

Visual-Sound Accident Protector

Worker Protection Service

Team. Eye Scream(No.4)

이현수, 조재훈, 최이윤

최근에 대두된 사회적 문제



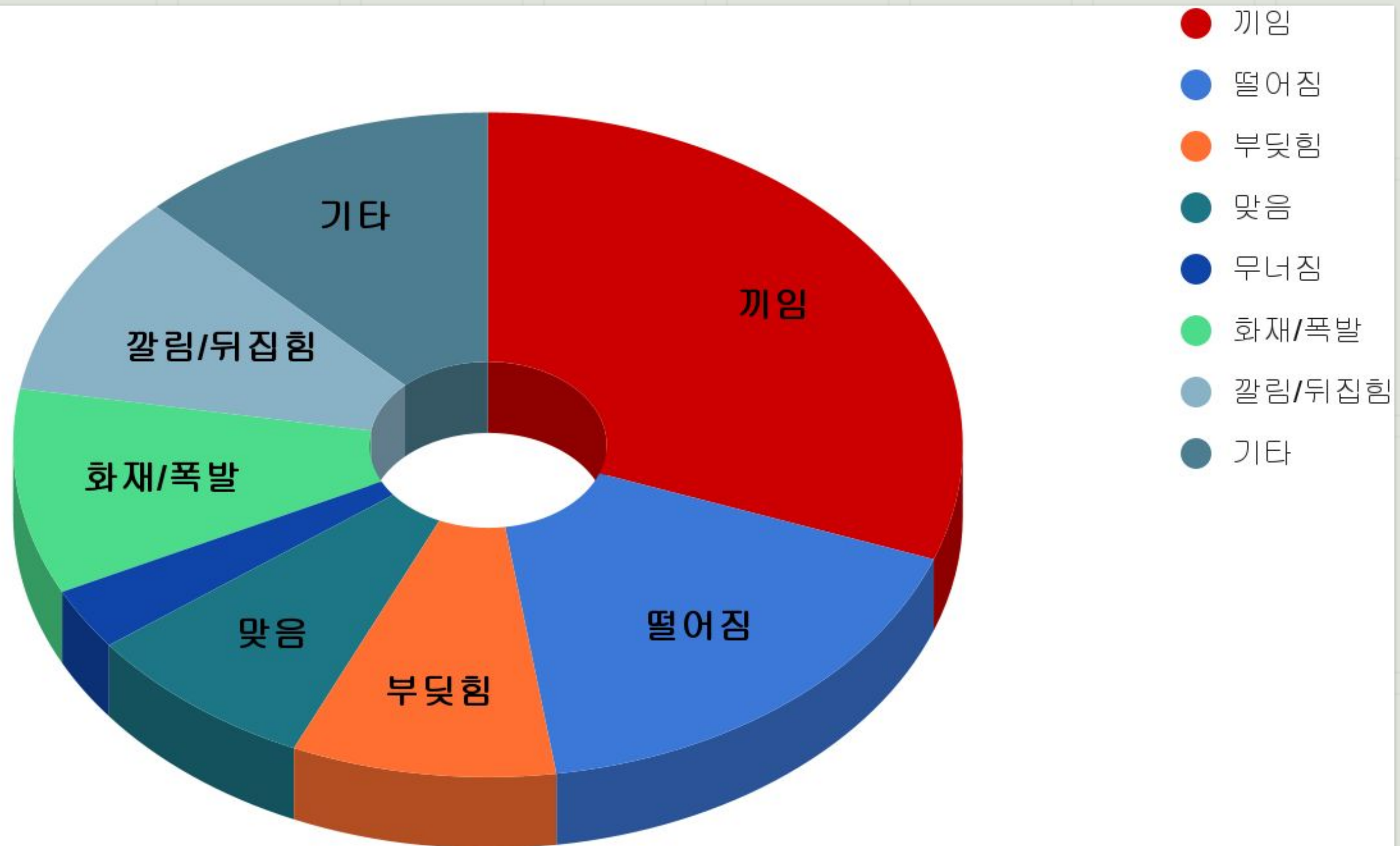
지난 15일, SPC 제빵공장 노동자가 교반기에 몸이 끼어 사망한 사건



다양한 요인 때문에 사고 발생 대처가 지체되는 경우가 많음

- 현장 휴대폰 반입 금지
- 119신고 금기시하는 암묵적 룰
- 수직적인 회사 보고 절차

제조업 사고성 사망자의 발생 유형 비율





대표적 유형인 끼임, 떨어짐, 부딪힘의 경우 위험한 장소, 물체에 접근하여 발생

=> 영상 인식 및 경고를 통해 안전사고 사전 방지가 어느정도 가능하다고 판단

산업 재해를 줄이는 방법은?

사고 예방 및 발생 대응 능력 중요

‘시각과 소리에 관점을 맞춘 아이디어’

시각을 활용한 사고 예방 및 대처

- 사고 예방

- CCTV & CV를 활용한 사람 및 물체 인식을 통해 위험한 상황이 생기기 전에 사전에 경고

- 사고 발생 시

- 사고 유형 판별 후 적절한 방식으로 응급 의료 기관과 119에 연락 조치
- 동료 작업자들에게 사고 발생 알림 및 응급 대처 방법 전달

소리를 이용한 신속한 사고 감지 및 대처

- 작업자의 휴대폰 어플리케이션이나 작업장 내 설치된 장치로 소리 인식
- 비명, 괴음, 폭발음 등 사람이나 기계의 비정상적인 소리 감지 시 기계 작동 중단 및 주변 작업자, 관리자에게 사고 발생 알림 및 응급 대처 방법 전달
- 사고 유형 판별 후 유형에 따라 적절한 방식으로 응급 의료 기관, 112, 119 등에 자동 연락 조치

비즈니스 모델

비즈니스 모델

가치전달

시장에서 고객에게
보이는 부분

가치생산

기업이 가치를
만들어내는 방법

가치 생산

가치 전달

<div><div>핵심 파트너</div><div>소방서 병원 CCTV 공급업체</div></div>			<div><div>핵심 활동</div><div>학습 데이터 구축 / 가공</div></div>		
			<div><div>핵심 자원</div><div>카메라 (모션데이터) 음성데이터</div></div>		
			<div><div>가치 제안</div><div>안전성 공익성</div></div>		
			<div><div>고객 관계</div><div>SW하나로 투자금 대비 고효율성</div><div>HW가 있는 경우 추가 구매 불필요</div><div>꾸준한 데이터 학습으로 지속적 피드백 가능</div><div>채널</div><div>모바일 어플리케이션 웨어러블 SW</div></div>		
<div><div>고객</div><div>산업체 제조업, 공장, 물류센터</div></div>					
<div><div>비용</div><div>서버 유지비 CCTV업체 or 카메라, 마이크 업체와의 제휴비용 데이터 클렌징 및 가공 비용</div></div>			<div><div>수익</div><div>수집되는 DATA SW사용(월 구독 서비스)</div></div>		

고객 관계 및 제공 채널

APP과 웨어러블 SW사용

HW보유시 추가 장치 필요X
반복적 학습으로 지속적 피드백 가능

안전성 보장
빠른 대처로 피해 최소화

가치 제안

가치 전달

산업체(공장), 공공기관, 기업

고객

수익

수집되는 DATA
SW사용료(월 구독 서비스)

