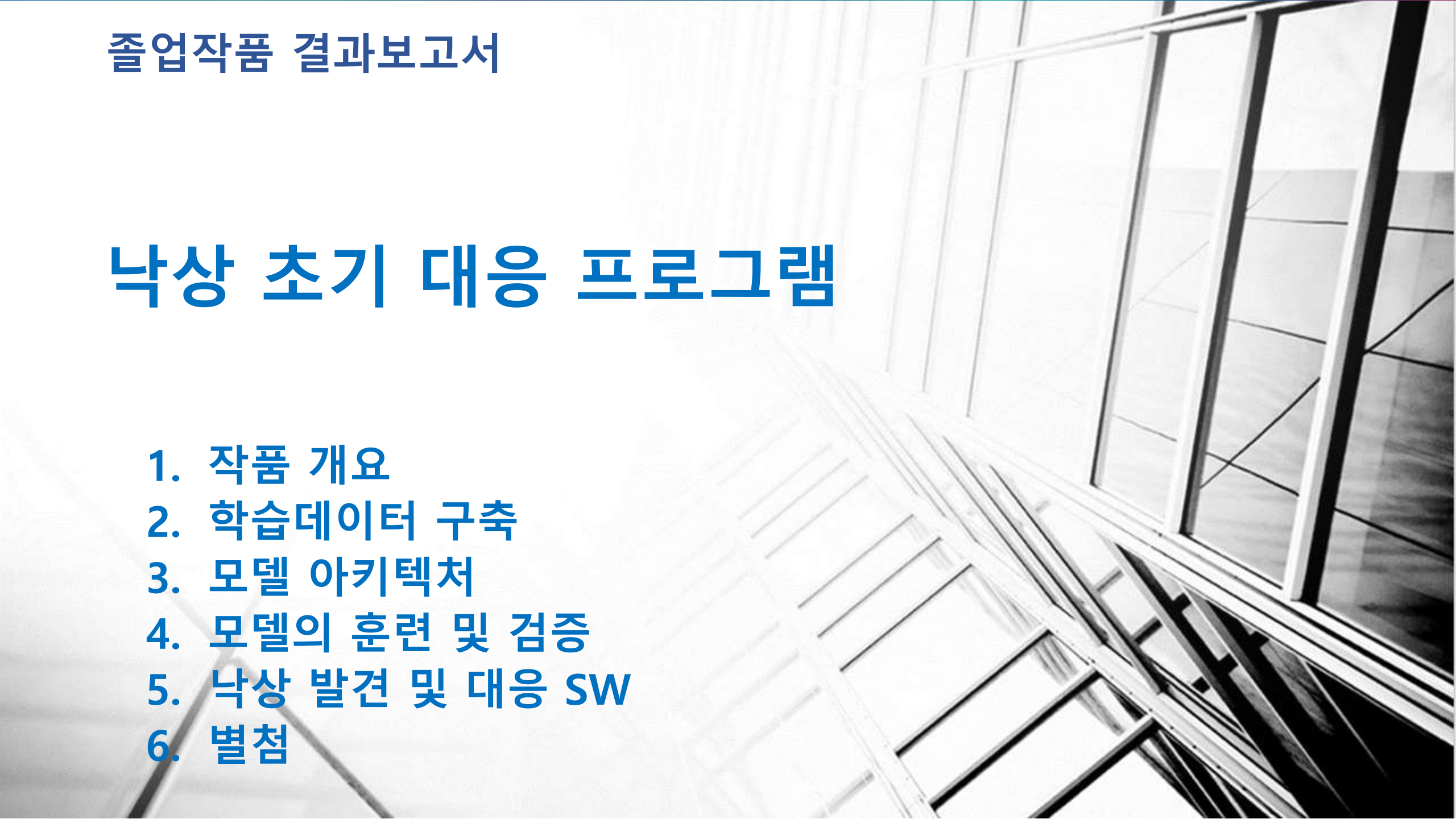
졸업작품 결과보고서

낙상 초기 대응 프로그램

1. 작품 개요
2. 학습데이터 구축
3. 모델 아키텍처
4. 모델의 훈련 및 검증
5. 낙상 발견 및 대응 SW
6. 별첨



1. 작품 개요

- 작품 소개
- 적용 기술
- 개발 및 사용 환경

작품 소개

대구대학교 DAEGU UNIVERSITY

목적

- 생활 속에서 일어날 수 있는 여러가지 사람들의 행동들 중에서 낙상 사고는 생각보다 흔하게 일어나는 사고 중 하나
- 신속한 대처는 사람의 목숨을 살릴 수도 있으나, 그 방법을 사람들이 잘 알지 못하는 경우가 다수
- 사고가 생긴 경우 빠른 초반 조치를 취하여 낙상 사고를 당한 사람의 목숨을 구할 수 있는 확률을 높일 수 있도록 하기 위함

프로그램 기능

- TTS 혹은 다른 신호를 통해 상황 전파, 응급 상황시, 구조 요청 신호 발생
- AI HUB 시니어 이상행동 데이터를 사용하여 YOLO5를 훈련시켜 얻은 학습 모델을 사용, 영상이 입력되면 실시간으로 이상행동(낙상)의 징후 및 사고를 찾아내어 주변 사람들에게 어떻게 도움을 주어야 하는지 알려줌
- 서버에 학습된 모델 내장 및 추론

대구대학교 DAEGU UNIVERSITY

작품 활용 절차

<학습단계>

시니어 이상행동 데이터
– YOLO5 학습

학습 모델 전송

<활용>

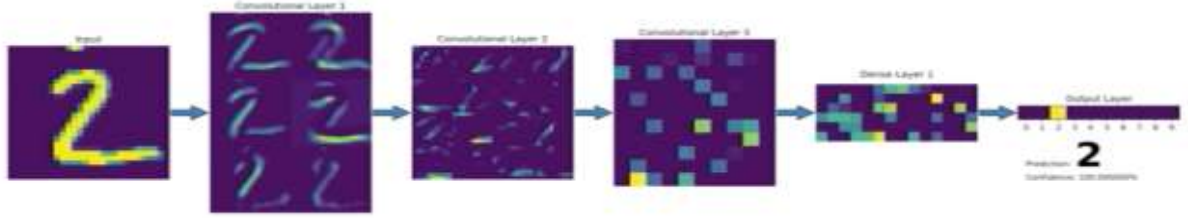
카메라를 통한
실시간 영상 입력

훈련된 모델을 통한
사람의 이상행동 파악

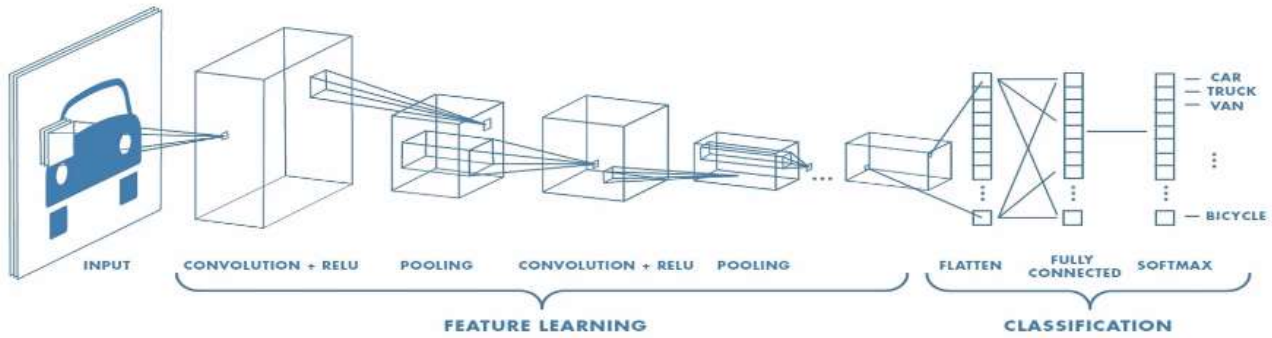
SW가 주변 사람들에게
응급상황 알림 및
도움 방법 알림

CNN의 구조 : 활용 측면과 학습 측면

- 활용 측면
  - CNN은 인간의 시신경 구조를 모방한 기술, 데이터를 학습하여 패턴을 찾아내어 이미지를 분류
  - 얼굴인식, 자율주행차와 같은 객체인식이나 그 외 다양한 분야에서 사용 중



