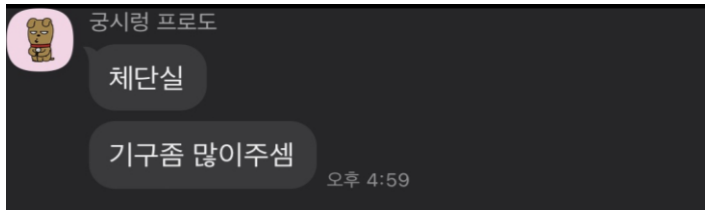
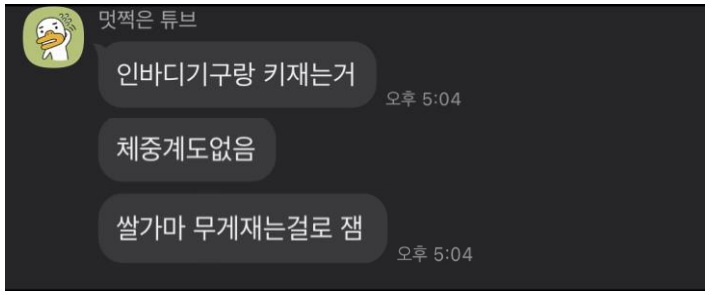
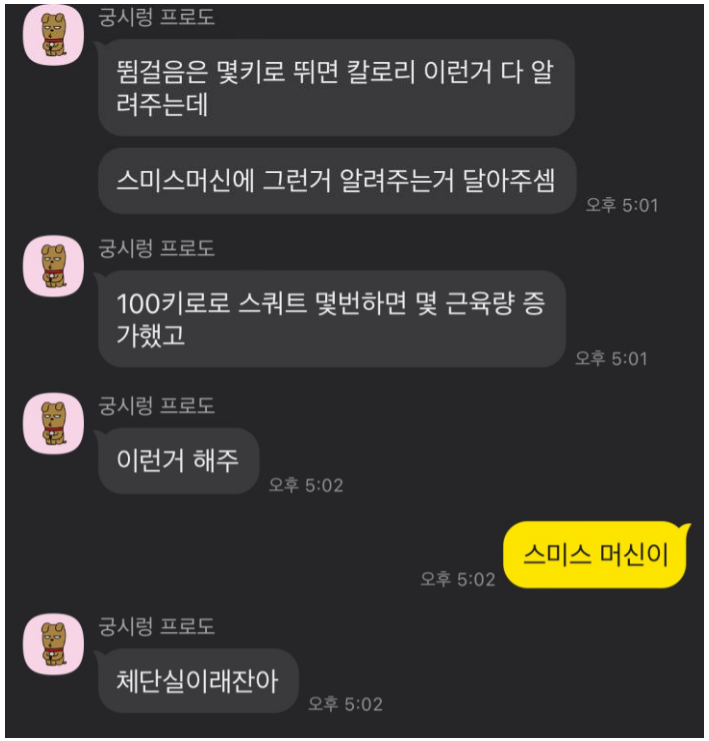
머신러닝을 활용한 운동자세보조 프로그램 개발

김민지, 박현준, 이무영, 조연우, 조하진

주제 선정 동기



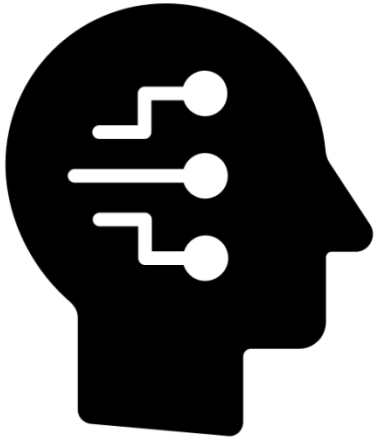




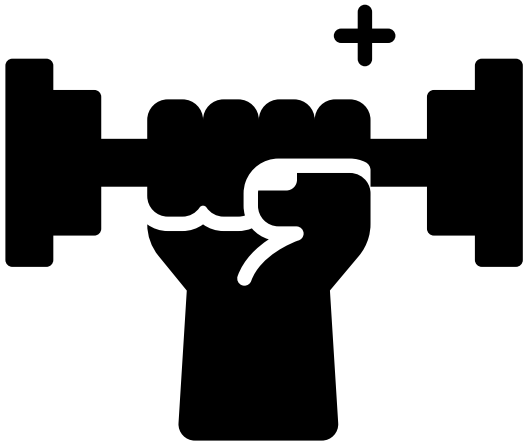
- 군 복무 중인 장병들 다수가 운동수행능력 증진을 목적으로 웨이트 트레이닝 등의 운동을 함. 군대 내에 있는 헬스장의 환경 개선을 요구하는 목소리도 들을 수 있었음.

