Tips for the Final Project

박사과정 김성빈 chengbinjin@inha.edu,
지도교수 김학일 교수 hikim@inha.ac.kr
인하대학교 컴퓨터비전 연구실

Grading Policy

Evaluation							
Mid Exam	Final Exam	Assignments	Class Activity (be present)	**Term Project**	Quiz	Presentation	Total
10 %	10 %	10%	5 %	**30 %**	25 %	10 %	100 %
2018.10.24	2018.12.05	Every week	Every week	**2018.12.12**	Every week	Every week	

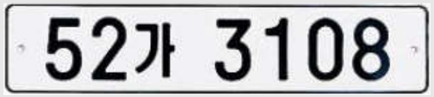
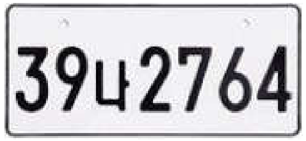
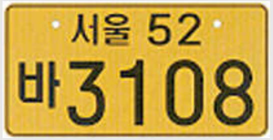
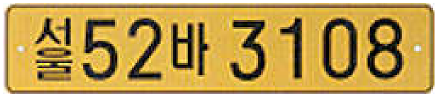
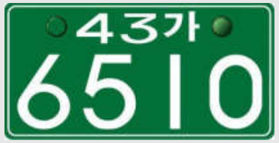
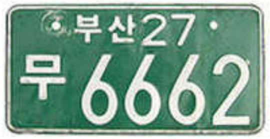
Note:

1. Mid & final exam: lectures (PPT) & coding parts in assignments
2. Assignments: assignment 1~3 of the CS231n 2016 http://cs231n.stanford.edu/2016/syllabus.html
3. Class activity: attendance check before class
4. Term project:
   - (1) license plate detection and recognition; (2) on-road object detection
   - Scores will be evaluated based on the team ranking
   - No open or commercial library is allowed.
5. Quiz: will be taken in the beginning of every class to check students' pre-studying the CS231n video
6. Presentation: 2 or 3 students every week

