<AI-실전프로젝트>

License Plate Recognition(LPR)

2020 AI 실전프로젝트

학부: 기계제어공학부

팀명: Judo

팀원1: 한선도 21400793

팀원2: 김주호 21600144

1. Project Goal & Plans

1.1 project Goal

YOLO라는 프로그램을 사용하여 차량번호판을 인식하고 Py-tesseract 라이브러리를 통해 OCR을 활용하여 차량번호판 내의 Text들을 인식하는 모델을 만드는 것이 기본 목표이며 심화 목표로 Object recognition을 실제에서 사용될 수 있는 business 모델로 학교로 들어오는 버스를 인식하고 버스의 차량번호판을 실시간으로 인식하여 해당 시간대에 출입한 버스에 대한 정보를 제공하여서 코로나 상황에서의 확진자 동선과 본인의 동선을 확인할 수 있도록 하는 단계까지 구현하는 것이다. 하지만 15분에 1대씩 들어오는 버스 간격과 낮은 노트북의 사양으로 인한 시스템 딜레이로 인해 실시간 구현을 직접 하는 것은 어려울 것으로 판단하였고, 실시간 model과 동일한 방식의 Video에서 구현하고 버스대신 가지고 있는 차량을 통해 움직이는 차량의 번호판이 버스가 들어올 때의 속도인 10~30km/h 구간에서 LPR(License Plate Recognition) 되는 것을 최종 project 목표로 설정하였다.

1.2 project plans

Week	계획	비고
4	YOLO를 Colab에서 실행하는 예제 코드 수행	
5	영문 차량 번호판 data 수집	
6	Image에서의 Object Detection	
7	Image에서 OpenCV를 활용한 ROI 및 Plate recognition	
8	Image 에서의 YOLO를 활용한 ROI 및 Plate Recognition	
9	Video에서 YOLO를 활용한 Object Detection	
10	Midterm Report	
11	영상으로부터 입력되는 Frame 수 변경 후 LPR 수행	
12	Local 환경에서 Webcam을 통한 Object Detection 수행	
13	Local 환경에서 Webcam을 통한 Plate Recognition 수행 버그 수정 및 모델 튜닝	
14	Business model을 적용 및 영상환경의 차이에 따른 LPR 결과 분석 및 개선점 정리	
15	Final Report	

2. 프로젝트 수행 과정 및 결과 요약

2-1. YOLO를 통한 Object Detection

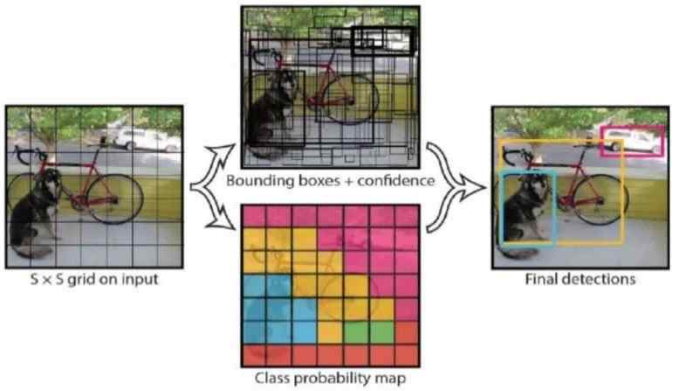

Figure1. YOLO의 원리

이번 프로젝트에서의 핵심기술인 YOLO(You Only Look Once)는 Object detection을 수행위해 사용된 프로그램이다. YOLO의 원리는 Input image 내의 bounding box와 class probability를 single regression problem으로 간주하여, 이미지를 한 번 보는 것으로 object의 종류와 위치를 추측하고 single convolutional network를 통해 다수의 bounding box에 대한 class probability(탐지된 객체가 어느 특정 클래스에 속하는지에 대한 확률)를 계산하여 출력하는 방식이다.

