JDCS
디지털콘텐츠학회논문지
Journal of Digital Contents Society
Vol. 25, No. 3, pp. 751-760, Mar. 2024

Check for updates

객체 탐지 알고리즘을 활용해 PDLC 필름을 제어하는 스마트 윈도우 시스템 구현

유 영 창[1] · 김 동 회[2*]
[1]강원대학교 IT대학 전기전자공학과 학사과정
[2*]강원대학교 IT대학 전기전자공학과 교수

Implementation of a Smart Window System Controlling a PDLC Film using an Object Detection Algorithm

Young-Chang Yu[1] · Dong-Hoi Kim[2*]

[1]Undergraduate, Electrical and Electronic Engineering, IT College, Kangwon National University, Chuncheon 24341, Korea
[2*]Professor, Electrical and Electronic Engineering, IT College, Kangwon National University, Chuncheon 24341, Korea

[요 약]

일반 가정의 보안을 위하여 기존의 스마트 윈도우에 추가할 수 있는 사생활 보호 기능을 제안한다. 제안하는 스마트 윈도우는 객체탐지 알고리즘을 기반으로 창 안쪽을 바라보는 사람을 감지해낸다. 감지 결과를 기반으로 라즈베리 파이가 창문에 붙은 PDLC(Polymer Dispersed Liquid Crystal) 필름에 인가되는 전압을 제어하여 투명도를 변화시켜 유리창의 투명도를 조절한다. 실시간 객체탐지 알고리즘으로는 YOLOv8(You Only Look Once version 8)를 활용한다. 목적에 맞게 얼굴 이미지를 활용하여 커스텀 데이터를 구성하고 알고리즘을 학습시킨다. 다양한 크기의 YOLOv8의 모델을 mAP(mean Average Precision)와 감지속도를 기준으로 비교하였다. 그 결과 다른 모델에 비해 mAP는 최대 2.3% 떨어지지만, 감지속도는 최대 845.9% 빠른 YOLOv8n이 가장 적절함을 확인하였다. 최종 모델을 기반으로 필름을 제어하는 시스템을 구현하였다.

[Abstract]

To enhance the security of ordinary households, a smart window with a privacy protection feature is proposed. The proposed smart window employs object detection algorithms to detect individuals inside the room. Based on the detection results, a Raspberry Pi controls the voltage applied to the PDLC film attached to the window, thus adjusting the transparency of the glass window. The YOLOv8 algorithm is utilized for real-time object detection. Custom data focusing on facial images are sourced to train the algorithm. Various sizes of YOLOv8 models are compared based on mAP and detection speed. YOLOv8n, which has a 845.9% faster detection speed compared to other models, is identified as the most suitable model despite a maximum decrease of 2.3% in the mAP. The final model is employed to implement a system that controls the film for the smart window.

색인어 : 욜로, 고분자 분산 액정, 실시간 객체 탐지, 딥러닝, 스마트 윈도우

Keyword : You Only Look Once, Polymer Dispersed Liquid Crystal, Real-time Object Detection, Deep Learning, Smart Window

http://dx.doi.org/10.9728/dcs.2024.25.3.751

Received 19 January 2024; **Revised** 16 February 2024
Accepted 26 February 2024

***Corresponding Author; Dong-Hoi Kim**

Tel: +82-33-250-6349
E-mail: donghk@kangwon.ac.kr

Ⅰ. 서 론

최근 1인 가구 수의 급격한 증가가 나타나고 있다. 2005년 1인가구의 비중은 20.0%에서 2021년에는 33.4%로 증가하였으며 2050년에는 39.6%에 이를 것으로 예상하고 있다[1]. 이에 따라 가정용 CCTV, 스마트 도어락 등 IT 기반의 가정용 보안 장비가 주목받고 있다. 또한 스토킹 범죄 신고건수도 증가하는 추세로 가정 보안에 대한 필요성이 증가하고 있다. 기존의 스마트 윈도우는 어플리케이션을 통한 창문의 개폐상태 조종이나 채광 조종, 환기 등 실내환경과 관련된 기능 위주로 이루어져 있으며, 보안 기능은 침입 경보 기능 정도에 그치고 있다. 이러한 배경 속에서, 사생활을 보호할 수 있는 기능을 가진 스마트 윈도우를 제안한다. 실시간 객체 탐지(Real-time Object Detection)를 통해 창 밖에서 창문을 바라보는 사람을 감지해내면, 창문 유리에 붙은 PDLC 필름의 투명도를 제어하여 실내의 모습이 보이지 않도록 한다.

기존에도 특수 필름을 이용하여 투명도를 조절해 내부의 모습이 보이지 않도록 하는 특수 창문이 존재하지만, 이는 사람이 직접 수동으로 조작해야되기 때문에 번거로움이 있으며 사생활 보호 목적으로는 빈틈이 존재한다. 시중에 존재하는 스마트 윈도우는 에너지, 실내 환경 관리, 빛 조절 등의 기능을 갖춘 형태에서 크게 벗어나지 않는다. 객체탐지 기능을 가진 스마트 윈도우는 사용자의 사생활까지 보호하는 새로운 형태의 솔루션을 제안할 수 있을 것으로 기대된다[2].

제안하는 스마트 윈도우는 웹캠을 통해 창밖의 모습을 영상 데이터로 입력 받으면, 실시간 객체 탐지 알고리즘을 통하여 사람의 얼굴을 감지해낸다. 사람의 얼굴을 감지중인 상황에는 PDLC 필름이 불투명해지도록 제어하여 창 밖에선 실내의 모습이 보이지 않도록 한다. 평소에 창문을 바라보는 얼굴이 감지되지 않는 상태에서는, 반대로 PDLC 필름이 투명한 상태를 유지하도록 하여 일반 유리창과 같이 작용한다.

얼굴을 감지해낼 실시간 객체 탐지는 YOLOv8 알고리즘을 활용한다. 목적에 맞게 클래스를 분리하여 라벨링한 이미지로 커스텀 데이터를 구성하고 이를 알고리즘에 학습시킨다. 이 과정에서 크기별로 다른 특성을 가지는 YOLOv8의 여러 모델들을 모두 학습시켜 비교해보며 적절한 모델을 선택한다. 커스텀 데이터로 학습이 완료된 YOLOv8 모델을 MS COCO (Microsoft Common Objects in Context) 데이터셋으로 사전학습 되어있는 기존의 YOLOv8 모델과 비교해보고 이를 PDLC 필름 제어에 활용하며 창문의 투명도를 조절하는 스마트 윈도우 시스템에 적용하여 가정 내 보안에 활용할 수 있을 것으로 기대한다.

본 논문의 Ⅱ장에서는 실시간 객체 탐지 알고리즘으로 YOLOv8을 선택한 이유와 학습 과정에 대해 설명하고 Ⅲ장에서는 제안하는 스마트 윈도우 시스템과 이를 구성하기 위한 방법에 대해 기술한다. Ⅳ장에서는 YOLOv8 알고리즘을 학습시키기 위한 준비를 다룬다. Ⅴ장에서는 YOLOv8를 다양한 방법으로 학습시켜보며 성능을 비교하여 적절한 학습 방법을 결정하고 모델의 크기별로도 성능을 비교한다. 최종적으로 가장 적절한 모델을 선정하여, 커스텀 데이터로 학습시키기 전의 YOLOv8과 비교해본다. Ⅵ장에서 본 논문에서 실험한 결과에 대해 고찰하며 결론을 맺는다.

