Report

YOLOv5 모델 학습

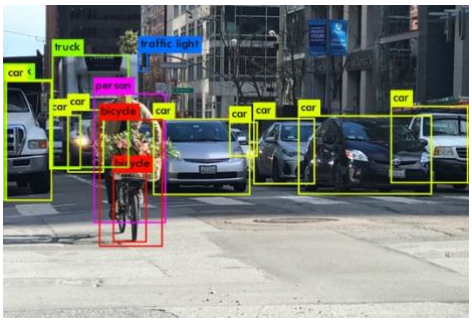
Spring, 2022

휴스타 ICT 설계팀프로젝트1 (ITEC0426001)

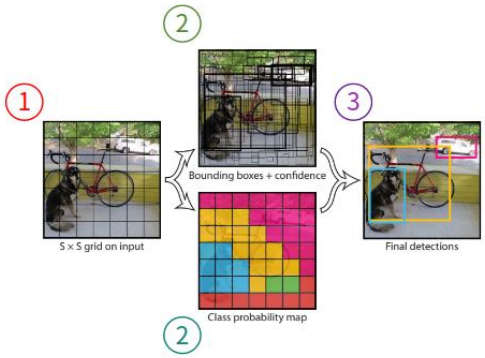
Student Name/Number: 이새봄(Lee SaeBom)/2019116247

1. YOLO v5?

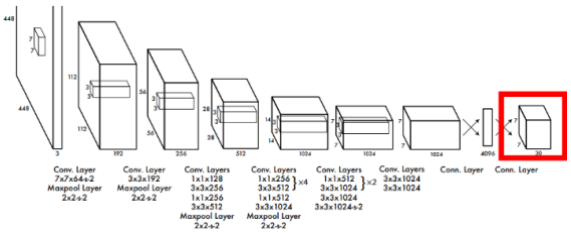
A. YOLO?



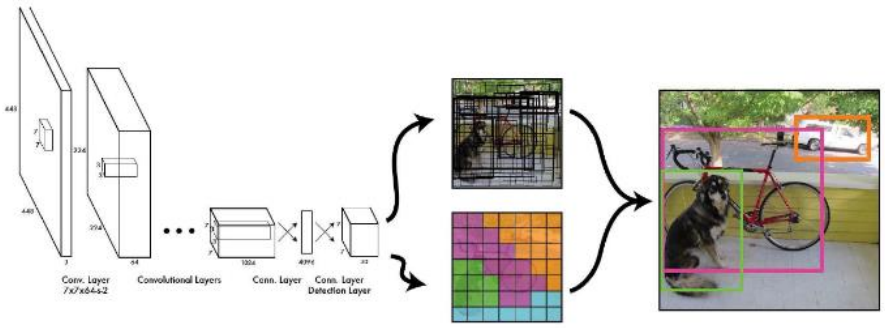
- Real time object detection 에 사용되는 알고리즘
- Josept Redmon이 2015년에 yolov1 논문을 발표하고 공개



- 각 이미지를 S x S 개의 그리드로 분할하고, 각 그리드의 신뢰도를 계산
- 처음에는 객체와 동떨어진 그리드가 설정되지만, 신뢰도를 계산하여 위치를 조정함으로써 가장 높은 객체인식 정확성을 가지는 그리드를 얻음
- 신뢰도는 주변의 그리드를 합쳐 높이고, 이후 임계값을 설정해 불필요한 부분을 제거



- Network Design
  - 총 24 개의 convolution layer 와 2 개의 fully connected layer 로 구성



Whole Network Pipeline

- 전체 pipeline
- 장점
  - 간단한 처리 과정으로 속도가 매우 빠르며 기존의 실시간 Object Detection 모델들과 비교하면 2 배 정도 높은 mAP 를 보임
  - 이미지 전체를 한 번에 바라보는 방식을 이용하므로 class 에 대한 맥락적 이해도가 다른 모델에 비해 높아 낮은 False-Positive 를 보임
  - 일반화된 Object 학습이 가능하여 자연 이미지로 학습하고 이를 그림과 같은 곳에 테스트 해도 다른 모델에 비해 훨씬 높은 성능을 보여줌

