Selected Topics in Visual Recognition using Deep Learning
Homework 1 grading

TA: 楊証琨, Jimmy

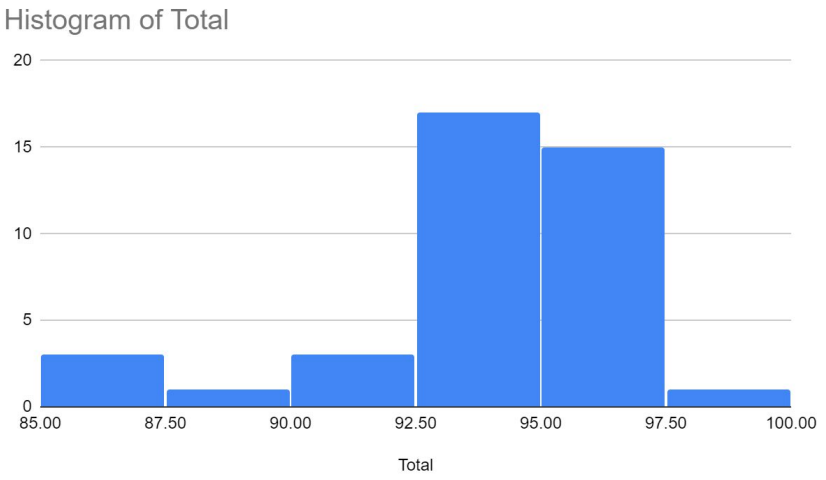
Ph.D. student at National Taiwan Universitiy

d08922002@ntu.edu.tw

HW1 scores

- We announce the score of HW1 here
- You should have received a e-mail at 11/6 which provide your code number

C	D	E	F	G	H	I
Code number	Accuracy	Model performance	Model perform	Reports	Code	Total
lZfwaKMc	0.89903	0.916998886	64.189922	20	10	94.189922
oxLMXben	0.9173	0.923332425	64.63326978	20	10	94.63326978
AYoKVkaj	0.9048	0.918999133	64.32993933	20	10	94.32993933
ADURbQxB	0.925	0.926001733	64.82012133	18	10	92.82012133
JyLqsTpx	0.9548	0.936332302	65.54326111	20	10	95.54326111
lSZmWhyK	0.94807	0.933999257	65.379948	20	10	95.379948
lRmPSZUH	0.92403	0.92566547	64.79658289	20	10	94.79658289
DuEQZHhG	0.91826	0.923665222	64.65656556	20	10	94.65656556
iFlzvfoA	0.95384	0.935999505	65.51996533	20	10	95.51996533
GtLrBZaS	0.95673	0.937001362	65.59009533	18	10	93.59009533
wpcPmJUB	0.97211	1	70	20	10	100



Good Reports

- Document your work (**in PDF**)
  - GitHub/ GitLab link of your code
  - Brief introduction
  - Methodology (Data pre-process, Model architecture, Hyperparameters,...)
  - **Findings or Summary**