가치 생산

핵심 파트너

소방서, 병원,
CCTV공급업체

핵심 활동

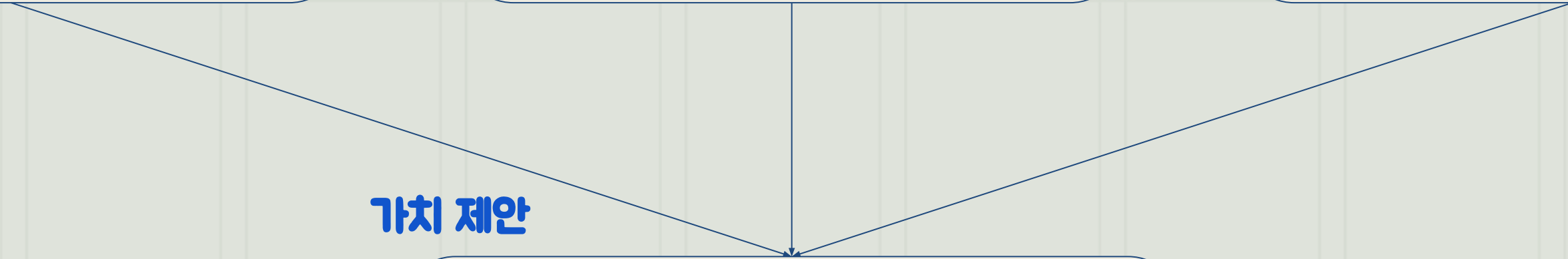
학습데이터 구축 / 가공

비용

서버 유지비,
CCTV업체 or 카메라,
마이크업체 제휴비용,
데이터 클렌징 및 가공비용

가치 제안

안전성, 공익성



핵심 파트너 소방서 병원 CCTV 공급업체	핵심 활동 학습 데이터 구축 / 가공	가치 제안 안전성 공익성	고객 관계 SW하나로 투자금 대비 고효율성 HW가 있는 경우 추가 구매 불필요 꾸준한 데이터 학습으로 지속적 피드백 가능	고객 산업체 제조업, 공장, 물류센터
비용 서버 유지비 CCTV업체 or 카메라, 마이크 업체와의 제휴비용 데이터 클렌징 및 가공 비용	핵심 자원 카메라 (모션데이터) 음성데이터		채널 모바일 어플리케이션 웨어러블 SW	
			수익 수집되는 DATA SW사용(월 구독 서비스)	

데이터 형태 - 음성

(필요한 데이터, 선택 이유)

위급상황 음성/음향 데이터 - AIHub

- 위급 상황 음성/음향 데이터
 - 성별 : 남성, 여성
 - 연령대 : 유아, 청소년, 노인, 기타
 - 상황 : 치안안전, 소방안전, 자연재해, 사고활동, 일반(위급), 일반(정상)
- 위급 상황 AI 학습용 데이터셋
 - 16개 클래스(중분류) 1,000건 이상의 상황·3,500시간
 - 음성/음향 단일 데이터셋, 음성/음향 복합 데이터셋

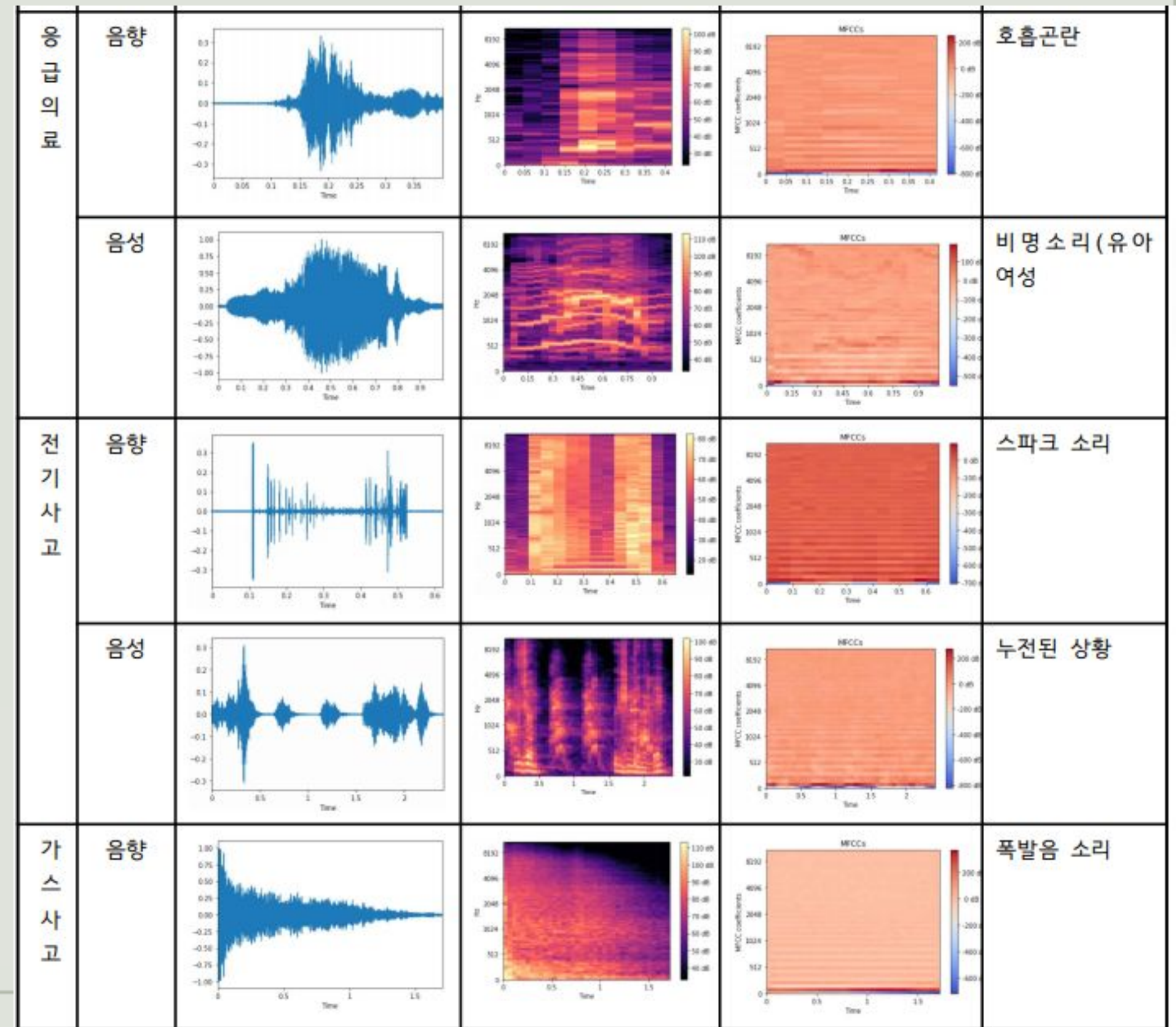
산업 현장에서 발생하는 다양한 종류의 위급 상황을 인지하기 위해
AIHub에서 제공하는 위급상황 음성/음향 데이터셋 활용

위급상황 음성/음향 데이터 - AIHub

- 공개 데이터
 - 공개 오디오 구축사이트
 - DCASE, AudioSet, UrbanSound, FreeSound

- 크라우드 소싱
 - 크라우드 워커 : 직접 녹음(가정), 수집 데이터(실내, 실외), 상황 연출(녹음실)

※ 저작권/개인정보 이슈 해결된 데이터 수집 : 녹음 동의서 활용




데이터 수집 방법 - 음성

위급상황 음성/음향 데이터 - AIHub

- AIHub에서 제공하는 위급사고
음성/음향 데이터 다운로드

- Youtube에서 소음 오디오 추출 및
다운로드

AI 학습용 다운로드

-  를 클릭하시면 하위 폴더와 파일 목록을 확인할 수 있습니다.
- 전체 파일을 한번에 다운로드 받고자 할 경우는 [전체 다운로드] 를,
일부만 선택하여 다운로드 받고자 하실 경우는 다운로드 받을 파일을 선택하신 뒤,
[선택 다운로드] 버튼을 눌러주세요.
- **주의 사항**
다운로드 진행 중 실행 창을 닫으면 다운로드가 자동적으로 중단됩니다.
이 때 다운로드 이력도 함께 삭제되기 때문에 파일을 다시 다운로드 받고자 할 경우
처음부터 다시 다운로드 받으셔야 합니다.