B. V5

i. YOLOv3

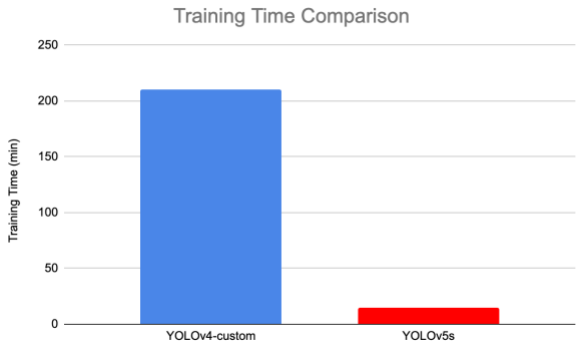
1. 2018년 4월 출시, 백본 아키텍쳐 Darknet 53을 기반으로 만들어져 있으며, YOLO를 만든 Josept Redmon이 발표했다.
2. 하지만 Josept Redmon 은 YOLOv3 을 마지막으로 더이상 개발하지 않는다고 밝혔다

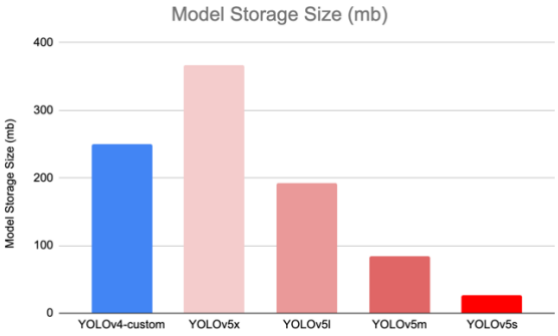
ii. YOLOv4

1. 2020년 4월 출시했다. v3에 비해 AP, FPS가 각각 10%, 12% 증가했다.
2. v3와 다른 개발자인 AlexeyBochkousky가 발표했다.
3. v3에서 다양한 딥러닝 기법(WRC, CSP ...) 등을 사용해 성능을 향상시켰다.
4. CSPNet 기반의 backbone(CSPDarkNet53)을 설계하여 사용했다.

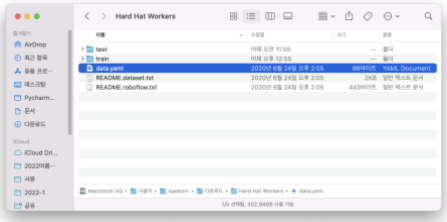
iii. YOLOv5

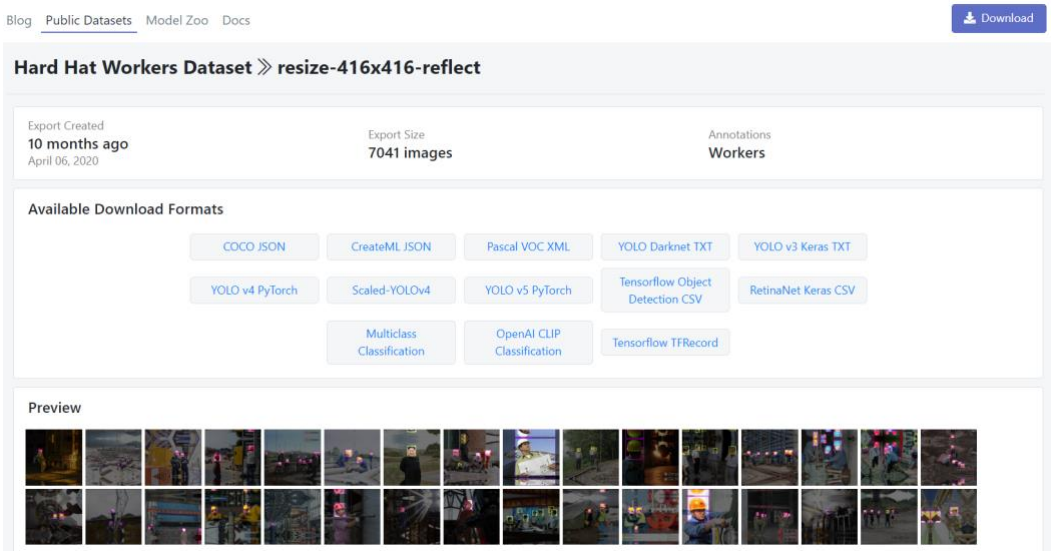
1. 2020년 6월 출시했다. v4에 비해 낮은 용량과 빠른 속도를 가지고 있다
2. YOLOv4와 같은 CSPNet 기반의 backbone을 설계하여 사용했다.
3. YOLOv3를 PyTorch로 implementation한 GlennJocher가 발표했다.
4. Darknet이 아닌 PyTorch 구현이기 때문에, 이전 버전들과 다르다고 할 수 있다.
5. 처음 출시될 때 논문이 함께 출시되지 않았고, 이름을 YOLOv5로 하는것에 대한 논란이 있다.