Anti-aliasing

From Kayo Yin 0845051

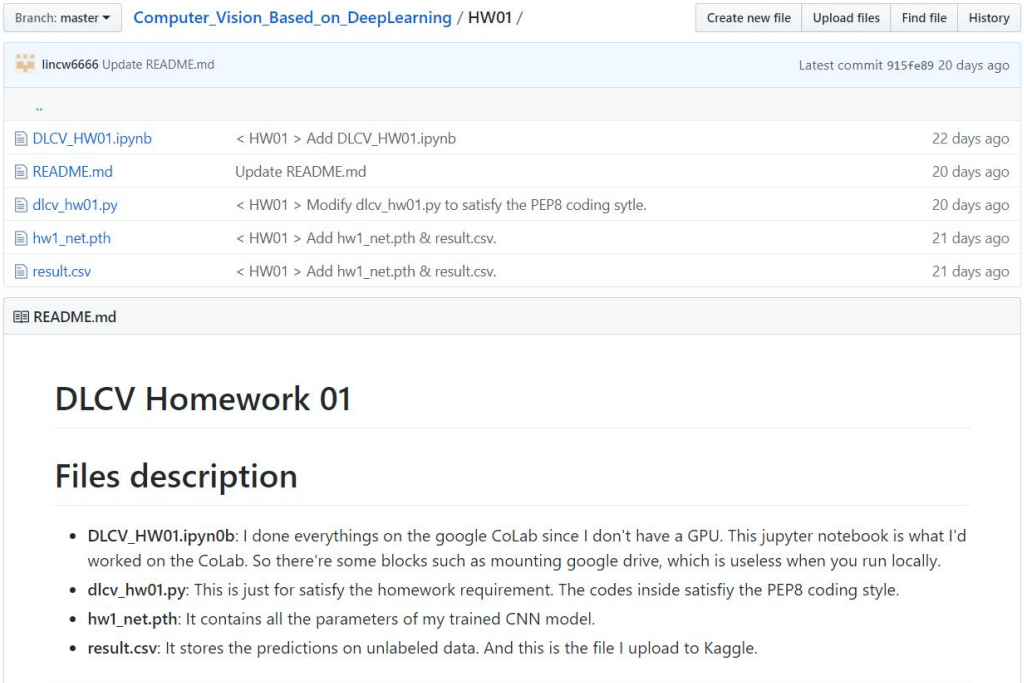
Most modern convolutional networks, such as ResNet18, are not shift-invariant. The network outputs can change drastically with small shifts or translations to the input. This is because the striding operation in the convolutional network ignores the Nyquist sampling theorem and aliases, which breaks shift equivariance.

I decided to apply an anti-aliasing method proposed in the recent April 2019 paper: "Making Convolutional Networks Shift-Invariant Again". This is done by simply adding a "BlurPool" layer, that is a blurring filter and a subsampling layer, after the convolution layers of

Code readability

- README.md is important! Because you will definitely forget what you have done after longtime

From 林正偉 0856030



Branch: master ▾ Computer_Vision_Based_on_DeepLearning / HW01 / Create new file | Upload files | Find file | History

lincw6666 Update README.md — Latest commit 915fe89 20 days ago

..

DLCV_HW01.ipynb	< HW01 > Add DLCV_HW01.ipynb	22 days ago
README.md	Update README.md	20 days ago
dlcv_hw01.py	< HW01 > Modify dlcv_hw01.py to satisfy the PEP8 coding sytle.	20 days ago
hw1_net.pth	< HW01 > Add hw1_net.pth & result.csv.	21 days ago
result.csv	< HW01 > Add hw1_net.pth & result.csv.	21 days ago

README.md

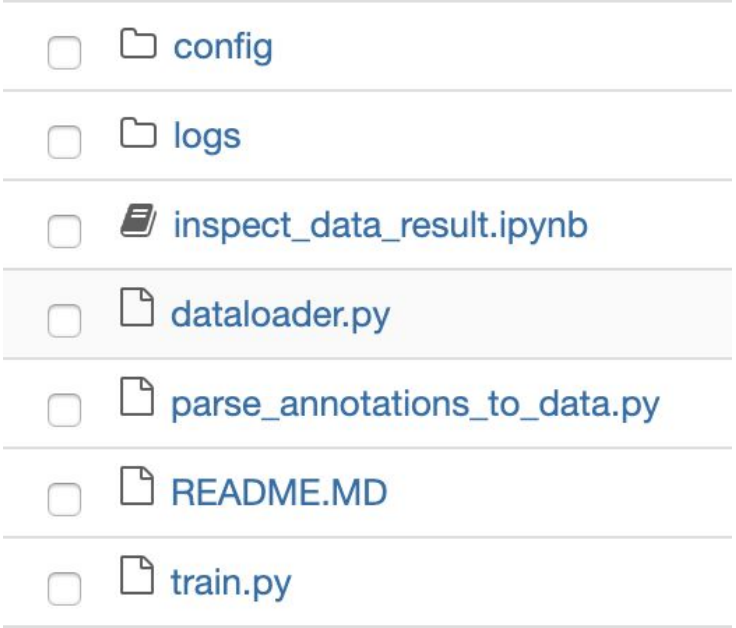
DLCV Homework 01

Files description

- **DLCV_HW01.ipyn0b**: I done everythings on the google CoLab since I don't have a GPU. This jupyter notebook is what I'd worked on the CoLab. So there're some blocks such as mounting google drive, which is useless when you run locally.
- **dlcv_hw01.py**: This is just for satisfy the homework requirement. The codes inside satisfiy the PEP8 coding style.
- **hw1_net.pth**: It contains all the parameters of my trained CNN model.
- **result.csv**: It stores the predictions on unlabeled data. And this is the file I upload to Kaggle.

