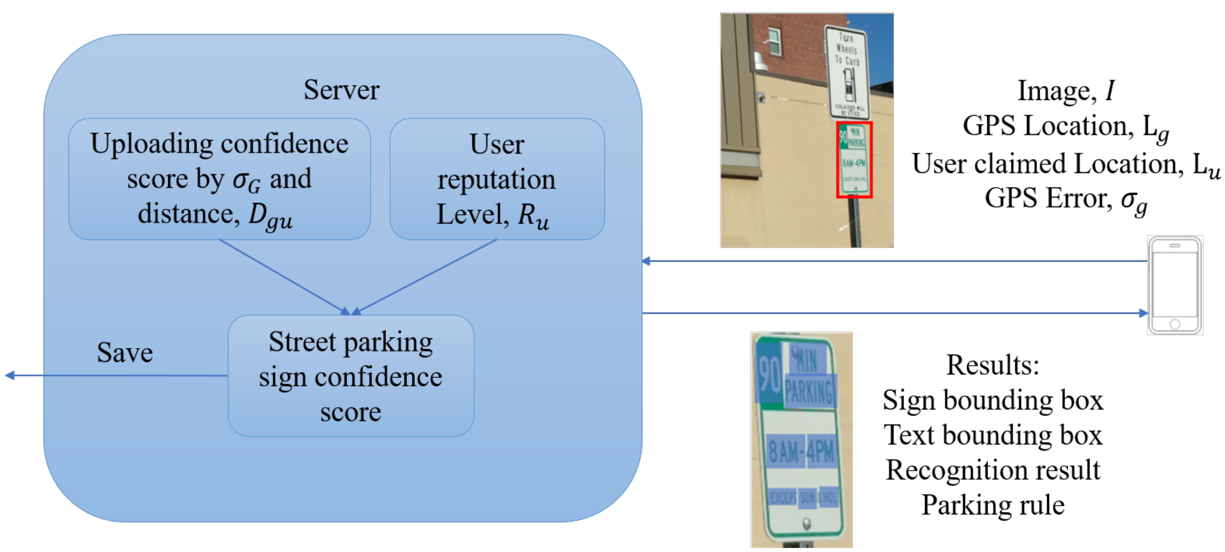

Figure 3.2: Trust system Pipeline

Chapter 4
METHOD

4.1 Street Parking Sign Detection

We test several object detection models on street parking sign detection. Finally, we choose the RetinaNet [17] as our backend detection method. RetinaNet is an one-stage object detection method, which is faster than two-stage methods and has reliable enough performance. We train our detection model with the pre-trained weight on ImageNet.

4.2 Content Recognition

4.2.1 Components of Parking Signs

Some features of parking signs make recognizing the content of them different from recognizing generic traffic signs or text. The first notable feature is that many parking signs have a rectangular block with inverted colors. Typically, this block locates in the left upper corner of a parking sign, indicating the length of allowed parking time, as Fig. 4.1a shows, or the word **NO** indicating no parking. The invert color scheme of the block makes the text in the block difficult to be recognized accurately.

In our tests, general OCR methods [35] cannot accurately reproduce the number or word in such blocks in more than half of test cases if we feed cropped out sign images into the model. Another important component of parking signs is the arrow. To specify the region where the parking sign is effective, we need to clarify if there are arrows on the signs and directions of those arrows. Arrows on the parking signs fall into one of three categories: pointing to the left, to the right, or bidirectional like Fig. 4.1a. In some cases, the rules on parking signs are not indicated by the text "No Parking," but by a no parking symbol like Fig. 4.1b instead. Also, in some cases, there is an allowed parking symbol like Fig. 4.1c on the sign. Based on the results of each OCR model mentioned, none of the models can provide stable and predictable output for the no parking symbols or allowed parking symbols. Other types of symbols also show on parking signs, as Fig. 4.1d is an example of a tow-away symbol, which means that vehicles violating the rule will be towed away. The figures of other types of symbols may vary from one to another, making it hard to be classified, while the semantic information of the symbols is always implied by the text on the parking sign. So, in our system, we set those symbols aside. We consider allowed parking symbols to be not informative similarly, and those symbols are ignored as well.