Ⅱ. 실시간 객체 탐지

2-1 실시간 객체 탐지 알고리즘

객체 탐지 알고리즘은 딥러닝 기반의 AI 기술 중 하나로, 주로 이미지 상에 존재하는 특정 객체를 탐지해내어 그 위치를 Bounding Box로 표시하는 방식을 이용한다[3]. 여기에 사용되는 알고리즘이 CNN(Convoltuional Nerual Network)이다. 하지만 CNN 방식은 한 이미지에 여러 개의 객체가 존재할 경우에는 객체 분류(Object Classification)에 한계가 있다. CNN 기반의 객체 탐지 알고리즘은 다시 지역 기반과 단일 회귀 방식(Single-Stage Methods)으로 분류할 수 있다. 먼저 지역 기반 방식은, 이미지에서 객체가 있을 법한 위치를 영역 제안한 뒤에 이를 바탕으로 특징을 추출하고 객체를 분류해낸다. 이와 같은 방식을 사용하는 알고리즘에는 R-CNN, Fast R-CNN, Faster R-CNN 등이 포함된다. 특징 추출과 분류의 단계가 나뉘어 있어 이 단계방식(Two-Stage Methods)으로 불리는 이 알고리즘들은 높은 정확도를 가지지만 비교적 처리 속도가 느리다는 단점이 있다[4]. 단일 회귀 방식은 이미지에서 개체의 위치 예측과 분류를 동시에 해낸다. 따라서 빠른 속도를 가진다는 장점을 가진다. 이와 같은 방식을 갖는 알고리즘에는 YOLO, SSD, RetinaNet 등이 있다[5]. 객체 탐지를 실시간으로 해내기 위해서는 빠른 속도가 필요하기 때문에, 실시간 객체 탐지 알고리즘에는 단일 회귀 방식이 적합하다.

2-2 YOLOv8

YOLO는 대표적인 단일 회귀 방식의 객체 탐지 알고리즘으로, 현존하는 가장 빠른 객체 검출 알고리즘 중 하나이다. 실시간으로 객체 탐지 기능을 해내는 데에 가장 적절한 알고리즘이라고 할 수 있다. YOLO는 크게 Backbone과 Head로 구성되어 있다. Backbone에서는 입력된 데이터의 특징을 추출하며, Head에서는 추출된 특징을 분류하여 감지해내는 역할을 한다. YOLO는 여러 개발자들에 의해 개발된 다양한 버전이 존재하며, 그 성능과 특징이 모두 다르다. 그림 1은 YOLO 이름을 갖는 여러 알고리즘들의 정확도 및 감지속도를 비교한 그래프이다. YOLOv8은 파라미터의 수에 따라 n, s, m, l, x의 다섯 가지 종류의 모델을 가지고 있다. 위 그래프를 보면 다른 YOLO와 비교하여 비슷한 파라미터 수에서 더 높

은 정확도를 가진다. 아래 그래프에서는 같은 감지속도에서 더 높은 정확도를 가지는 것을 알 수 있다. 본 논문에서는 가장 최근에 발표되고 가장 빠른 처리 속도와 높은 정확도를 가지는 YOLOv8 알고리즘을 이용한다.

YOLOv8은 2023년 1월 Ultralytics에서 개발되었다. 개체 감지, 인스턴스 세분화 및 이미지 분류 모델을 train하기 위한 통합 프레임워크로 구축되었다는 특징이 있다. Backbone은 modified CSPDarknet32 backbone을 사용하며, Head에서는 process objectness, classification, regression task의 작업을 수행한다. YOLOv8은 다시 YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, YOLOv8x의 다섯 가지 사이즈로 구분된다. 모델의 사이즈에 따라 Backbone을 구성하는 depth_multiple과 width_multiple 변수의 크기가 다르다. YOLOv8n은 3.2M, YOLOv8s는 11.2M, YOLOv8m은 25.9M, YOLOv8l은 43.7M, YOLOv8x는 68.2M의 파라미터 크기를 갖는다. 사이즈가 작은 모델일수록 정확도가 떨어지지만, 처리 속도가 빨라진다.

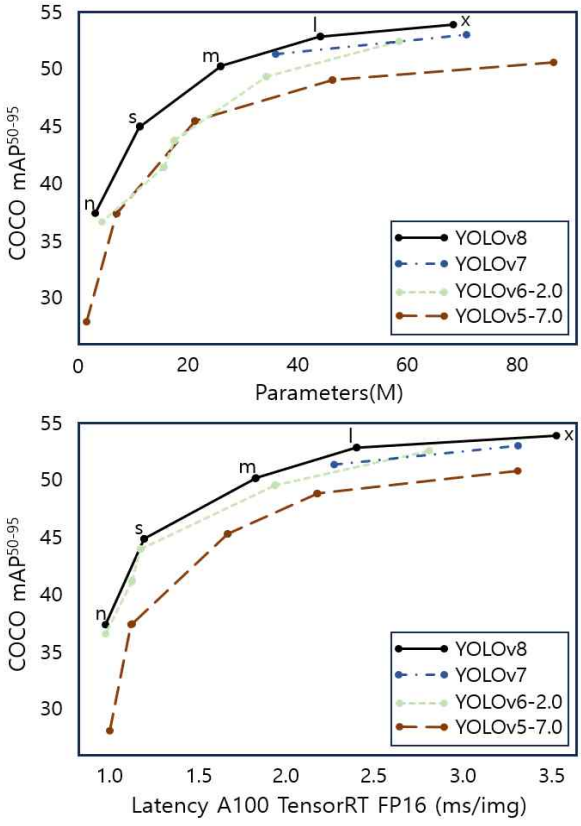

그림 1. YOLO 시리즈의 성능 비교[6]
Fig. 1. Performance comparison of YOLO series

2-3 YOLOv8의 학습

YOLOv8을 이용한 학습 프로세스는 크게 다섯 단계로 구분할 수 있다. 첫 번째로 Custom Data를 준비하는 Data Preparation 단계, 다음으로 데이터를 Colab으로 보내고 학습시키기 위한 데이터를 담은 YAML file을 준비하는 Loading Data 단계, YOLOv8 실행에 필요한 라이브러리를 불러오는 Install YOLOv8 단계, 데이터를 바탕으로 모델을 학습시키는 Train model 단계, 마지막으로 테스트 이미지를 이용해 학습된 모델을 실행해보고 성능을 평가하는 Prediction 단계이다.

