RSP Project

RSP (Rock Scissors Paper)
: Image Classification, Save Dataset

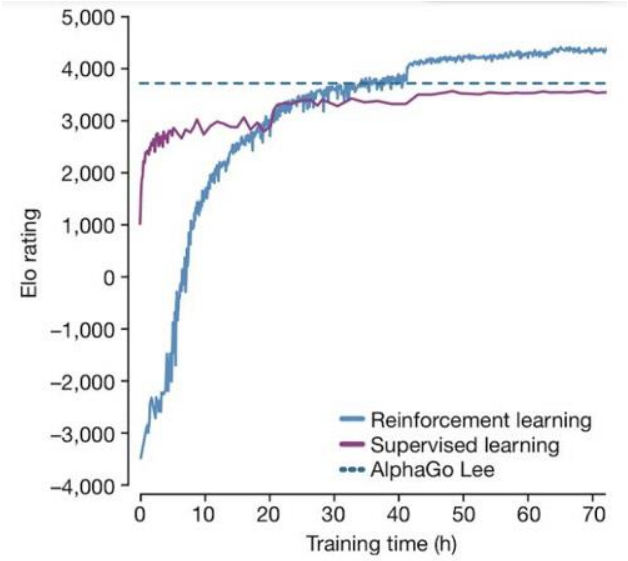
김기영 옥진해 전고은

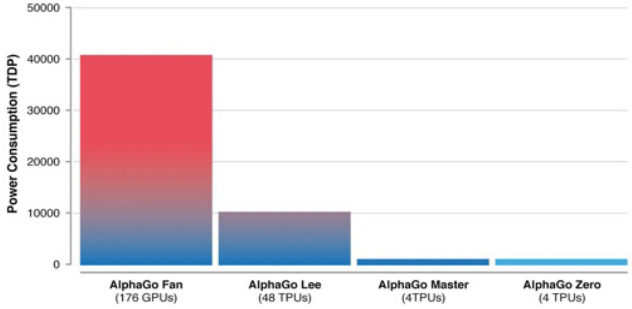
INDEX

Effect

Brain-Storming

Alpha Zero





알파고 제로와 기존 알파고 버전의 컴퓨팅 파워를 비교한 그림. DeepMind 제공

강화학습 방식으로 만들어진 알파고 제로는 지금까지 나온 알파고 버전들 중 가장 강력하다. 알파고 제로는 72시간 독학을 한 후 '알파고 리'와 대국한 결과 100전 100승을 기록했다. 한 수에 0.4초가 걸리는 '초속기' 바둑으로 490만판을 혼자 두고 쌓은 결과다. 40일에 걸쳐 2900만 판을 혼자 둔 후에는 올해 5월 세계랭킹 1위 커제 9단을 3대 0으로 꺾었던 '알파고 마스터'의 실력마저 압도했다. 알파고 제로는 알파고 마스터에 100전 89승 11패를 거뒀다. 알파고 제로는 강화학습으로 바둑의 이치를 스스로 깨달았을뿐만아니라 새로운 정석을 개발하기도 했다.

Introduce

01

Label Image

이미지 클래스 분류

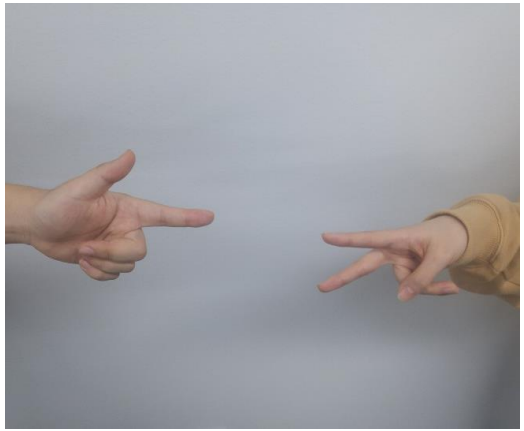
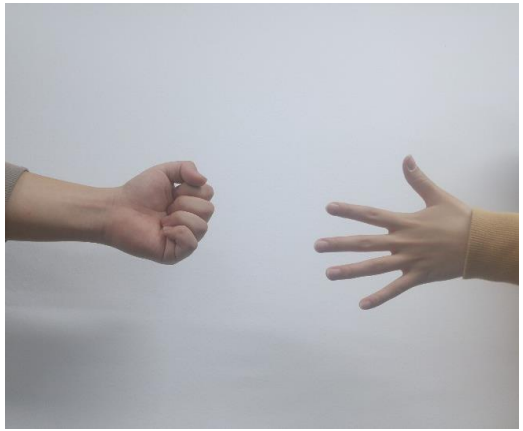
02