4.2.2 Blocks in Invert Color

The fact that general OCR models perform relatively bad on the text in the blocks in the invert color scheme shows the necessity of special process on such blocks. We further find that if we crop out the region that only contains the blocks and feed that region to models, the accuracy will improve significantly. In our pipeline, we first detect regions of the blocks in invert color if they exist and then feed them into OCR models. We notice the fact that one parking sign may have multiple such blocks in invert color, and the blocks do not necessarily occur in the left upper corner, as Fig. 4.3. Fortunately, as we collected more than 2000 different parking signs across over 20 cities, all such blocks only occur against the left or right edge of the parking signs, which we observed. That property eases the difficulty of detecting and extracting the blocks. Another property is that such blocks are always rectangular, for our detection algorithm gives a rectangle region for every detected block as output.

After all, we notice that no block spans more than half of the parking sign in width, that is, every block locates either in the left half of the sign or the right half, and all the blocks in the same side of one sign have the same width. The detection of the blocks locating in the left half includes three steps: first, finding the possible right edges of the blocks, second, vertically splitting the region bounded by the proposed right edges and the left edge of the sign, and third, determining the best right edge of the blocks. A symmetric process

Figure 4.1: Figure 4.1a: example for block in invert color indicating the length of allowed parking time (2 hours) and bidirectional arrow. Figure 4.1b: example for no parking symbol. Figure 4.1c: example for allowed parking symbol. Figure 4.1d: example of other types of symbols (indicating that violators will be towed away).

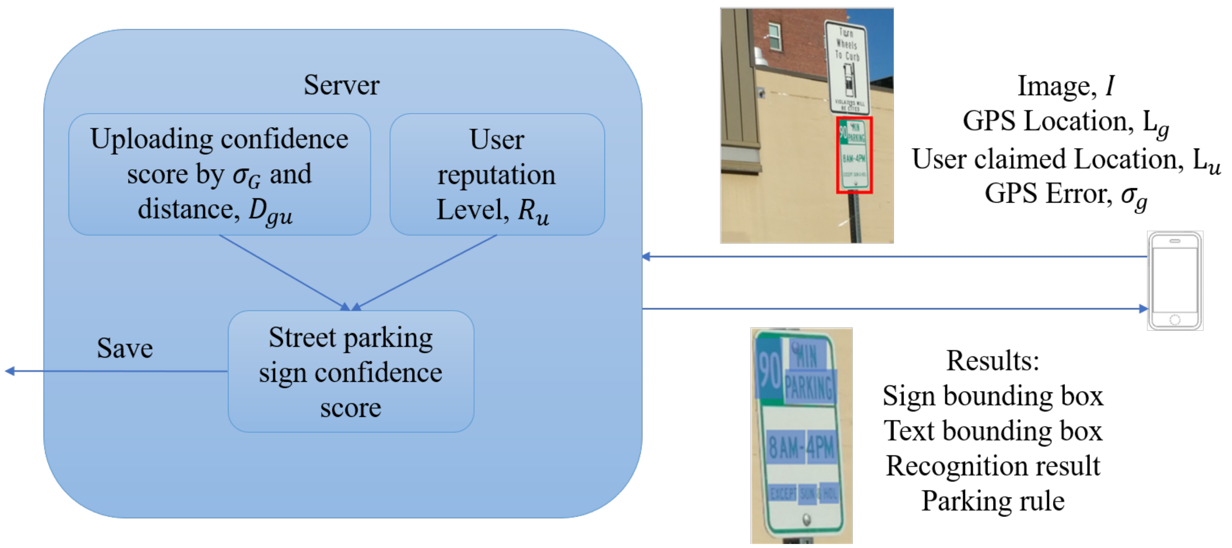

Figure 4.2: Trust system Pipeline

is performed to the right side of the sign. We binarize the input image and scan from the vertical midline of the image, which could be denoted as $u = \frac{W}{2}$. Decrease u by one pixel every time until the mean of the pixels in that vertical line is smaller than a threshold δ_1.

$$\sum_{v=1}^{H} \frac{p_{v,u_j}}{H} < \delta_1 \tag{4.1}$$

We may find multiple u_j meet (4.1). For consecutive u_j, remove all the smaller ones and only leave the rightmost ones. List the remaining u_j as $u_1, u_2, ..$ as the proposed right edge of the blocks. For each remaining u_j, scan from $v = 0$ down to $v = H$ horizontally. If the pointer v_i is initialized or in a white region, *i.e.* not in the block in revert color, scan for a v_i that meets

$$\sum_{u=1}^{u_j} \frac{p_{v_i,u}}{u_j} < \delta_2. \tag{4.2}$$

That denotes an upper edge of a proposed block. Then search for a lower edge by

$$\sum_{u=1}^{u_j} \frac{p_{v_i,u}}{u_j} > \delta_3. \tag{4.3}$$

Figure 4.3: Example for multiple blocks in invert color, and neither of them locates in the left upper corner.