<input type="checkbox"/>	NAME ▲	SIZE ▲
<input checked="" type="checkbox"/>	[라벨]1.강제추행(성범죄).zip	18MB
<input checked="" type="checkbox"/>	[라벨]9.가스사고.zip	19MB
<input checked="" type="checkbox"/>	[라벨]12.태풍-강풍.zip	20MB
<input checked="" type="checkbox"/>	[라벨]13.지진.zip	23MB
<input checked="" type="checkbox"/>	[라벨]15.실내.zip	23MB
<input checked="" type="checkbox"/>	[라벨]11.붕괴 사고.zip	25MB
<input checked="" type="checkbox"/>	[라벨]8.전기사고.zip	32MB
<input checked="" type="checkbox"/>	[라벨]14.도움요청.zip	34MB

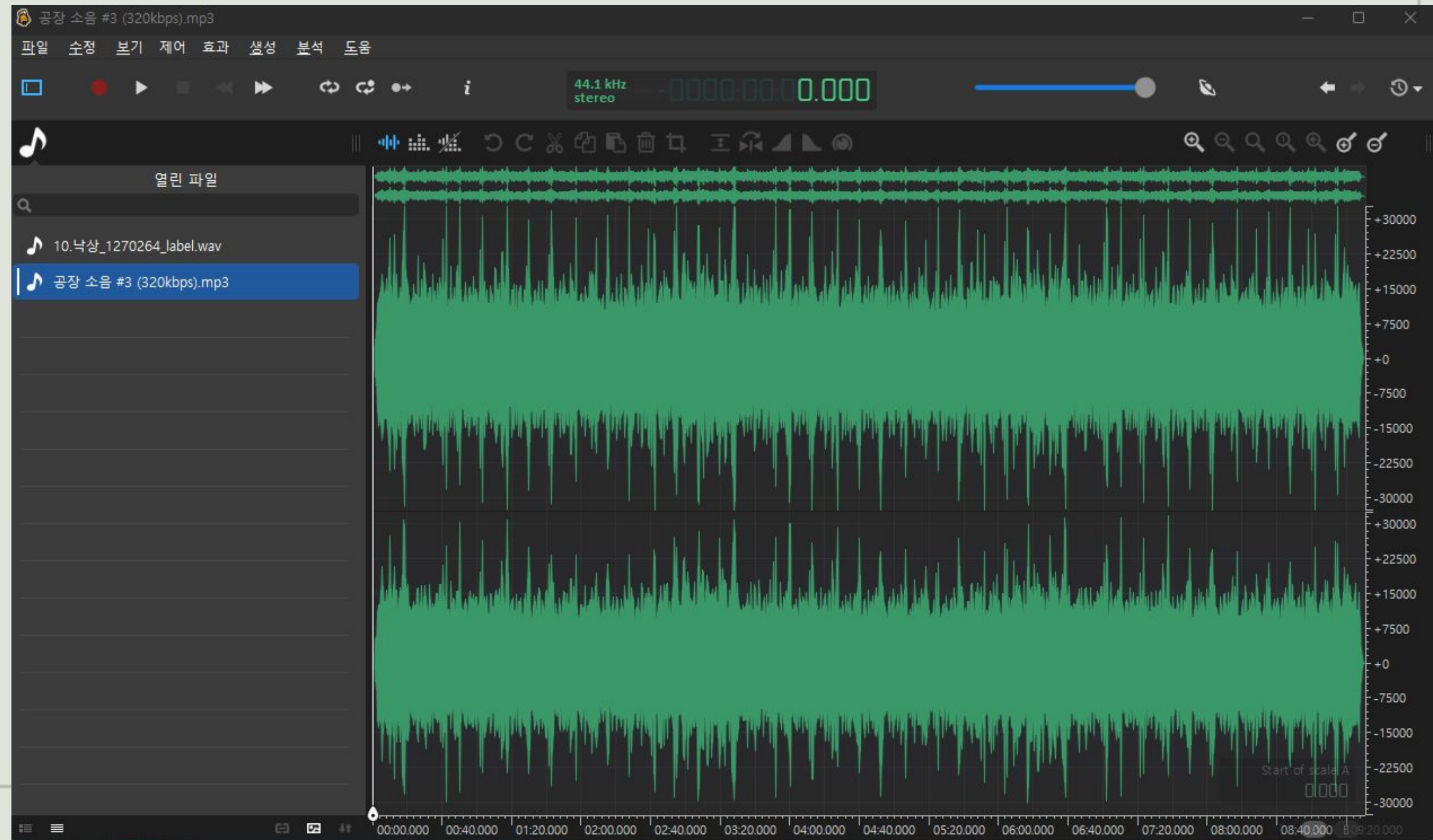
선택 다운로드

전체 다운로드

위급상황 음성/음향 데이터 - Ocenaudio

- 공사장/공장 소음 등 소리 인식을 어렵게 하는 요소들을 해결하기 위해, 소음 파일을 합성시키는 방식으로 데이터를 가공