Code readability: suggestion

- It’s hard to reproduce results without proper config management
- Use configuration tools such as yacs_config or config.py to track your model config
- Develop your model code by jupyter notebook is good. But train your model with python scripts is recommended



Selected Topics in Visual Recognition using Deep Learning
Homework 2 grading

TA: 楊証琨, Jimmy

Ph.D. student at National Taiwan Universitiy

d08922002@ntu.edu.tw

Grading policy: voting criteria

1. Completeness of human faces

Failure example

Good example

2. Variation of gerenerated faces

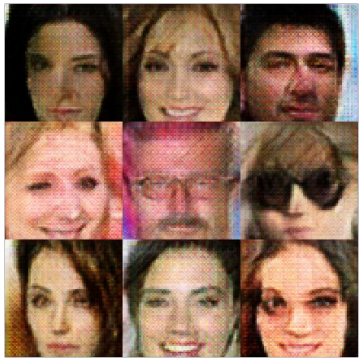
Failure example

Good example

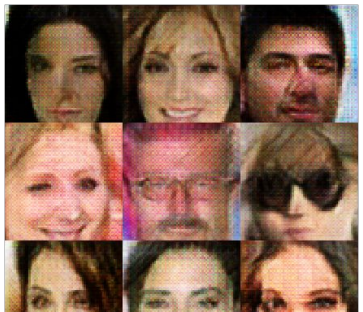
Voting

- We will random select two images from your results and compare each to the baseline

VS.



VS.



Votes

- Bring a pen next week
- Vote for the better one

index	Left	Right
1	v	
2		v
….		v
82	v	

Selected Topics in Visual Recognition using Deep Learning
Homework 3 announcement

TA: 楊証琨, Jimmy

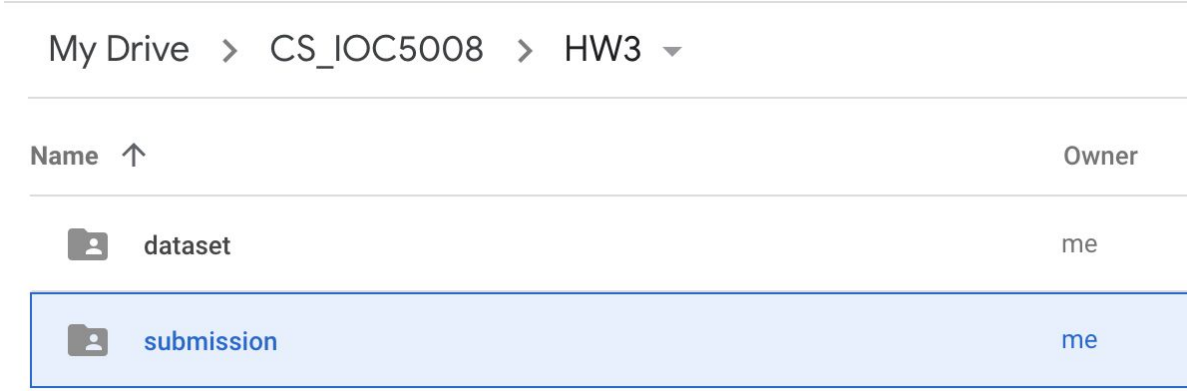
Ph.D. student at National Taiwan Universitiy

d08922002@ntu.edu.tw

Homework 3: Digits detection

- **Deadline: 11/28, Thr at 23:59**
  1. Upload your **report.pdf** and **submission file** in this Google drive

  2. Test your model inference speed by the Google colab



HW3 Introduction: Street View House Numbers

- SVHN dataset contains 33,402 trianing images, 13,068 test images
- Train a not only **accurate** but **fast** digit detector!