Stop when $v_i = H$. Pair the upper edges with the lower edges and for each proposed right edge, some blocks of invert color have been segmented. Denote the upper edge of the block as $v = v_u$ and the lower edge $v = v_l$. For each proposed right edge u_j, first remove the blocks which

$$\frac{v_l - v_u}{u_j} < \frac{1}{3} \tag{4.4}$$

to remove the proposed blocks which are actually horizontal rules like Fig. 4.1c. Also remove the blocks which

$$\sum_{v=v_u}^{v_l} \frac{p_{v,u_j}}{v_l - v_u} > \delta_4 \tag{4.5}$$

which fixes the prediction on right edge. Sort the proposed right edges by the number of remaining proposed blocks and choose the one with most blocks (the rightmost one if draw) as the right edge.

4.2.3 Arrow and No Parking Symbol

Since there are some signs in the same images representing different rules for different directions, we need to identify the different types of arrows in signs. As a result, we add an arrow classification network to our system. We adopt SqueezeNet [12] as our classification network since this network architecture has fewer parameters than other classification networks and can achieve a reliable enough performance. Signs are divided into four categories: containing a left arrow, a right arrow, a bidirectional arrow, or no arrow.

Also, since our text recognition method can not recognize some special symbols(figure 5.1) in parking signs, for example, the No Parking symbol, we adopt the same network architecture as the arrow classification and divide the signs into two categories, with or without No Parking symbol.

Both of these two classification results will be fed into the text processing module to merge the text and information from different signs.

4.2.4 Text Recognition

For text recognition, based on the performance reported by [15], [23], [35] and [30], models with convolutional neural networks generally outperform the models without CNN. Convolutional recurrent neural network, shortly CRNN, is a model proposed by Shi *et al.* [30]. CRNN is a model using convolutional layers to extract feature maps and classifying by a recurrent network with LSTM [10] and CTC loss [19]. CRNN model takes a image of a single word or text line as input. The height of the input image is fixed to be 32 and the width could be flexible. The output gives the predicted text on the input image.

To crop out the region of text lines or words to be fed into CRNN model, we choose connectionist text proposal network, or shortly CTPN, proposed by Tian *et al.* [34], a model leveraging features from convolutional neural networks to detect and locate all the text lines by a recurrent network with LSTM. The main drawback of CTPN model is that it can only detect a text line, *i.e.* a rectangular region that contains a line of text. The width of the detected region spans from the width of a single character to the width of the input image of the sign. The output region of CTPN model usually includes more than one word, and CRNN model will not recognize the space between words in a text line.

A solution proposed by Zhou *et al.* [37] is a multichannel text detector using convolutional neural networks called EAST model. It performs better than CTPN model on a smaller scale, and the main feature of EAST is to detect the space between words in a single text line. Its performance is not so good as CTPN in detecting the text lines, *i.e.* it is not accurate enough in the upper and lower edge of the region of a text line, but it can give the region of a single word. So our text recognition solution is to feed the image of parking sign into CTPN first to get the bounding boxes of text lines and to crop out the region of each detected text line and leverage EAST to split the text line and output the bounding boxes of single words. The region of every word and every block in invert color is cropped and resized to the height of 32. The ratio is the same as the input of CRNN, and the output is the predicted word. For further usage, we ordered the output of CRNN by the bounding boxes provided by EAST model and blocks in invert color detection. All the bounding boxes involved is a rectangular region bounded by the coordinate of four angles.

