

YOLO 네트워크를 이용한 단자 구분

정다운, *정성훈, **김재윤, ***정지훈, ****공경보
부경대학교

ekdnsdl15@naver.com, *tlqwrk915@gmail.com, **1704wodbs@naver.com,
***ljm1222@naver.com, ****kbkong@pknu.ac.kr

Classification of terminal using YOLO network

Daun Jeong *Jeong Seong-Hun **Jaeyun Gim ***jihoon Jung ****Kyeongbo Kong
Pukyong National University

요약

최근 인공지능 기반 객체 탐지 기술이 발전함에 따라 영상 감시, 얼굴 인식, 로봇 제어, IoT, 자율주행, 제조업, 보안 등 다양한 분야에 활용되고 있다. 이에 본 논문은 발전된 객체 탐지 알고리즘을 이용하여 비전문가에게 생소한 컴퓨터나 전기 장치 등의 '단자(terminal)' 모양을 구별하는 방법을 제안한다. 이를 위해 객체 탐지 프로그램인 You Only Look Once (YOLO) 알고리즘을 이용하여 입력한 단자들의 모양을 검출하는 알고리즘을 구성하였다. 일상에서 쉽게 볼 수 있는 단자들의 이미지(VGA, DVI, HDMI, DP, USB-A, USB-C)를 라벨링하여 데이터셋을 구축하였고, YOLOv4와 YOLOv5 두 버전의 알고리즘을 사용하여 성능을 검증하였다. 실험 결과 mean Average Precision(mAP) 기준 최대 92.9%의 정확도를 얻을 수 있었다. 전기 장치에 따라 단자의 모양이 다양하고, 그 종류 또한 많기 때문에 본 연구가 방송 기술 등의 여러 분야에 응용될 것으로 기대된다.

I. 서론

디지털 서비스들이 발전하면서 서비스들을 구현할 디스플레이와 기기들도 나날이 발전하는 추세다. 사람들은 다종의 진보된 서비스를 사용하기 위해 TV, 노트북, 휴대폰을 비롯해 태블릿 PC까지 소유하기도 한다. 발전된 서비스가 즐비한 요즘 디지털 기기들이 없어서는 안 될 존재가 되어버린 것이다. 한 사람당 사용하는 기기의 수가 많아지고 디스플레이의 해상도도 HD, 4K, 8K 등 굉장히 다양하고 선명해졌다. 화면 비율도 필요에 따라서 16:9, 21:9 등으로 그 선택의 폭이 넓어졌다. 이렇듯 다양한 기기들이 생겨나면서 서로 연결시키거나 충전하기 위해 다양한 유형의 케이블과 단자들이 나타났다. 케이블과 단자를 많이 다루는 전문가들에 비해 비 전문가에게는 단자들의 생김새가 익숙지 않아 구분하는 데에 어려움을 겪는다.

C 타입, HDMI, 8핀 등 직관적으로 이해하기 어려운 이름도 단자를 구분하기 어려운 요소 중 하나다. 그래서 본 논문에서는 이런 단자들을 구별하기 위하여 객체 탐지 기술 (Object Detection)을 활용한다. Object Detection 기술은 2개 이상의 객체를 탐지해 분류하는 기술이다. 최근 자율 주행 자동차, 얼굴 인식 등에 Object Detection 기술을 활용하면서^[1] 정확도뿐만 아니라 '실시간'으로 객체의 위치와 class를 구별해 내는 것이 중요해졌다. Object Detection을 위해 구축된 알고리즘은 Convolutional Neural Networks (CNN), You Only Look

Once (YOLO), Faster Region based Convolutional Neural Networks (FASTER R-CNN) 등 여러 알고리즘이 있다. 본 논문에서는 그 중 정확도를 유지하면서 탐지 속도가 아주 빠른 YOLO 알고리즘을 이용하여 단자를 구별한다.

실생활에서 자주 쓰이는 단자 6개를 임의로 선정하여 데이터셋을 구축하고 단자 별로 평균 120-140개의 이미지 데이터를 수집했다. 분류될 단자들의 쓰임은 아래의 표 1에 기술한다.