2. Dataset





-> 안전모 detection dataset, 다음과 같이 bbox 정보가 라벨링됨.

3. Colab 실습

- dataset 경로 -> google drive에 업로드해서 가져옴

```
[13] %cat /content/drive/MyDrive/HardHatWorkers/data.yaml

     names:
     - head
     - helmet
     - person
     nc: 3
     train: /content/drive/MyDrive/HardHatWorkers/train.txt
     val: /content/drive/MyDrive/HardHatWorkers/val.txt
```

```
[14] %cd /
     from glob import glob
     img_list = glob('/content/drive/MyDrive/HardHatWorkers/test/images/*.jpg')
     print(len(img_list))

     /
     1766
```

```
[15] from sklearn.model_selection import train_test_split
     train_img_list, val_img_list = train_test_split(img_list,test_size=0.2,random_state=2000)
     print(len(train_img_list),len(val_img_list))

     1412 354
```

```
     with open('/content/drive/MyDrive/HardHatWorkers/train.txt','w') as f:
       f.write('\n'.join(train_img_list)+'\n')

     with open('/content/drive/MyDrive/HardHatWorkers/val.txt', 'w') as f:
       f.write('\n'.join(val_img_list)+'\n')
```

⇨ Image 개수 확인 및 train set, test set 분리

```
!pip install pyyaml==5.4.1

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Requirement already satisfied: pyyaml==5.4.1 in /usr/local/lib/python3.7/dist-packages (5.4.1)
```

```
import yaml

with open('/content/drive/MyDrive/HardHatWorkers/data.yaml','r') as f:
  data = yaml.load(f)

print(data)

data['train'] = '/content/drive/MyDrive/HardHatWorkers/train.txt'
data['val']= '/content/drive/MyDrive/HardHatWorkers/val.txt'

with open('/content/drive/MyDrive/HardHatWorkers/data.yaml','w') as f:
  yaml.dump(data, f)

print(data)

{'names': ['head', 'helmet', 'person'], 'nc': 3, 'train': '/content/drive/MyDrive/HardHatWorkers/train.txt', 'val': '/content/drive/MyDrive/HardHatWo
{'names': ['head', 'helmet', 'person'], 'nc': 3, 'train': '/content/drive/MyDrive/HardHatWorkers/train.txt', 'val': '/content/drive/MyDrive/HardHatWo
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: YAMLLoadWarning: calling yaml.load() without Loader=... is deprecated, as the default
  after removing the cwd from sys.path.
```