Google Colaboratory & YOLO

yolov3 활용
coco dataset 이미지 분류 학습

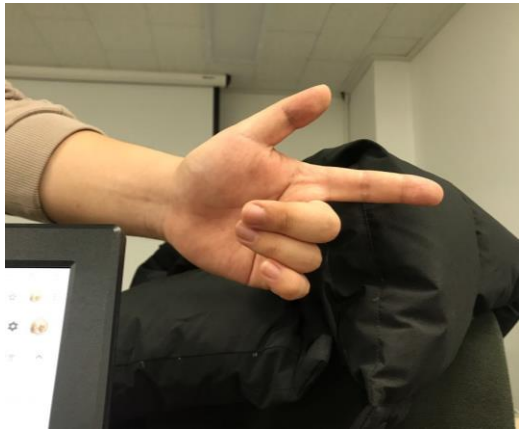
클래스(이미지) 처리 (1)

- 단순 분류


클래스(이미지) 처리 (2)

- 혼합 분류



Training (Google Colab)

```
!./darknet detector train game_yolo/custom_data.data game_yolo/custom-train-yolo.cfg backup/custom-train-yolo_backup.weights -dont_show
```

```
learning_rate=0.001
burn_in=1000
max_batches = 6000
policy=steps
steps=4800,5400
scales=.1,.1
```

```
learning_rate=0.001
burn_in=1000
max_batches = 9000
policy=steps
steps=7200,8100
scales=.1,.1
```

```
learning_rate=0.001
burn_in=1000
max_batches = 12000
policy=steps
steps=9600,10800
scales=.1,.1
```

```
384 x 384
 try to allocate additional workspace_size = 52.43 MB
 CUDA allocate done!
Loaded: 0.000059 seconds

 11991: 0.018834, 0.014940 avg loss, 0.000010 rate, 2.085105 seconds, 383712 images
Loaded: 0.000055 seconds

 11992: 0.012876, 0.014733 avg loss, 0.000010 rate, 2.162809 seconds, 383744 images
Loaded: 0.000045 seconds

 11993: 0.011104, 0.014370 avg loss, 0.000010 rate, 2.164764 seconds, 383776 images
Loaded: 0.000059 seconds

 11994: 0.016126, 0.014546 avg loss, 0.000010 rate, 2.169024 seconds, 383808 images
Loaded: 0.000071 seconds

 11995: 0.010130, 0.014104 avg loss, 0.000010 rate, 2.167438 seconds, 383840 images
Loaded: 0.000048 seconds

 11996: 0.013662, 0.014060 avg loss, 0.000010 rate, 2.172126 seconds, 383872 images
Loaded: 0.000069 seconds

 11997: 0.016249, 0.014279 avg loss, 0.000010 rate, 2.171730 seconds, 383904 images
Loaded: 0.000045 seconds

 11998: 0.019076, 0.014759 avg loss, 0.000010 rate, 2.166644 seconds, 383936 images
Loaded: 0.000040 seconds

 11999: 0.008589, 0.014142 avg loss, 0.000010 rate, 2.171869 seconds, 383968 images
Loaded: 0.000040 seconds

 12000: 0.009018, 0.013629 avg loss, 0.000010 rate, 2.079949 seconds, 384000 images
Saving weights to backup/custom-train-yolo_12000.weights
Saving weights to backup/custom-train-yolo_last.weights
Saving weights to backup/custom-train-yolo_final.weights
```