자세히 설명하면, Input image가 들어왔을 때 이미지를 SxS Grid cells로 나누고, 최종 결과로 출력되는 (7, 7, 30) 텐서를 예측하는 CNN 네트워크를 구축하고 각각의 grid cell에서는 대상 이미지들의 데이터 정보를 담고 있는 weight file을 data기반으로 bounding box들을 생성하고, 각각의 box들은 Confidence score를 가지게 되어 예측된 boxes의 수와 관계없이 가장 높은 Confidence score를 가지는 단 하나만의 객체만을 탐지한 box를 출력하게 된다. Single convolutional network를 통한 간단한 처리과정과 매우 빠른 속도는 기존의 real-time detection system들과 비교할 때, 2배 정도의 높은 속도를 가지는 장점이 있으며 각 cell마다 하나의 객체만을 예측할 수 있기 때문에 여러 겹쳐 있는 객체에 대한 탐지 정확도가 상대적으로 낮은 것이 단점이다. 아래 Figure는 이번 프로젝트를 진행했을 때의 Object detection 결과이다.

Figure2. YOLO Object detection 결과

2-2. 이미지 전처리과정

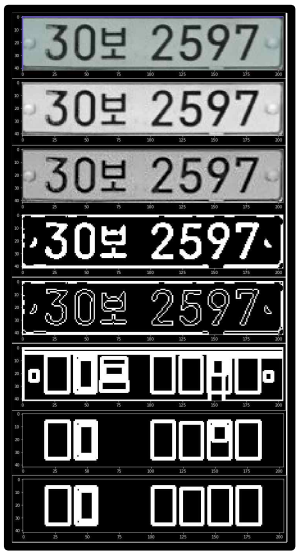

Figure3. 이미지 전처리 과정 및 결과

위의 Figure는 이번 프로젝트를 진행하면서 이미지 전처리를 수행한 결과이다. 이미지 전처리는 OCR의 정확도를 높이기 위해 OCR을 수행하기 전 OCR에 적합한 이미지로 전처리해주는 과정으로 아래와 같은 순서로 이루어진다. 원본 이미지의 다양한 상태는 매우 다양하므로 컴퓨터가 이미지를 쉽게 인식할 수 있게 여백 제거, 레이아웃, 밝기, 대비, 해상도 등을 조정하는 이미지 정제작업이다. 이미지 전처리의 주목적은 Contours를 찾아내는 것이다. Contours란 이미지의 연속된 점과 색깔을 분석한 등고선 형태의 경계로, 간단하게 사물의 경계선으로 이해할 수 있으며 OCR을 수행되는 ROI라고 할 수 있다. Contours 추출을 통해 박스 형태의 경계선을 찾아내고 문자가 있을 만한 box만을 대상으로 인식 범위를 좁혀 OCR의 정확도를 높이는 것이다. 박스 형태로 Contours를 찾는 이유는 글자의 각도와 비틀림을 고려하여 이미지들을 다시 반듯하게 맞추어서 인식하기 위함이다.

1. Grayscale
2. TopHat, BlackHat, plus Grayscale
3. Thresholding, GaussianBlur
4. Find contours(특징점 찾기)
5. 사각형 형태의 multiple box 그리기
6. 최소 사각형 크기를 제한하여 문자가 존재할 만한 사각형만을 추려냄.