- 오디오 파일 합성을 위해
ocenaudio 프로그램 사용



인공지능 모델 형태 - 음성

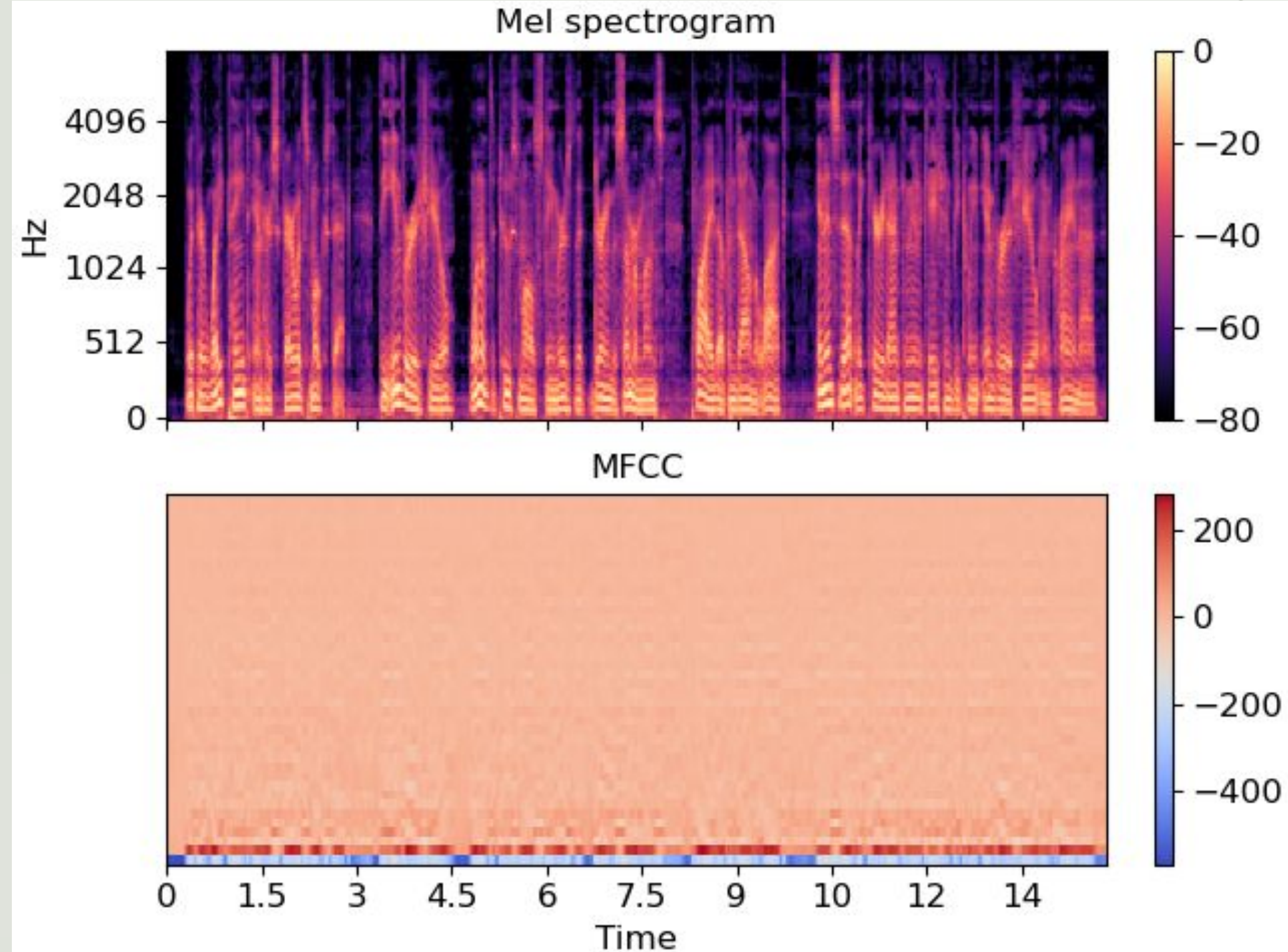
인공지능 모델 및 알고리즘 - Resampling

- Librosa를 사용해 오디오 리샘플링

```
[9] 1 def get_signal(file_name):  
2     audio, sample_rate = librosa.load(file_name, res_type='kaiser_fast')  
3     return audio, sample_rate  
4  
5 X_train_signal = []  
6 for audio in X_train:  
7     signal, sr = get_signal(audio)  
8     X_train_signal.append([signal, sr])  
9
```

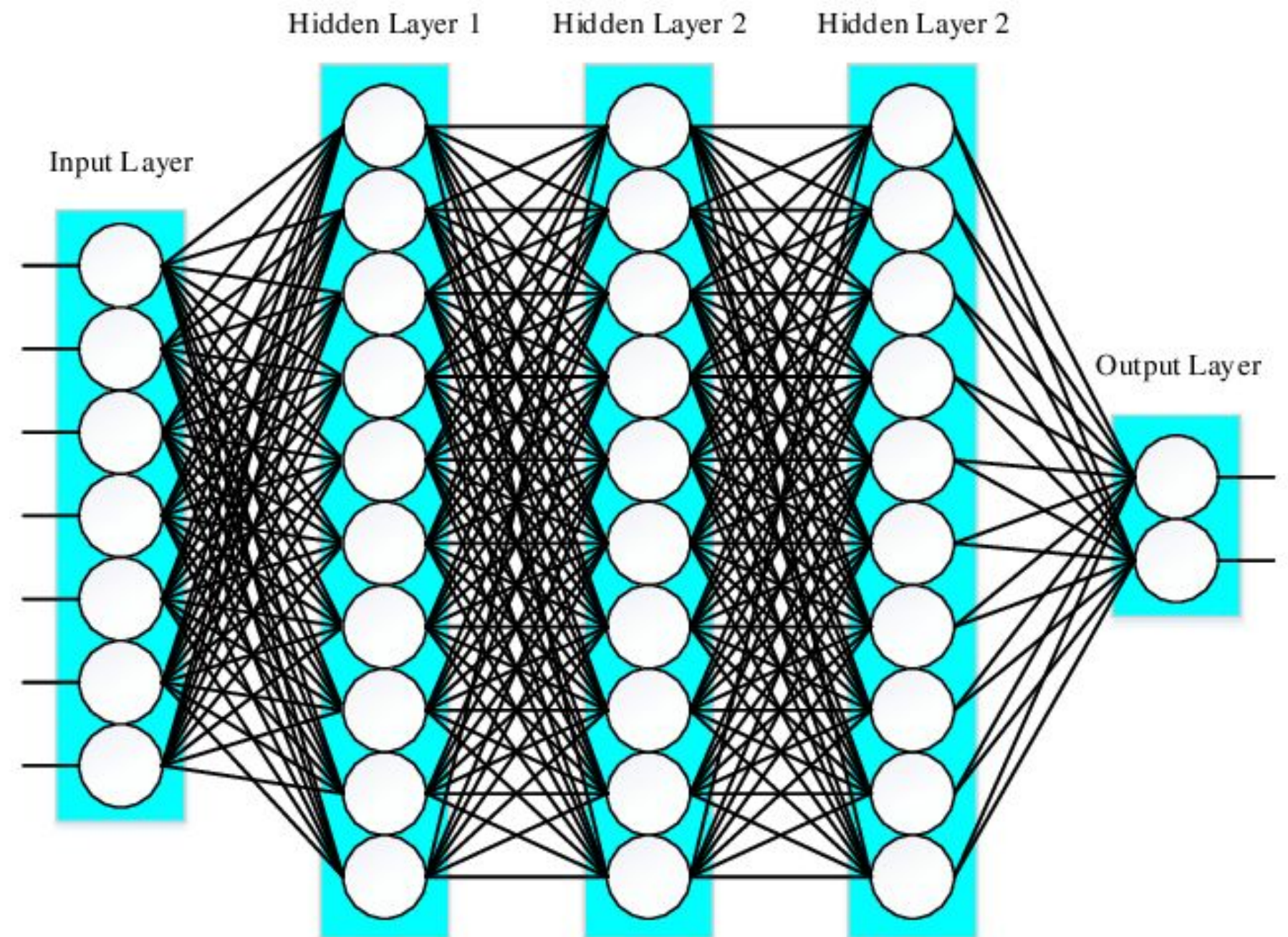

인공지능 모델 및 알고리즘 - MFCC

- Feature 추출을 위해 MFCC 알고리즘 사용
- MFCC: 소리의 특징을 추출하는 기법.
일정 구간씩 나누어 각 구간에 대한
스펙트럼을 분석하여 특징을 추출하는 기법
- 여러 차례 테스트 후 60 구간으로 나누기로
결정