Python Numpy Tutorial:
http://cs231n.github.io/python-numpy-tutorial/

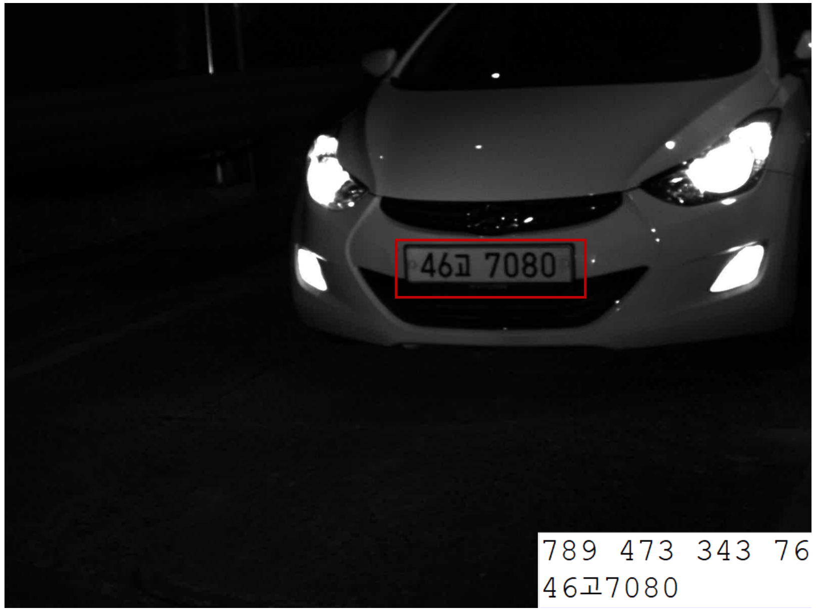
Dataset

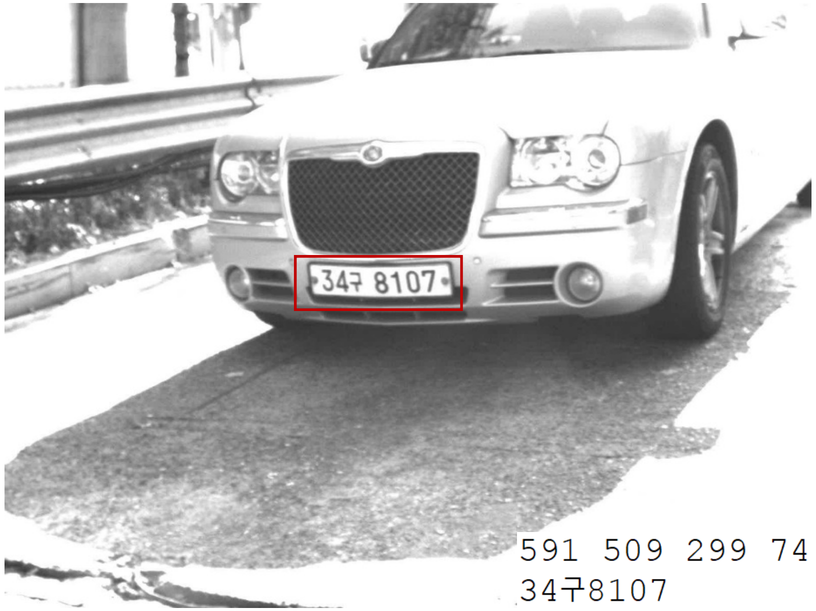
License Plate Type

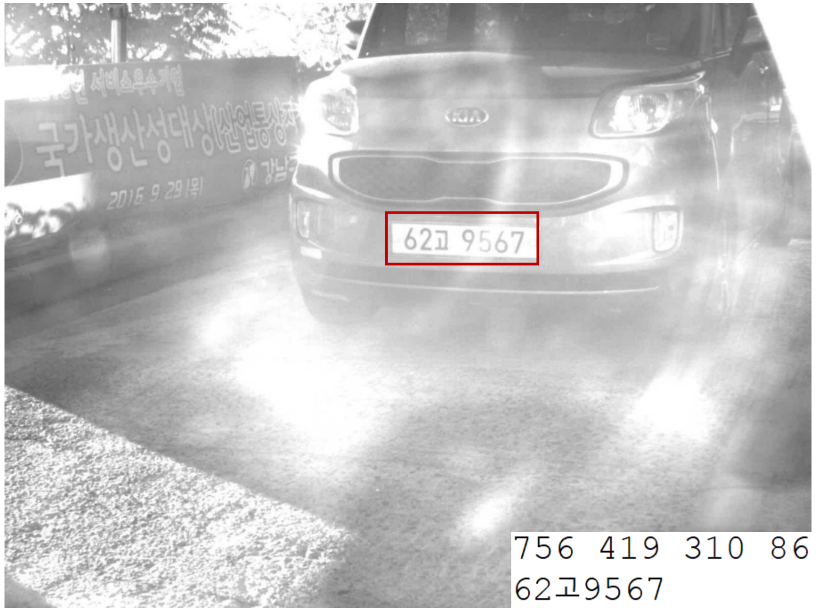
Type	License Plate		Type
1			P1
2			P2
3		서울52바3108	P3
4		서울52바3108	P4
5		43가6510	P5
6		부산27무6662	P6

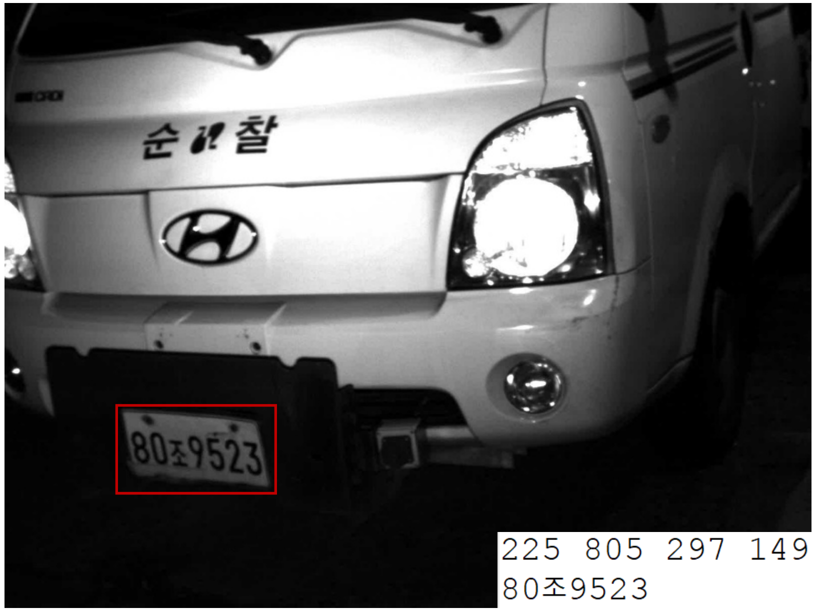
Recognizing plate type is not our task

Parking Data (285 Test Samples)