2-3. Py-tesseract를 활용한 OCR

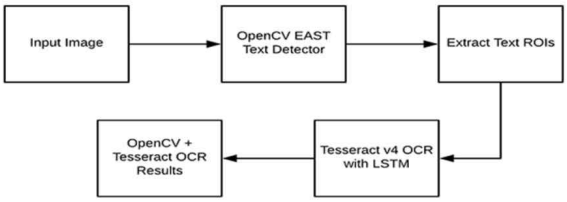

Figure4. OCR 수행 과정

이번 프로젝트에서는 Python 환경에서의 OCR 프로그램인 Py-tesseract를 통해 YOLO를 통해 Object detection된 차량번호판을 ROI로 설정하여 그 ROI 내에서 OCR을 수행하였다. 이미지 전처리를 통해서 추출된 사각형 박스에서 각 언어에 대한 정보를 가지고 있는 Tassdata를 다운받아 문자 이미지에서 OCR을 수행할 수 있다. Py-Tesseract 옵션을 통해서 OCR의 engine mode를 설정할 수 있으며 읽어드리는 Text 형식에 따라 이미지에 대한 Layout 분석 방법을 선택하여 설정할 수 있으며, 이번 프로젝트에서는 일반적인 차량 번호판 Text의 모양인 'Single text line'으로 설정하였다.

OCR을 수행할 때는 대상 이미지의 다수가 유사한 특성(사이즈, 해상도 등)을 가지고 있으므로, 인식할 텍스트의 위치가 규칙적인 경우 등 인식 조건이 특정된 경우에는 정확도가 높게 나타나는 반면 실제 규칙성 없고 다양한 환경에서의 이미지에 적용하는 데에는 여러 변수가 작용하여 인식률이 낮게 나타나기도 한다. 실제 이미지에서 글자 인식이 어려운 이유는 아래와 같은 변수 특성들 때문이다. 아래의 OCR 결과에서 볼 수 있듯이 이미지 자체 data를 기반으로 인식하기 때문에 '고'를 '끄'로 인식하는 등 한글 첨자에 대해서 잘 못 인식하는 경우도 발생한다. 이러한 문제점에 대해서는 첨자에 대한 한글 데이터의 추가적인 학습 데이터가 필요로 된다.

대상	인식 대상이 어디에 위치해 있을지 예상할 수 없음
초점	초점에 따라 피사체의 경계가 뚜렷하지 않을 수 있음
화질	카메라의 성능에 따라 전반적인 화질 차이가 존재함
노이즈	카메라의 센서 노이즈는 일반적으로 스캐너보다 많음
각도	텍스트가 평행하지 않은 각도를 가질 수 있고 이미지 자체가 회전될 수 있음
흐림	렌즈가 깨끗하지 않거나 손떨림 등으로 이미지가 흐려질 수 있음
조명	심하게 어둡거나 카메라의 플래시가 켜져 있는 등 잘 안 보이거나 번질 수 있음
해상도	이미지의 각 해상도는 정해져 있지 않음
폰트	다양한 형태의 폰트가 존재함
비평면	사물 위의 텍스트는 표면 형태에 의해 왜곡됨

Figure5. OCR 작업의 변수들

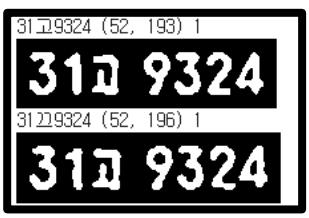

Figure6. Object detection된 차량번호판 OCR 결과

2-4. Local 환경에서 실시간으로 LPR 구현 및 한글 Text 출력

Local 환경에서의 실시간 LPR을 구현하기 위해 먼저, python에서 YOLO를 활용한 Object detection을 먼저 수행해보았는데 Real-Time에서 webcam으로부터 정해진 프레임을 입력 받아 LPR이 수행되는 방식은 동영상에서의 LPR이 수행되는 방식과 동일하게 이루어진다. 실시간 이미지 처리에서는 컴퓨터의 성능에 따라서 인식 속도에 대해 time delay가 발생하며, 노트북의 성능과 webcam의 성능, 그리고 Bus에 대한 model test가 여건상 연속적으로 test하기 어려우므로 동영상에 대한 LPR 모델 구현을 수행하였다.