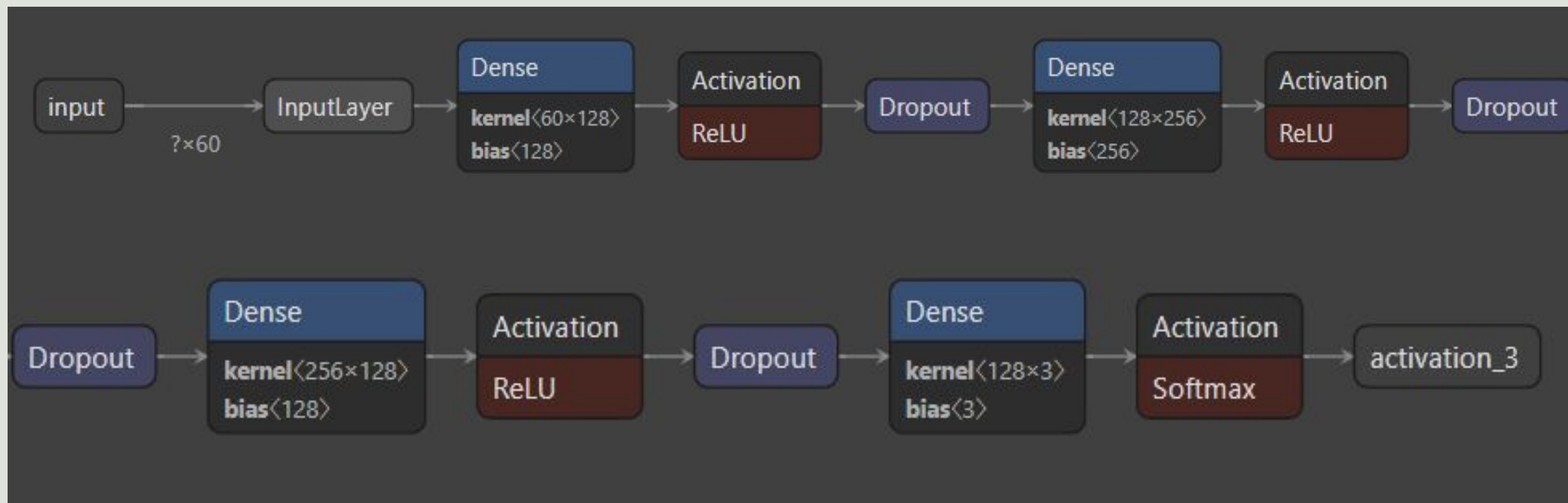
인공지능 모델 및 알고리즘 - FCN

- 앞서 추출한 60개의 Feature를 input layer에 넣고, 3개의 hidden layer를 거쳐 최종적으로 classification을 할 수 있도록 구성