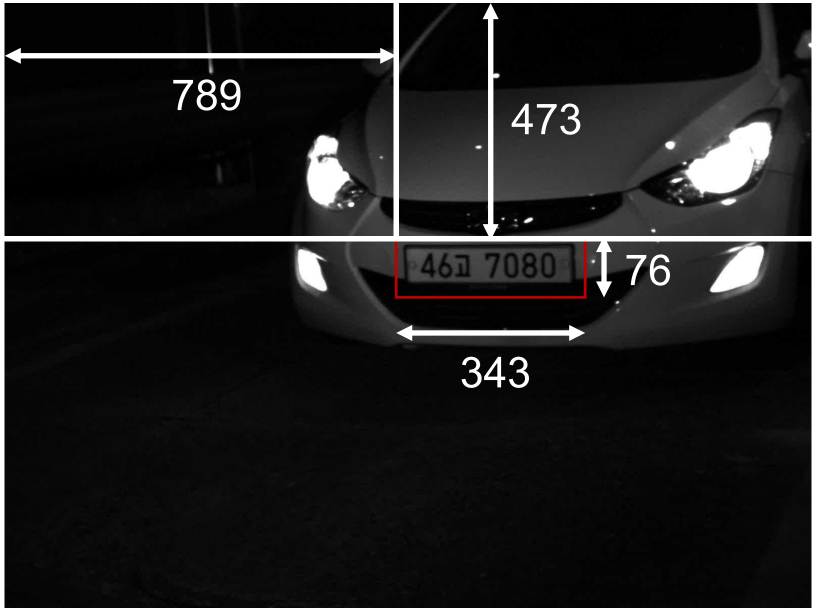


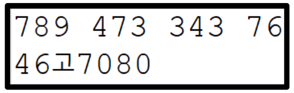






I. Parking Dataset

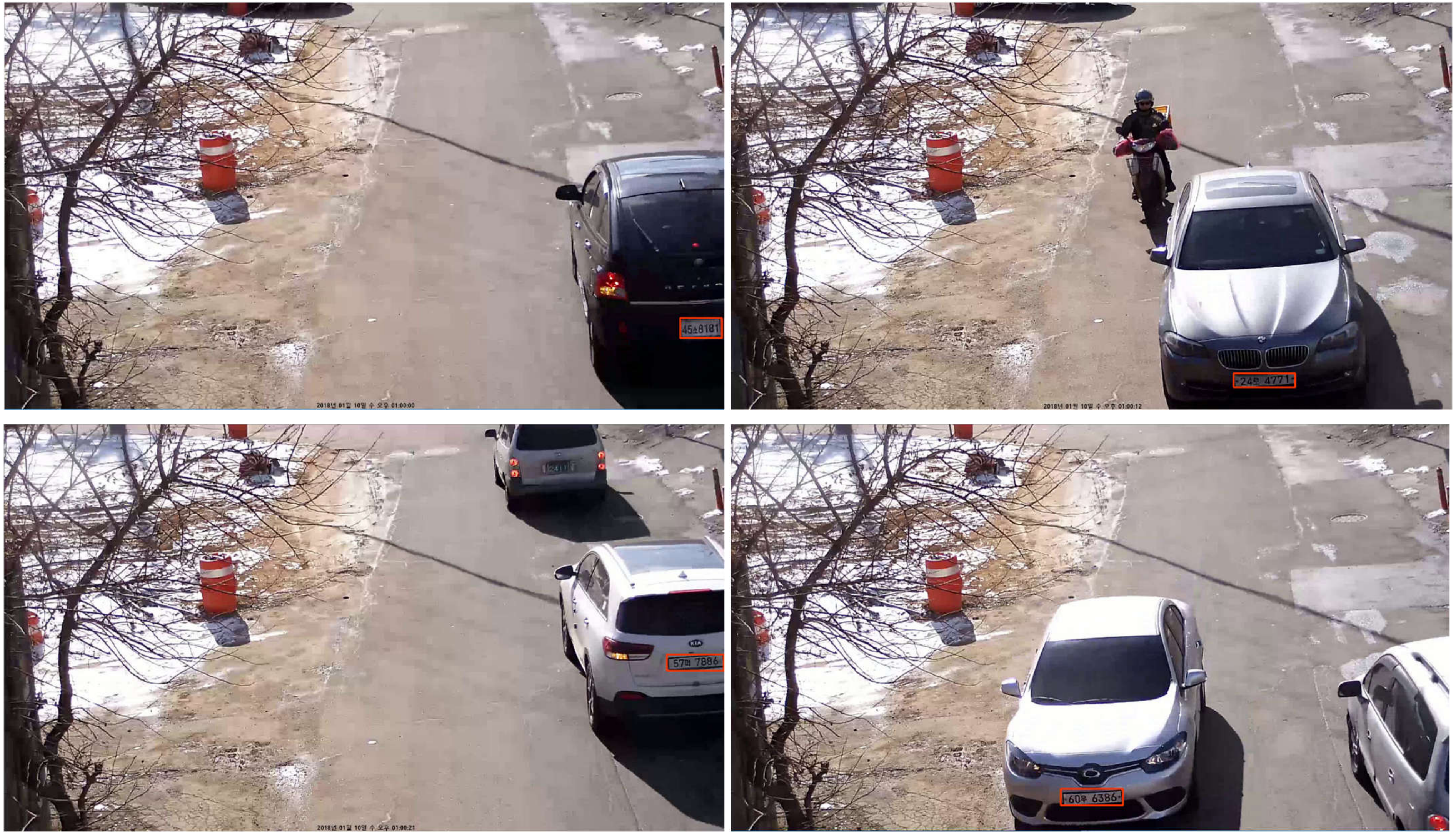




```
789 473 343 76
46고7080
```