Data Preparation 단계에서는 커스텀 데이터를 구축하기 위한 이미지 파일과 정답 데이터가 필요하다. 딥러닝은 충분한 양의 학습 데이터가 있을 때, 신뢰도와 처리속도가 향상된다. 따라서 충분한 데이터를 수집하기 위하여 Roboflow에서 제공하는 Dataset을 활용하였다. 학습 데이터는 Train, Validation, Test의 세 범주로 구분할 수 있다. Train은 모델을 학습 시키는 데에 사용되는 데이터이며, Test는 학습된 모델의 성능을 평가할 때 사용된다. Validation은 모델을 학습시키는 과정에서 모델이 학습되고 있는 정도를 검증하는 과정에 사용된다. Loading Data 단계에서 앞서 확보한 데이터를 Google Colab으로 불러와 준다. 그 후 학습 데이터의 위치(Directory) 및 인식(Detection)하고 싶은 클래스의 종류와 이름 정보가 들어있는 YAML파일을 형성해야 한다.

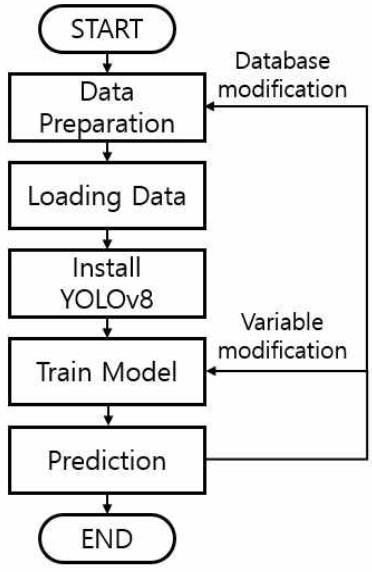

그림 2. YOLOv8 학습 프로세스
Fig. 2. YOLOv8 training process

YOLOv8을 Install 해주고, 모델을 학습해준다. 이후 test image를 이용하여, 학습된 모델이 객체 탐지를 수행하도록 하여 모델의 성능을 평가한다. 이때 평가된 결과를 바탕으로 Data Preparation이 제대로 이루어졌는지 확인할 수 있다. 학습 결과 충분한 성능을 얻지 못하였을 때는, Train model 단계로 돌아가 학습과 관련된 변수를 수정하여 다시 학습을 진행한다. 또는 Data Preparation 단계로 돌아가 데이터셋을 다시 구성하는 방법이 있다.

Ⅲ. 제안하는 스마트 윈도우

3-1 YOLOv8과 PDLC 필름을 이용한 보안 시스템 구현

제안하고자 하는 스마트 윈도우의 보안 시스템 기능은 '바깥으로부터 창문을 통해 들어오는 시선의 차단'이다. 이를 위해선 크게 두 가지 요소가 필요하다. 첫번째는 시선의 감지이고, 두번째는 시선의 차단이다. 창문 방향을 바라보는 사람의 시선을 감지해내기 위해서 객체 탐지 알고리즘 중 하나인 YOLOv8을 활용한다. YOLOv8은 이미지 데이터를 딥러닝하여 학습된 데이터를 바탕으로, 입력받은 이미지 속에서 미리 학습받은 특정한 개체를 구별해 감지해낸다. YOLOv8이 이러한 역할을 할 수 있도록 YOLOv8에 '사람의 눈'을 학습시켰으나, 이미지를 입력받는 카메라의 성능에 따라 정확도가 크게 떨어지는 것을 확인하였다. 이를 해결하기 위한 대안으로 YOLO에 '정면을 바라보는 사람의 얼굴'을 학습시켜 창문을 바라보는 사람의 얼굴을 감지할 수 있도록 한다. 이를 통해 사람이 창문을 들여다보는 상황을 감지해낼 수 있는 일종의 센서 역할을 할 수 있을 것으로 기대한다.

창문 안쪽을 바라보는 것을 막기 위해서 창문의 유리창에 PDLC 필름을 부착한다. PDLC는 고분자 분산형 액정 유리를 의미하는데, 필름에 걸리는 전압의 크기에 따라 액정의 배열이 변화하며 투명도가 바뀐다. YOLOv8에서의 감지결과에 따라 PDLC에 걸리는 전압을 제어하면, 창문의 투명도를 제어할 수 있게 된다[7]. YOLOv8 알고리즘을 실행할 하드웨어는 라즈베리 파이(Raspberry Pi 3 Model B)를 사용하였다. 라즈베리 파이의 USB 포트에는 웹캠을 연결하여 실시간 영상 데이터를 입력받는다. 영상 데이터는 프레임별로 각각 하나의 이미지로써 YOLOv8에 이미지 입력으로 사용된다. YOLOv8의 객체탐지는 수십~수백 ms 단위의 매우 빠른 속도로 이루어지기 때문에 실시간으로 탐지가 가능하다.

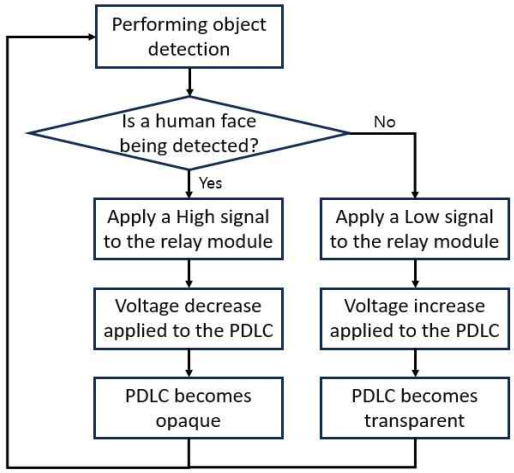

그림 3. 제안하는 스마트 윈도우 시스템의 순서도
Fig. 3. Flowchart of the proposed smart window system

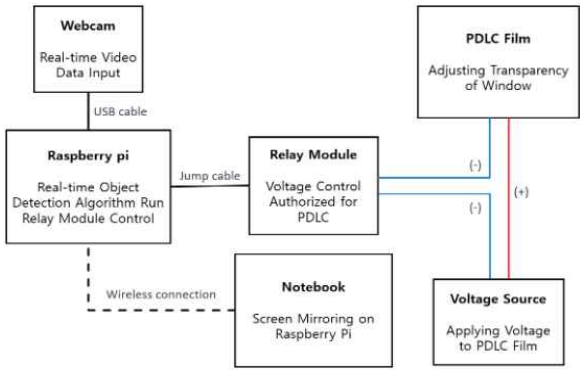

그림 4. 제안하는 스마트 윈도우 시스템의 구상도
Fig. 4. Proposed smart window system schematic diagram