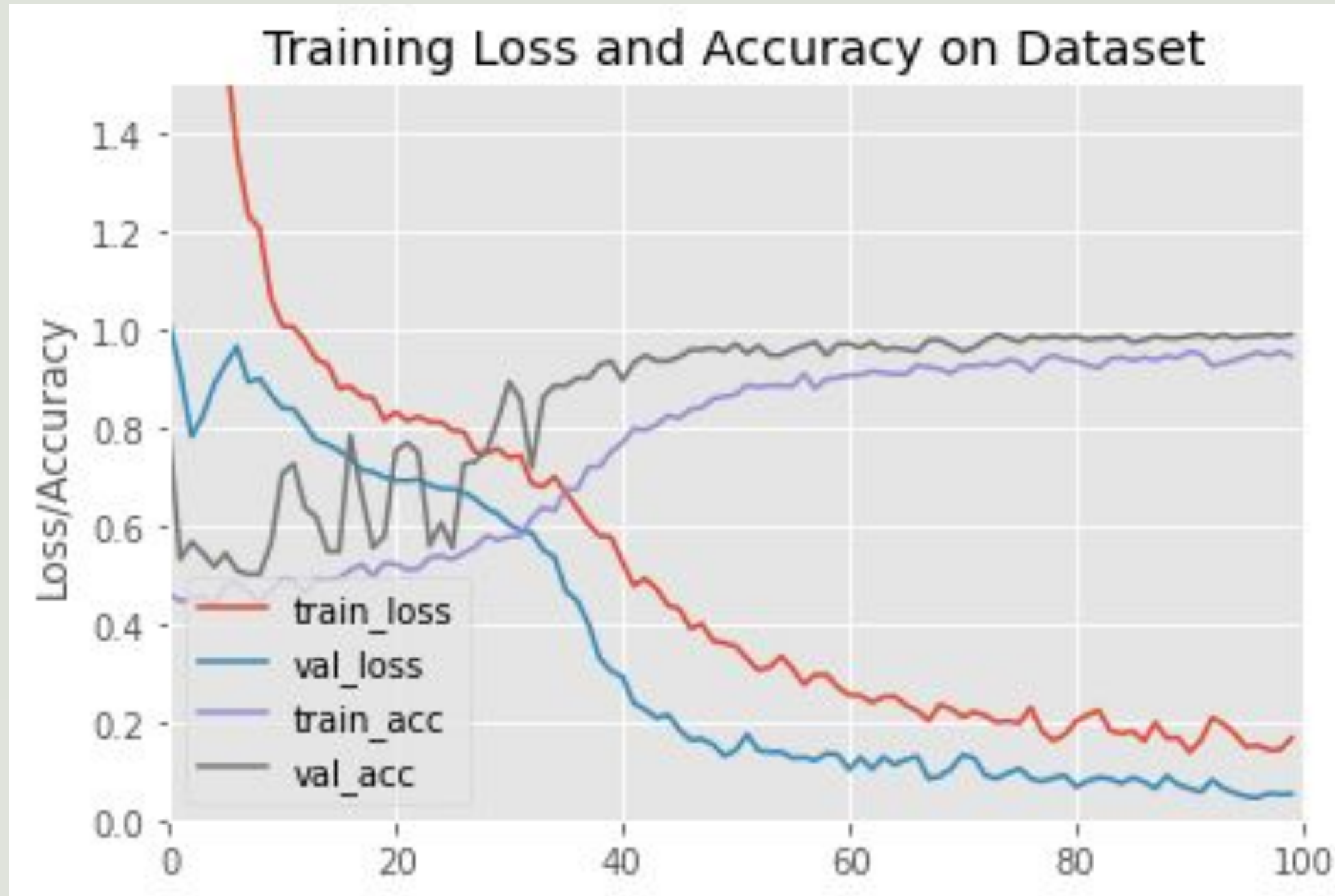


인공지능 모델 및 알고리즘 - 모델 구조

- Netron 프로그램으로 시각화



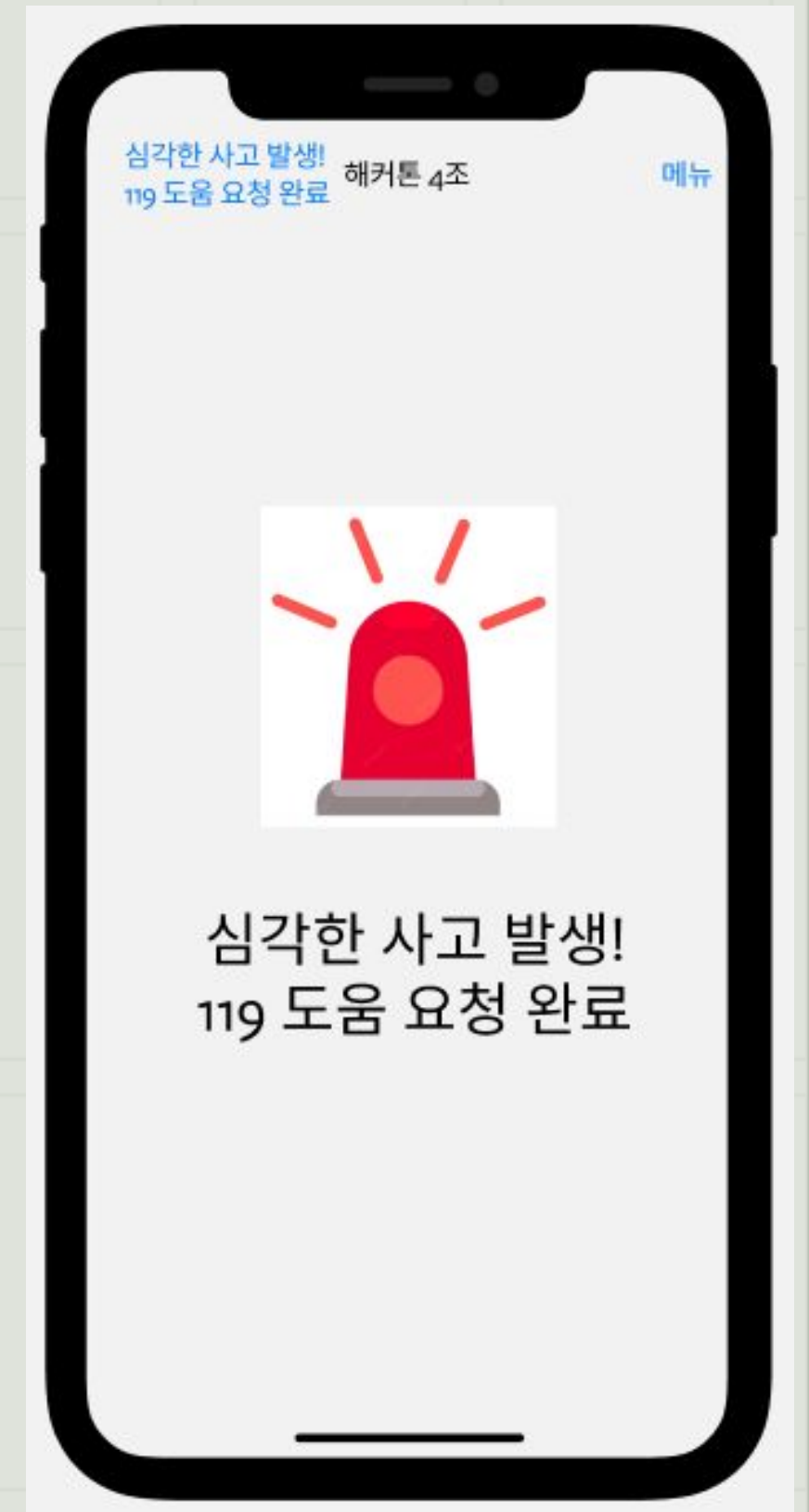
인공지능 모델 및 알고리즘 - 학습



- 소음을 합성한 사고 상황 vs vs 소음을 합성한 다른 유형의 사고 상황 vs 소음만 있는(안전한) 상황 분류

프론트 UI/UX

프론트 UI/UX

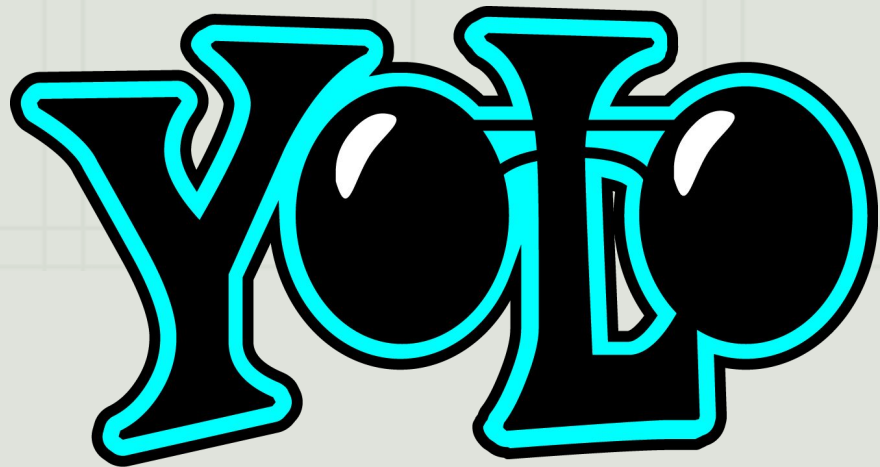


데이터 형태 - 영상

(필요한 데이터, 선택 이유)

데이터 형태 - 필요 데이터

- 사람과 공장 기계들의 구분을 위한 이미지나 영상 데이터
 - > 머신러닝 학습, YOLO나 Data Set
- 리얼 타임으로 작동하게 하기 위한 실시간 영상 데이터
 - > Open CV



데이터 형태 - 데이터 선택 이유

- 실시간 영상을 통해 상황을 인지하는 모델이기 때문에,
영상 내에서 객체들(사람, 기계)을 구분할 수 있도록 데이터 세트를 구성해야 한다.
- 기계의 위험 정도에 따라 안전 거리를 조정해야 하기 때문에,
여러 종류의 기계들을 구분할 수 있을만큼의 데이터가 필요하다.
- 실시간으로 정보를 받아오며 위험한 상황을 탐지해야 하기 때문에,
사용되는 공간의 실시간 영상 정보가 필요하다.

데이터 수집 방법 - 영상

데이터 수집 방법

1. 수집 방법

사람의 경우 YOLO알고리즘을 이용하여 구분하고,
기계의 경우 공장에서 자주 쓰이는 기계들의 사진을
반복 학습시켜 구분할 수 있게 한다.



그림 4: YOLO 객체 탐지 알고리즘

2. 사용 알고리즘 설명

-YOLO 알고리즘은 단일 단계 방식의 객체 탐지 알고리즘으로,
이미지를 동일한 크기의 그리드(grid)들로 나누어, 객체가 포함되어 있다고 예상되는 구역들을 분류한다.
이후 나누어진 구역들을 앵커 박스라 불리는 미리 정의된 형태(preddefined shape)의 직사각형 경계박스로
묶어 객체를 탐지한다.

-Open CV는 실시간 컴퓨터 비전을 목적으로 한 프로그래밍 라이브러리이다.
원래는 인텔이 개발하였다. 실시간 이미지 프로세싱에 중점을 둔 라이브러리이다.