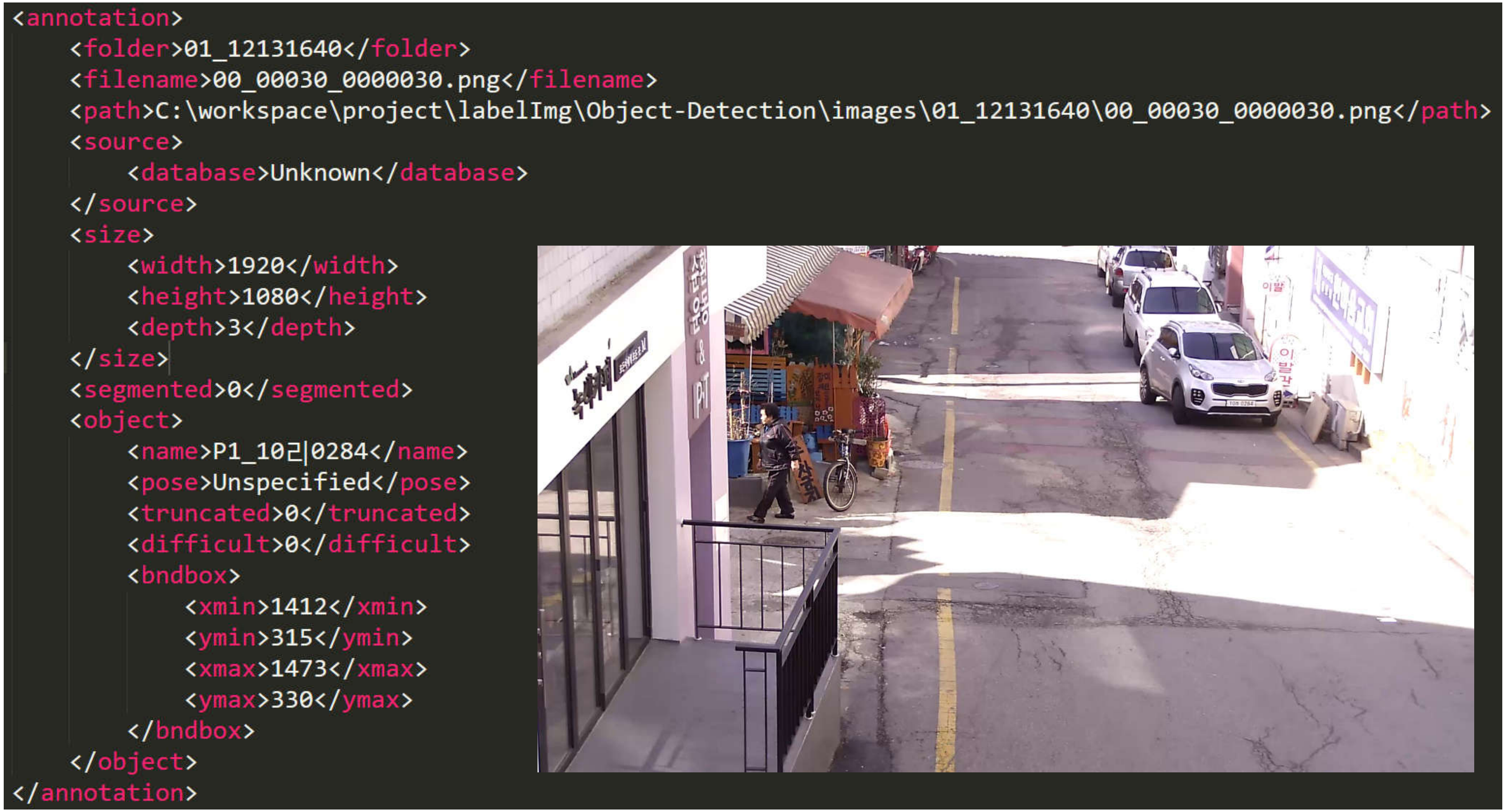
- Ground-truths for parking dataset are saved as txt file.

CCTV Data (451 Test Samples)



- Every test image includes one plate that near to the camera only!