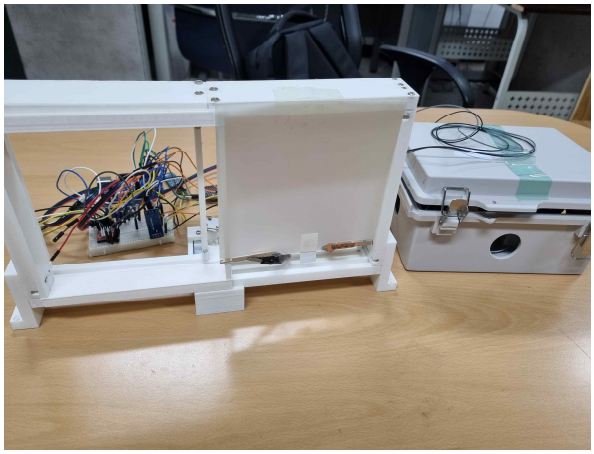

그림 5. 구현된 스마트 윈도우 시스템
Fig. 5. Implemented smart window system

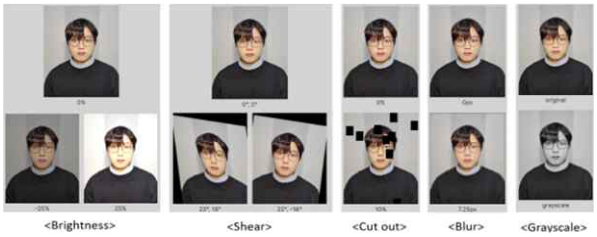

그림 6. 이미지 증식 적용 예
Fig. 6. Application example of image augmentation

그림 3은 제안하는 스마트 윈도우 시스템의 작동 알고리즘을 간단하게 설명하는 순서도이다. 본 논문에서의 목적에 맞도록 YOLOv8 모델을 커스텀 학습시켜 PDLC필름을 제어하는 시스템을 그림 4와 같은 형태로 구상하였다. 그림 5는 구상도를 바탕으로 구현한 스마트 윈도우 시스템의 모습이다.

3-2 YOLOv8의 학습 준비

YOLOv8를 학습시키기 위해서는 이미지 데이터가 필요하

다. 일반적인 이미지 데이터가 아니라, 이미지에 어떤 종류의 객체가 포함되어 있는지 설명하는 클래스(Class)와 이미지에서 어느 위치에 객체가 존재하는지 설명하는 Bounding Box 데이터가 포함되어 있어야 한다. 이미지에 이러한 정보를 입력하는 것을 라벨링(Labeling)이라 하며, 이렇게 구성된 데이터의 집합을 데이터셋(Dataset)이라 한다.

딥러닝의 특성상, 데이터는 많을수록 정확도가 상승한다. 한정된 이미지 데이터 수를 이용하여 보다 큰 효과를 얻기 위해 이미지 증식(image augmentation) 기법을 활용한다. 이미지 증식은 이미지에 왜곡을 적용하여 원본과 서로 다른 이미지인 것처럼 활용할 수 있는 기법이다.

그림 7. 특징에 따라 구분된 클래스
Fig. 7. Classes classified according to characteristics

YOLO에서는 감지해내는 객체를 클래스로 나눠 구분한다. 모자를 쓴 사람이나 마스크를 쓴 사람, 안경을 쓴 사람 등을 모두 하나의 클래스로 묶어 학습하면 특징이 모호해져 정확도가 떨어지게 된다. 따라서 특징에 따라 클래스를 나눠 학습시킨 뒤에 객체를 감지해내도록 하여 정확도를 높히는 방식을 제안한다.

3-3 PDLC 필름 제어

PDLC 필름은 고분자 분산형 액정으로, 필름에 가해진 전압에 따라서 액정 분자가 나열되거나 흩어지면서 빛을 투과시키거나 막는다. 이러한 원리를 이용하여, 유리창에 PDLC 필름을 부착하고 객체 탐지 결과에 따라 필름에 가해지는 전압을 제어하여 유리창의 투명도를 제어해낸다. 필름에 고전압이 가해지면 필름의 액정이 정렬되며 투명해지고, 전압이 낮아지면 액정의 배열이 흐트러지며 불투명해진다[8]. 실험에 사용된 PDLC 필름은 60V AC 전압을 사용한다. 반면에 라즈베리 파이의 출력 핀은 5V DC 전압으로 되어있다. 비교적 약한 전압으로 높은 전압을 이용하는 필름을 제어하기 위하여 릴레이 모듈을 활용한다. 릴레이 모듈은 NC(Nomally Closed) 터미널을 이용할 때는 High 신호가 입력되었을 때 전류를 차단하는 역할을 한다. 반대로 NO(Nomally Open)터미널을 이용할 때는 Low 신호가 입력되었을 때 전류를 차단하는 역할을 한다. 제안하는 시스템은 YOLOv8이 사람의 얼굴을 감지하였을 때에 High의 신호를 출력하여 필름이 불투명해지도록(전압이 0이 되도록) 제어하기 때문에 NC 터미널을 사용하여 회로를 구성하였다. 평상시 창문에 부착된 필름에는 60V의 전압이 인가되어 있다. 이때 필름은 투명한 상태를 유지하게 된다. YOLOv8에서 사람의 얼굴을 감지해내었을 때, 라즈베리 파이의 출력 핀을 통해 릴레이 모듈에 High 신호를 입력해준다. 릴레이 모듈은 High 신호를 인가받고 개방(Open)된다. 이 릴레이 모듈이 필름에 인가되는 전압을 차단하여 필름이 불투명해지도록 한다. 위와 같은 작동원리에 따라 제안하는 사생활 보호 시스템을 수행하도록 한다.

Ⅳ. 실험 환경

4-1 실험 장비

실시간 객체 탐지를 구동하는 하드웨어는 라즈베리 파이를 사용하였으며 웹캠을 통해 데이터를 입력 받는다. 웹캠은 라즈베리 파이의 USB 포트에 연결되어 실시간 영상 데이터를 수집한다. 라즈베리 파이는 학습된 YOLOv8 모델을 통해 영상 데이터를 실시간으로 처리한다. 이를 통해 인간의 얼굴을 감지하는 역할을 한다.

표 1. 라즈베리 파이의 스펙 시트
Table 1. Spec sheet of Raspberry P

Raspberry Pi 3 Model B	
Processor	Quad-core 64-bit ARM cortex A53 CPU
Memory	1GB LPDDR2-900 SDRAM
GPU	Dual Core VideoCore IV
USB	4 USB 2 ports
Power	Micro-usb socket 5V/2.5A
Network	802.11n Wireless LAN
Deminsion	85×56×17mm

표 2. 웹캠의 스펙 시트
Table 2. Spec sheet of Webcam

TB189	
Definition	300k pixele
Chip type	Color CMOS image sensor
Resolution	640×480
Video format	24-bit RGB
Frame rate	640×480 up to 30 frame/sec(VGA)
Sensor size	4.86×3.64 mm

4-2 구글 코랩

구글 코랩(Google Colab)은 구글이 제공하는 클라우드 기반의 Jupyter Notebook 환경이다. 웹 브라우저 상에서 Python 스크립트를 작성 및 실행할 수 있다. 작업을 수행할 때 Goolge 클라우드 서버 상에서 GPU나 TPU 등의 하드웨어

가속기를 제공하기 때문에 컴퓨터의 성능과 관계없이 머신러닝 작업을 수행할 수 있다.