4.3 Trust System

To build a trust system which can detect invalid data, we design two parameters, User Reputation Level, R, and Uploading Confidence Score, C. Both of these two parameters are between 0 and 1. The final trust scores T is the product of R and C.

$$T = RC \tag{4.6}$$

4.3.1 Uploading Confidence Score

Since we allow users to upload new street parking signs to the server, like a crowdsourcing system, we need to build a trust system to judge whether the coming data or users should be trusted or not. We define the confidence score, C, of a data as:

$$C = \lambda_r \lambda_t C_b \tag{4.7}$$

The basic confidence of that data, C_b, is generated when the data is uploaded. The data existing time factor λ_r, and the number of reports factor λ_t will also influence the final confidence. Users can report a sign if they think the image or the recognition result of the sign is invalid.

When users try to upload an image of a new sign, since the GPS location L_g isn't that reliable. We allow users to claim their locations, L_u, near that sign. However, allowing users to claim their locations will let users upload some invalid locations. So, since the sensor will give us current localization error σ_g, we consider the users' possible location is like a 2D Gaussian distribution with $\mu = L_g$ and $\sigma = \sigma_g$ in each axis. According to the distance D_{gu}, between users' current locations and their claimed location and GPS error σ_g, basic confidence, denoted as C_b, is:

$$C_b = \frac{1}{\sigma_g} e^{-\frac{D_{gu}}{2\sigma_g^2}} \tag{4.8}$$

When users try to update an existed sign, the basic confidence scores C_b of updating is formed by the distance between the original location and the user claimed location, D_{ou}.

$$C_b = e^{-\lambda_r Diff_r - \lambda_d D_{ou}} \tag{4.9}$$

It is reasonable to consider data that exists for a long time and receives few reports from users as more reliable data. So, we define two factors, λ_t for existing time t in weeks and λ_r for the number of reports from users r. In our experiment, these two parameters are set as follows:

$$\lambda_t = 0.5 \times (1 + min(\frac{t}{52}, 1)) \tag{4.10}$$

$$\lambda_r = e^{\frac{r}{50}} \tag{4.11}$$

4.3.2 User Reputation Level

The reputation level of a certain user u, denoted as $R(u)$, is the probability of a correct uploading by u. $R(u)$ can be set as a value between [0, 1] as initialization which is 0.5 in our experiment. A user who receives more number of reports r and has smaller account age a is obviously less trusty than other users. So, $R(u)$ is influenced by these two factors. In this experiment, $\lambda_{nr} = 0.99$, $\lambda_a = 1.001$.

$$R(u) = min(0.5\lambda_{nr}^{r}\lambda_a^{a}, 1) \tag{4.12}$$

However, users can upload tons of signs to lower the average number of received reporting. Additionally, to make sure users' behavior isn't abnormal, we introduce another term, behavior score R_b, for limiting user reputation level. The user's average number of uploading per week in the past three months is x.

$$R_b(u, x) = R(u)\frac{f(x|\mu_{up}, 4\sigma_{up})}{f(\mu_{up}|\mu_{up}, 4\sigma_{up})} \tag{4.13}$$

$f(\mu, \sigma)$ is the Gaussian distribution.

μ_{up} is the average number of users' uploading per week in the past three months, and σ_{up} is the standard deviation of that uploading number. If the user's average number of uploading per week in the past three months is x, then the final user reputation score will be:

$$R(u, x) = min(0.5\lambda_{nr}^{r}\lambda_a^{a}, 1)R_b(u, x) \tag{4.14}$$

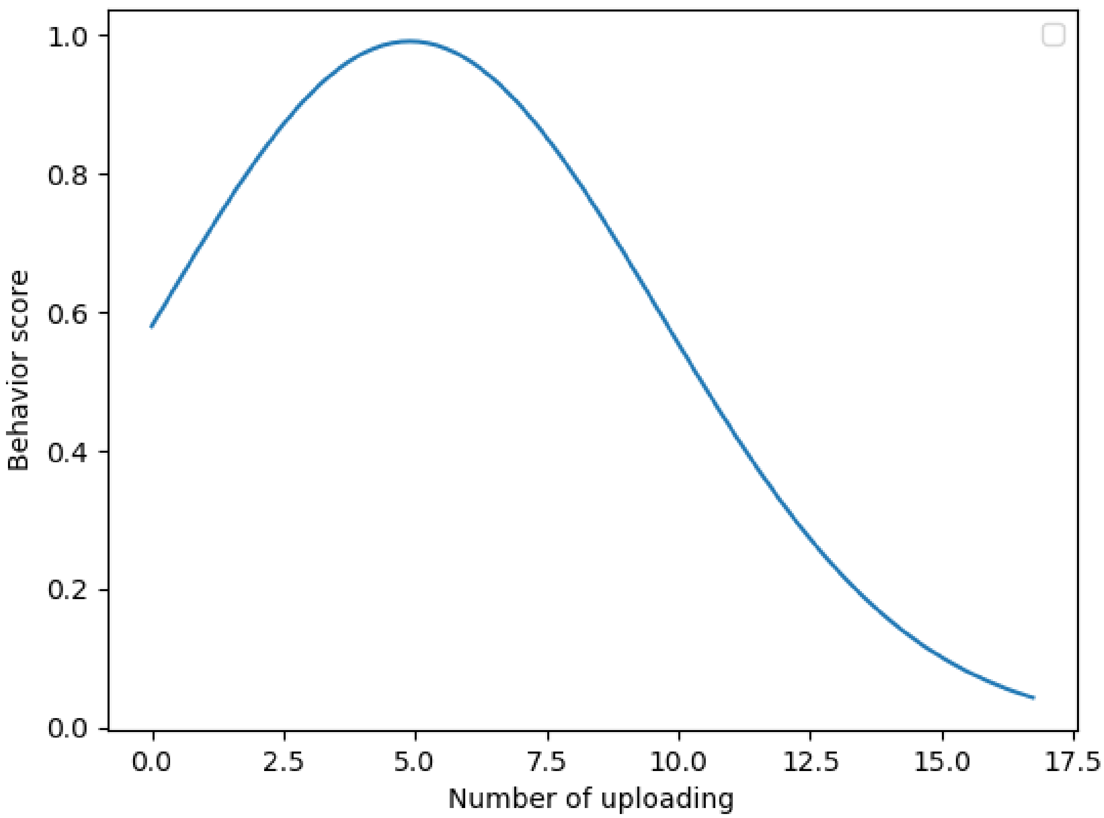

Figure 4.4: $R_b(u, x)$ with $\mu_{up} = 4.9$, $\sigma_{up} = 1.4$ and $(u) = 1$

Chapter 5

STREET PARKING SIGN DETECTION AND RECOGNITION DATASET

Since there are only a few datasets about parking signs, we collect our own street parking sign detection and recognition dataset(Fig. 5.1).

In our street parking sign detection dataset, there are in total 1764 images(1532 images for training, 232 images for testing). Those images are taken from 10 states around the US. We've annotated all the signs for training, testing, and evaluation purpose.

Besides, we also crop all the words in those signs out and annotate them as our parking sign recognition dataset(Fig. 5.2) to train our CRNN model [30]. This dataset contains 22462 and 2895 words for training and testing. To crop and label all these words, we adopt Google OCR API [20] to provide a rough result, and then manually modify the OCR results from Google OCR for improving the label accuracy.

To train the arrow and No Parking symbol classification dataset, we also label two small datasets for these two models. In the arrow classification dataset(Fig. 5.3), we crop all the signs with arrows, and there are 2088 signs, divided into 4 categories, for training and 224 signs for testing.

In the No Parking symbol classification dataset(Fig. 5.3), signs are divided into two categories: with or without No Parking symbol, and there are 1822 signs for training and 284 signs for testing.

Figure 5.1: Some examples in our street parking sign detection dataset

Figure 5.2: Some examples in our parking sign text recognition dataset

(a)

(b)

(c)

(d)

Figure 5.3: Some examples in our arrow and No Parking symbol classification dataset. Figure 5.3a and Figure 5.3b are two signs with right and bidirectional arrows. Figure 5.3c and Figure 5.3d are two signs with and without No Parking symbol.

Chapter 6

EXPERIMENT

6.1 Experiment Environment

We test these models 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 test several different models on our parking sign detection dataset. During training, we combine our dataset and Irshad's dataset [13]. As table 6.1, RetinaNet with ResNet50 [9] as backbone has the second-highest performance and the second-fastest speed. So, considering the trade-off of performance and speed, we choose the RetinaNet as our detection model.

6.3 Text Recognition

We retrain the CRNN model in our street parking sign text recognition dataset. As table 6.2, comparing with Google OCR, after training on our dataset, CRNN can achieve a better performance, since the model is fine-tuned on a street parking sign dataset.

6.4 Arrow and No Parking Classification

We train two SqueezeNet [12] on our arrow and no parking symbol classification dataset. As table 6.3, both of the two models can have a good performance in this experiment.