II. CCTV Dataset



```
<annotation>
    <folder>01_12131640</folder>
    <filename>00_00030_0000030.png</filename>
    <path>C:\workspace\project\labelImg\Object-Detection\images\01_12131640\00_00030_0000030.png</path>
    <source>
        <database>Unknown</database>
    </source>
    <size>
        <width>1920</width>
        <height>1080</height>
        <depth>3</depth>
    </size>
    <segmented>0</segmented>
    <object>
        <name>P1_10리0284</name>
        <pose>Unspecified</pose>
        <truncated>0</truncated>
        <difficult>0</difficult>
        <bndbox>
            <xmin>1412</xmin>
            <ymin>315</ymin>
            <xmax>1473</xmax>
            <ymax>330</ymax>
        </bndbox>
    </object>
</annotation>
```

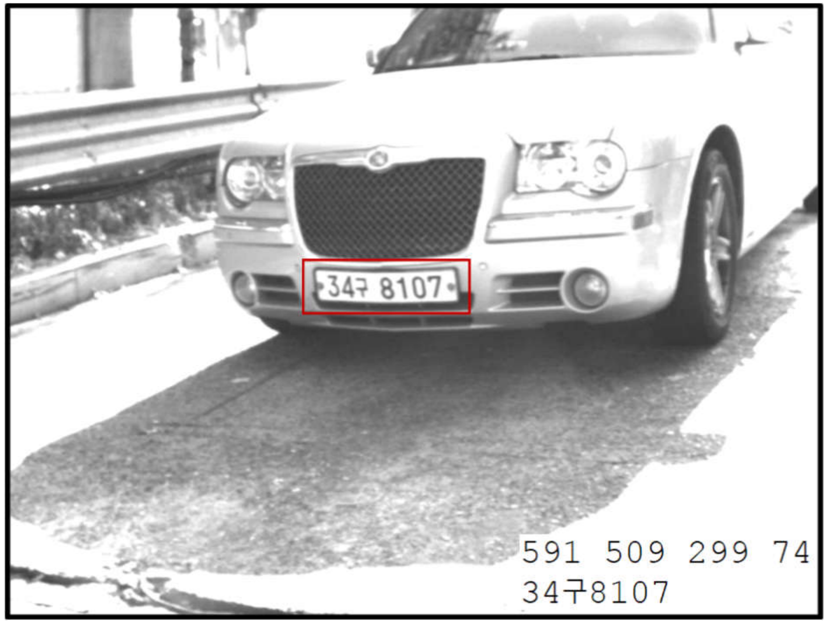
- Ground-truths for cctv dataset are saved as xml file that follows PASCAL VOC GT format.

Metrics for Final Project

Final Project

Subject:

- License Plate Recognition (LPR)



Deep Learning Platform:

TensorFlow PYTORCH K Keras

Definition of the Score

$$Score = Score_{park} + Score_{cctv} + 0.1 \times \left(100 - PT_{park}\right) + 0.1 \times \left(100 - PT_{cctv}\right)$$

$$PT = m\sec./image\,(average)$$

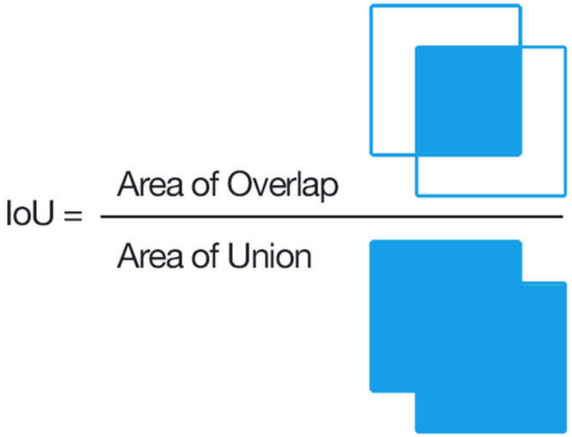
$$Score_i = Accuracy_{det} + Accuracy_{rec} \qquad (i = park\;or\;cctv)$$

$$Accuracy_{det} = \frac{1}{n}\sum_{i=0}^{n-1}\frac{\#TP_{det} - \#FP_{det}}{\#GT} \times 100\%$$

$$Accuracy_{rec} = \frac{1}{n}\sum_{i=0}^{n-1}\frac{\#TP_{rec}}{\#GT} \times 100\%$$

Range: [-Inf ~ 420]

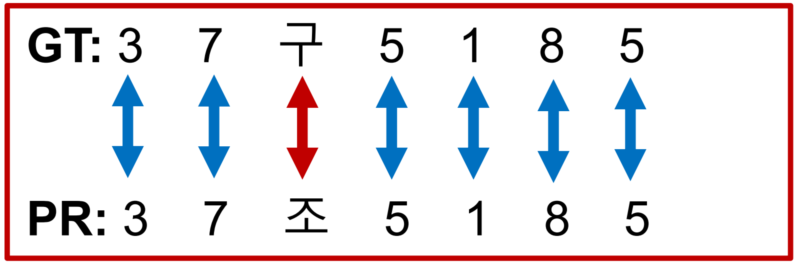
- PT: average processing time of the model (unit: msec.)
- $\#TP_{det}$: number of true positive for detection
- $\#FP_{det}$: number o false positive for detection
- $\#TP_{rec}$: number of true positive for recognition
- #GT: number of ground-truth



IoU = Area of Overlap / Area of Union

- TP_{det}: IoU >= θ, θ=0.7
- FP_{det}: IoU < θ

Wrong!