4-3 로보플로우

로보플로우(Roboflow)는 딥러닝 분야에서 컴퓨터 비전(Computer Vision) 작업을 쉽게 하도록 도와주는 도구이다. 인터넷 상에 이미지 데이터를 업로드하고 이미지 라벨링(Image Labeling)을 통해 클래스를 나눠 정답 데이터를 입력해줄 수 있다. 이렇게 형성된 데이터셋을 저장하고 관리할 수 있다. 그림 8은 face 클래스로 이미지를 라벨링한 모습이다. 로보플로우의 UI를 활용하면 간단하게 이미지를 라벨링 할 수 있어 많은 양의 이미지도 빠르게 처리할 수 있다.

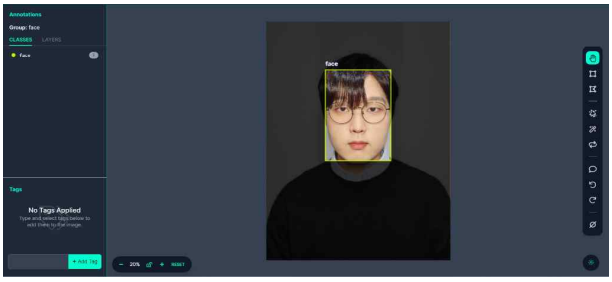

그림 8. 로보플로우를 이용하여 라벨링한 이미지
Fig. 8. Images labeled using RoboFlow

4-4 VNC

VNC(Virtual Network Computing)는 RFB 프로토콜을 활용하여 컴퓨터에 원격으로 액세스하여 제어할 수 있는 프로그램이다. 이를 이용하여 라즈베리 파이를 외부에서 접속하여 원격으로 제어할 수 있다. 라즈베리 파이와 노트북을 같은 네트워크 상에 연결하고, 노트북에 VNC Viewer를 설치한 뒤에 라즈베리 파이의 ip를 입력하여 사용한다. 이를 통해 라즈베리 파이를 별도의 디스플레이 및 입력장치 연결 없이 노트북의 GUI를 통해 제어할 수 있다.

V. 제안하는 시스템의 성능 분석

본 논문에서는 YOLOv8모델에 인간의 얼굴 이미지 자료를 데이터셋으로 하여 학습을 진행하였다. 인간의 얼굴만을 대상으로 학습하였을 때는 마스크를 쓴 사람의 얼굴을 제대로 인식하지 못하는 문제가 발견되었다. 따라서 데이터셋에는 마스크를 착용하지 않은 얼굴(face), 마스크를 착용한 얼굴(mask), 마스크를 반만 착용한 얼굴(half_mask)의 세 가지로 구분된 클래스를 적용하였다. 학습(train) 데이터 5,930장, 검증(valid) 데이터 568장, 시험(test) 데이터 601장으로 구성되어 있다.

5-1 학습 설정

같은 데이터셋을 이용하여 같은 객체탐지 알고리즘 모델을 학습시킨다고 하여도, 사용자가 학습과 관련된 변수를 어떻게 설정하느냐에 따라 그 결과는 바뀐다. 본 논문에서 YOLOv8을 학습시키는 과정에서 임의로 설정한 변수는 epochs, patience, batch이다. 데이터셋을 객체 탐지 알고리즘에 학습시킬 때, 전체 데이터셋에 대해 한 번의 학습을 한 번의 epoch가 진행되었다고 한다. 만약 epoch 값을 너무 작게 설정하면 충분한 학습이 이루어지지 않아 과소적합(underfitting)이 발생할 수 있다. 반대로 너무 크다면 과대적합(overfitting)이 발생할 수 있다. 따라서 적절한 epoch값을 설정하는 것이 중요하다[9]. 또한 너무 큰 epoch로 학습하려면 학습에 소요되는 시간이 길어진다. 이와 관련해 patience를 활용할 수 있다. patience는 학습을 진행하는 동안 설정한 patience값 만큼의 학습 횟수동안 성능의 증가가 일어나지 않으면 학습을 중단하는 변수이다. batch는 한번에 처리할 데이터의 크기를 뜻한다. 이 값을 크게 설정할수록 메모리 용량을 많이 차지하므로 주의해야 한다.

표 3. Epoch값에 따른 성능변화
Table 3. Performance changes based on epoch values

Model	Epochs	mAP
YOLOv8n	1	0.622
YOLOv8n	10	0.877
YOLOv8n	50	0.902
YOLOv8n	100	0.889

본 논문에서는 epochs = 128, patience = 0, batch = 32으로 학습을 진행하였다. batch값의 설정은 학습에 이용되는 메모리의 크기를 고려하여 임의의 수를 설정하였다. epochs 값의 설정은 epochs 값을 변경해보며 실험을 진행한 결과를 나타낸 표 3을 보면 epochs의 크기가 커질수록 성능 향상이 정체되거나 심지어 하락하는 것을 확인하였다. 따라서 학습에 걸리는 시간에 비해 성능의 향상이 크지 않게 되어버린다. 실험의 결과를 바탕으로 epochs값을 128로 설정하였다. patience = 0으로 변수를 지정하면 학습의 중단 없이 128번의 학습을 모두 시행하게 된다. patience값을 너무 작게 설정하면 학습 진행시에 성능 향상이 조금만 정체되거나 성능이 하락하는 경우 학습이 중단되는 문제가 발생하였다. YOLOv8은 학습 과정에서 가장 높은 성능을 냈던 상태를 'best.pt' 파일로, 가장 마지막 학습까지 끝낸 상태를 'last.pt' 파일로 저장하여 두 상태의 모델을 모두 얻을 수 있다. 이를 활용하여, patience = 0으로 설정해서 학습을 모두 진행시킨 후에 가장 성능이 높았던 상태의 모델을 추출할 수 있다. 이후 실험에서 사용된 모델은 위와 같은 설정으로 커스텀 데이터를 활용하여 학습시킨 YOLOv8의 best.pt 파일을 활용하였다.

5-2 모델 성능 평가 지표

객체 감지 알고리즘의 성능의 평가에는 mAP(Mean Average Precision)이 사용된다. mAP를 측정하기 위한 요소 중 하나인 혼동행렬(Confusion Matrix)은 아래와 같은 네 가지 속성을 포함한다.

True Positives(TP) : 모델이 실제 분류값과 일치하는 예측을 함.
True Negatives(TN) : 모델이 실제 분류값이 아닌 것을 일치하지 않는다는 예측을 함.
False Positives(FP) : 모델이 실제 분류값이 아닌 것을 일치한다고 예측을 함(제 1종 오류).
False Negatives(FN) : 모델이 실제 분류값을 일치하지 않는다는 예측을 함(제 2종 오류).