HW3 Get the dataset

- Download the dataset from this Google Drive
- The annotations are save in **.h5** file. Here are Python code to parse the annotation file

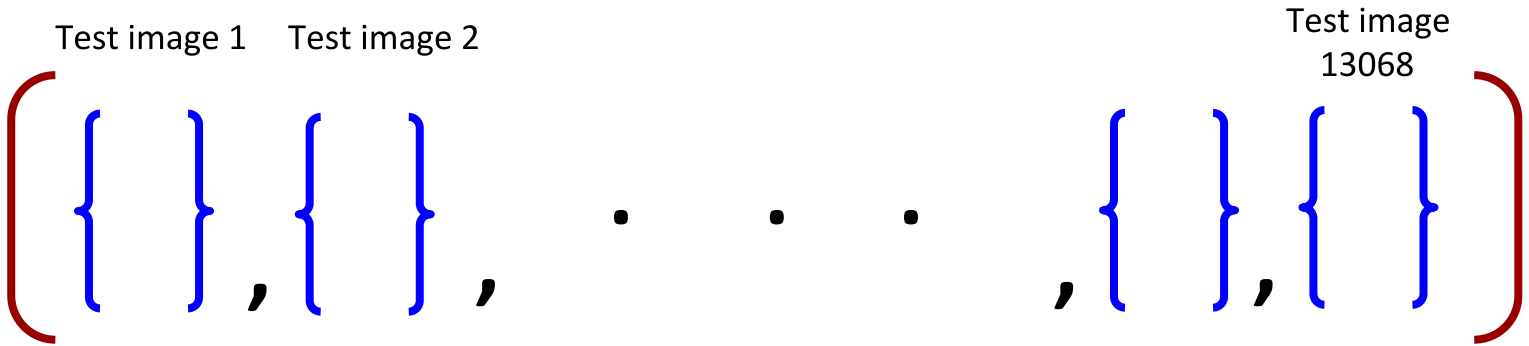
```
def get_name(index, hdf5_data):
    name = hdf5_data['/digitStruct/name']
    return ''.join([chr(v[0]) for v in hdf5_data[name[index][0]].value])

def get_bbox(index, hdf5_data):
    attrs = {}
    item = hdf5_data['digitStruct']['bbox'][index].item()
    for key in ['label', 'left', 'top', 'width', 'height']:
        attr = hdf5_data[item][key]
        values = [hdf5_data[attr.value[i].item()].value[0][0]
                  for i in range(len(attr))] if len(attr) > 1 else [attr.value[0][0]]
        attrs[key] = values
    return attrs
```

Upload your submission.json file here

- Free version Kaggle doesn't provide detection metrics :(
- Upload you submission file into the Google Drive, I will inference it and return the performance on your filename for every midnight
- filename should be STUDENTID.json

Submission.json file format

- List of dictionaries, len(list) = number of test images. Order matters!
- Each dictionary contains three keys
  - **“bbox”**: list of bounding boxes in (y1, x1, y2, x2). y1x1=topleft
  - **“score”**: list of probability for the class
  - **“label”**: list of label



Submission.json file format

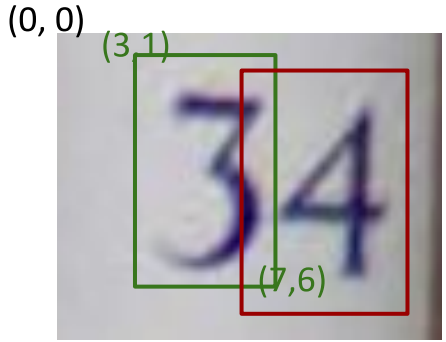
- E.g., your model output two boxes on image, the dictionary will be