⇨ yaml.load가 오류나서 pyyaml==5.4.1로버전 변경

```
%cd /content/yolov5/
!python train.py --img 416 --batch 16 --epochs 50 --data /content/drive/MyDrive/HardHatWorkers/data.yaml --cfg ./models/yolov5s.yaml --weights yolov5s.pt

/content/yolov5
train: weights=yolov5s.pt, cfg=./models/yolov5s.yaml, data=/content/drive/MyDrive/HardHatWorkers/data.yaml, hyp=data/hyps/hyp.scratch-low.yaml, epochs=50
github: up to date with https://github.com/ultralytics/yolov5 ✅
YOLOv5 🚀 v6.1-246-g2dd3db0 Python-3.7.13 torch-1.11.0+cu113 CUDA:0 (Tesla T4, 15110MiB)

hyperparameters: lr0=0.01, lrf=0.01, momentum=0.937, weight_decay=0.0005, warmup_epochs=3.0, warmup_momentum=0.8, warmup_bias_lr=0.1, box=0.05, cls=0.5,
Weights & Biases: run 'pip install wandb' to automatically track and visualize YOLOv5 🚀 runs (RECOMMENDED)
TensorBoard: Start with 'tensorboard --logdir runs/train', view at http://localhost:6006/
Downloading https://ultralytics.com/assets/Arial.ttf to /root/.config/Ultralytics/Arial.ttf...
100% 755k/755k [00:00<00:00, 28.5MB/s]
Downloading https://github.com/ultralytics/yolov5/releases/download/v6.1/yolov5s.pt to yolov5s.pt...
100% 14.1M/14.1M [00:00<00:00, 155MB/s]
```

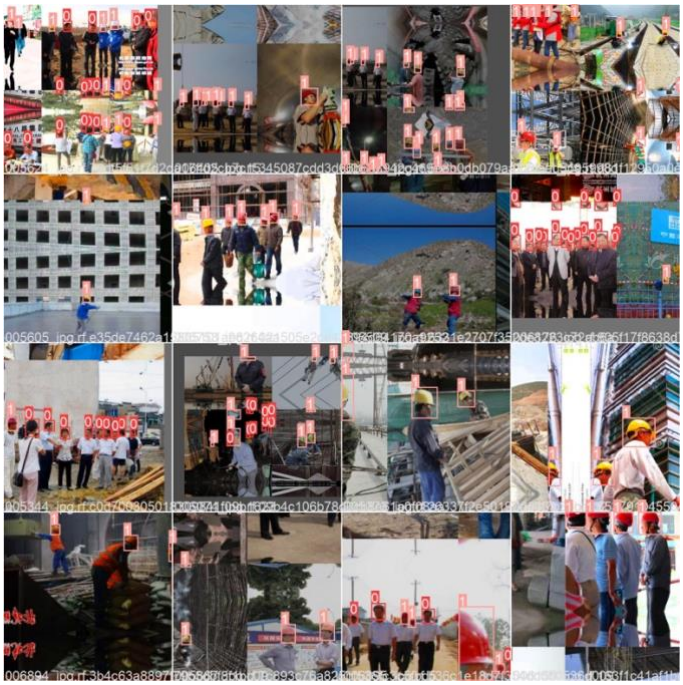
⇨ 학습 코드, **yolov5s** 사용

```
Overriding model.yaml nc=80 with nc=3

                 from  n    params  module                                  arguments
  0                -1  1      3520  models.common.Conv                      [3, 32, 6, 2, 2]
  1                -1  1     18560  models.common.Conv                      [32, 64, 3, 2]
  2                -1  1     18816  models.common.C3                        [64, 64, 1]
  3                -1  1     73984  models.common.Conv                      [64, 128, 3, 2]
  4                -1  2    115712  models.common.C3                        [128, 128, 2]
  5                -1  1    295424  models.common.Conv                      [128, 256, 3, 2]
  6                -1  3    625152  models.common.C3                        [256, 256, 3]
  7                -1  1   1180672  models.common.Conv                      [256, 512, 3, 2]
  8                -1  1   1182720  models.common.C3                        [512, 512, 1]
  9                -1  1    656896  models.common.SPPF                      [512, 512, 5]
 10                -1  1    131584  models.common.Conv                      [512, 256, 1, 1]
 11                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 12          [-1, 6]  1         0  models.common.Concat                    [1]
 13                -1  1    361984  models.common.C3                        [512, 256, 1, False]
 14                -1  1     33024  models.common.Conv                      [256, 128, 1, 1]
 15                -1  1         0  torch.nn.modules.upsampling.Upsample    [None, 2, 'nearest']
 16          [-1, 4]  1         0  models.common.Concat                    [1]
 17                -1  1     90880  models.common.C3                        [256, 128, 1, False]
 18                -1  1    147712  models.common.Conv                      [128, 128, 3, 2]
 19         [-1, 14]  1         0  models.common.Concat                    [1]
 20                -1  1    296448  models.common.C3                        [256, 256, 1, False]
 21                -1  1    590336  models.common.Conv                      [256, 256, 3, 2]
 22         [-1, 10]  1         0  models.common.Concat                    [1]
 23                -1  1   1182720  models.common.C3                        [512, 512, 1, False]
 24      [17, 20, 23]  1     21576  models.yolo.Detect                      [3, [[10, 13, 16, 30, 33, 23], [30, 61, 62, 45, 59, 119], [116, 90, 156
YOLOv5s summary: 270 layers, 7027720 parameters, 7027720 gradients, 15.9 GFLOPs
```