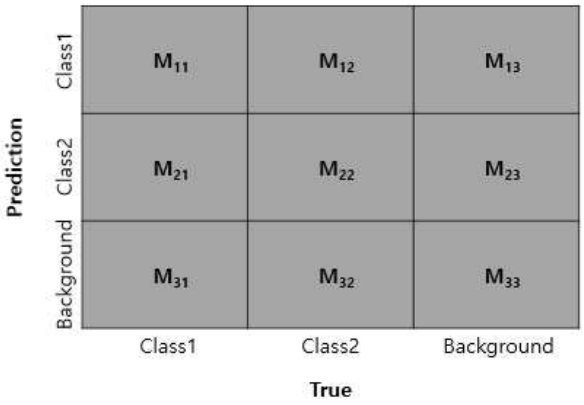

그림 9. 다중 분류 모델의 혼동 행렬
Fig. 9. Confusion matrix for multi-class classification

그림 9는 두 가지 클래스에 대한 학습을 진행한 모델의 혼동행렬을 나타낸 것이다. 그림에서 Class1에 대한 TP는 실제 Class1을 Class1로 예측한 경우인 M11이다. TN은 Class1이 아닌 것을 Class1이 아닌 것으로 예측한 경우인 M22, M23, M32, M33이다. FP는 실제로 Class1이 아닌 것을 Class1로 예측한 경우인 M12, M13이다. FN은 실제로 Class1인 것을 Class1이 아닌 것으로 예측한 경우인 M21, M31이다. 정답에 대한 기준은 IoU(Intersection over Union)를 이용한다. IoU는 정답 데이터에서 설정한 객체의 Bounding Box와 예측한 Bounding Box가 어느정도 이상이어야 정답으로 판단할지를 설정하는 임계값이다. 혼동행렬을 이용하여 정밀도(Precision)와 재현율(Recall)를 구할 수 있다. 정밀도 및 재현율은 아래와 같은 요소를 이용해 수식 (1), (2)와 같이 구한다.

$$\mathrm{Pr}ecision = \frac{TP}{TP+FP} \tag{1}$$

$$Recall = \frac{TP}{TP+FN} \tag{2}$$

모델이 어떠한 이미지를 판별할 때, 어느정도 이상의 일치율을 가져야 되는지에 대한 기준에 따라 그 결과는 달라진다. 이 기준을 confidence레벨의 threshold값이라고 한다. 이 값에 따라 정밀도와 재현율 또한 변화하게 된다. confidence레벨이 낮아지면, 안정적이지 않은 특징을 기반으로 객체 존재를 예측하게 되어 거짓긍정(FP)이 많아져서 정밀도가 낮아진다. 반대로 confidence레벨이 높아지면, 정확한 조건을 만족할 때만 객체가 탐지된 것으로 간주하여 거짓부정(FN)이 많아져서 재현율이 낮아진다. 이러한 변화에 따른 정밀도와 재현율의 변화를 그래프로 나타낸 것이 PR곡선(Precision-Recall curve)라고 한다. PR 곡선의 아래면적을 구하면 AP(Average Precision)을 계산할 수 있다. 각 클래스에 대한 평균 정밀도(AP)가 바로 mAP(mean Average Precisio)가 된다.

$$mAP = \frac{1}{N}\sum_{i=1}^{N} AP_i \tag{3}$$

PR 곡선의 아래면적을 구하면 AP(Average Precision)을 계산할 수 있다. 각 클래스에 대한 평균 정밀도(AP)가 바로 mAP(mean Average Precisio)가 된다[10]. 이때 IoU를 0.5로 설정하였을 때 mAP를 mAP50와 같이 표기하며, IoU를 0.5 ~ 0.95로 0.05씩 높이면서 측정한 mAP는 mAP50-95와 같이 표기한다.

5-3 모델의 크기와 mAP

아래는 같은 조건에서 YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, YOLOv8x의 다섯 모델의 학습 결과로 도출된 혼동행렬과 그를 바탕으로 구한 정확도 및 재현율, 그리고 PR곡선이다. 혼동행렬 및 PR곡선은 본 논문에서 최종 선택된 YOLOv8n 모델을 대표로 나타내었다.

mAP 값은 YOLOv8n의 경우에 0.902, YOLOv8s의 경우에 0.911, YOLOv8m의 경우에 0.913, YOLOv8l의 경우에 0.910, YOLOv8x의 경우에 0.923의 결과를 나타내었다. 모델의 크기가 클수록 성능이 높아지는 경향을 갖고 있지만 그 차이는 최대 2.3% 수준으로 나타난다. YOLOv8을 기존의 YOLO모델과 비교하였을 때 가장 두드러지는 차이점은 바로 작은 모델들의 성능 향상이다. 특히, 가장 작은 모델인 'Nano'의 성능이 가장 크게 향상되었다. 사이즈가 가장 작은 모델인 YOLOv8n과 사이즈가 가장 큰 모델인 YOLOv8x의 성능 차이가 작은 편이라는 것을 의미한다. YOLOv8의 이러한 특성 때문에 사이즈별 mAP 차이가 비교적 적게 나타났다.

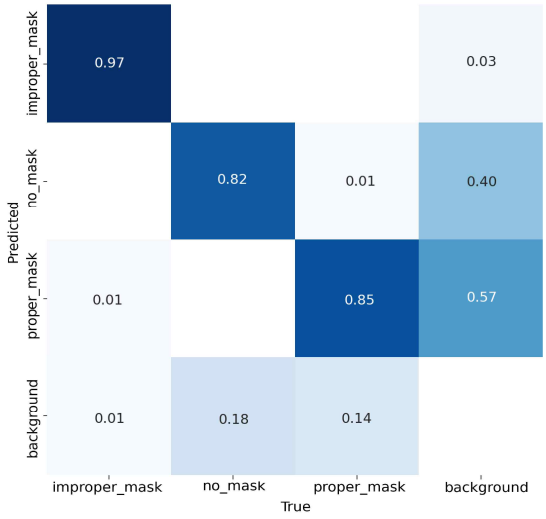

그림 10. YOLOv8n의 혼동행렬
Fig. 10. Confusion matrix of YOLOv8n

표 4. 모델의 크기에 따른 정확도
Table 4. Precision according to the size of model

Model	Face	Mask	Half_mask	Average
YOLOv8n	0.6838	0.5890	0.9510	0.7413
YOLOv8s	0.5882	0.6532	0.9510	0.7308
YOLOv8m	0.6566	0.6056	0.9600	0.7407
YOLOv8l	0.6484	0.6232	0.9327	0.7348
YOLOv8x	0.6259	0.6350	0.9510	0.7373

표 5. 모델의 크기에 따른 재현율
Table 5. Recall according to the size of model

Model	Face	Mask	Half_mask	Average
YOLOv8n	0.8000	0.8600	0.9700	0.8767
YOLOv8s	0.8000	0.8100	0.9700	0.8600
YOLOv8m	0.8400	0.8600	0.9600	0.8867
YOLOv8l	0.8384	0.8600	0.9700	0.8895
YOLOv8x	0.8700	0.8700	0.9604	0.9001