y1, x1, y2, x2

dict = {“bbox”: [(1, 3, 6, 7), (4, 5, 8, 12)] ,

“score”:[0.87, 0.61],

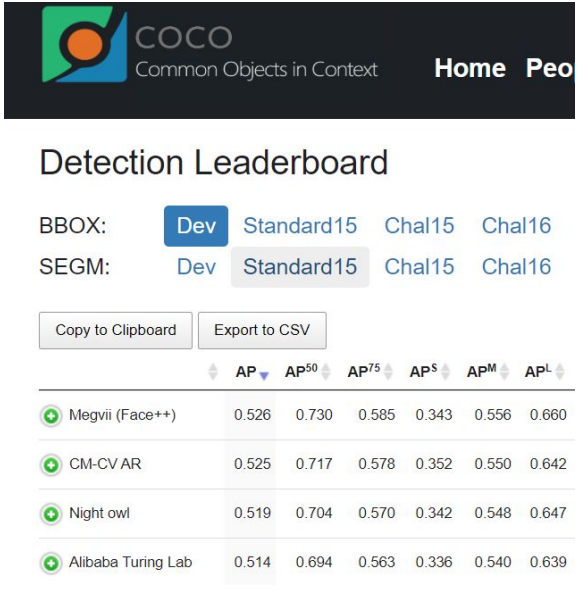
“label”: [3, 5]}



[{ } , { } , . . . , { } , { }]

Evaluation metrics: mean Average Precision

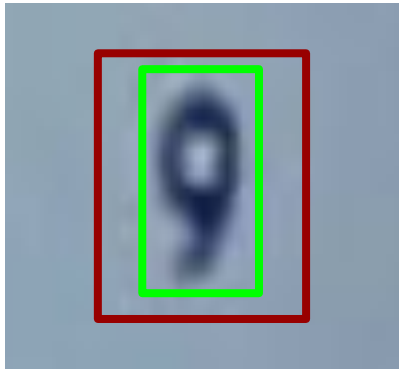
- Most common metric for object detection
- Measure the average precision on different threshold and also the IOU between GT and prediction



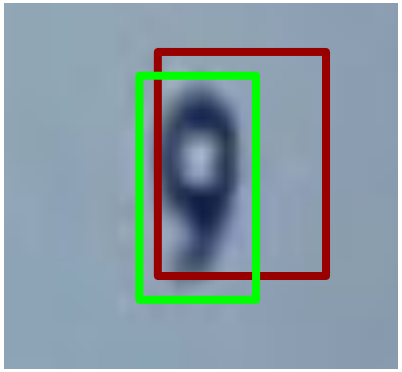
	AP	time
[A] YOLOv2† [27]	21.6	25
[B] SSD321 [22]	28.0	61
[C] DSSD321 [9]	28.0	85
[D] R-FCN‡ [3]	29.9	85
[E] SSD513 [22]	31.2	125
[F] DSSD513 [9]	33.2	156
[G] FPN FRCN [20]	36.2	172
RetinaNet-50-500	32.5	73
RetinaNet-101-500	34.4	90
RetinaNet-101-800	37.8	198

Evaluation metrics: mean Average Precision

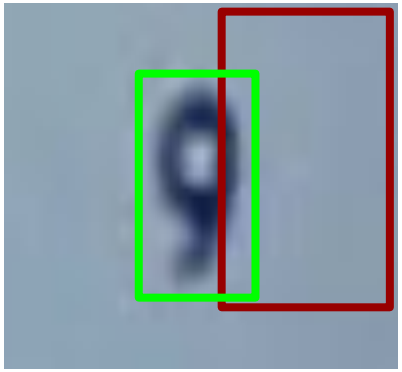
- To calcaute precision, we need to first define what's **True positive**
- TP depends on the IOU threshold, COCO use range(0.5, 0.95, 0.05) then average the results



TP?



TP?



TP?

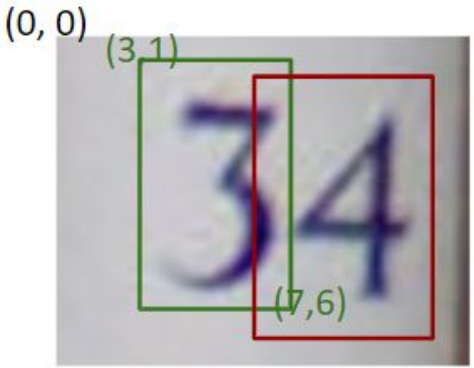
Evaluation metrics: mean Average Precision