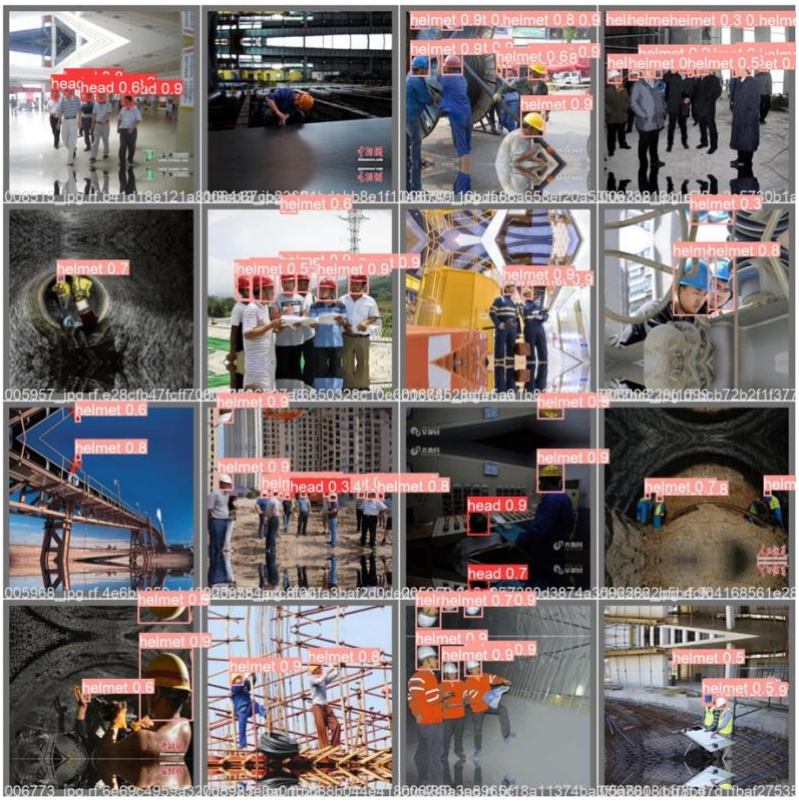
⇨ Yolov5s model summary

```
      Epoch   gpu_mem       box       obj       cls    labels  img_size
      46/49     1.94G   0.02824   0.02102  0.001294        26       416: 100% 89/89 [00:26<00:00,  3.38it/s]
                 Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 12/12 [00:03<00:00,  3.50it/s]
                   all        354       1678      0.627      0.564      0.607      0.394

      Epoch   gpu_mem       box       obj       cls    labels  img_size
      47/49     1.94G   0.02784   0.02054  0.001226        44       416: 100% 89/89 [00:26<00:00,  3.35it/s]
                 Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 12/12 [00:03<00:00,  3.40it/s]
                   all        354       1678       0.63      0.558      0.608      0.392

      Epoch   gpu_mem       box       obj       cls    labels  img_size
      48/49     1.94G   0.02796   0.02092  0.001433        32       416: 100% 89/89 [00:26<00:00,  3.37it/s]
                 Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 12/12 [00:03<00:00,  3.55it/s]
                   all        354       1678      0.627      0.564      0.608      0.392

      Epoch   gpu_mem       box       obj       cls    labels  img_size
      49/49     1.94G   0.02767   0.02103  0.001459        57       416: 100% 89/89 [00:26<00:00,  3.39it/s]
                 Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 12/12 [00:03<00:00,  3.40it/s]
                   all        354       1678       0.63      0.559      0.608      0.396

50 epochs completed in 0.431 hours.
Optimizer stripped from runs/train/hat_yolov5s_results/weights/last.pt, 14.3MB
Optimizer stripped from runs/train/hat_yolov5s_results/weights/best.pt, 14.3MB

Validating runs/train/hat_yolov5s_results/weights/best.pt...
Fusing layers...
YOLOv5s summary: 213 layers, 7018216 parameters, 0 gradients, 15.8 GFLOPs
                 Class     Images     Labels          P          R     mAP@.5 mAP@.5:.95: 100% 12/12 [00:05<00:00,  2.18it/s]
                   all        354       1678       0.63      0.559      0.608      0.396
                  head        354        312      0.942      0.838      0.897      0.595
                helmet        354       1310      0.947      0.838      0.925      0.591
                person        354         56          0          0    0.00312    0.00162
Results saved to runs/train/hat_yolov5s_results
```