- 국군장병의 헬스케어를 도울 수 있는 방법을 고안함.

연구 목적



딥러닝/머신러닝

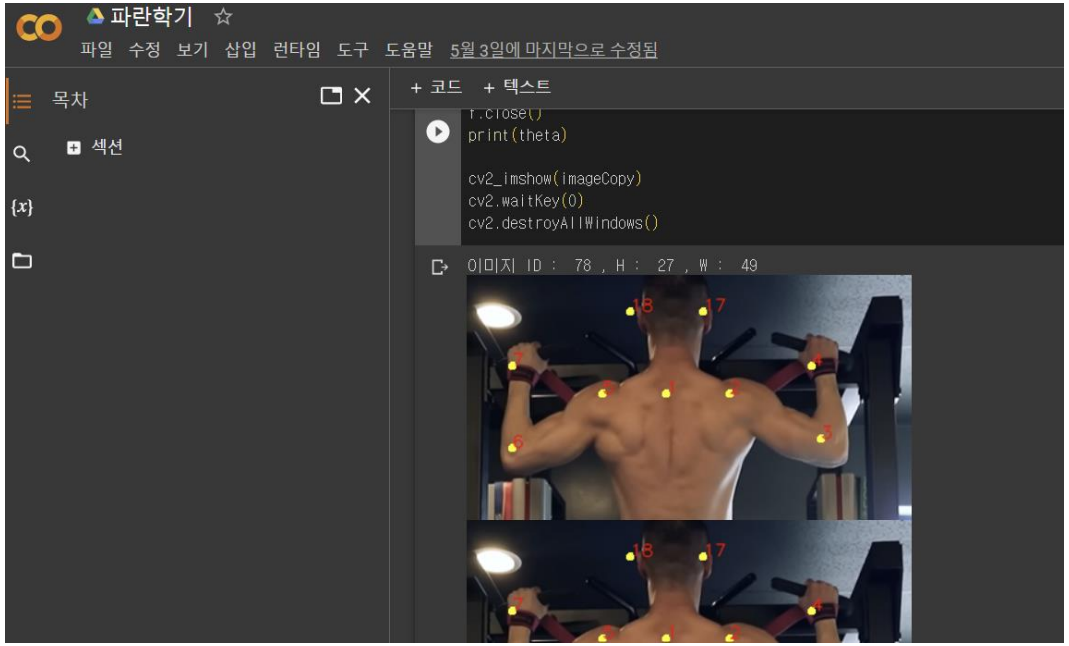


맞춤형 헬스 트레이닝

군인의 신체능력 향상

Open Pose

- 카메라를 통해 사람의 몸, 얼굴, 손가락마디 등 신체 키포인트들을 추출해내는 프로그램
- 신체의 특징점을 추론하고 이 관절들을 이어주는 방식
- CNN 네트워크 : BODY-25, COCO, MPII



Open Pose

- Openpose를 통해 얻은 관절의 좌표에서 원하는 관절(풀업 자세)의 각도를 구하고자 함.
- 좌표를 이용해 점 사이의 거리를 구하는 함수를 구현함.
- 점 사이의 거리를 활용하여 코사인법칙을 통해 관절의 각도를 구하는 함수를 구현함.
- 약 500개의 데이터(AI-HUB 운동자세이미지)로 부터 각도를 추출하여 csv 파일로 저장

```
def lens(a_x,a_y,b_x,b_y):
  len = abs(math.sqrt(math.pow(b_x-a_x,2)+math.pow(b_y-a_y,2)))
  return len

def angle(a_x,a_y,b_x,b_y,c_x,c_y):
  len1 = lens(a_x,a_y,b_x,b_y)
  len2 = lens(b_x,b_y,c_x,c_y)
  len3 = lens(a_x,a_y,c_x,c_y)
  tmp = (math.pow(len2,2)+math.pow(len1,2)-math.pow(len3,2))/(2*len2*len1)
  return math.degrees(math.acos(tmp))

theta = [] #각도
theta1 = angle(points_x[4],points_y[4],points_x[3],points_y[3],points_x[2],points_y[2])
theta2 = angle(points_x[8],points_y[8],points_x[2],points_y[2],points_x[3],points_y[3])
theta3 = angle(points_x[5],points_y[5],points_x[6],points_y[6],points_x[7],points_y[7])
theta4 = angle(points_x[6],points_y[6],points_x[5],points_y[5],points_x[8],points_y[8])

theta.append(theta1)
theta.append(theta2)
theta.append(theta3)
theta.append(theta4)
cv2_imshow(imageCopy)
cv2.waitKey(0)
cv2.destroyAllWindows()

f= open("/content/drive/MyDrive/2023 BLUESEMESTER JOJOJO/up.csv",'a',newline='')
wr = csv.writer(f)
wr.writerow([theta1, theta2, theta3, theta4])
f.close()
print(theta)
```