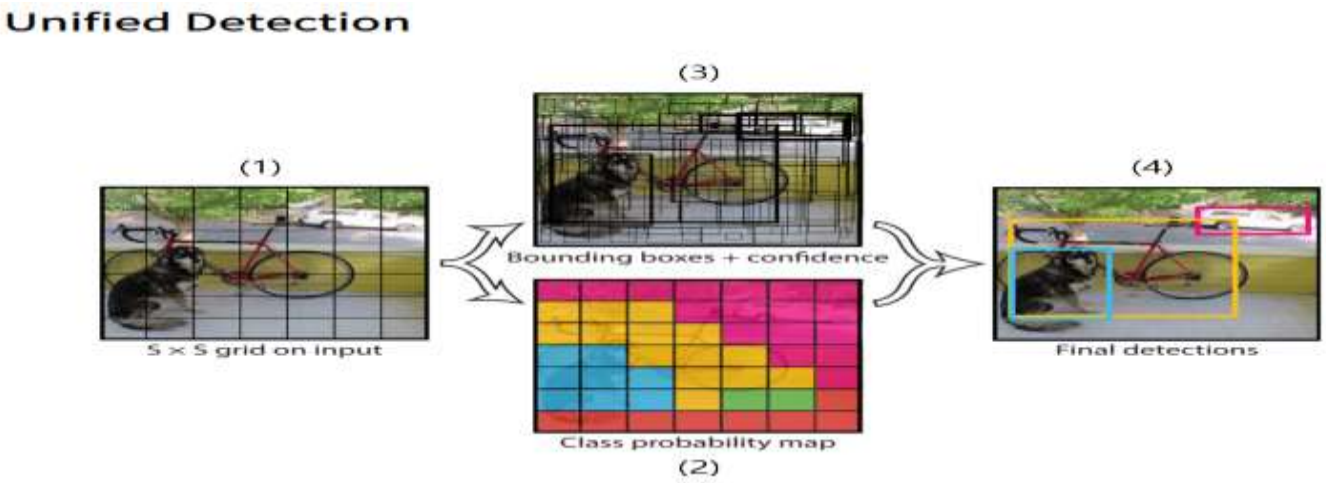
- 학습 측면
  - Convolution과 Pooling을 반복적으로 이용하여, 찾으려는 객체 또는 이미지의 특징을 찾아내고, 찾아낸 특징을 입력데이터로 Fully-connected 신경망에 보내어 Classification을 수행한다.



YOLO5(You Look Only Once)

- CNN을 이용한 객체 인식 알고리즘 중 하나이며, 객체 인식 알고리즘 분야에서 가장 유명한 알고리즘 중 하나

- 주요 특징
  - Single Stage Methods(Detection without Proposals)이며, 객체의 위치를 먼저 찾은 후 객체의 종류를 파악하는 Two-state Methods(ex:R-CNN) 와는 다르게 한번에 객체의 위치와 종류를 검출
  - 빠른 처리 시간을 이용한 실시간 객체 검출 가능



개발 및 실행 환경

하드웨어 및 OS

- OS : 윈도우 OS
- 옵션 선택으로 GPU 또는 CPU 모드로 동작 가능

소프트웨어

- 기본사항
  - 파이썬 기본 패키지 이외 나머지 아나콘다에서 설치
  - 서버구축은 Django 사용 예정
  - 작품에 사용하는 패키지 목록

구분	비고
python	버전 3.9
Python packages	os, sys, utils
numpy	Ver 1.21.2
matplotlib	Ver 3.4.3
pytorch	Pytorch 버전 311 이상
torchvision	Torchvision 버전 0.11

개발 업무 분장

개요

구분	하드웨어	소프트웨어
홍길동(팀장)	-	▪ 프로젝트 관리 ▪ 모델 설계 및 학습알고리즘 구현
홍길순	-	▪ 학습알고리즘 구현 지원 ▪ 응용 소프트웨어 구현

2. 학습 데이터 구축

- 학습데이터 구축
- 데이터셋 및 데이터로더 클래스 제작
- 진행현황 및 향후계획

대구대학교 DAEGU UNIVERSITY

개요

구분	제목
개요	외부 또는 내부에서 발생하는 이상행동(낙상) 영상 및 이미지 모음
출처	https://aihub.or.kr/aidata/34140
규모	전체 영상 250시간 동영상 2500클립, 이미지 250만장
파일구성	구분 / JSON 포맷 데이터셋 정보 이미지 정보 이미지 라이선스 정보
샘플 이미지	

대구대학교 DAEGU UNIVERSITY

학습 샘플의 구성

- 원본의 샘플의 이미지

- 라벨(label)

[senior posture-based and movements-based monitoring methods]

Level 1	Level 2
Action	Standing
	Sitting
	Bending
	Lying
	Lying toward
Movement	Walking/Running
	Jump
	Active
	Inactive

*** Nasution, A.H.; Zhang, P.; Emmanuel, S. Video surveillance for elderly monitoring and safety. In Proceedings of the TENCON 2009-2009 IEEE Region 10 Conference, Singapore, 23-26 January 2009; pp. 1-6