- Set IOU threshold=0.5
- We got 1 TP, 1 FP, precision=0.5, recall=0.5
- But if we change the probability threshold to 0.7, the red box will disappear, then we got 1 TP, precision=1, recall=0.5
- We also miss one GT, so when recall=1, precision=0.5

- E.g., your model output two boxes on image, the dictionary will be

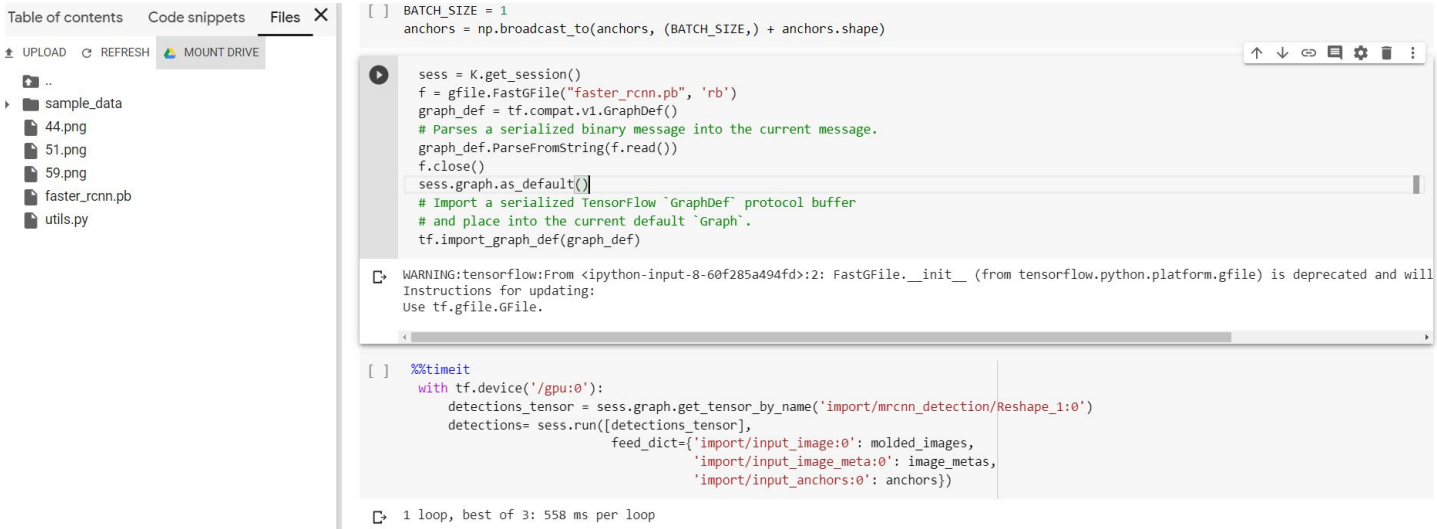
y1,x1, y2, x2

```
dict = {"bbox": [(1, 3, 6, 7), (4, 5, 8, 12)] ,
        "label": [3, 5],
        "score":[0.87, 0.61]}
```



Model speed benchmark by Google Colab

- To evaluate your model by same GPU, you need to transfer the code and weights of your model into Colab. Then run inference to test the speed
- **Please include this part in your reports**



Grading policy: Model performance (70 points)

- 40 points for the accuracy ranking
- 30 points for the speed benchmark ranking
- Pass the each baseline will get 80% of that points