인공지능 모델 형태 - 영상

인공지능 모델 알고리즘

1. YOLO 알고리즘을 통해 사람과 기계를 인식한다.

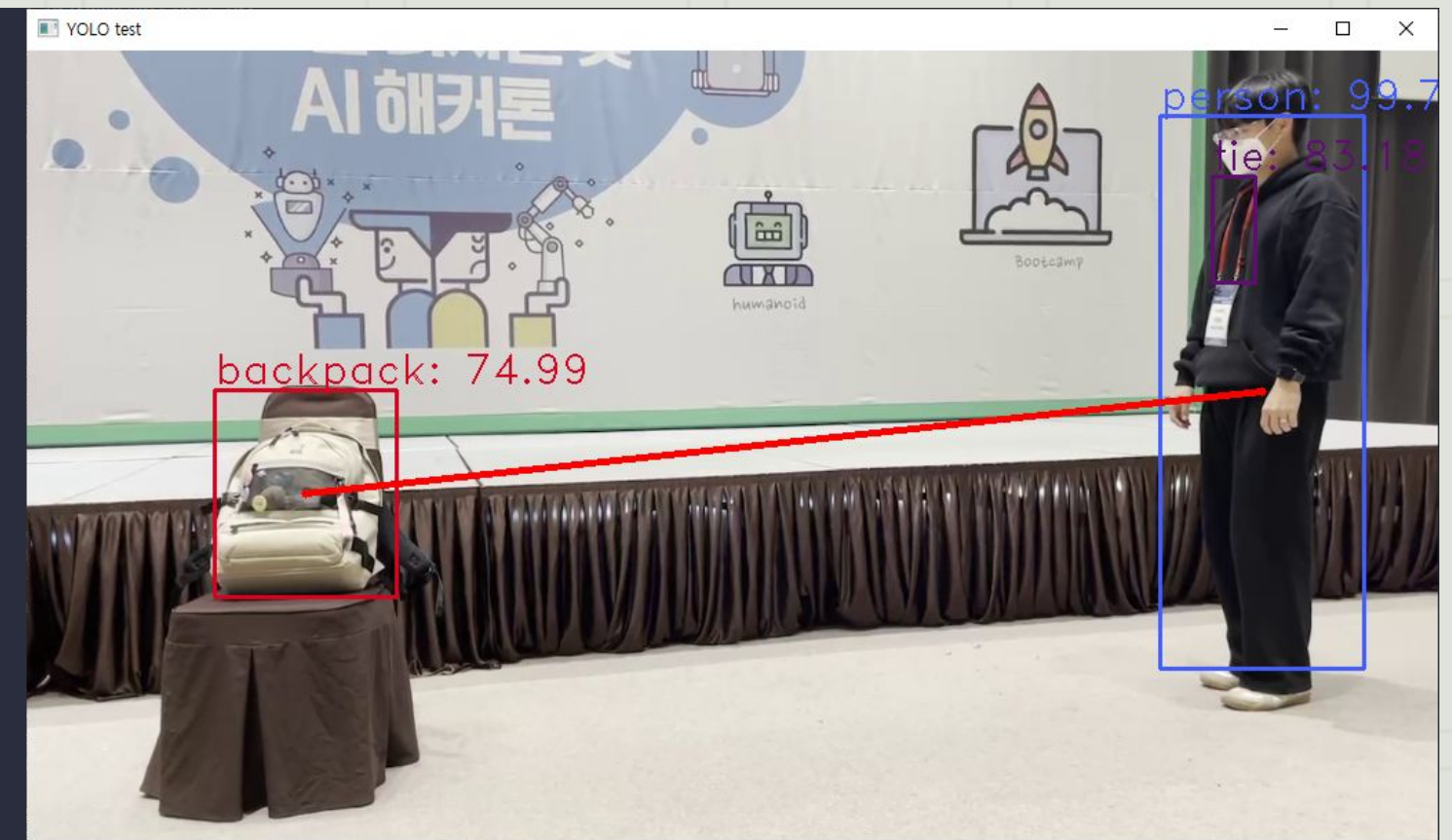
```
9 def detectAndDisplay(frame):
10     #녹화된 동영상 전처리
11     start_time = time.time()
12     img = cv2.resize(frame, None, fx=0.8, fy=0.8)
13     height, width, channels = img.shape
14
15     #-- 창 크기 설정
16     blob = cv2.dnn.blobFromImage(img, 0.00392, (416, 416), (0, 0, 0), True, crop=False)
17
18     net.setInput(blob)
19     outs = net.forward(output_layers)
20
21     #-- 탐지한 객체의 클래스 예측
22     class_ids = []
23     confidences = []
24     boxes = []
25
26     #거리 측정용 list
27     distance = [[],[]]
28     j = 0
```

```
30 for out in outs:
31     for detection in out:
32         scores = detection[5:]
33         class_id = np.argmax(scores)
34         confidence = scores[class_id]
35         if confidence > min_confidence:
36             # 탐지한 객체 박스
37             center_x = int(detection[0] * width)
38             center_y = int(detection[1] * height)
39             w = int(detection[2] * width)
40             h = int(detection[3] * height)
41
42             x = int(center_x - w / 2)
43             y = int(center_y - h / 2)
44
45             boxes.append([x, y, w, h])
46             confidences.append(float(confidence))
47             class_ids.append(class_id)
48
49 indexes = cv2.dnn.NMSBoxes(boxes, confidences, min_confidence, 0.4)
50 font = cv2.FONT_HERSHEY_DUPLEX
```


인공지능 모델 알고리즘

2. 두 객체의 중간 좌표를 계산하여 두 객체 사이의 거리를 구한다.

```
indexes = cv2.dnn.NMSBoxes(boxes, confidences, min_confidence, 0.4)
font = cv2.FONT_HERSHEY_DUPLEX
for i in range(len(boxes)):
    if i in indexes:
        x, y, w, h = boxes[i]
        label = "{}: {:.2f}".format(classes[class_ids[i]], confidences[i]*100)
        print(i, label)
        color = colors[i] #-- 경계 상자 컬러 설정 / 단일 색상 사용시 (255,255,255) 사용(B,G,R)
        cv2.rectangle(img, (x, y), (x + w, y + h), color, 2)
        cv2.putText(img, label, (x, y - 5), font, 1, color, 1)
        if ("person" in label) or ("backpack" in label):
            cv2.line(img, ((x+(x+w))/2, (y+(y+h))/2), ((x+(x+w))/2, (y+(y+h))/2), (0,0,255), 5)
            distance[j].append((x+(x+w))/2)
            distance[j].append((y+(y+h))/2)
            j += 1
j = 0
if distance[1]:
    length_a = (distance[0][0] - distance[1][0]) * (distance[0][0] - distance[1][0])
    length_b = (distance[0][1] - distance[1][1]) * (distance[0][1] - distance[1][1])
```



인공지능 모델 알고리즘

3. 데이터셋을 통해 공장의 기계별 위험성 정도를 모델에게 학습시킨다.