데이터셋 및 데이터로더 제작

대구대학교 DAEGU UNIVERSITY

제작방법

- AI HUB에서 이미 구축이 완료된 데이터셋 사용 예정
- JSON 포멧이므로 YOLO 포멧으로 CONVERT 필요

진행현황 및 향후계획

대구대학교 DAEGU UNIVERSITY

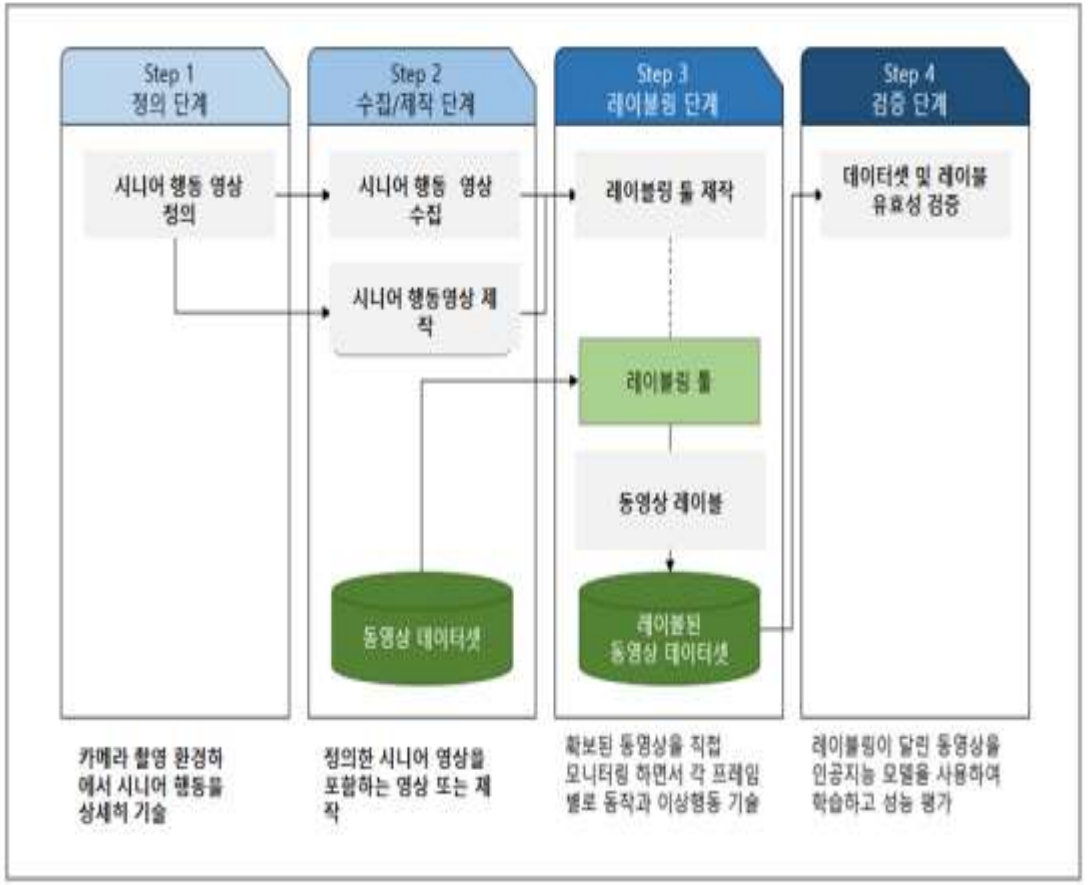
학습데이터 구현 현황

- 구축 규모
  - 훈련 : 동영상 2200 클립
    이미지 250만장
  - 상태 : 완료
- 구축 계획
  - JSON 포멧 YOLO 포멧 CONVERT(5월 중순)

데이터셋 및 데이터로더 제작

- JSON 포멧 YOLO 포멧 CONVERT 시 완료예정

데이터 구축은 2020년 10월 1일부터 참여형 크라우드 소싱 인력들의 직접 촬영, CCTV 촬영영상에 대해서 분야별 분류에 균등 분포가 이루어지게 했음



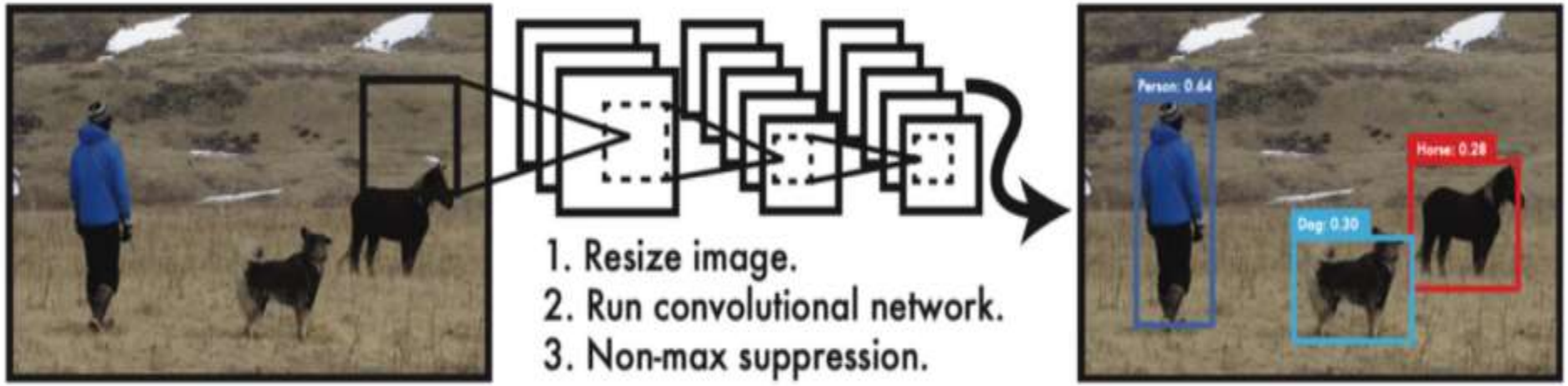
3. 모델 아키텍처

- YOLO 모델의 아키텍처
- YOLO 모델의 구현 : BACKBONE
- YOLO 모델의 구현 : HEAD
- Nvidia trt_pose 구현
- 현황 및 향후계획

YOLO 모델의 아키텍처

대구대학교 DAEGU UNIVERSITY

- 개요
  - YOLO의 구조는 크게 BACKBONE와 HEAD로 나누어져 있음
  - BACKBONE 부분은 이미지로부터 Feature map을 추출
  - HEAD 부분은 추출된 Feature map을 바탕으로 물체의 위치를 찾음



YOLO 모델의 아키텍처

YOLO 모델의 구현

대구대학교 DAEGU UNIVERSITY

- 모듈 구성

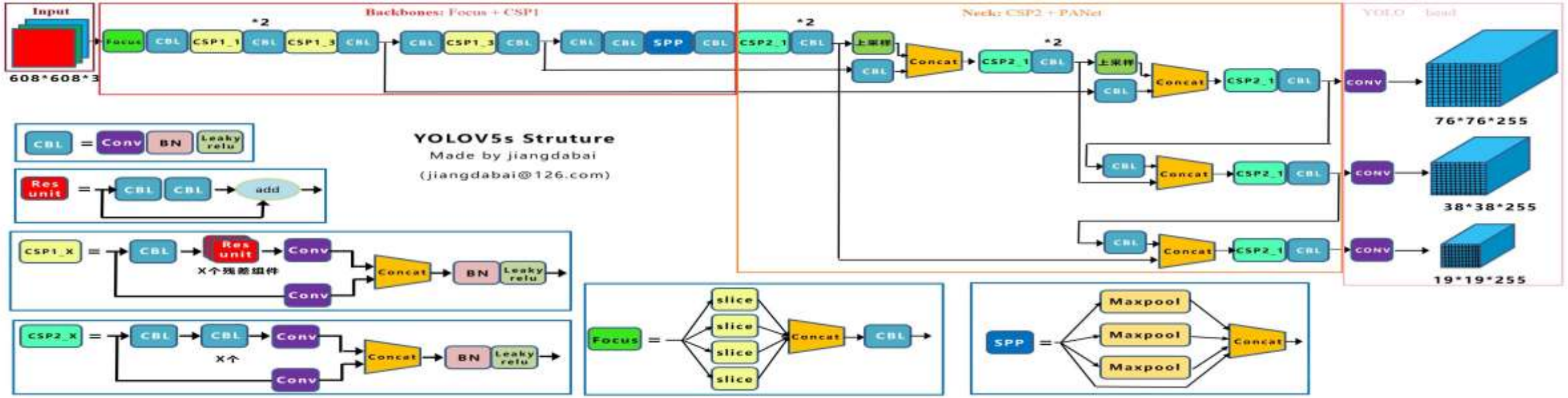
```
Using CUDA device0 _CudaDeviceProperties(name='GeForce GTX 1080 Ti', total_memory=11178MB)
         device1 _CudaDeviceProperties(name='GeForce GTX 1080 Ti', total_memory=11178MB)

                 from  n    params  module                                  arguments
  0                -1  1      3520  models.common.Focus                     [3, 32, 3]
  1                -1  1     18560  models.common.Conv                      [32, 64, 3, 2]
  2                -1  1     19904  models.common.BottleneckCSP             [64, 64, 1]
  3                -1  1     73984  models.common.Conv                      [64, 128, 3, 2]
  4                -1  1    161152  models.common.BottleneckCSP             [128, 128, 3]
  5                -1  1    295424  models.common.Conv                      [128, 256, 3, 2]
  6                -1  1    641792  models.common.BottleneckCSP             [256, 256, 3]
  7                -1  1   1180672  models.common.Conv                      [256, 512, 3, 2]
  8                -1  1    656896  models.common.SPP                       [512, 512, [5, 9, 13]]
  9                -1  1   1248768  models.common.BottleneckCSP             [512, 512, 1, False]
 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.BottleneckCSP             [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     95104  models.common.BottleneckCSP             [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    313088  models.common.BottleneckCSP             [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   1248768  models.common.BottleneckCSP             [512, 512, 1, False]
 24      [17, 20, 23]  1    229245  Detect                                  [80, [[10, 13, 16, 30, 33, 23], [30

Model Summary: 191 layers, 7.46816e+06 parameters, 7.46816e+06 gradients
```