6.5 Trust System

To evaluate our trust system, we do some adversary simulation. For testing the uploading confidence score, we set several uploading with different GPS errors. As figure 6.1, with

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	14.00
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 6.1: Performance and FPS(frame per second) of different models

Model	Accuracy
Google OCR [20]	90
CRNN [30]	97
CRNN with data augmentation	98

Table 6.2: Performance and FPS(frame per second) of different models

Model	Accuracy
SqueezeNet(No Parking)	98.6
SqueezeNet(Arrow)	96.3

Table 6.3: Performance and FPS(frame per second) of different models

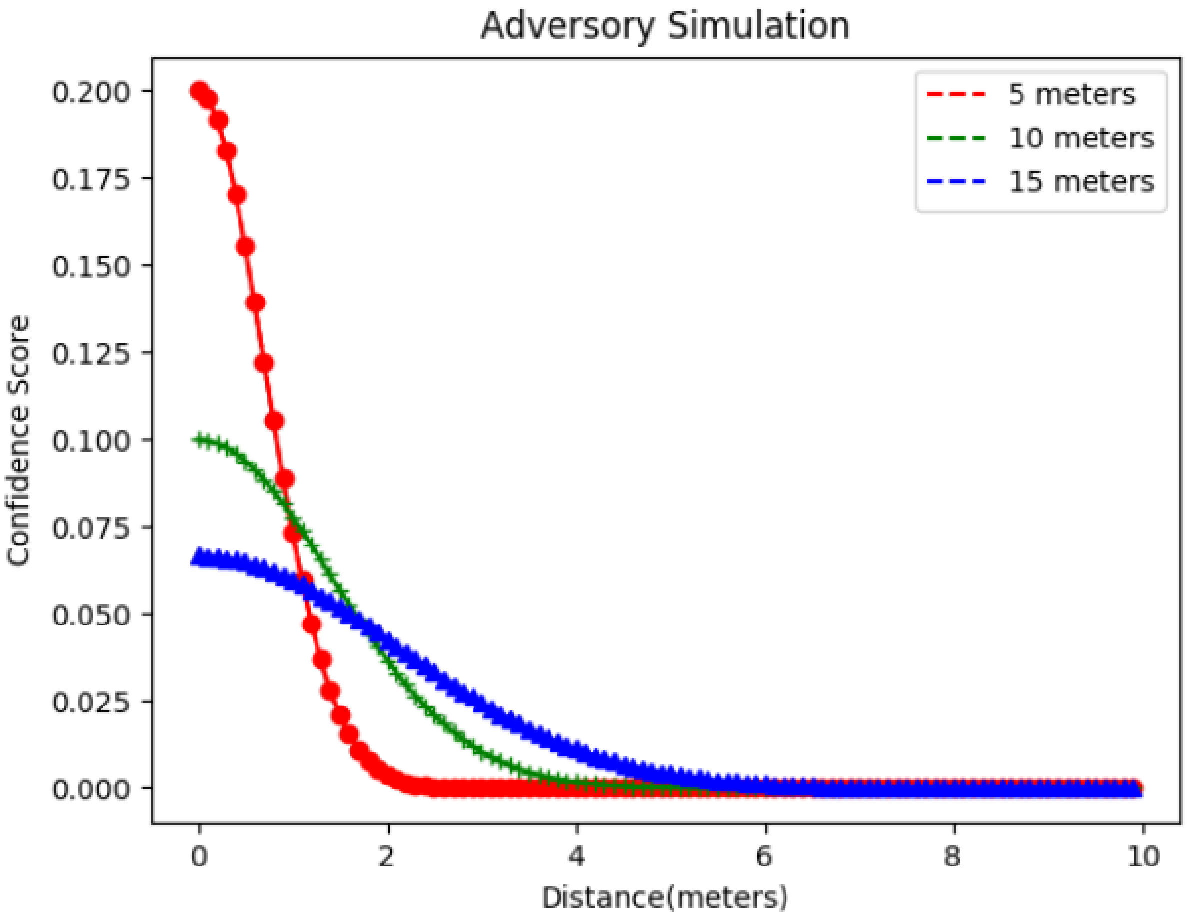

Figure 6.1: Adversary simulation with different GPS errors

different GPS localization errors, the system has different tolerance of distance between user claimed location and GPS location from the device.