표 1 라벨링에 사용한 단자들에 대한 설명

VGA	Video Graphics Array. 아날로그 컴퓨터 영상 단자이며, HDMI나 DVI로 대체 가능. 오래됐지만 저가형 모니터가 VGA 케이블을 장착함에 따라 아직 쓰이는 단자.
DVI	Digital Visual Interface. 압축하지 않은 디지털 영상을 세 가지 방식(DVI-D, DVI-I, DVI-A) 중 하나로 전송함. DVI-D는 디지털 신호만 전송 가능하며 DVI-I는 이전의 VGA와의 호환을 위해 아날로그 신호를 전달함.
HDMI	High Definition Media Input. 히타치(Hitachi), 파나소닉(Panasonic), 필립스(Philips), 실리콘 이미지(Silicon Image), 소니(Sony), 테크니컬러 SA(Technicolor SA), RCA, 도시바(Toshiba) 여덟 개 회사가 모여 만든 독점 기술. 기존의 아날로그 케이블보다 고품질 음향 및 영상을 감상 가능.
DP	Display Port. HDMI와 마찬가지로 영상신호뿐 아니라 음성 신호도 전송 가능.
USB-A	USB Type-A(Universal Serial Bus Type A) 컴퓨터와 주변

	기기를 연결하기 위해 사용되는 입출력 표준인 범용직렬버스(USB)의 표준 단자 형태 중 하나. PC 메인 보드나 USB 메모리, TV 등 흔히 볼 수 있는 단자.
USB-C	USB Type-A보다 크기가 작으며 앞뒤의 구분이 없어 요즘 최신 기기들에서 잘 볼 수 있는 타입. 노트북이 경량화되고 얇아지며 2015년 Apple사의 맥북을 선두로 USB Type-C를 채택하여 씀.

본 논문에서 사용한 YOLO 알고리즘은 이미 Object Detection 분야에서 널리 알려져 있다. 이미지를 한 번만 보는 특성으로 인해 그 속도가 매우 빠르고, 높은 정확도를 보이기 때문이다. 객체 인식을 하나의 회귀(Regression) 문제로 접근해 객체를 찾고 객체가 어떤 클래스로 분류되는지 맞혀낸다.^[2] 앞에서 Object Detection을 위한 알고리즘 중 하나로 언급했던 Faster R-CNN은 0.5 Frames per second(FPS)의 성능을 가진 반면 YOLO 모델은 45 FPS의 성능을 가진다. 완벽한 정확도를 가진 모델은 아니지만 실시간으로 많은 이미지의 객체를 탐지할 수 있는 부분에서 메리트가 가진다. 단자들은 각각 고유한 규격을 가지고 있기 때문에, 다른 객체에 비해 쉽게 분류될 수 있을 것이라 예상된다. 따라서 우리는 정확도보다는 속도에 초점을 맞춰 실시간으로 객체 탐지가 가능한 YOLO 알고리즘을 사용한다.

II. Background

2.1. Object Detection

Object Detection이란 이미지 내에서 객체의 종류와 위치를 찾아낸다. 즉, classification과 localization의 개념이 포함되어 있다. Classification이란 분류로서 이미지를 보고 해당 이미지 내의 객체가 어떤 물체인지 구분하는 것이다. 예컨대 새 사진을 보고 '새'라고 답하고 사람 사진을 보고 '사람'으로 분류해 주는 것을 들 수 있다. Localization이란 이미지 내의 객체가 어디에 위치하는지 위치 정보를 출력해 주는 것으로, 주로 Bounding Box로 나타난다. 여기서 Bounding Box란 객체 전체를 포함한 가장 작은 직사각형을 의미한다. X, Y 좌표와 가로, 세로 크기 정보, 정확도를 알려준다. 객체를 얼마나 잘 검출했는지 알 수 있는 지표는 Intersection over Union(IoU)인데, 정답 영역과 예측 영역이 일치하는 정도를 나타낸다. 0에서 최대 1의 값이 나오고, IoU의 값이 1에 가까울수록 객체 검출의 성능이 높다고 할 수 있다.

Deep Learning을 활용한 Object Detection은 크게 one-stage detector와 two-stage detector로 나눌 수 있다. one-stage detector는 앞서 언급했던 classification과 localization을 동시에 하는 방법이며 속도는 비교적 빠르지만, 정확도는 낮을 수 있다. 반면 two-stage detector는 classification과 localization을 순차적으로 진행하여 속도가 비교적 느리지만, 정확도가 더 높다. two-stage detector는 R-CNN 계열이 대표적이며 one-stage detector에는 YOLO 계열이 있다.

본 논문에서는 one-stage detector 중 하나인 YOLO 알고리즘을 사용한다. 그중 YOLOv4와 YOLOv5 두 가지 버전을 사용하여 Object Detection 실험을 진행해 보고자 한다. YOLO 알고리즘의 동향과 간단한 설명은 2.2에 기술한다.