SOURCE

BACKBONE

- 전체 구조
  - YOLO5 의 BACKBONE은 4가지의 종류가 존재 (S(small), M(medium), L(large), X(extra))
  - BACKBONE은 2가지 변수로 결정됨 (depth_multiple, width_multiple)

  - depth_multiple의 값이 클수록 BottlebeckCSP모듈이 더 많이 반복되어, 더 깊은 모델이 됨
  - width_multiple의 값이 클수록 해당 레이어의 Conv 필터수가 많아짐



YOLO 모델의 구현 : BACKBONE

대구대학교 DAEGU UNIVERSITY

BACKBONE 모듈 구성

```
number : 3
depth_gain : 1
module : BottleneckCSP(
  (cv1): Conv(
    (conv): Conv2d(64, 32, kernel_size=(1, 1), stride=(1, 1), bias=False)
    (bn): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (act): Hardswish()
  )
  (cv2): Conv2d(64, 32, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (cv3): Conv2d(32, 32, kernel_size=(1, 1), stride=(1, 1), bias=False)
  (cv4): Conv(
    (conv): Conv2d(64, 64, kernel_size=(1, 1), stride=(1, 1), bias=False)
    (bn): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
    (act): Hardswish()
  )
  (bn): BatchNorm2d(64, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
  (act): LeakyReLU(negative_slope=0.1, inplace=True)
  (m): Sequential(
    (0): Bottleneck(
      (cv1): Conv(
        (conv): Conv2d(32, 32, kernel_size=(1, 1), stride=(1, 1), bias=False)
        (bn): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (act): Hardswish()
      )
      (cv2): Conv(
        (conv): Conv2d(32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1), bias=False)
        (bn): BatchNorm2d(32, eps=1e-05, momentum=0.1, affine=True, track_running_stats=True)
        (act): Hardswish()
      )
    )
  )
)
```

SOURCE

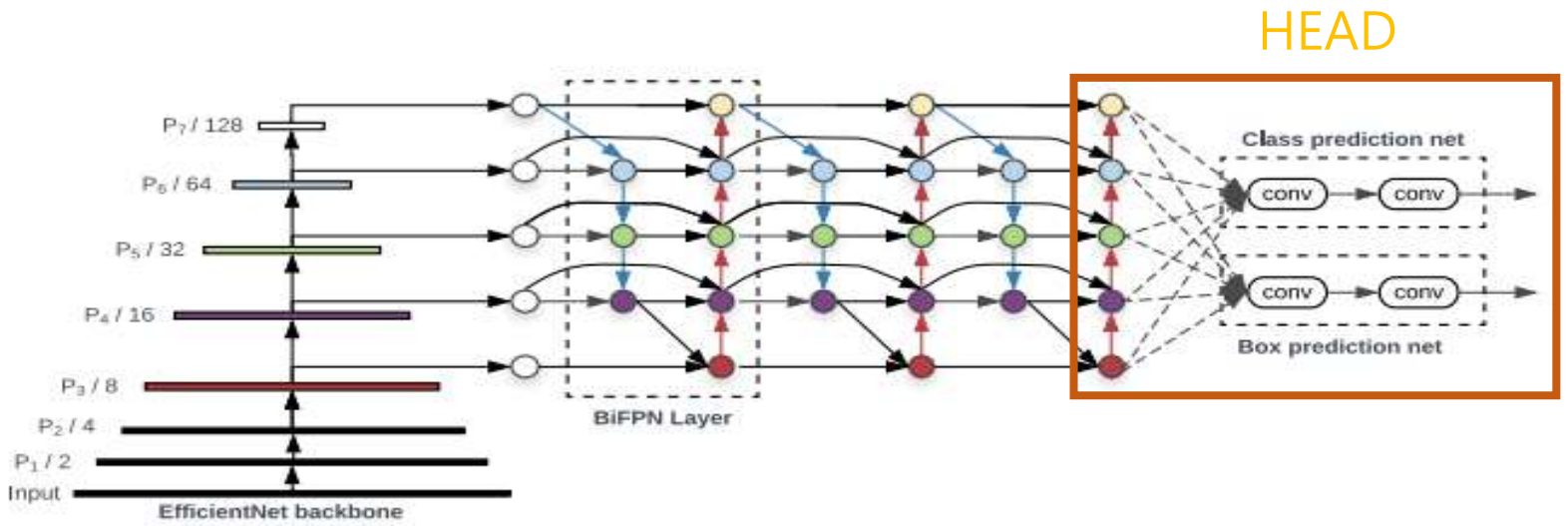
YOLO 모델의 구현 : BACKBONE

- BACKBONE 모듈 구성

SOURCE

HEAD

- 전체 구조
  - HEAD는 크게 보자면 이미지의 객체 위치 및 종류를 예측하는 곳
  - from, number, module, args 으로 구성
  - Conv, Upsample, Concat, BottleneckCSP은 Detect부분에서 연결



YOLO 모델의 구현 : HEAD

대구대학교 DAEGU UNIVERSITY

HEAD 모듈 구성

```
# yolov5s.yaml
# YOLOv5 head
head:
  [[-1, 1, Conv, [512,
   [-1, 1, nn.Upsample
   [[-1, 6], 1, Concat
   [-1, 3, BottleneckC
  ]
```

```
# yolo.py result (YOLOv5-s'HEAD)
```