To evaluate our trust system, we do some adversary simulations. For testing the uploading confidence score, we set several uploading with different GPS errors. As figure 6.1, with different GPS localization errors, the system has different tolerance of distance between user claimed location and GPS location from the device.

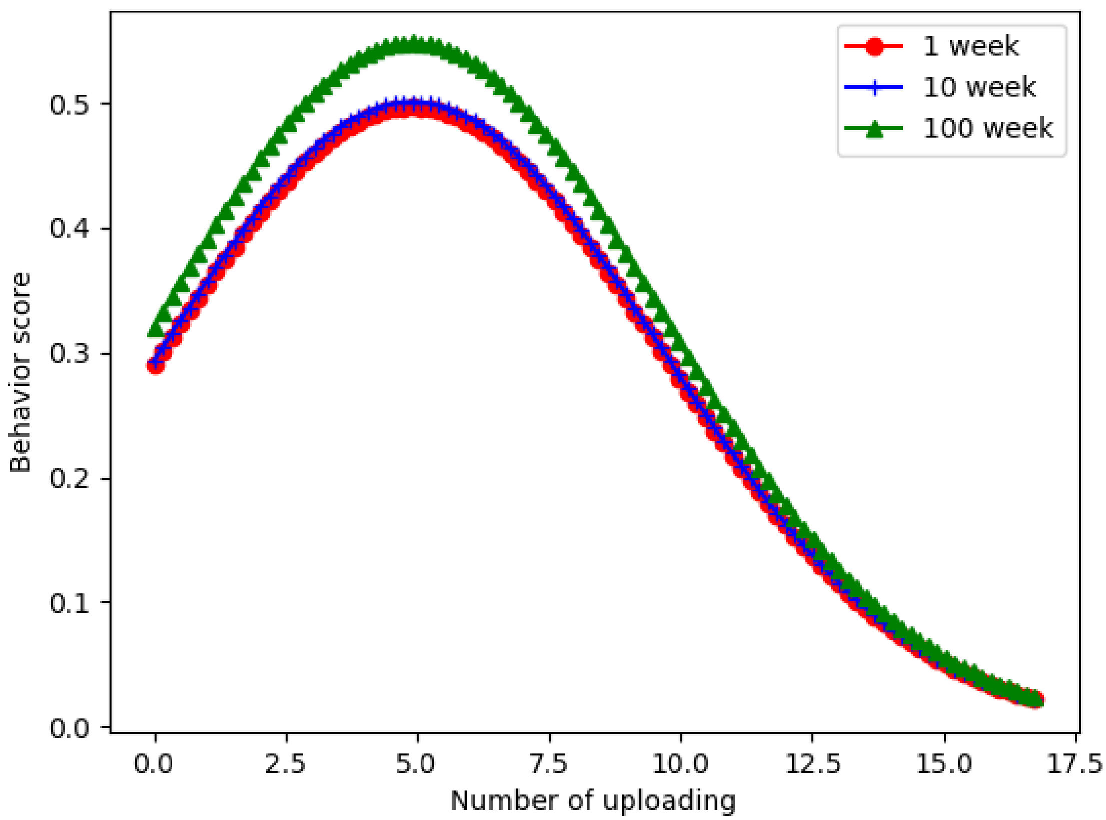

Figure 6.2: Adversary simulation with different account ages

6.6 Parking Sign Detction and Recognition System

After developing our PSDR pipeline, we deploy that pipeline on a server for processing uploaded signs and storing them in a database and develop an android app for accessing the pipeline and the stored parking signs and rules.

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 provide several web APIs for getting saved parking signs in our database. For example, through web APIs, users can access any parking signs' information, including positions, parking rules, and upload time., in any specific region.

We embed the Google Map API in our app. By accessing our app, users can upload and look for any parking signs around them.

6.7 Conclusion

We propose a novel pipeline to automatically detect and analyze street parking signs, which may help autonomous cars find their parking lots. Besides that, we also build a dataset to manage the collected data. During our testing, we achieve a favorable performance on understanding signs in images.

However, since the leak of data, we can not train a perfect model to detect text in street parking signs. As a result, a dictionary is introduced, and we filter some noise which is generated by the OCR pipeline. In the future, we plan to label more words and their bounding boxes in street parking signs and train a more robust OCR model. Also, although the street parking signs are taken from more than ten states around the United States, there are still many kinds of parking signs which are not included in our dataset. Further street parking sign data collection is also needed.