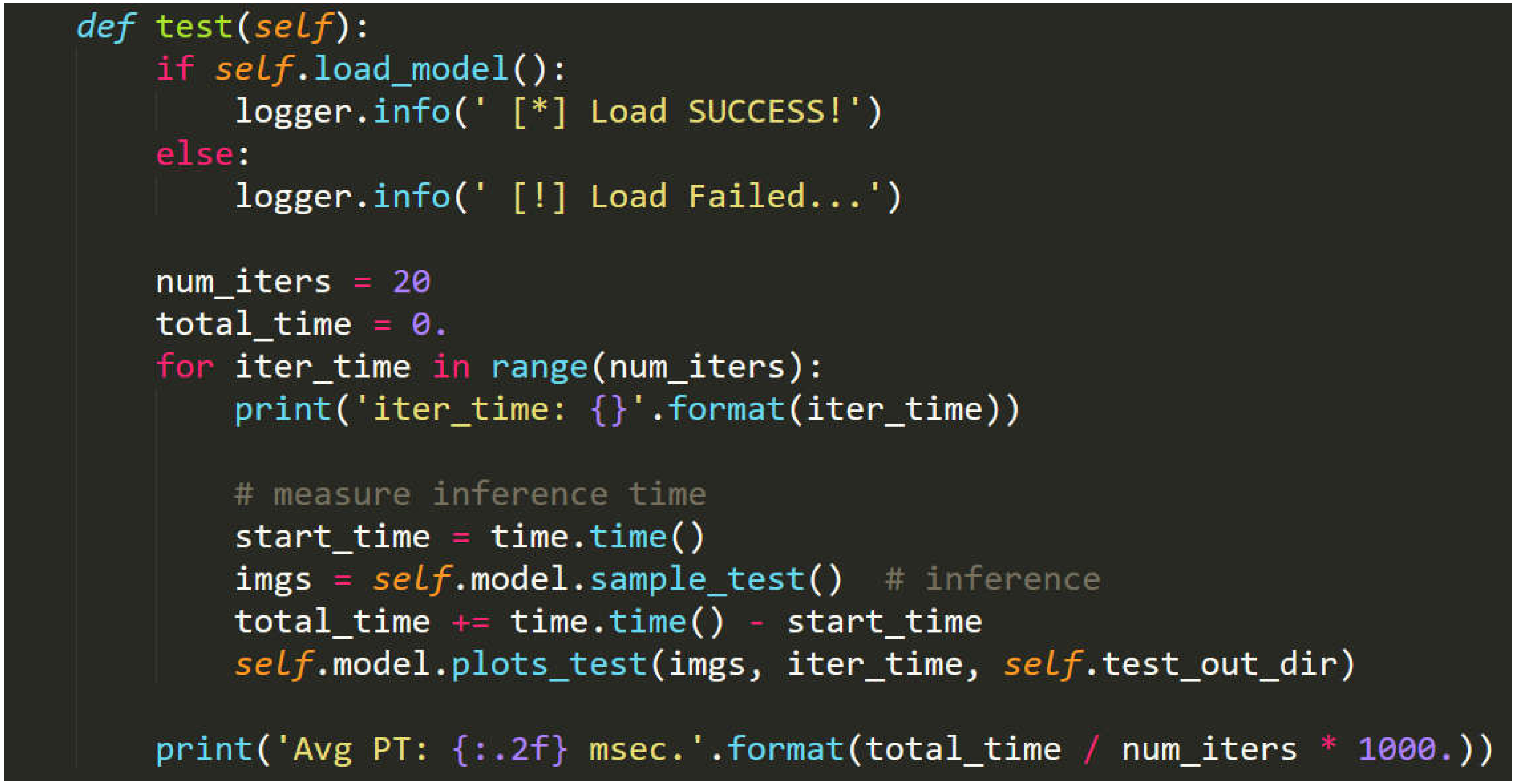
GT: 3 7 구 5 1 8 5

PR: 3 7 조 5 1 8 5

Rules

Processing Time

- Insert time check function in the for loop
- Start tic after read frame
- End toc after model forward
- But, if you have preprocess stage, preprocessing time is also included in time tic toc