SOURCE

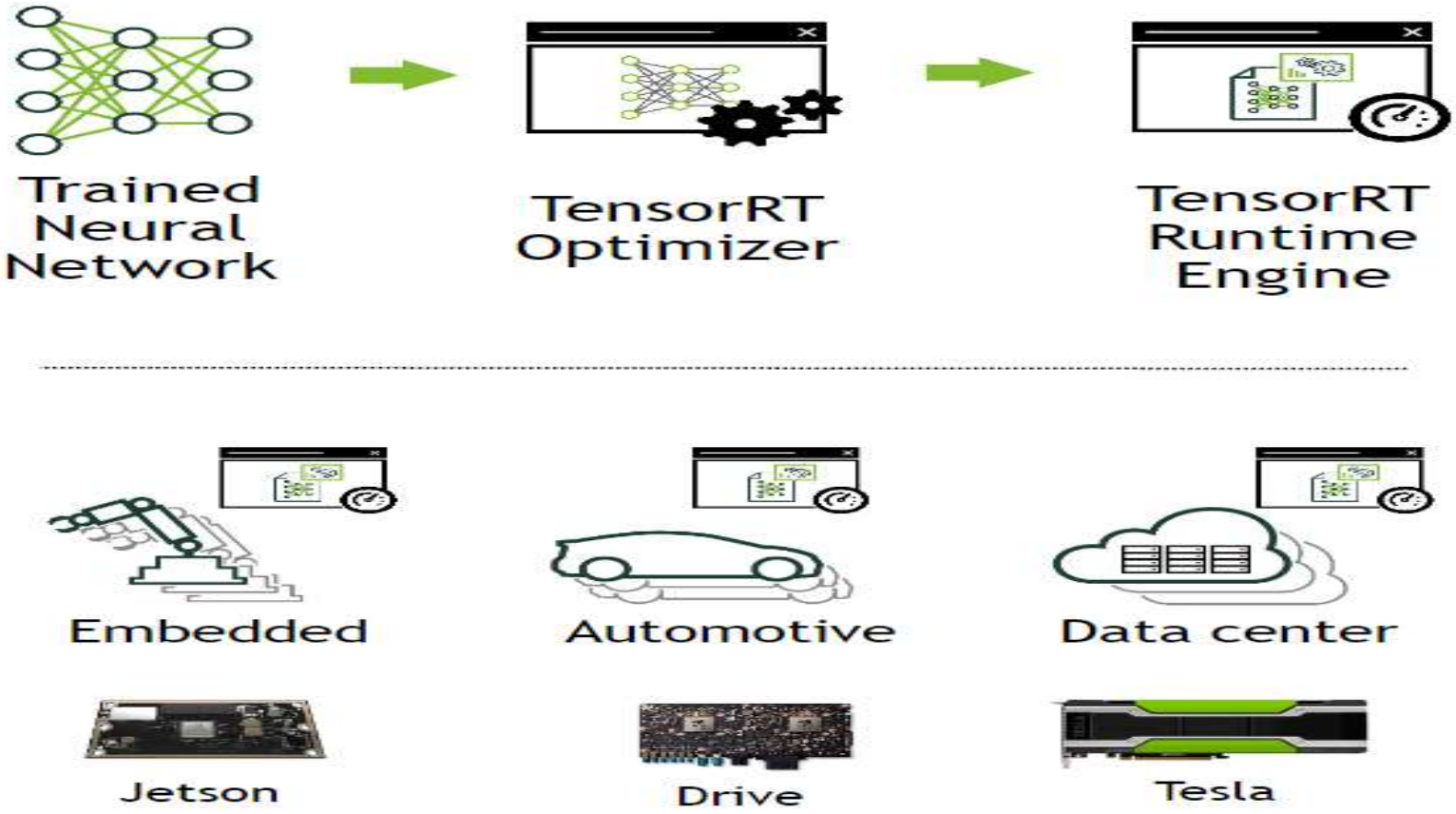
Nvidia trt_pose 모델의 구현

Nvidia trt_pose 란

trt_pose는 NVIDIA Jetson에서 실시간 포즈 추정을 가능하게 하는 것을 목표로 함
얼굴 어깨 팔 다리와 같은 주요 키포인트 부분을 학습하여 표현할 수 있다

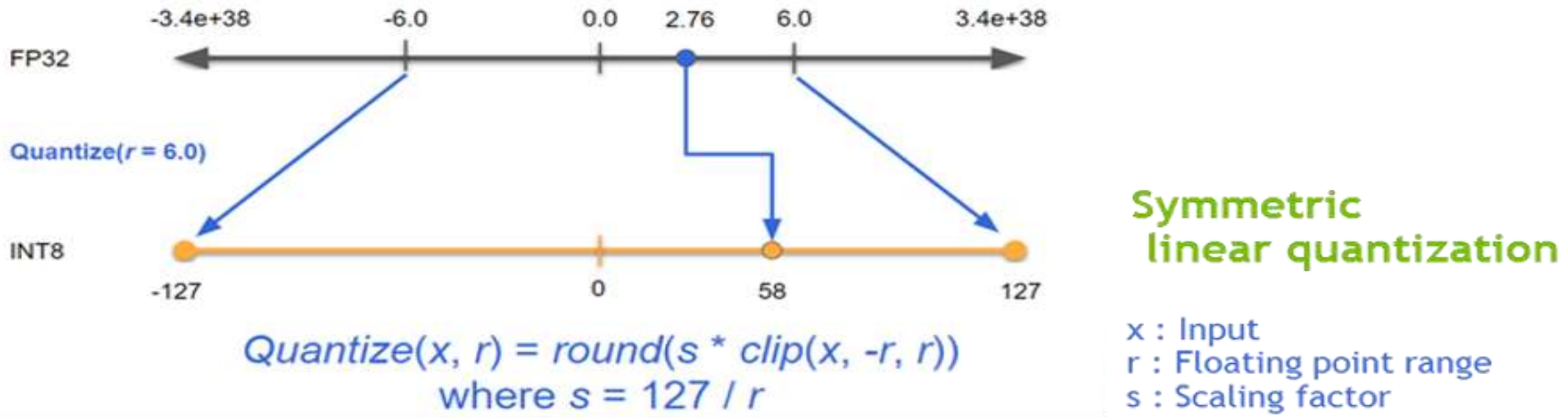
Nvidia trt_pose 모델의 구현 : TensorRT

TensorRT란



TensorRT는 학습된 딥러닝 모델을 최적화하여 NVIDIA GPU 상에서의 추론 속도를
수배 ~ 수십배 까지 향상시켜 딥러닝 서비스를 개선하는데 도움을 줄 수 있는 모델 최적화 엔진이다

Quantization & Precision Calibration (양자화 및 정밀도 캘리브레이션)

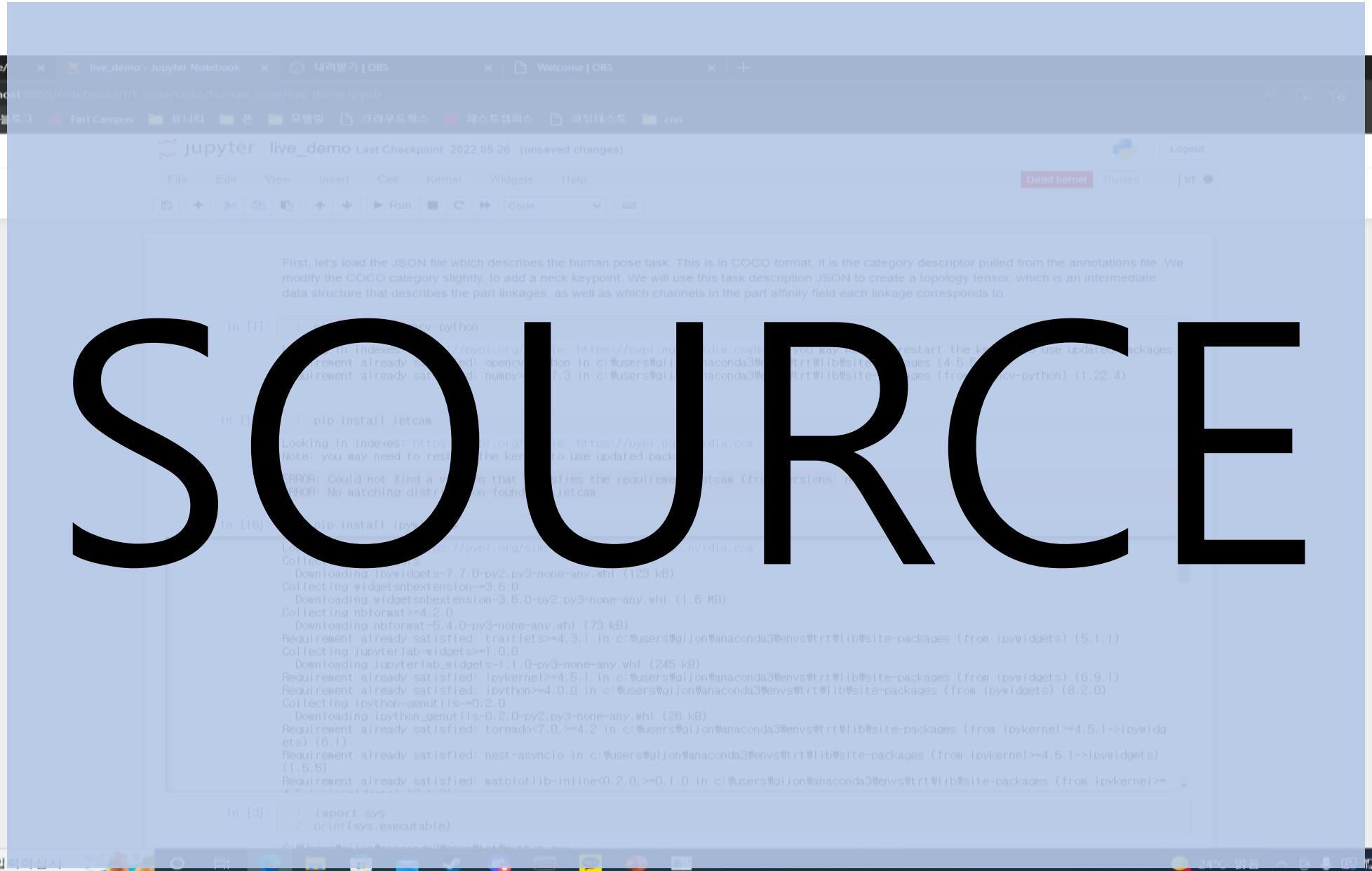


딥러닝의 학습 및 추론에서 정밀도(Precision)를 낮추는 일은 거의 일반적인 방법이 되었다.
낮은 정밀도를 가지는 신경망일수록 데이터의 크기 및 가중치들의 bit 수가 작기 때문에
더 빠르고 효율적인 연산이 가능하다.
이를 위한 양자화 기법중 TensorRT는 Symmetric Linear Quantization 을 사용하고 있다.