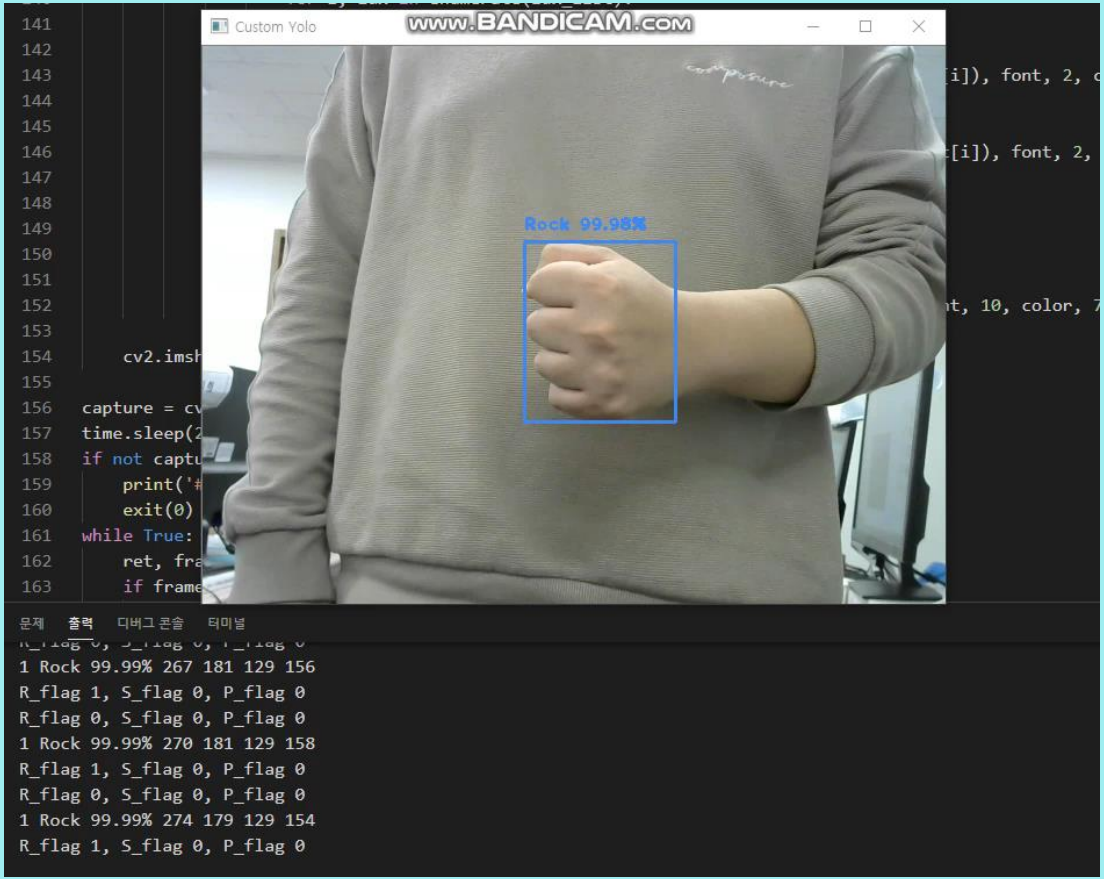
CODE

```
def RSP_flag_function(rsp_flag, names):
    if names == 'Rock':
        rsp_flag[0] = 1
    elif names == 'Scissors':
        rsp_flag[1] = 1
    elif names == 'Paper':
        rsp_flag[2] = 1
    return rsp_flag


def RSP_output(win, lose, img, indexes, names, boxes, font, colors):
    for idx in indexes:
        x, y, w, h = boxes[idx[0]]
        color = colors[idx[0]]
        if names[idx[0]] == win:
            cv2.putText(img, win_label, (x, y+h), font, 2, color, 2)
        if names[idx[0]] == lose:
            cv2.putText(img, lose_label, (x, y+h), font, 2, color, 2)
```