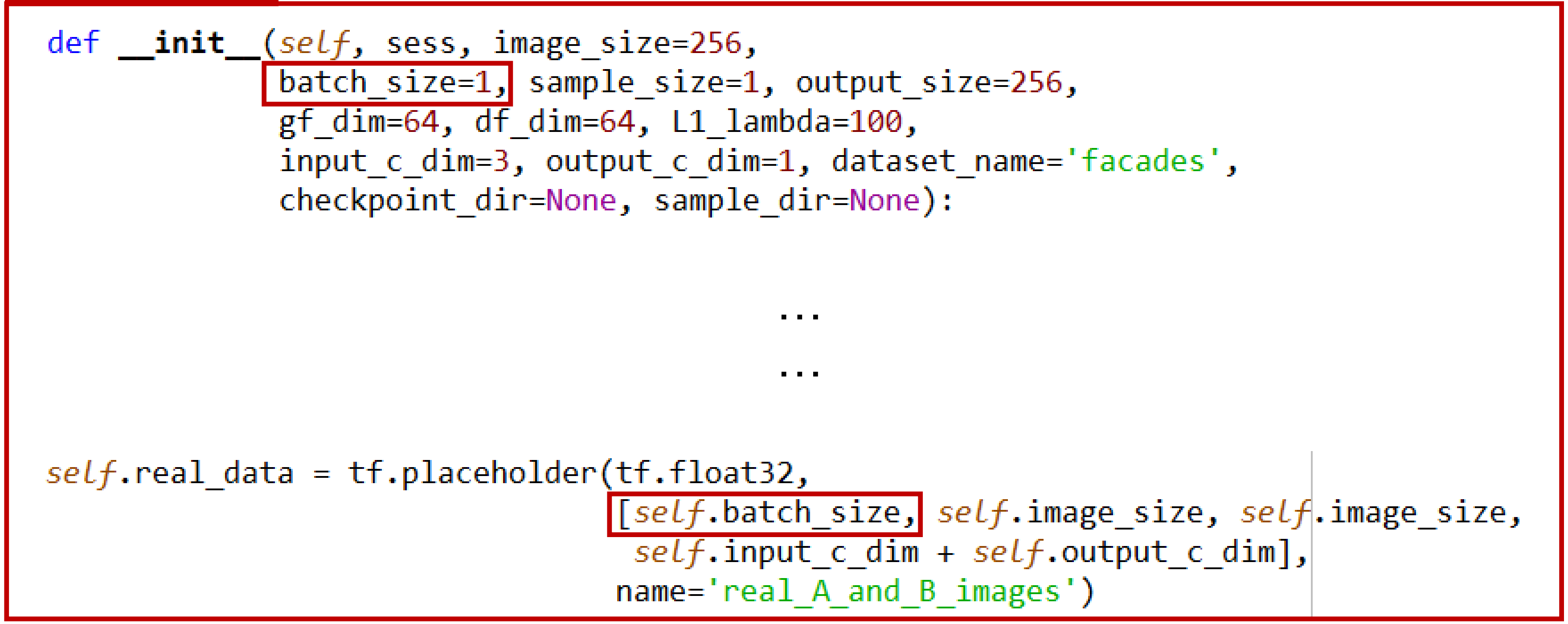


```python
def test(self):
    if self.load_model():
        logger.info(' [*] Load SUCCESS!')
    else:
        logger.info(' [!] Load Failed...')

    num_iters = 20
    total_time = 0.
    for iter_time in range(num_iters):
        print('iter_time: {}'.format(iter_time))

        # measure inference time
        start_time = time.time()
        imgs = self.model.sample_test()  # inference
        total_time += time.time() - start_time
        self.model.plots_test(imgs, iter_time, self.test_out_dir)

    print('Avg PT: {:.2f} msec.'.format(total_time / num_iters * 1000.))
```

Batch_Size in Test Mode

- Batch_size can be bigger than 1 in training mode
- **But in test mode batch_size have to 1!**