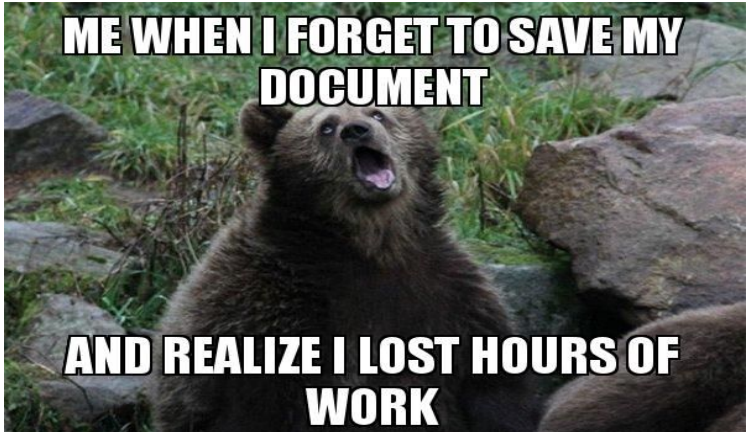
- Accuracy baseline: 0.36898
- Speed baseline: 558 ms per image

```
[ ]  %%timeit
      with tf.device('/gpu:0'):
          detections_tensor = sess.graph.get_tensor_by_name('import/mrcnn_detection/Reshape_1:0')
          detections= sess.run([detections_tensor],
                               feed_dict={'import/input_image:0': molded_images,
                                          'import/input_image_meta:0': image_metas,
                                          'import/input_anchors:0': anchors})
```

```
1 loop, best of 3: 558 ms per loop
```

Grading policy: Reports (20 points)

- Document your work (**in PDF**)
  - GitHub/ GitLab link of your code
  - **reference if you used code from GitHub**
  - **Speed benchmark**
  - Brief introduction
  - Methodology (Data pre-process, Model architecture, Hyperparameters,...)
  - Findings or Summary



Grading policy: Code readability (10 points)

- Write beautiful Python code with [PEP8 guidelines] for readability. Base requirement: use whitespace correctly!

Python

```
# Recommended
def function(default_parameter=5):
    # ...


# Not recommended
def function(default_parameter = 5):
    # ...
```

Python

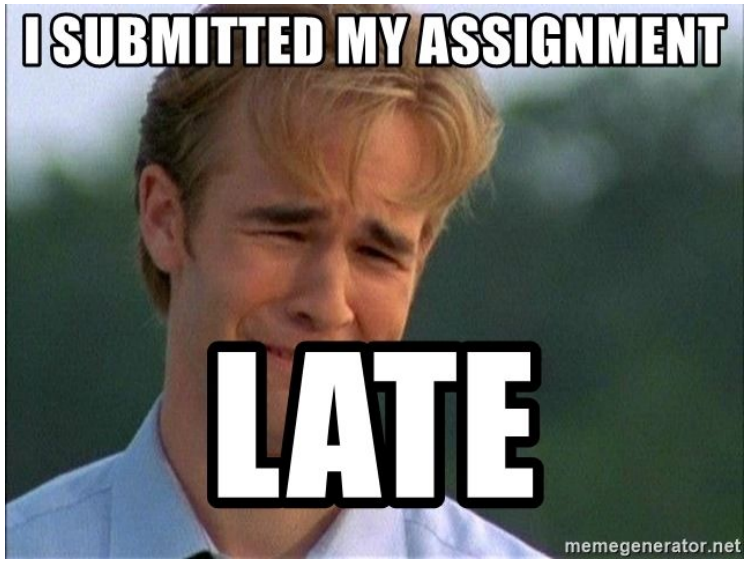
```
# Recommended
my_list = [1, 2, 3]

# Not recommended
my_list = [ 1, 2, 3, ]
```

Python

```
x = 5
y = 6

# Recommended
print(x, y)

# Not recommended
print(x , y)
```

Late Policy

- We will deduct a late penalty of 20 points per additional late day
- For example, If you get 90% of HW but delay for two days, your will get only 90 points- (20 points x 2) = 50 points!



Keywords

- Beat the baseline
  - yolo, SSD, Retina-Net, Faster RCNN

- Rank Top 3!
  - Read some paper from CVPR2019, ICCV2019 and try to implement it!