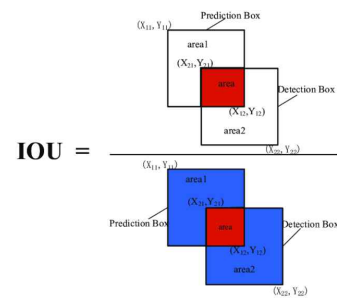


그림 1 Object Detection의 성능을 확인할 수 있는 지표, IoU

2.2. YOLO 알고리즘의 동향

YOLO는 You Only Look Once의 약자로 Object Detection에 사용되는 알고리즘이다. 처음으로 one-stage Detection 방법을 고안해 실시간으로 Object Detection이 가능하게 만들었다. Joseph Redmon이 2015년에 YOLOv1^[3] 논문을 발표하고 공개했다. 본 논문에서 사용할 알고리즘인 YOLOv4^[4]는 Alexey Bochkousky가, YOLOv5는 Glenn Jocher가 발표했다. 두 알고리즘 모두 원저자와 다른 저자가 발표했다.

YOLO는 이미지 전체를 한 번만 본다. 이전의 CNN 모델들은 이미지 하나를 여러 개로 쪼개서 하나의 이미지여도 여러 장의 이미지를 분석하는 방법을 사용했다. 또 다른 YOLO 알고리즘의 특징은 통합된 모델을 사용한다는 것이다. 이 특징들로 YOLO는 실시간으로 객체 탐지가 가능한 모델이다. YOLO는 속도와 정확도 두 가지 지표에서 높은 성능을 보이는 알고리즘이다. 현재는 YOLOv7까지 발표된 상태이다.

III. 방법

3.1. 데이터셋 수집 및 라벨링

VGA, DVI, HDMI, DP, USB-A, USB-C 총 여섯 개의 단자를 class로 설정하고, 이에 대한 이미지를 구글에 검색하여 수집하였다. YOLO 알고리즘은 지도학습으로 시행하기 때문에 레이블링이 잘 된 양질의 데이터 확보가 필요하다. 객체 탐지 문제의 경우, 정답 레이블은 각 객체의 레이블 명과 Bounding Box의 쌍으로 구성되며 이를 Annotation이라고 부른다.^[5] 즉 정답이 될 데이터가 필요하다.

학습에 필요한 데이터 셋을 구축하기 위해 Labeling Image Tool(labelimg) 프로그램을 사용하여 정답이 될 데이터 셋을 만들었다. Class 별로 모은 사진 자료의 수는 표 2와 같다. 웹상에서 이미지를 검색할 때, 특정 단자를 단일로 촬영한 이미지뿐만 아니라 하나의 이미지 안에 여러 단자가 함께 있는 이미지 또한 수집했다. 따라서 그림 2와 같이 하나의 이미지에 class로 지정된 모든 단자들에 대해 라벨링을 진행했다.

labeling은 Object Detection 학습을 위해 모은 사진 자료들



그림 2 labelimg를 활용한 단자 라벨링

표 2 라벨링을 위해 수집한 이미지 수

단자 명	VGA	DVI	HDMI	DP	USB-A	USB-C
개수	200	102	120	150	150	150

에 Bounding Box를 지정하여 라벨링을 수행할 수 있는 도구이다. .xml 또는 .txt파일로 저장할 수 있다. labelimg tool에 사전에 모은 이미지 데이터를 불러온 후 데이터를 입력할 부분, 즉 객체가 위치한 오브젝트 영역에 직접 드래그하여 Bounding Box를 형성한다. YOLO 알고리즘을 활용하기 위해 확장자를 .txt로 지정하여 이미지와 텍스트 파일을 짝지어 데이터 셋을 준비했다.

3.2. YOLO 알고리즘의 객체 탐지 원리

YOLO 알고리즘이 주목을 받은 이유 중 하나는 모든 학습 과정이 이미지 전체를 파악하고, 단일 대상의 특징뿐 아니라 이미지 전체의 맥락을 학습하기 때문이다. YOLO 알고리즘은 예측하려고 하는 원본 이미지를 같은 크기의 $S \times S$ 그리드로 나눈다. Joseph Redmon의 연구에서는 S 를 7로 예시를 두었다. 각 그리드에 대해 그리드 중앙을 중심으로 미리 정의된 형태(predefined shape)로 지정된 B 개의 Bounding Box의 개수를 예측하고 이를 기반으로 각 Bounding Box마다 Confidence Score를 예측한다. 이는 Bounding Box 내에 검출하고자 하는 객체가 존재할 확률을 이야기하며 0에서 1 사이의 값을 가진다. 각 Bounding Box는 $x, y, w, h, confidence$ 를 예측한다. (x, y) 는 Bounding Box의 중심점을 뜻하며 (w, h) 는 width(폭)와 height(높이)를 뜻하는데 전체 이미지에 대해 상대적인 값으로 표현한다.