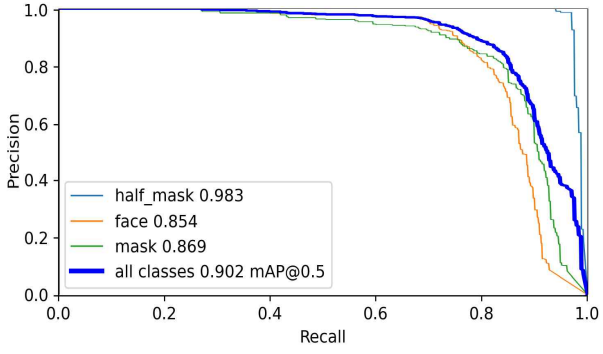

그림 11. YOLOv8n의 PR커브
Fig. 11. PR curve of YOLOv8n

5-4 모델의 크기와 객체탐지 속도

지금까지 제안하는 시스템을 구현하기 위하여 객체탐지 알고리즘인 YOLOv8을 학습시켰다. 이를 위해 사람의 얼굴에 특화한 객체탐지를 해내도록 데이터셋을 구성하였다. 그리고 성능 평가 지표인 mAP를 기준으로 학습 파라미터를 변경해 보며 학습시키면서 적절한 학습 횟수를 찾아내었다. 이후 이 파라미터를 기준으로 같은 조건 하에 모델을 크기별로 학습시켰다. YOLOv8이 수행해야되는 역할은 실시간 객체탐지이므로, 객체를 탐지하는데 드는 속도 역시 중요하다. 탐지속도를 비교하기 위해 웹캠으로 촬영한 영상을 객체탐지하며 평균 탐지속도를 측정하였다.

표 6. 결과 정리
Table 6. Summary of results

Model	mAP	Speed(ms)
YOLOv8n	0.902	50.1
YOLOv8s	0.911	89.7
YOLOv8m	0.913	197.5
YOLOv8l	0.910	340.3
YOLOv8x	0.923	423.8

위는 640×480 해상도와 초당 30의 프레임률을 갖는 웹캠으로 약 5분 30초간 촬영한 영상을 YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, YOLOv8x의 다섯 모델로 같은 조건에서 각각 객체탐지한 결과에서 10000번의 탐지속도 평균을 나타낸 것이다. 가장 작은 모델인 YOLOv8n이 50.1ms으로 가장 따른 탐지속도를 보였으며, 가장 큰 모델인 YOLOv8x가 423.8ms으로 가장 느린 탐지속도를 보였다.

본 논문에서는 실시간 객체탐지 알고리즘을 실행할 하드웨어로 라즈베리파이 3 B모델을 사용한다. 라즈베리파이는 일반적인 PC나 구글 코랩에서 제공하는 런타임 GPU 성능과 비교하여 매우 낮은 성능의 처리 능력을 가지고 있다. 또한 인간의 눈으로 들어온 시각정보가 뇌로 전달되는 데에는 수십 ms의 시간이 필요하며 이를 바탕으로 판단을 내리려면 약 150ms의 시간이 소요된다[11]. 그러므로 수십 ms의 빠른 탐지속도를 가지고 있어야 보다 높은 보안성을 가질 수 있다. 따라서 앞서 진행한 실험 결과를 바탕으로 낮은 처리 성능으로 최대한 빠른 객체 탐지를 수행한다는 목적에 부합하도록 가장 모델 사이즈가 작은 YOLOv8n 모델을 사용하기로 결정하였다.

5-5 COCO 데이터셋으로 학습된 기존의 YOLOv8과의 비교

Ultalytics에서 제공하는 YOLOv8은 기본적으로 MS COCO 2017 Dataset으로 사전 학습(pre-trained)되어 있다. COCO 데이터셋이란 객체 탐지(object detection), 세그먼테이션(segmentation), 키포인트 탐지(keypoint detection) 등의

컴퓨터 비전(computer vision) 분야의 task를 목적으로 만들어진 데이터셋이다. 일상적인 사물과 인간의 이미지 뿐만 아니라, 개체를 인식과 레이블 지정 및 설명하도록 기계 학습 모델 학습에 사용할 수 있는 주석을 포함하고 있다. 객체 탐지(object detection) 분야에서 성능평가 목적으로 주로 사용하는 MS COCO 2017 데이터는 학습(train) 데이터 118,000장, 검증(valid) 데이터 5,000장, 시험(test) 데이터 41,000장으로 구성되어 있다. YOLOv8의 성능 검증 또한 COCO val2017 데이터셋을 기준으로 나타내었다. 표 7은 Ultalytics에서 제공하는 Detection 성능표이다.

표 7. YOLOv8의 감지 성능[6]
Table 7. Detection of YOLOv8[6]

Model	Size (pixels)	mAP50-95	Speed CPU ONNX (ms)	Speed A100 TensorRT (ms)	Params (M)
YOLOv8n	640	37.3	80.4	0.99	3.2
YOLOv8s	640	44.9	128.4	1.20	28.6
YOLOv8m	640	50.2	234.7	1.83	78.9
YOLOv8l	640	52.9	375.2	2.39	165.2
YOLOv8x	640	53.9	479.1	3.53	257.8

표 8. COCO 데이터셋으로 사전 학습된 기존의 모델과 커스텀 데이터셋으로 학습된 제안하는 모델의 mAP 비교
Table 8. Comparison of mAP between pre-trained existing model on COCO dataset and proposed model trained on custom dataset

Model	Provided model	Proposed model
YOLOv8n	0.373	0.588
YOLOv8s	0.449	0.597
YOLOv8m	0.502	0.601
YOLOv8l	0.529	0.598
YOLOv8x	0.539	0.612

표 9. COCO 데이터셋으로 사전 학습된 기존의 모델과 커스텀 데이터셋으로 학습된 제안하는 모델의 탐지속도 비교
Table 9. Comparison of detection speed between pre-trained existing model on COCO dataset and proposed model trained on custom dataset

Model	Provided Model(ms)	Proposed Model(ms)
YOLOv8n	80.4	50.1
YOLOv8s	128.4	89.7
YOLOv8m	234.7	197.5
YOLOv8l	375.2	340.3
YOLOv8x	479.1	423.8

커스텀 데이터로 학습시킨 YOLOv8의 성능과 COCO 데이터셋으로 학습되어 있는 기존의 YOLOv8의 성능을 각각 mAP와 탐지속도 측면에서 비교, 정리하여 아래 표에 나타내었다. Ultalytics에서 제공한 mAP값은 mAP50-95로 제공되었기 때문에 아래 표에서 mAP50-95를 기준으로 하였다.

실제 제안하는 시스템에 적용한 YOLOv8n을 기준으로 비교하였을 때 mAP는 약 1.73배 더 높았으며, 탐지속도는 약 1.60배 더 빠르다. COCO 데이터셋은 일상적인 사물 및 인간의 이미지를 감지할 수 있도록 80개의 클래스(사람, 고양이, 버스, 사과, 책 등)로 사물을 구분한다. 반면에 커스텀 데이터의 경우 사람의 얼굴만을 감지하기 위한 목적으로 데이터셋이 구성되어 있기 때문에 3가지의 클래스(face, mask, half_mask)만을 감지한다. 따라서 어떠한 클래스의 물체를 다른 클래스로 잘못 인식할 확률이 감소하는 효과를 가진다.