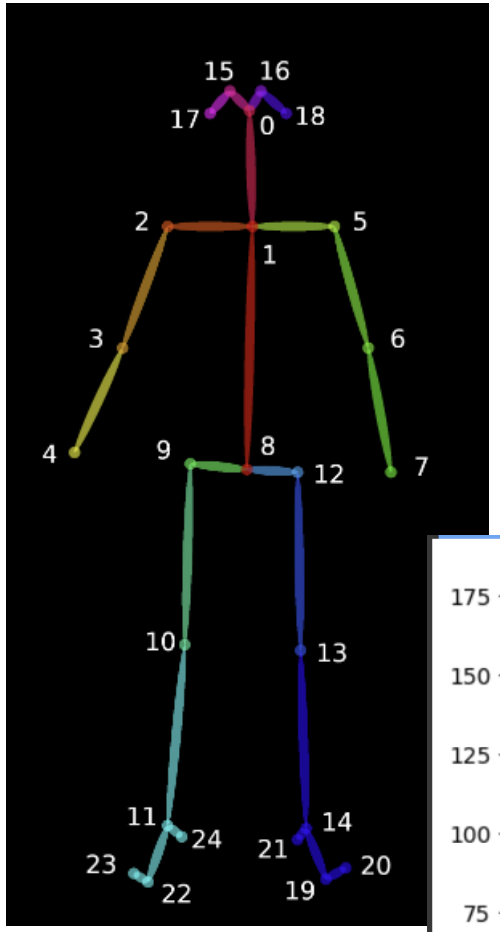
upright.csv

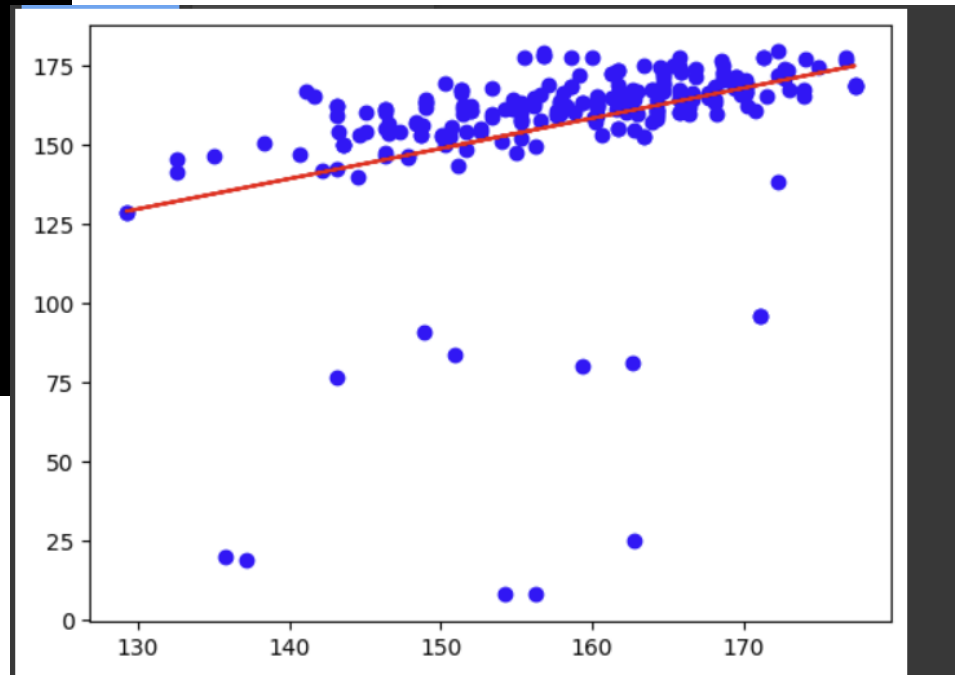
	A	B
1	50.71059314	60.2551187
2	57.99461679	76.42956561
3	53.13010235	72.42014857
4	56.30993247	73.00917671
5	52.12501635	68.82426058
6	55.15426658	65.51331907
7	55.15426658	65.51331907
8	55.15426658	65.51331907
	60.9453959	78.47096427
	50.71059314	66.80140949
	50.59933934	70.93080646
	46.41442321	67.38013505
	46.41442321	67.38013505
	67.59016263	59.90825801
	40.60129465	61.45575268
	66.03751103	83.68763525
	58.24051992	80.36246189
	73.47420361	79.38034472
	73.47420361	76.8907918
	73.47420361	78.47096427
	69.44395478	133.1556927
	90	110.2248594
	69.44395478	132.6034452
	55.15426658	65.51331907
	64.33480854	77.28466301
	60.9453959	77.64464013
	59.4702941	69.07549826
	54.52466797	64.75947074
	68.19859051	88.42344994
	61.92751306	74.7448813
	49.39870535	53.56914188
	45	64.29004622
	41.58341181	48.34465032
34	57.72435569	81.86989765
35	49.75172907	73.75236043