객체의 위치를 찾은 후, 각각의 그리드는 class의 확률인 C 를 예측한다. 식은 조건부 class 확률인 $Pr(Class_i|Object)$ 를 사용하며 이는 예측한 Bounding Box가 배경이 아닌 Object를 포함하는 경우의 각 class 별 확률이다. YOLO는 Bounding Box의 개수와 상관없이 하나의 class 확률만 예측한다. 그 후 조건부 확률과 각 Bounding Box 가 confidence score를 곱해주어 해당 class 객체가 Bounding Box를 올바르게 예측했는지를 나타낸다. 그 식은 아래와 같다.

$$Pr(Class_i|Object) \cdot Pr(Object) \cdot IOU_{pred}^{truth} = Pr(Class_i) \cdot IOU_{pred}^{truth}$$

IV. 실험 결과

4.1. 환경 세팅

Google에서 제공하는 Colab을 이용하여 알고리즘을 구성한다. Colab(코랩)이란 웹 브라우저에서 프로그램 코드를 작성할 수 있는 클라우드 기반의 Jupyter Notebook(주피터 노트북) 개발 환경이다. 본 논문에서는 CPU와 RAM을 제공해 주고 GPU 사용이 가능한 Google Colab을 사용한다.

4.1.1. YOLOv4를 활용한 실험과 결과

구축한 데이터 쌍들의 70%, 20%, 10%를 각각 Train, Validation, Test에 사용하기 위해 무작위로 나눠주었다. 그림 3은 비율을 정해 나눈 데이터 셋을 시각화한 자료이다. 일반적으로 Training data는 학습만을 위해 존재하고, Test data는 학습이 이미 완료된 모델의 성능을 평가하기 위해 나눠두는 것이다. Validation data 또한 학습이 완료된 모델을 검증하기 위해 나눠두는 데이터이다. 이는 Test 이전에 모델 검증을 위해 관여하며 overfitting 등이 발생했을 때 Validation data를 이용하여 최적의 모델 parameter를 찾는다.



그림 3 YOLOv4 실험 시 나누었던 데이터셋

아래 그림 4는 YOLOv4-tiny-custom을 활용하여 데이터 셋을 학습시킨 후 나온 결과물이다. YOLOv4 알고리즘을 활용한 실험은 YOLOv5를 활용한 실험보다 결과가 좋지 않았다. YOLOv4를 활용한 실험에서는 mAP@0.5 기준 최대 0.696의 성능을 냈다. 그림 4의 좌측 사진을 보면 USB-A class 구별에서 정확도가 떨어지는 모습이 보이며, 우측 사진에서는 USB-C class의 이미지에 정답인 USB-C 일 확률보다 다른 class인 DP 일 확률이 0.81이나 된다는 결과가 나오는 등 비교적 부정확한 결과를 보였다.



그림 4 YOLOv4를 활용한 객체 탐지 결과

4.1.2. YOLOv5를 활용한 실험

Training data와 test data를 그림 6과 같이 전체 이미지 수의 각각 80%, 20%로 맞춰주고, YOLOv5 알고리즘을 활용한 실험은 총 9번 실행했다. epoch, weight size 등 학습 parameter를 계속 변경하며 최적의 parameter 값을 찾기 위해 transfer running을 시행했다.



그림 6 YOLOv5 실험 시 나누었던 데이터셋

이미지 개수가 728장으로 적은 편이었지만 상대적으로 규격화되어 있는 단자들의 이미지였기 때문에 정확도보다는 속도에 좀 더 초점을 두고 실험했다. YOLOv5의 경우 s, m, l, x, s6, m6, l6, x6과 같은 총 8가지의 모델 중에서 관심도에 따라 선택할 수 있다. 따라서 정확도에 초점을 둔다면 정확도가 가장 높지만, 검출속도가 상대적으로 느린 x6를, 검출 속도에 초점을 둔다면 정확도는 다소 높지 않지만, 검출속도가 빠른 s를 선택하여 객체 탐지를 할 수 있다는 장점이 있다.^[6] 따라서 weight size를 s size와 m size 위주로 실험을 진행했다. 또한, 이미지 수가 적다 보니 과도하게 정답 데이터 셋에 맞춰 학습하는 overfitting 현상이 일어날 것을 대비해 epoch 수를 최대 200으로 잡았다. 아래 표 3은 변경하면서 학습한 parameter 별 mean Average Precision(mAP)을 내림차순으로 정렬해둔 것이다. mAP@0.5에서 0.5는 IoU의 기준값을 나타낸 것이다. IoU≥0.5의 조건을 만족하는 것을 맞는다고 정의하고 평균값을 낸 것을 의미한다.