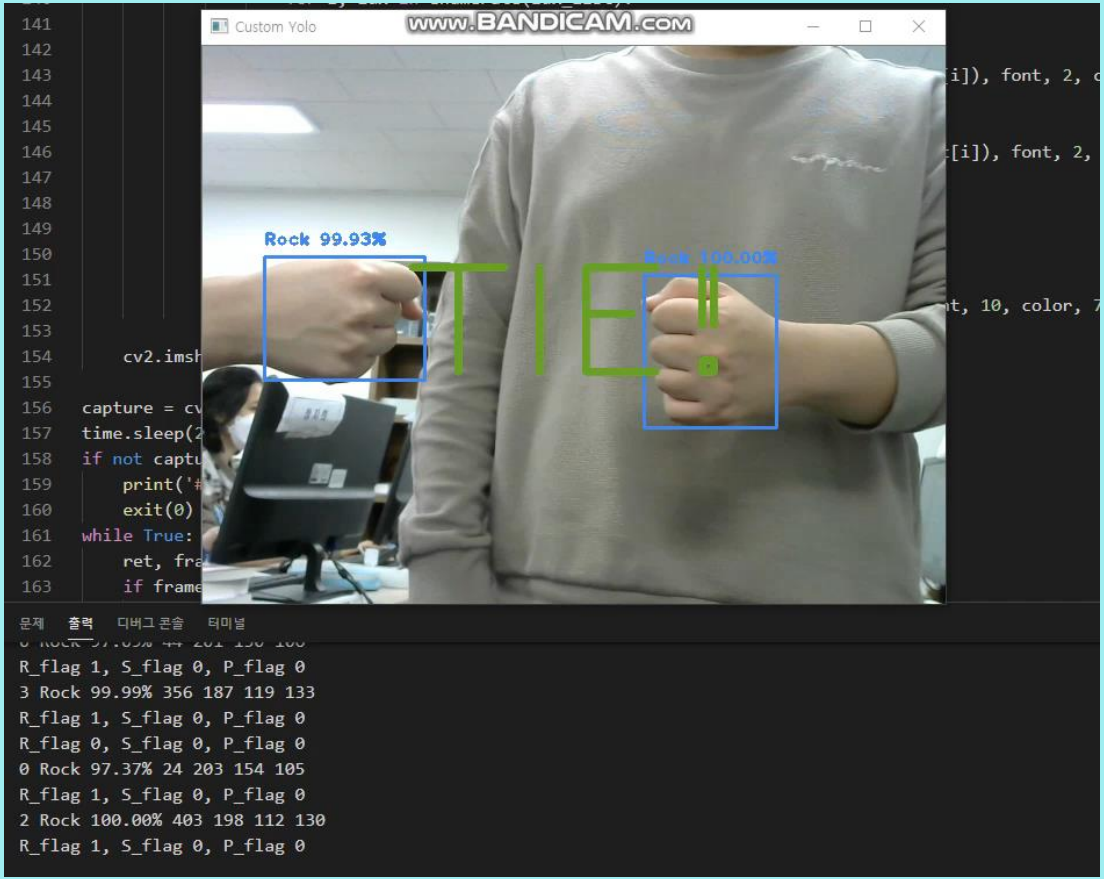
CODE

```
if len(indexes) > 1:
    if sum(rsp_flag) != 2:
        color = np.random.uniform(0, 255, size=(3,))
        cv2.putText(img, tie_label, (int(width/2)-150, int(height/2)+50), font, 10, color, 7)

    else:
        # 묵vs찌
        if rsp_flag[0] == 1 and rsp_flag[1] == 1 and rsp_flag[2] == 0:
            win = 'Rock'
            lose = 'Scissors'
            RSP_output(win, lose, img, indexes, names, boxes, font, colors)

        # 묵vs빠
        elif rsp_flag[0] == 1 and rsp_flag[1] == 0 and rsp_flag[2] == 1:
            win = 'Paper'
            lose = 'Rock'
            RSP_output(win, lose, img, indexes, names, boxes, font, colors)

        # 찌vs빠
        elif rsp_flag[0] == 0 and rsp_flag[1] == 1 and rsp_flag[2] == 1:
            win = 'Scissors'
            lose = 'Paper'
            RSP_output(win, lose, img, indexes, names, boxes, font, colors)
```