About the trust system, we propose a reliable trust system based on user reputation and uploading confidence scores. By applying this trust system, we can detect and eliminate invalid data.

(a) (b) (c) (d)

Figure 6.3: Some screen shots of our app

BIBLIOGRAPHY

[1] C. Bahlmann, Y. Zhu, Visvanathan Ramesh, M. Pellkofer, and T. Koehler. A system for traffic sign detection, tracking, and recognition using color, shape, and motion information. In *IEEE Proceedings. Intelligent Vehicles Symposium, 2005.*, pages 255–260, June 2005.

[2] A. Bosch, A. Zisserman, and X. Munoz. Image classification using random forests and ferns. In *2007 IEEE 11th International Conference on Computer Vision*, pages 1–8, Oct 2007.

[3] Keunwoo Choi, György Fazekas, Mark Sandler, and Kyunghyun Cho. Convolutional recurrent neural networks for music classification. In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2392–2396. IEEE, 2017.

[4] Arturo De La Escalera, Luis E Moreno, Miguel Angel Salichs, and José María Armingol. Road traffic sign detection and classification. *IEEE transactions on industrial electronics*, 44(6):848–859, 1997.

[5] Jingyao Fan, Qinghua Li, and Guohong Cao. Privacy-aware and trustworthy data aggregation in mobile sensing. In *2015 IEEE Conference on Communications and Network Security (CNS)*, pages 31–39. IEEE, 2015.

[6] Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Pictorial structures for object recognition. *International Journal of Computer Vision*, 61(1):55–79, Jan 2005.

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

[8] 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.

[9] 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.

[10] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9:1735–80, 12 1997.

[11] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861*, 2017.

[12] 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.

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

[14] Y. Joshi, P. Gharate, C. Ahire, N. Alai, and S. Sonavane. Smart parking management system using rfid and ocr. In *2015 International Conference on Energy Systems and Applications*, pages 729–734, Oct 2015.

[15] Kai Wang, B. Babenko, and S. Belongie. End-to-end scene text recognition. In *2011 International Conference on Computer Vision*, pages 1457–1464, Nov 2011.

[16] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[17] 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.

[18] 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.

[19] Marcus Liwicki, Alex Graves, Horst Bunke, and Jürgen Schmidhuber. A novel approach to on-line handwriting recognition based on bidirectional long short-term memory networks. In *In Proceedings of the 9th International Conference on Document Analysis and Recognition, ICDAR 2007*, 2007.

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

[21] Jiri Matas and Karel Zimmermann. A new class of learnable detectors for categorisation. In Heikki Kalviainen, Jussi Parkkinen, and Arto Kaarna, editors, *Image Analysis*, pages 541–550, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[22] 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.

[23] L. Neumann and J. Matas. Real-time scene text localization and recognition. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 3538–3545, June 2012.

[24] Mohammad Osiur Rahman, Fouzia Asharf Mousumi, Edgar Scavino, Aini Hussain, and Hassan Basri. Real time road sign recognition system using artificial neural networks for bengali textual information box. *European journal of scientific research*, 25(3):478–487, 2009.

[25] 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.

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

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

[28] Robert E. Schapire and Yoram Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336, Dec 1999.

[29] Pierre Sermanet and Yann LeCun. Traffic sign recognition with multi-scale convolutional networks. In *IJCNN*, 2011.

[30] 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.

[31] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[32] Johannes Stallkamp, Marc Schlipsing, Jan Salmen, and Christian Igel. The german traffic sign recognition benchmark: A multi-class classification competition. In *IJCNN*, volume 6, page 7, 2011.

[33] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[34] 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.

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

[36] Xinlei Oscar Wang, Wei Cheng, Prasant Mohapatra, and Tarek Abdelzaher. Artsense: Anonymous reputation and trust in participatory sensing. In *2013 Proceedings IEEE INFOCOM*, pages 2517–2525. IEEE, 2013.

[37] 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.

VITA

Zhongyu Jiang received the B.E. degree in Computer Science and Technology from the Tsinghua University, Beijing, China, in 2018. He is a Computer Science and Systems M.Sc. student at the University of Washington, Tacoma.