실험 결과, parameter를 epoch=100, weight size=s로 설정하여 학습시켰을 때 0.929mAP로 최대의 성능을 나타냈다. 모든 실험에서 일정 수준 이상의 정확도를 보였지만 weight size가 작은 s와 m에서 성능이 mAP 0.92를 웃도는 높은 수치를 보이는 경향이 있었다. 수치로 따지면 epoch 100, weight size s 일 때 가장 성능이 좋았지만 class 전반적으로 수치가 떨어지지 않은 것은 epoch 50, weight size m인 경우였다.

표 3 parameter별 학습 결과(mAP@0.5)

	parameter		mAP@0.5 ▽
	epoch	weight size	
1	100	s	0.929
2	50	m	0.926
3	50	s	0.925
4	100	m	0.922
5	200	s	0.913
6	100	l	0.909
7	200	l	0.905
8	20	m	0.893
9	200	m	0.881

4.2. YOLOv4-tiny-custom과 YOLOv5를 이용한 실험 결과 비교

그림 7은 하나의 같은 이미지를 각각 YOLOv4-tiny-custom 알고리즘과 YOLOv5 알고리즘에 test image를 각각 입력하여 결과를 살펴보았다. YOLOv4-tiny-custom을 사용한 실험에서는 표 4와 같이 USB-C 단자를 다른 class인 DP라고 0.81의 확률로 예측하는 등의 아쉬운 결과를 보였다. 그러나 YOLOv5m을 이용한 실험에서는 모든 단자를 일정 수치 이상의 정확도로 알맞게 예측한 결과가 나타났다.



그림 7 YOLOv4-tiny-custom과 YOLOv5m 알고리즘을 이용한 실험 결과 비교

표 4 실험 결과 수치 비교

그림 6 기준	class	USB-C (정답)	DP (오답)
예측값	YOLOv5m	0.95	-
	YOLOv4-tiny-custom	0.48	0.81

V. 결론 및 향후 연구 방향

본 논문에서는 Object Detection 기술을 활용하여 컴퓨터나 전기장치 등의 단자(terminal)를 구별하는 방법을 제안했다. 최근 들어 Object Detection 분야가 특히 주목받아 빠르게 발전하고 있는 분야이고, 다른 분야와의 협업을 기대할 수 있다.

본 연구에서는 데이터의 수가 비교적 부족했다는 한계점이 있었다. 하지만 비교적 규격화되어 있는 ‘단자’를 구별한 것, 이미 한 장 내에 여러 개의 Bounding Box를 생성하여 수집한 이미지 수 보다는 더 많은 수의 데이터로 진행한 것, 이 두 가지 이유로 한계점이 보완될 수 있었다. 거기에 방대한 데이터를 통해 학습되어있는 YOLO 알고리즘을 이용했기 때문에, 위 결과와 같이 좋은 성능이 나온 것으로 해석될 수 있다. 다음 연구에서는 더 큰 데이터 셋으로 실험을 진행할 예정이다.

참고 문헌 (References)

[1] 이진수, 이상광, 김대욱, 홍승진, 양성일.(2018). 딥러닝 기반 객체 인식 기술 동향.[ETRI] 전자통신동향분석,33(4),0-0.

[2] 오다솜, 이강복, 홍상기. (2021). 소방시설 점검 지원을 위한 yolov4 기반 소방시설 객체인식. 한국정보과학회 학술발표논문집, (), 512-514

[3] Joseph Redmon, Santosh Divvala, Ross Girshick, Ali Farhadi, "You Only Look Once: Unified, Real-time Object Detection", IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, pp.779-788, June 2016.

[4] Alexey Bochkovskiy, Chien-Yao Wang, Hong-Yuan Mark Liao. "Yolov4: Optimal speed and accuracy of object detection", arXiv preprint arXiv:2004.10934, 2020

[5] 오진선, 천인국. (2020). 딥러닝 객체 탐지 기술을 사용한 스마트 쇼핑카트의 구현. 한국산학기술학회 논문지, 21(7), 262-269.

[6] 김익수, 이문구, & 전용호. (2021). 결함검출 적용을 위한 YOLO딥러닝 알고리즘 비교. 한국생산제조학회지, 30(6), 514-519.