Nvidia trt_pose 모델의 구현 : 코드

대구대학교 DAEGU UNIVERSITY

코드



Nvidia trt_pose 모델의 구현 : 코드

대구대학교 DAEGU UNIVERSITY

코드

SOURCE

리눅스 환경에서 만들어진 코드를
윈도우로 구현하기 위한
오류 수정 및 최적화 작업 완료

젯슨 카메라 환경에서 구동하기
위한 코드를 윈도우 웹캠으로
구동하기 위한 변환작업 완료

Nvidia trt_pose 모델의 구현 : 코드

대구대학교 DAEGU UNIVERSITY

코드 동작 영상

진행현황 및 향후 계획

대구대학교 DAEGU UNIVERSITY

진행현황

- YOLO 와 Nvidia trt_pose 모델 비교
- 데이터 추가 수집

향후계획

- 최적 모델 도출
  - 모델의 하이퍼 파라미터 변경(계층 개수 및 계층별 필터개수 등)이 변경이 가능하도록 모델 코드 변경
    → 학습 성능 최대화를 위한 적합한 하이퍼 파라미터 설정 (~ 6월)
  - 새로운 모델인 Nvidia trt_pose 가 YOLO와 비교해서 어떤 장단점이 있는지 확인(~6월)
  - 좀 더 자세한 포즈 분석을 위한 추가 모델 도입 예정(~7월)
- 모델의 성능 분석
  - 계층 필터개수, 필터 크기 등의 설계사항이 성능에 미치는 영향 분석 (~6월)

4. 모델의 훈련 및 검증

- Yolo 학습 데이터 세트 및 학습 방법
- Yolo v5 학습 SW 구현
- Nvidia trt_pose 구현
- 진행현황 및 향후계획

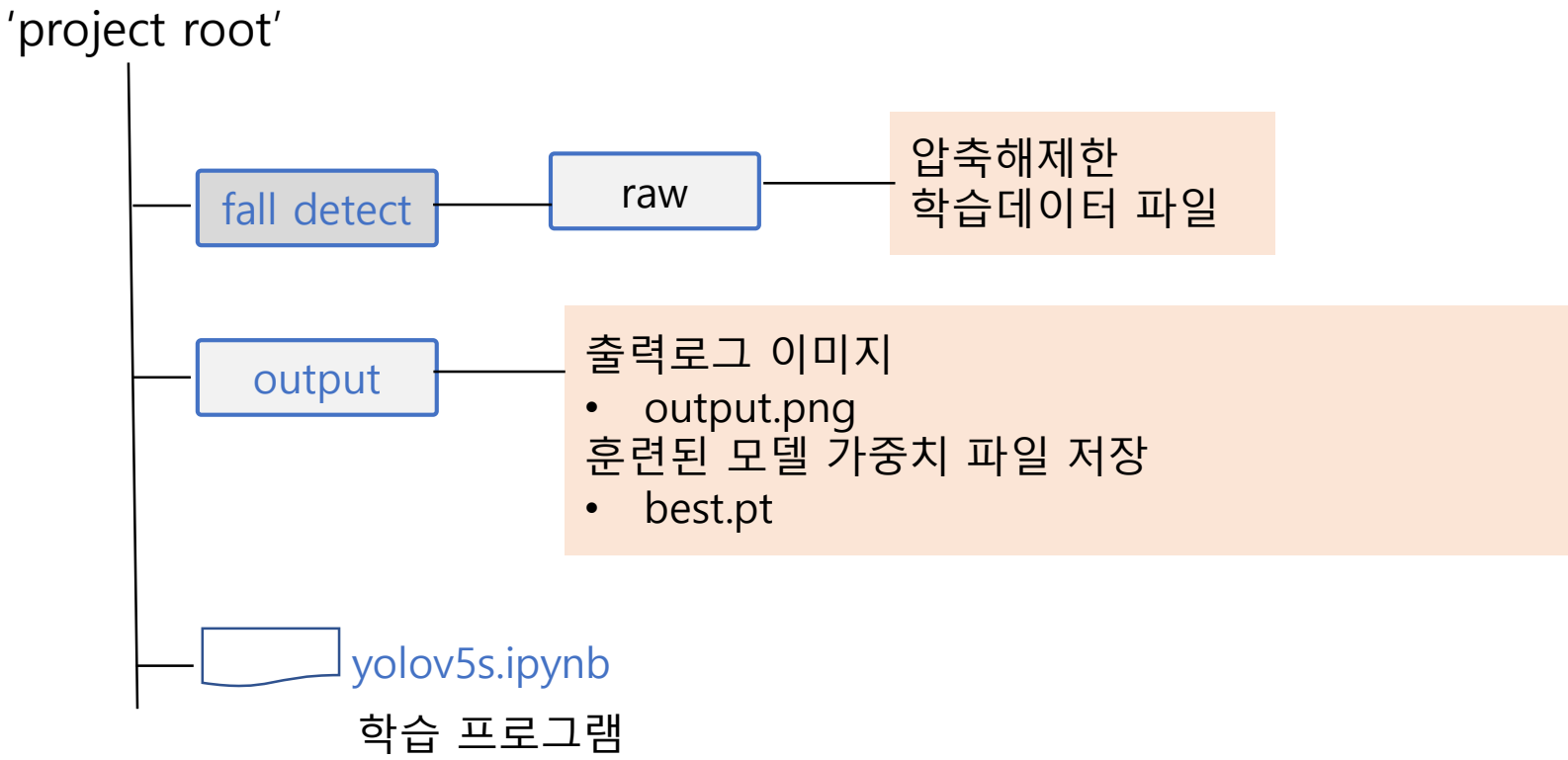
Yolo 학습 데이터 세트 및 학습 방법

대구대학교 DAEGU UNIVERSITY

개요

구분	제목	비고
학습 구현내용	▪ 훈련 과정만 구현 • 테스트와 검증 과정은 특별히 필요가 없어 따로 구현하지 않았음	
훈련데이터	동영상 2200 클립, 이미지 220만장	
배치크기	128	설정사항
훈련에포크	100	설정사항
로깅	▪ 1에포크마다 로깅 • G와 D의 손실(오류) 화면 출력 • Box, obj, class, label 추론 값을 제공하며, • 1에포크 학습마다 Precision, Recall, mAP 값을 제공. ▪ 훈련을 마치면 G와 D의 학습된 가중치를 파일에 저장	저장디렉토리 C:₩Users₩OWNER₩Desktop₩yolo₩yolov5₩runs₩train₩yolov5s_results4₩weights₩