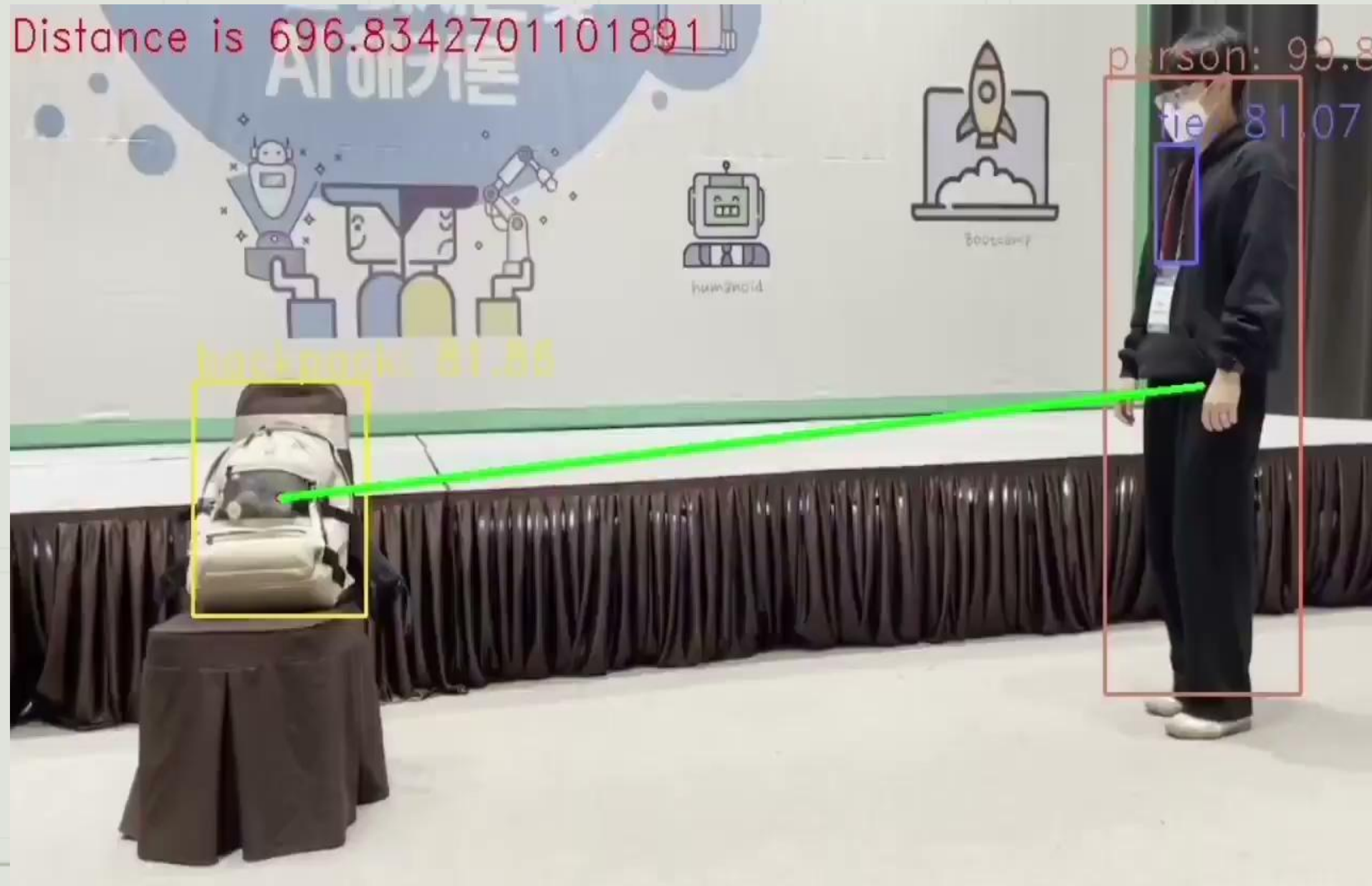
-> 위험성 정도에 따라 거리를 나타내는 선의 색을 다르게 한다.



인공지능 모델 알고리즘

4. 데이터셋을 통해 공장의 기계별 위험성 정도를 모델에게 학습시킨다.

-> 위험성 정도에 따라 거리를 나타내는 선의 색을 다르게 한다.



인공지능 모델 알고리즘

5. 데이터셋을 통해 공장의 기계별 위험성 정도를 모델에게 학습시킨다.

-> cmd 창에 영상 처리에 관련된 정보와 거리별 주의 메시지를 프린트한다.

```
if distance[1]:
    length_a = (distance[0][0] - distance[1][0]) * (distance[0][0] - distance[1][0])
    length_b = (distance[0][1] - distance[1][1]) * (distance[0][1] - distance[1][1])
    length_sum = math.sqrt(length_a + length_b)
    length_text = "Distance is " + str(length_sum)
    if length_sum >= 550:
        cv2.line(img, distance[0], distance[1], green, 3)
        print("Status : Safe Distance")
    elif length_sum >= 400 and length_sum < 550:
        cv2.line(img, distance[0], distance[1], blue, 3)
        print("Status : Average Distance")
    else:
        cv2.line(img, distance[0], distance[1], red, 3)
        print("Status : Caution! Stay away from the machine!")
    cv2.putText(img, length_text, (0,30), font, 1, (0,0,255), 1)
end_time = time.time()
process_time = end_time - start_time
print("=== A frame took {:.3f} seconds".format(process_time))
cv2.imshow("YOLO test", img)
```

2 person: 100.00
5 backpack: 75.59
Status : Safe Distance
=== A frame took 0.570 seconds

0 person: 99.52
2 backpack: 87.63
Status : Average Distance
=== A frame took 0.562 seconds

1 person: 99.98
4 backpack: 86.04
Status : Caution!
Stay away from the machine!
=== A frame took 0.621 seconds

한계점 & 개선점

한계점 & 개선점

1. 영상처리 - 공장 기계 학습

- 공장에 설치되어 있는 각종 기계들을 학습시켜, 사람과 기계간의 거리에 따른 결과값을 내려고 했으나, 기계들의 사진 등을 모아둔 데이터셋을 찾지 못해 YOLO 알고리즘이 구분할 수 있는 다른 사물로 대체하였다.
- 기계의 종류를 나누어 덜 위험한 기계와 더 위험한 기계의 안전 거리 기준을 서로 다르게 만드려 했는데, 위와 같은 이유로 공장 기계들의 외관을 학습시키지 못해 해당 기능이 누락되었다.

2. 음성 처리

- 데이터가 매우 큰 이유로 확보한 데이터셋 전체를 사용하지 못하고 우선 일부 사고 유형 음성에 해당하는 데이터만 모델 학습에 사용하였다.

3. UI 디자인

- 시간 관계 상 디자인에 많은 공을 들이지 못했다.

활용방안 및 기대효과

활용 방안

산업체를 넘어 가정, 기업 그리고 국가적 등 모든 방향의 안전사고에 대비 가능

기대 효과

가정 : 가정에서 발생하는 안전사고에 대비

기업 : 사내 발생 가능한 안전사고에 대비

국가 : 공공기관에서 발생 가능한 안전사고에 대비/ 경찰서, 소방서와의 협업을 통한
발빠른 대응으로 국민의 신뢰성 회복

결과 : 안전사고 예방 및 대응에 강한 국가적 이미지 형성

향후 계획

향후 계획

가정, 기업, 산업, 국가에 따라 세부적인 분야로 확장 가능

EX)

가정 : 가정 내 사망 인식(자살, 고독사)

기업 : 합선 / 사각지대 응급 환자 발생 대처

산업 : 비정상적 전류에 대한 전기공 감전사고 예방

국가 : 국가 행사 시 대테러 예방

글로벌 진출 가능

치안이 안 좋은 국가의 총기사고, 테러사고, 범죄의 예방

Q & A

Reference

1. 제작 계기, PPT 제작 관련 참고 자료

-SPC 사고 대응 관련 기사

-제조업 사고성 사망자의 발생 유형 비율 차트(재구성)

-픽토그램 출처

1. 음성 처리 알고리즘 관련 참고 자료

-클라우드 소싱 데이터 수집 사이트

-위급상황 음성/음향 (AI Hub)

-MFCC 음성 인식 알고리즘

1. 영상처리 알고리즘 관련 참고 자료

-Python과 OpenCV를 이용한 실시간 객체 탐지 알고리즘 구현, Deep.I

-딥러닝을 활용한 객체 탐지 알고리즘 이해하기, SAS Korea Blog