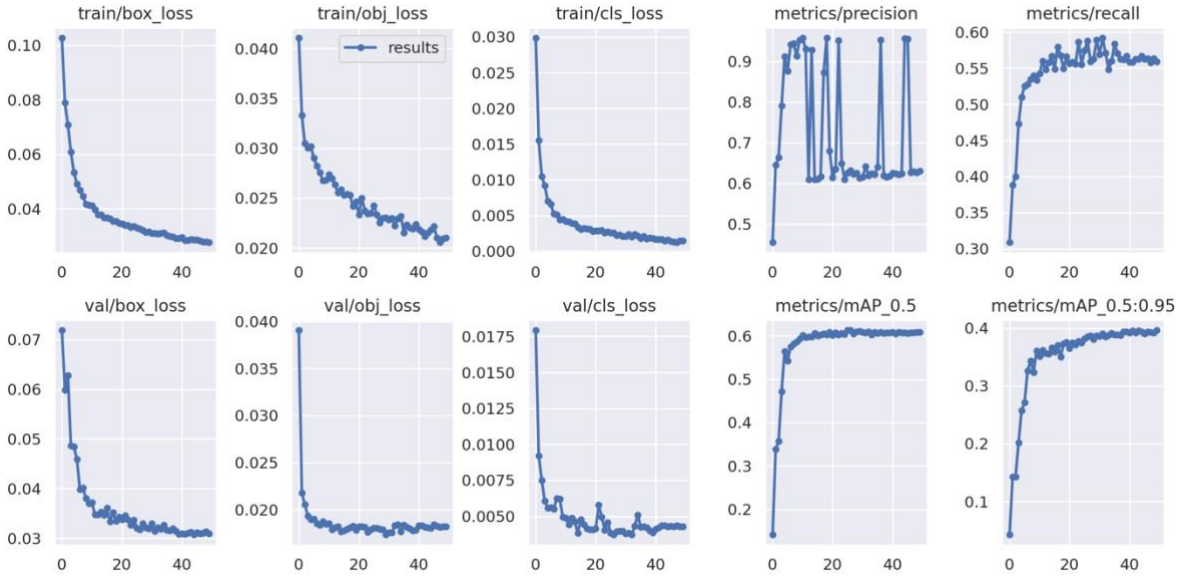
⇨ 모델 학습 완료



⇨ Train batch



⇨ Val batch



⇨ Results

⇨ Helmet 인식 잘 하는 것 확인하였음, 성능 지표에 대해서도 전반적으로 좋은 결과가 나타났지만 precision에 대해서는 좋은 결과를 내지 못함.