Sklearn을 활용한 학습 알고리즘

- 선형회귀는 입력 변수(X)와 연속적인 종속 변수(Y) 사이의 선형 관계를 모델링하는 통계적 기법이다. sklearn은 파이썬에서 머신러닝 모델을 구축하기 위한 인기 있는 라이브러리이다. sklearn의 linear_model 모듈은 선형회귀 분석을 수행하기 위한 다양한 클래스와 메서드를 제공한다.

- 2,3,4 포인트(입력변수)에 대한 1,2,8(종속변수) 사이 선형관계를 모델링함(좌, 우 마찬가지)





Sklearn을 활용한 학습 알고리즘

- 500개의 운동자세 이미지의 데이터를 각도에 대한 데이터로 가공하는 과정에서 body-25가 관절을 제대로 인식하지 못하는 사진데이터도 존재하여 이상치에서 벗어나는 데이터들을 발견할 수 있었음

- 수백개의 csv데이터로부터 오류데이터를 일일이 지울 수 없기 때문에 이상치 탐지 알고리즘을 활용함

- 이상치인 데이터를 1로 반환하고 이상치에서 벗어나는 데이터는 0으로 반환하여 이를 dataFrame으로 다루어 구분해낼 수 있음

```
clf.fit(X)

# 이상치 탐지 (inlier: 이상치가 아닌 데이터, outlier: 이상치)
inlier_mask = clf.predict(X) == 1

# 이상치가 아닌 데이터로만 다시 X, y 추출
X = X[inlier_mask]
y = y[inlier_mask]

x_train,x_test,y_train,y_test = train_test_split(X,y,test_size = 0.2)
model3 = LinearRegression()
model3.fit(x_train,y_train)
y_predict = model3.predict(x_test)

mse = mean_squared_error(y_test,y_predict)
mse**0.5
#print(mse);

score = model3.score(X, y)


#print("성능 평가: {:.3f}".format(score))

#plt.scatter(X, y, color = 'blue')         # 독립변수와 종속변수의 데이터 점의 산포도를 플로트
#plt.plot(X, model3.predict(X), color = 'red') # 회귀직선을 플로트
#plt.show
error3 = max_image_theta[1] - model3.predict([[max_image_theta[0]]])
print(error3)
```

영상 프레임 분석

- 사용자의 운동(풀업)영상을 프로그램에 입력하면 프로그램이 운동자세에 대한 피드백을 제공해주는 것을 목표로 하였다. 따라서 사용자의 운동자세 영상을 분석할 필요가 있다.

- 운동자세 영상에서 운동 전(팔을 폈을 때)의 타겟 관절의 각도는 최대가 되고, 팔을 당겼을 때 타겟 관절의 각도는 최소가 되는 점을 이용한다. 이때 openpose(body-25)와 연계하여 프로그램을 작성한다.