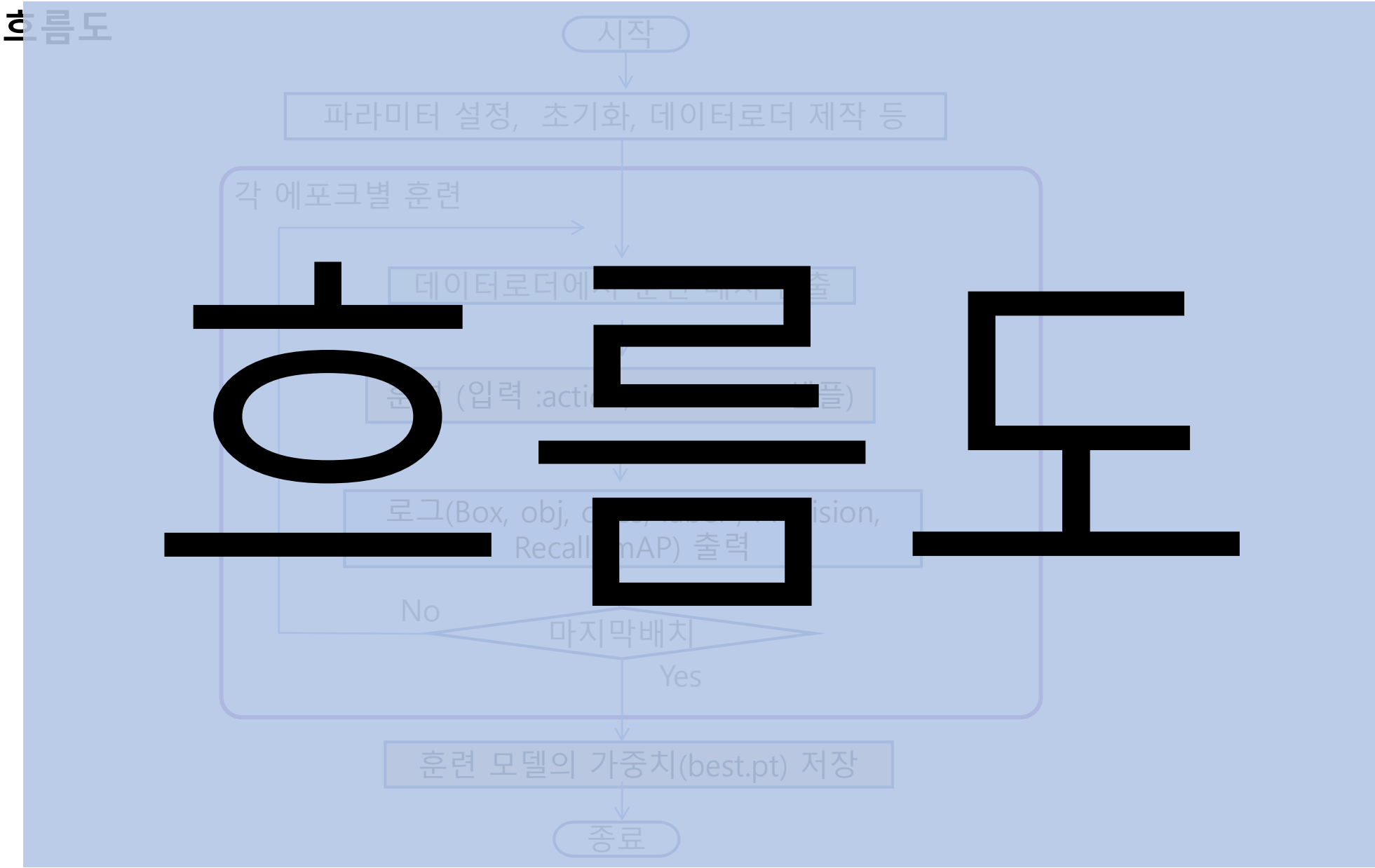
파일 및 폴더 구조


'project root'
fall detect
raw
압축해제한
학습데이터 파일
output
출력로그 이미지
• output.png
훈련된 모델 가중치 파일 저장
• best.pt
yolov5s.ipynb
학습 프로그램

Yolo v5 학습 SW 구현

대구대학교 DAEGU UNIVERSITY

- 전체 흐름도



진행현황 및 향후계획

대구대학교 DAEGU UNIVERSITY

진행 현황

- 훈련부 코드 완성 (75%)

향후 계획

- 로그출력 오류 해결(~6월말)
  - 학습시, 로그 출력부분에 오류가 발생하는 버그가 있음.
  - 원인을 찾는 중에 있음.
- 학습 정확도 증가 (~6월말)
  - 현재 간단한 모델은 구현이 가능하나, 정확한 추론이 불가능함.
  - 스켈레톤을 통해 자세를 추정하거나, 배치사이즈, 에포크 등 다양한 파라미터를 조정하여 학습 정확도를 증가시킬 예정.

5. 낙상 발견 및 대응 SW

- 낙상 발견 및 대응 SW 개요
- 낙상 발견 및 대응 SW 구현
- 진행현황 및 향후계획

낙상 발견 및 대응 SW 개요

낙상 발견 및 대응 SW 개념

① CCTV 혹은 카메라를 통한 실시간 낙상 영상 입력

② 받아온 영상을 디렉토리로 전송

학습된 Yolo v5 모델

③ 낙상 판별 시, 판별 영상 및 낙상 판별 가능한 문자열 전송

낙상 감지 SW

④ 낙상 판별 이후 n초(시간 정해야함) 동안 그대로 자세를 유지시, 경보

음성 신호

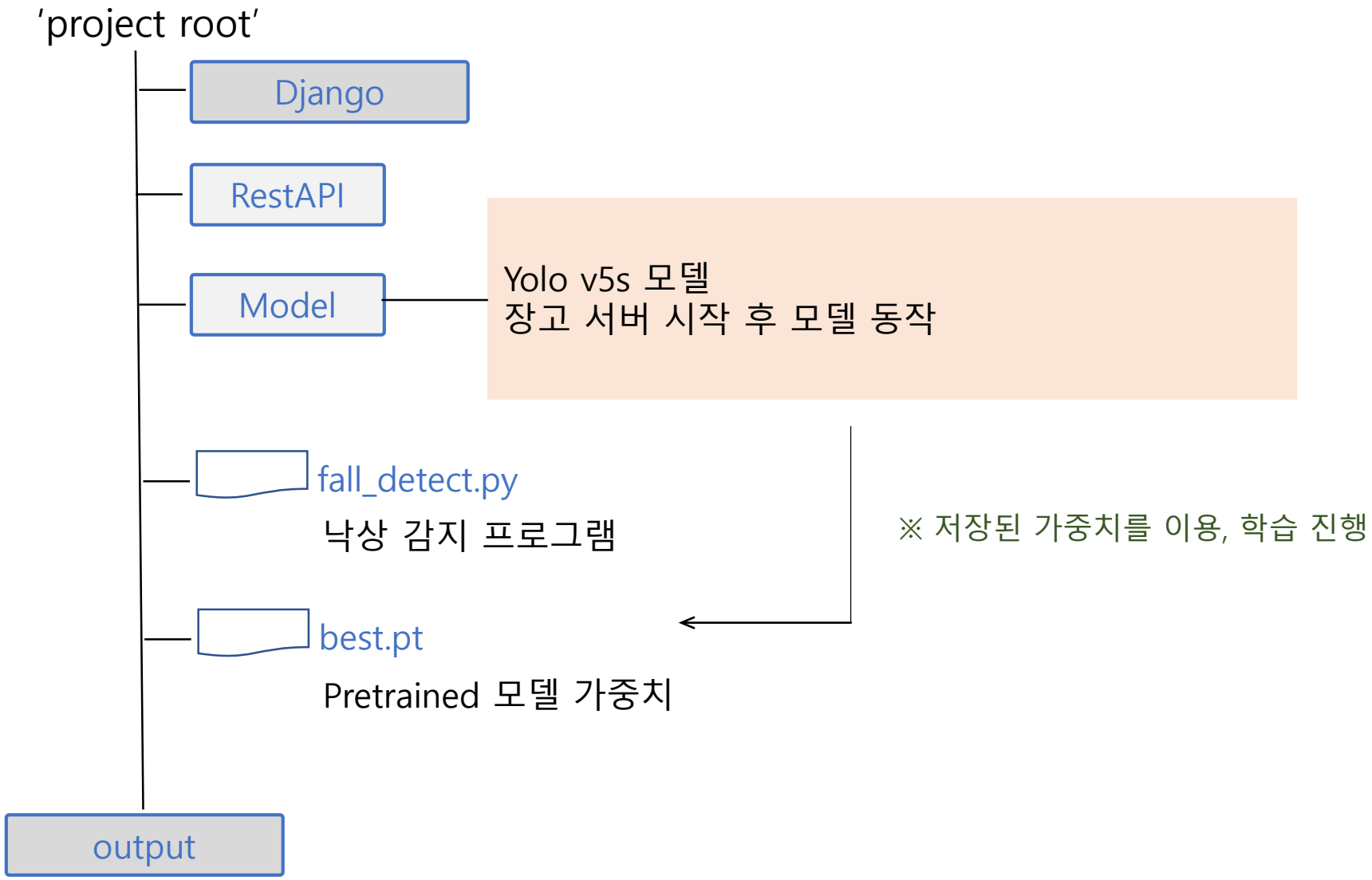
⑤ TTS, 부저 알림등으로 주위에 위험 사항 전파 및 119 신고(미정)