이에 따라 커스텀 데이터로 학습시킨 YOLOv8 모델이 사전 학습된 YOLOv8 모델보다 높은 mAP값을 가질 수 있게 되었다. 위와 같은 결과는 커스텀 데이터로 학습된 YOLOv8 모델이 기존의 모델보다 스마트 창문의 보안 기능에 필요한 얼굴 감지 부분에 있어 보다 뛰어난 성능을 보인다는 것을 보여준다.

Ⅵ. 결 론

본 논문에서 YOLOv8에 인간의 눈과 얼굴 이미지 데이터셋을 이용하여 얼굴을 감지해낼 수 있도록 학습시키고, YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, YOLOv8x의 다섯 가지 모델을 각각 평가하였다. 모델의 크기는 오름차순으로 n, s, m, l, x로 나타낼 수 있으며 mAP(mean Average Precision) 및 PR 곡선(Precesion-Recall curve)을 통해 정밀도와 재현율을 평가하였을 때 크기가 큰 모델일수록 높은 정확도를 가지는 것을 확인할 수 있었다. mAP50을 기준으로 하여 가장 작은 모델인 YOLOv8n은 0.902, 가장 큰 모델인 YOLOv8x는 0.923으로 YOLOv8x가 2.1%p 더 높은 성능을 보였다. 반대로 처리속도는 작은 모델일수록 빠른 결과를 나타냈다. 평균 감지속도가 가장 작은 모델인 YOLOv8n은 50.1ms, 가장 큰 모델인 YOLOv8x는 423.8ms으로 YOLOv8n가 373.7ms더 빠른 결과를 보였다.

학습이 완료된 YOLOv8 모델을 라즈베리 파이에 옮긴 후, 연결된 웹캠을 통해 실시간 탐지를 수행하도록 하였다. 라즈베리 파이는 비교적 낮은 처리성능을 가지고 있으며 실내 보안을 위해 빠른 반응이 필요하기 때문에 가장 작은 모델인 YOLOv8n을 선택하였다.

Ultalytics에서 사전학습시켜 제공하는 기존의 YOLOv8 모델과 비교했을 때는 약 1.57배 더 높은 mAP50-95와 약 1.60배 더 빠른 탐지속도를 가지는 것을 확인하였다. 이를 통해 본 논문에서 커스텀 데이터를 통해 학습시킨 YOLOv8 모델이 제안하는 스마트 윈도우 보안 시스템에 있어서 보다 적합한 객체 탐지 모델로 활용될 수 있음을 알 수 있다.

본 논문의 얼굴 감지 모델은 비교적 학습 이미지 자료가 풍부한 서양인에 대한 데이터 비율이 가장 높았다. 이 때문에

한국인에 대해서는 정밀도가 다소 낮아지는 경향이 있었다. 또한 선글라스나 모자, 마스크 착용 등 악세서리를 착용한 모습에 대해서도 탐지 정밀도가 떨어지는 문제가 있었다. 추가로, 날씨가 흐리거나 안개 혹은 비가 내리는 경우 입력되는 이미지의 선명도가 떨어져 감지 성능이 저하되는 문제가 말생할 수 있다. 또한 사용 환경에 따라 감지 성능에 영향을 미치는 요인이 존재할 수 있음을 인지하여, 본 논문에서 마스크를 착용한 이미지에 대해 클래스를 분리하여 학습한 것처럼 다른 경우에 대해서도 클래스를 세분화하여 추가하고, 데이터셋을 보강하여 충분한 학습을 진행한다면 더욱 높은 신뢰도를 가지는 보안 시스템을 구축할 수 있을 것으로 기대된다.

참고문헌

[1] Statistics Korea. 2022 Single-member Households by the Statistics [Internet]. Available: https://kostat.go.kr/board.es?mid=a10301010000&bid=10820&tag=&act=view&list_no=422143&ref_bid=.

[2] P. Martine, S. K. Hong, and H. M. Oh, “Liquid Crystal Based Smart Window with Integrated Luminescent Solar Concentrator,” *Abstracts Presented at the 56th KSIEC Meeting*, Vol. 2017, No. 1, p. 335, November 2017.

[3] L. Jiao, F. Zhang, F. Liu, S. Yang, L. Li, Z. Feng, and R. Qu, “A Survey of Deep Learning-based Object Detection,” *IEEE Access*, Vol. 7, pp. 128837-128868, September 2019. https://doi.org/10.1109/ACCESS.2019.2939201

[4] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 6, pp. 1137-1149, June 2017. https://doi.org/10.1109/TPAMI.2016.2577031

[5] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. Berg, “SSD: Single Shot MultiBox Detector,” in *Proceedings of the 14th European Conference on Computer Vision (ECCV 2016)*, Amsterdam, Netherlands, pp. 21-37, October 2016. https://doi.org/10.1007/978-3-319-46448-0_2

[6] Ultralytics. YOLOv8 [Internet]. Available: https://docs.ultralytics.com/models/yolov8/.

[7] R. Hasegawa, M. Sakamoto, and H. Sasaki, “Dynamic Analysis of Polymer-Dispersed Liquid Crystal by Infrared Spectroscopy,” *Applied Spectroscopy*, Vol. 47, No. 9, pp. 1386-1389, September 1993. https://doi.org/10.1366/0003702934067441

[8] S. C. Jain and D. K. Rout, “Electro-Optic Response of Polymer Dispersed Liquid-Crystal Films,” *Journal of Applied Physics*, Vol. 70, No. 11, pp. 6988-6992, December 1991. https://doi.org/10.1063/1.349828

[9] H. K. Jabbar and R. Z. Khan, “Methods to Avoid Over-Fitting and Under-Fitting in Supervised Machine Learning (Comparative Study),” in *Proceedings of International Joint Conference on AFT, ComNet and PCIE 2014*, Kochi, India, pp. 163-172, December 2014. https://doi.org/10.3850/978-981-09-5247-1_017

[10] I. Kim and C.-H. Lee, “An Efficient Gradient-Based Approach to Optimizing Average Precision through Maximal Figure-of-Merit Learning,” *Journal of Signal Processing Systems*, Vol. 74, No. 3, pp. 285-295, March 2014. https://doi.org/10.1007/s11265-013-0748-0

[11] F. H. Durgin, K. Gigone, and R. Scott, “Perception of Visual Speed While Moving,” *Journal of Experimental Psychology: Human Perception and Performance*, Vol. 31, No. 2, pp. 339-353, 2005. https://doi.org/10.1037/0096-1523.31.2.339

유영창(Young-Chang Yu)

2018년～현 재: 강원대학교 IT대학 전기전자공학과 재학
※관심분야 : 인공지능(AI), 딥러닝, 객체 탐지

김동회(Dong-Hoi Kim)

2005년 : 고려대학교 전파공학과 (공학박사)

1989년 1월～1997년 1월: 삼성전자 전임연구원
2000년 8월～2005년 8월: 한국전자통신연구원 선임연구원
2020년 6월～2020년 6월: 강원대학교 정보화본부장 등
2006년 3월～현 재: 강원대학교 IT대학 전기전자공학과 교수
※관심분야 : 무선 네트워크, 인공지능, 사물인터넷(IoT) 등