- 영상에서 초당 10장의 사진을 뽑아 타겟 관절의 각도가 최소인 프레임과 최대인 프레임을 저장한다. 이후 이를 학습한 알고리즘에 predict함수를 이용하여 선형회귀로 만든 그래프에 대입한다.

```
###카메라랑 연결
capture = cv2.VideoCapture("/content/drive/MyDrive/2023 BLUESEMESTER JOJOJO/pullpupfinal.mp4")  # 카메라 정보 받아옴
# capture.set(cv2.CAP_PROP_FRAME_WIDTH, 640) #카메라 속성 설정
# capture.set(cv2.CAP_PROP_FRAME_HEIGHT, 480) # width:너비, height: 높이
image_array =[]
inputWidth = 320;
inputHeight = 240;
inputScale = 1.0 / 255;

# 반복문을 통해 카메라에서 프레임을 지속적으로 받아옴
while True:  # 아무 키나 누르면 끝난다.
    # 웹캠으로부터 영상 가져옴
    ret, frame = capture.read()
    # 영상이 커서 느리면 사이즈를 줄이자
    # frame=cv2.resize(frame,dsize=(320,240),interpolation=cv2.INTER_AREA)

    # 웹캠으로부터 영상을 가져올 수 없으면 웹캠 중지
    if not ret:
        break
    if (int(capture.get(1)) % 10 == 0):
      count = count + 1

      frameWidth = frame.shape[1]
      frameHeight = frame.shape[0]

      inpBlob = cv2.dnn.blobFromImage(frame, inputScale, (inputWidth, inputHeight), (0, 0, 0), swapRB=False, crop=False)

      imgb = cv2.dnn.imagesFromBlob(inpBlob)
```

```
            points_y.append(y)

      theta1 = angle(points_x[4],points_y[4],points_x[3],points_y[3],points_x[2],points_y[2])
      theta2 = angle(points_x[3],points_y[3],points_x[2],points_y[2],points_x[8],points_y[8])
      theta3 = angle(points_x[5],points_y[5],points_x[6],points_y[6],points_x[7],points_y[7])
      theta4 = angle(points_x[6],points_y[6],points_x[5],points_y[5],points_x[8],points_y[8])

      theta11.append(theta1)
      theta22.append(theta2)
      theta33.append(theta3)
      theta44.append(theta4)

          # 키포인트 검출한 결과가 0.1보다 크면(검출한곳이 위 BODY_PARTS랑 맞는 부위면) points에 추가, 검출했는데 부위가
      if prob > 0.1:
          cv2.circle(frame, (int(x), int(y)), 3, (0, 255, 255), thickness=-1,
                     lineType=cv2.FILLED)  # circle(그릴곳, 원의 중심, 반지름, 색)
          cv2.putText(frame, "{}".format(i), (int(x), int(y)), cv2.FONT_HERSHEY_SIMPLEX, 0.5, (0, 0, 255), 1,
                      lineType=cv2.LINE_AA)
          points.append((int(x), int(y)))
      else:
          points.append(None)

      image_array.append(frame)
max_image = 0
min_image = 0
for i in range(0,count):
  if(max_angle < theta11[i]):
    max_angle = theta11[i]
    max_image = i

for i in range(0,count):
  if(min_angle > theta11[i]):
    min_angle = theta11[i]
```

결과 분석

-동영상 데이터에서 관절의 각도 데이터를 뽑아 기존의 선형회귀 모델을 통해 오차율을 계산한다. 그 후 각각 양쪽 팔의 각도의 오차율을 통해 팔의 높낮이에 대한 피드백을 제공한다

-오차율 피드백 제공 출력물을 그저 txt파일 형식으로 제공하는 방식은 프로그램으로 보기 어렵기 때문에 GUI를 따로 작성하여 출력하고자 한다.