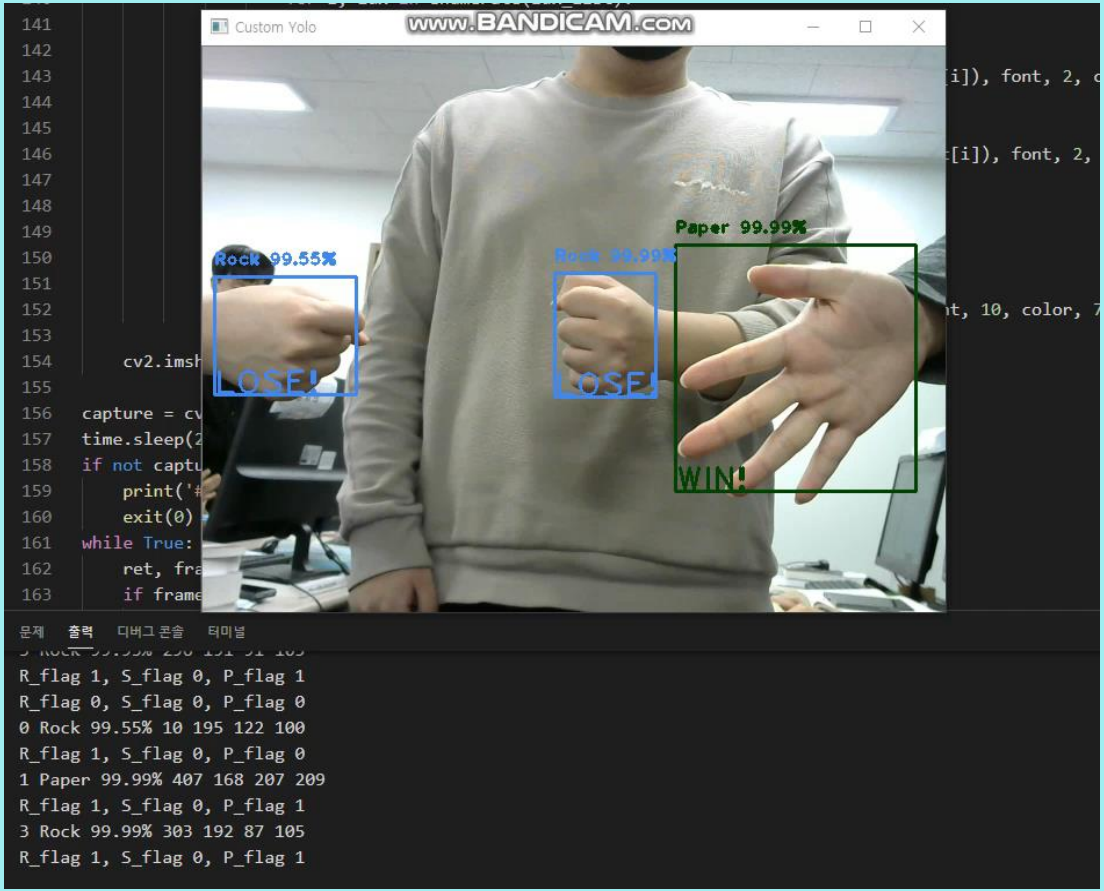
DEMO


Custom Yolo
www.BANDICAM.com
Rock 99.98%
문제
출력
디버그 콘솔
터미널
1 Rock 99.99% 267 181 129 156
R_flag 1, S_flag 0, P_flag 0
R_flag 0, S_flag 0, P_flag 0
1 Rock 99.99% 270 181 129 158
R_flag 1, S_flag 0, P_flag 0
R_flag 0, S_flag 0, P_flag 0
1 Rock 99.99% 274 179 129 154
R_flag 1, S_flag 0, P_flag 0

DEMO


Custom Yolo
www.BANDICAM.com
Rock 99.93%
Rock 100.00%
TIE
문제
출력
디버그 콘솔
터미널
R_flag 1, S_flag 0, P_flag 0
3 Rock 99.99% 356 187 119 133
R_flag 1, S_flag 0, P_flag 0
R_flag 0, S_flag 0, P_flag 0
0 Rock 97.37% 24 203 154 105
R_flag 1, S_flag 0, P_flag 0
2 Rock 100.00% 403 198 112 130
R_flag 1, S_flag 0, P_flag 0

DEMO


Custom Yolo
www.BANDICAM.com
Rock 99.55%
Paper 99.99%
LOSE!
WIN!
문제
출력
디버그 콘솔
터미널
R_flag 1, S_flag 0, P_flag 1
R_flag 0, S_flag 0, P_flag 0
0 Rock 99.55% 10 195 122 100
R_flag 1, S_flag 0, P_flag 0
1 Paper 99.99% 407 168 207 209
R_flag 1, S_flag 0, P_flag 1
3 Rock 99.99% 303 192 87 105
R_flag 1, S_flag 0, P_flag 1

Thank you