낙상 판별이 안되면,
실시간으로 영상 입력 받으며,
실시간으로 모델에서 추론 반복

구성요소 및 기능

구성요소	기능	비고
Yolov5를 이용한 낙상 감지 모델	▪ 낙상 감지 프로그램 ▪ 외부에서 받은 영상을 실시간 추론하여 낙상 감지 진행.	
낙상 감지 대응 SW	▪ 낙상 감지 모델에서 제공한 문자열 데이터를 입력받아 낙상 상태 감지 및 일정 시간동안 변화가 없을 시, 구조 요청 신호 발생 ▪ TTS 음성신호를 통한 상황 전파 ▪ 서버를 이용하여 실시간으로 웹 페이지에 낙상 상태 업로드	

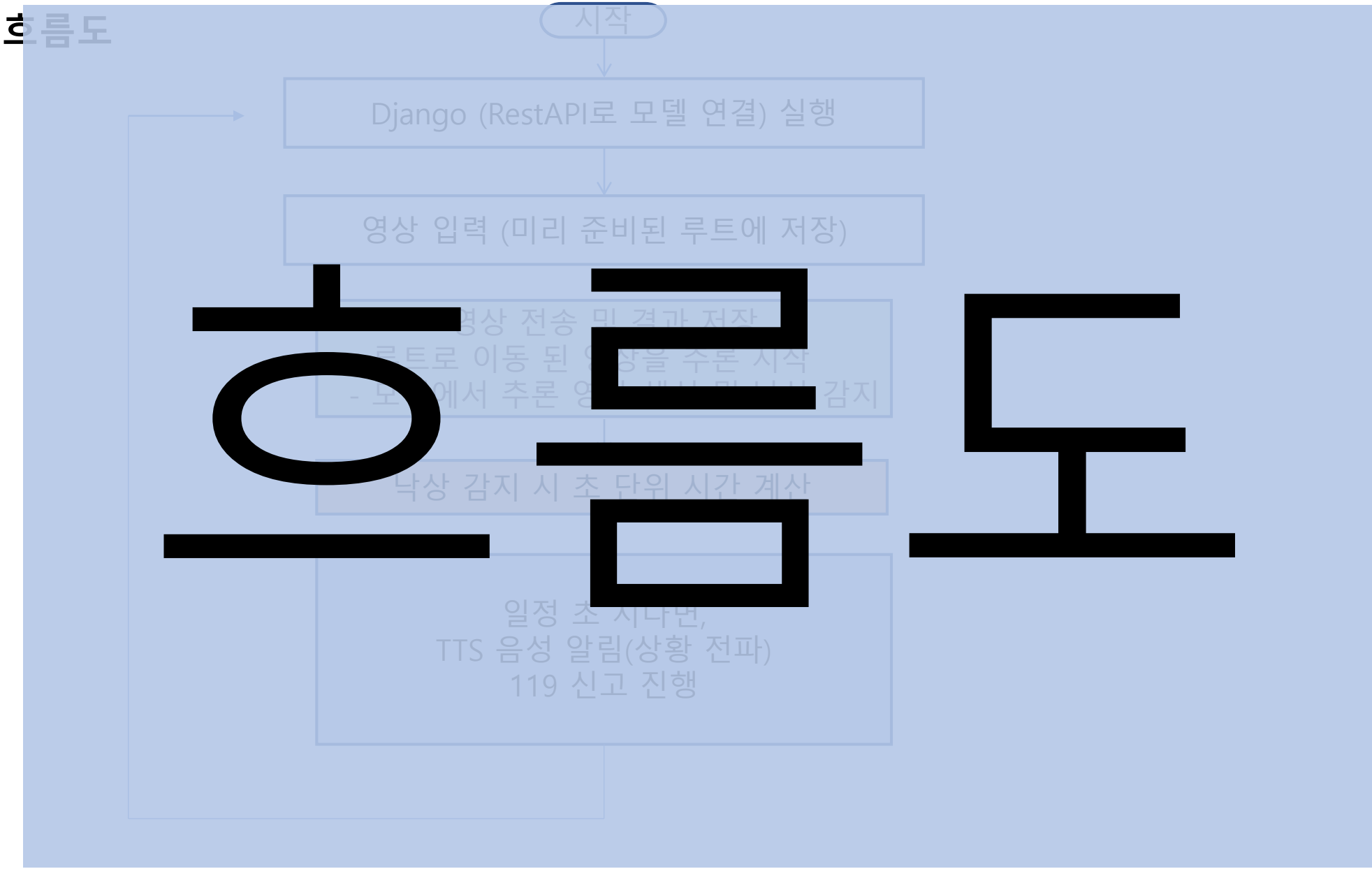
파일 및 폴더 구조


'project root'
Django
RestAPI
Model
Yolo v5s 모델
장고 서버 시작 후 모델 동작
fall_detect.py
낙상 감지 프로그램
※ 저장된 가중치를 이용, 학습 진행
best.pt
Pretrained 모델 가중치
output

낙상 발견 및 대응 SW 구현

대구대학교 DAEGU UNIVERSITY

- 전체 흐름도



Django 활용

- 웹 프레임워크
  - Python 기반 웹 프레임 워크
  - Django에서 지원하는 debug 용 local server를 이용해 테스트 할 예정

- REST API
  - 월드 와이드 웹(www)과 같은 분산 하이퍼 미디어 시스템을 위한 소프트웨어 개발 아키텍처의 한 형식
  - HTTP URI(Uniform Resource Identifier)를 통해 자원(Resource)을 명시하고, HTTP Method(POST, GET, PUT, DELETE)를 통해 해당 자원에 대한 CRUD Operation을 적용
  - REST API를 이용해 모델 동작 여부를 클라이언트가 결정하고 모델 동작 결과를 웹페이지로 전송해 시각화 할 예정

진행현황 및 향후계획

대구대학교 DAEGU UNIVERSITY

진행 현황

- 낙상 감지 프로그램 코드 (~6월말)
- 감지 시간 카운트 다운 및 감지 문자열 전송(~6월말)

향후 계획

- 서버 구축 및 모델 연동
  - 서버 구축 (완)
  - 웹 페이지 제작 (진행중)
  - Restful API를 이용한 모델 동작 및 데이터 입출력(진행중)

6. 별첨

- 필요시 행정 또는 기술 분석 등 추가 자료 별첨 요망 !!

Q & A

More information?