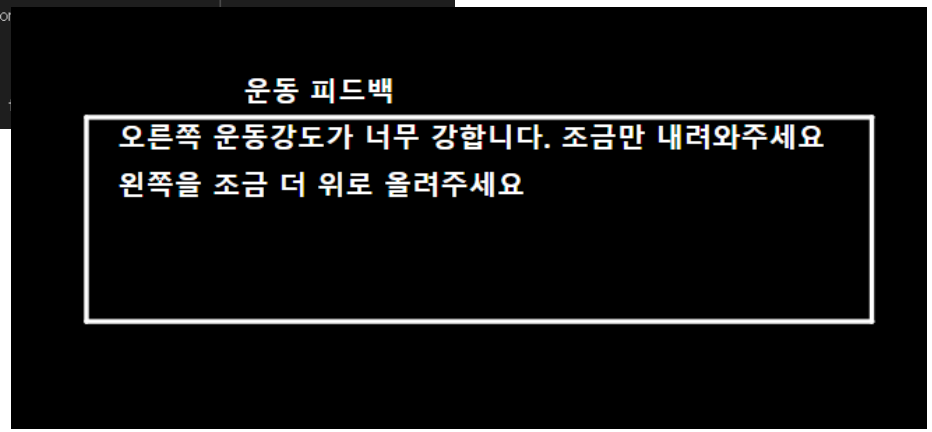
-UI를 비교적 쉽고 간단하게 출력할 수 있는 Colab Turtle을 사용하여 피드백 제공 출력물의 GUI를 작성했다.

```
import Colabturtle.Turtle as turtle
turtle.initializeTurtle()
turtle.hideturtle()
turtle.pensize(3)   # 펜의 두께를 3 픽셀
turtle.penup()
turtle.bgcolor("black")
turtle.pencolor("white")
turtle.goto(80,80)
turtle.pendown()
turtle.right(90)
turtle.fd(500)
turtle.right(90)
turtle.fd(130)
turtle.right(90)
turtle.fd(500)
turtle.right(90)
turtle.fd(130)
turtle.right(90)
turtle.penup()

turtle.goto(180,70)
turtle.pendown()
turtle.write("운동 피드백", font = (18,"맑은고딕", "bold"))    # 문자 쓰기
turtle.penup()


turtle.goto(100, 100)

if(error1>0):
  if(error1<=10):
    turtle.pendown()
    turtle.write("오른쪽을 조금 더 위로 올려주세요", fo
    turtle.penup()
  elif(error1>10):
    turtle.pendown()
    turtle.write("오른쪽을 충분히 위로 더 올려주세요",
```



발전 방향

1. 데이터 추가 학습을 통한 운동 솔루션 정확도 향상
현재 본 프로그램은 약 500여개의 데이터를 학습하고, 이에 기반해 운동 분석 및 솔루션 제공을 진행중이지만, 데이터를 추가로 학습시킨다면 더욱 정확하고 세세한 운동 솔루션을 제공해 줄 수 있으리라 생각된다.

2. 다양한 운동 분석
지금 데이터 가공 및 머신러닝을 진행한 운동은 풀업 하나뿐이지만, openpose를 통해 스켈레톤과 관절의 각도를 추출하고 sklearn을 통해 학습시켜 솔루션을 제공하는 프로그램의 구조는 다른 다양한 운동들에도 능히 적용 가능한 높은 확장성을 지닌다. 데이터만 제공된다면 기존 프로그램의 틀 아래 풀업뿐만 아니라 다양한 여러 운동들의 분석과 솔루션 제공 또한 가능하다.

3. 접근성 향상
현재 본 프로젝트는 영상 파일 입력 및 솔루션 출력 과정 모두 windows환경에 기반한 프로그램이다. 이를 모바일 환경에서도 구동 가능하도록 발전시킨다면, 많은 사람들이 널리 사용할 수 있는 프로그램이 될 것이다.

4. 군 적용 가능성
프로젝트 동기에서 언급했듯, 현재 군대에서는 퍼스널 트레이닝에 대한 높은 수요가 존재하지만, 군대의 환경상 이는 쉬이 충족지 않음. 이에 본 프로그램을 통해 군 장병들에게 운동 솔루션을 제공함으로서 이러한 수요를 충족시킬수 있기를 예상함.

우리 프로그램이 기존의 상용화된 프로그램에 대해 가지는 장점

보안

통신이 필요가 없다!

기존의 상용화된 프로그램들은 모두 네트워크, 통신 시스템 아래에서 운동 분석 및 솔루션 제공을 진행한다.
그러나 본 프로젝트가 주 타겟으로 삼는 군 장병들의 경우, 상용화된 프로그램을 사용하기에는 보안 등 여러 문제점이 존재한다.

그러나 우리의 프로젝트의 경우 영상 분석, 솔루션 제공 등의 모든 작업을 통신 대신 내부적으로 존재하는 학습된 머신러닝 모델들을 통해서 처리하므로 외부와의 통신, 정보 유출 등의 보안 이슈로부터 자유롭다.

또한 이는 보안이 중요한 군 장병들 뿐만 아니라, 여러가지 이유로 통신이 자유롭지 못한 지역의 이용자를 대상으로 큰 이점으로 작용할 것이다.

감사합니다