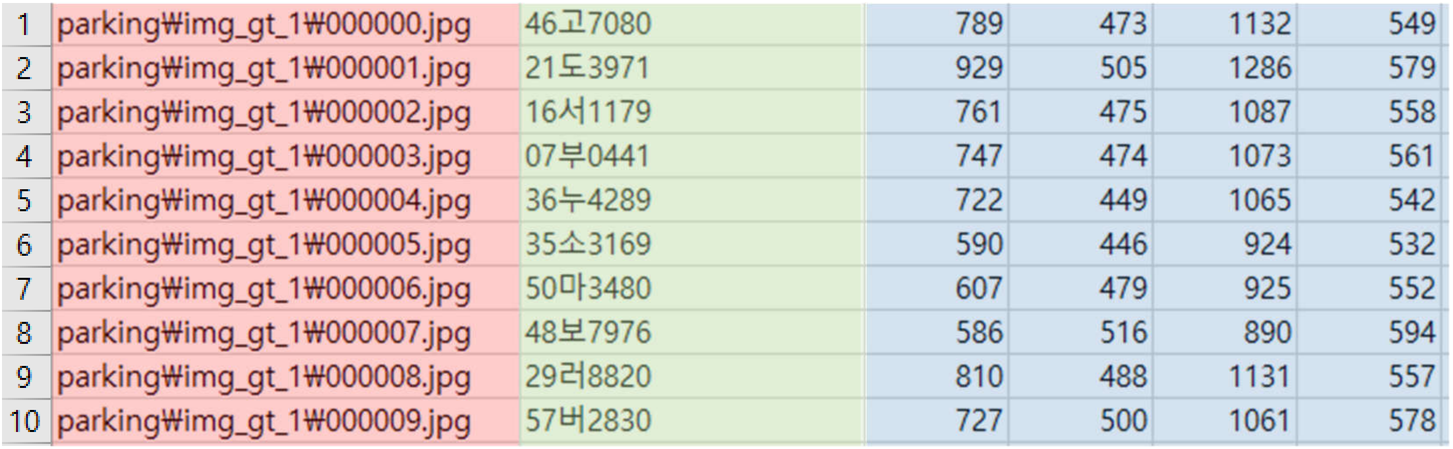
Example:



```
def __init__(self, sess, image_size=256,
             batch_size=1, sample_size=1, output_size=256,
             gf_dim=64, df_dim=64, L1_lambda=100,
             input_c_dim=3, output_c_dim=1, dataset_name='facades',
             checkpoint_dir=None, sample_dir=None):

                              ...

                              ...

self.real_data = tf.placeholder(tf.float32,
                                [self.batch_size, self.image_size, self.image_size,
                                 self.input_c_dim + self.output_c_dim],
                                name='real_A_and_B_images')
```

Write Prediction Results in Log File

- Prediction results should be written in “dataset_name.csv” file **(cctv.csv or parking.csv)**
- Please refer to the write_csv.py function
  - ➢ https://github.com/ChengBinJin/License_plate_recognition
  - ➢ There are read_xml.py, write_csv.py, and eval.py function to help you



1	parking₩img_gt_1₩000000.jpg	46고7080	789	473	1132	549
2	parking₩img_gt_1₩000001.jpg	21도3971	929	505	1286	579
3	parking₩img_gt_1₩000002.jpg	16서1179	761	475	1087	558
4	parking₩img_gt_1₩000003.jpg	07부0441	747	474	1073	561
5	parking₩img_gt_1₩000004.jpg	36누4289	722	449	1065	542
6	parking₩img_gt_1₩000005.jpg	35소3169	590	446	924	532
7	parking₩img_gt_1₩000006.jpg	50마3480	607	479	925	552
8	parking₩img_gt_1₩000007.jpg	48보7976	586	516	890	594
9	parking₩img_gt_1₩000008.jpg	29러8820	810	488	1131	557
10	parking₩img_gt_1₩000009.jpg	57버2830	727	500	1061	578

img_file **recognition** **bounding box**
(x1, y1, x2, y2)

Test examples:
https://www.dropbox.com/s/s6ns7dgmpcu5sjb/test_data.zip?dl=0

Specification of the Evaluation Server

- **Equipment:** Use same server in the CVLab. (Room 525, Hi-Tech Building)

(Use same server to check **processing time** and **run evaluation function** to get accuracy on a test set)

- **OS:** Ubuntu 16.04
- **CPU:** Intel(R) Core™ i9-7900X CPU @3.30GHz x 20
- **RAM:** 64GB
- **GPU:** GeForce GTX 1080Ti x 4 (just use one to evaluate)

- Ubuntu system and Windows system are different. Highly recommend to test your code on Ubuntu first.

Teams

No.	Team Name	Members	Num. of Students	Deep learning library
1	ICVL1	배규호, 수랴	2	
2	Team 1	송광호, 이윤선 이화선	3	
3	Team 2	윤정언, 이반	2	
4	Team 3	이명오, 알만	2	
5	AI Lab	아자맛, 자와힐, 아지즈	3	
6	SBS	쇼크루, 뷔뇨벡, 사이드라술콘	3	
7	Geeks	네마트전, 코필전	2	

Competition

- **Time: December 10-11 (Mon. and Tue.)**
- 09:30-11:30, 14:00-17:00, and 20:00-24:00
- Each team has the limited **1 hour (very strict)** to finish all of the process including install necessary libraries and test.
- Each team can try **two times for one dataset** and select the most good one to record the score.

Thank you for your attention!