FAQ

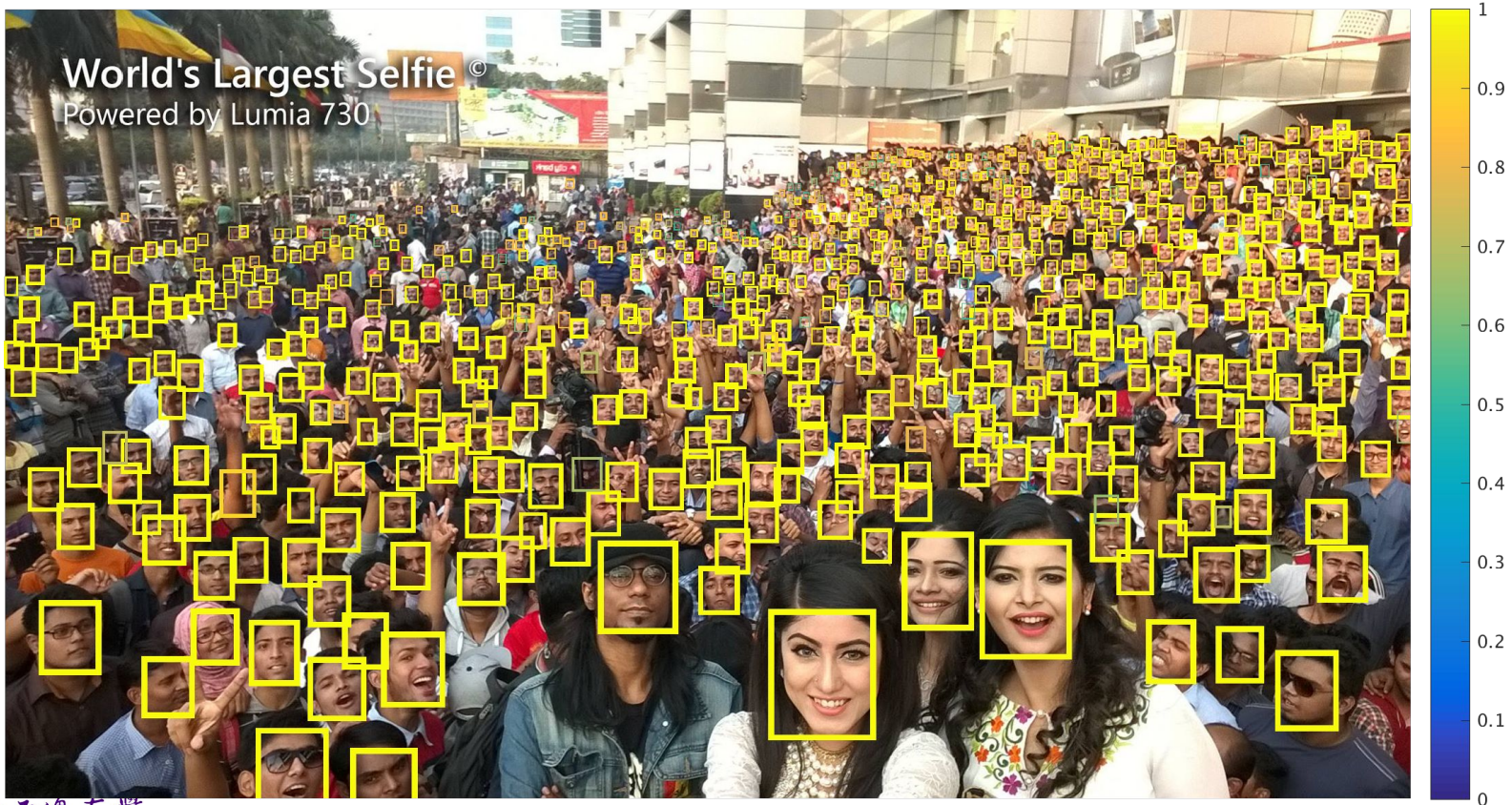
- Can I use any code/tools/Library from GitHub or other resources?
  - Yes! We encourage you to learn how to apply existing tools on your own task, such as Keras-Retinanet, Pytorch-mmdetection, TF-object-detection-API

    But DO NOT copy code from your classmate!
- Why my testing results are so bad?
  - If you have done any image translation (resize, padding), you will need to transfer the coordinates into original image dimension

Notice

- Check your email regularly, we will mail you if there are any updates or problems of the homework
- If you have any questions or comments for the homework, please mail me and cc Prof. Lin
  - Prof. Lin: **lin@cs.nctu.edu.tw**
  - Jimmy: **d08922002@ntu.edu.tw**

Have fun!


World's Largest Selfie ©
Powered by Lumia 730
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0