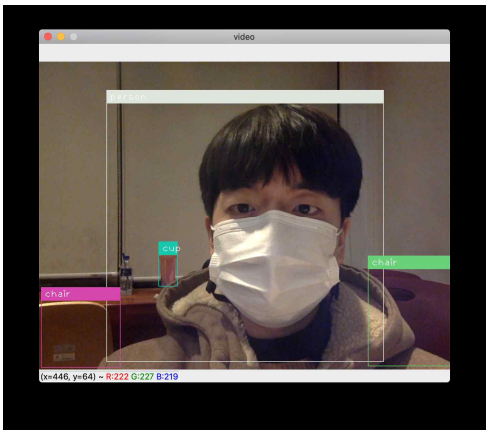

Figure7. Local 환경에서의 Webcam에서 실시간 Object detection

동영상을 가지고 프로젝트를 진행하면서 인식과 동시에 영상에 OCR된 결과 값을 출력 화면에 바로 출력시키는 작업에서 Opencv가 유니코드를 지원하지 않아서 Opencv 기반 함수인 put text를 통해서 한글 문자에 대해 출력하지 못하는 사실을 깨닫게 되었고, 유니코드를 지원하는 PIL 라이브러리에 있는 draw.text 함수를 사용하여 한글 출력하여 아래와 같은 결과를 얻어낼 수 있었다.

Figure8. 동영상에서의 LPR 결과 (OCR 결과 한글 출력)

2-5. 카메라 재원별 LPR 성능 비교

해상도	차량 속도(km/h)	최초 OCR 인식 해상도	OCR 신뢰도(%)
FHD 1920x1080 (30FPS)	10	**49x168**	88.67
	20	47x163	83.33
	30	47x166	58.27
FHD 1920x1080 (60FPS)	10	48x170	86.33
	20	52x186	80.00
	30	49x172	78.79
QHD 2560x1440 (30FPS)	10	**55x204**	87.27
	20	57x220	93.33
	30	60x239	93.33
UHD 3840x2160 (30FPS)	10	**73x290**	90.63
	20	73x307	88.00
	30	69x277	93.75
UHD 3840x2160 (60FPS)	10	68x281	76.92
	20	66x268	82.35
	30	72x308	66.67

Figure9. 카메라 재원에 따른 최초 OCR 인식 해상도와 신뢰도

LPR의 성능에 많은 영향을 끼치는 original image file에 대해서 LPR 성능이 최적화되기 위한 조건을 알아보기 위해 현재 휴대폰에서 지원하는 다양한 해상도에서 10~30km/h 속력별로 동영상을 촬영하여 최초로 OCR이 인식되었을 때의 해상도와 OCR의 결과값들을 통해 신뢰도를 계산하여 표로 정리했다.

표에서 확인할 수 있듯이 카메라의 최초 OCR 인식 해상도는 같은 프레임(30FPS)을 기준으로 하였을 때 FHD, QHD, UHD 순으로 증가하는 것을 볼 수 있었다. 이는 YOLO에서 차량번호판이 Object detection될 때 일정 비율로 Gird를 형성하여 Object가 탐색하기 때문에 해상도가 높아질수록 하나의 Grid에 할당되는 해상도 또한 증가하게 되어 해상도의 증가에 따라 최초 OCR가 수행된 해상도도 증가하게 되는 것이다. 이는 FHD를 통해서도 충분히 다음과 같이 주어진 환경에서 LPR을 수행하는데 문제가 없다는 것을 알 수 있었다.

다음은 카메라의 해상도와 초당 프레임 수(FPS)에 대한 OCR의 신뢰도에 대한 결과를 차량 속도 30km/h일 때 값으로 비교해 보았는데 여건상 여러 차량에 대해서 실행시켜보지 못한 점과 data의 수가 작아서 카메라의 성능에 비례하여 OCR의 신뢰도가 높아지는 등 정확한 결과를 얻지는 못하였으나 test sample이 많아진다면 카메라 성능이 좋아질수록 OCR의 정확도가 높아질 것이다. 또한 이러한 결과는 이번 프로젝트에서 LPR 시스템의 목적과 용도에 비추어 보았을 때 모든 카메라 환경에서 실시간으로 입력되는 여러 프레임에 대한 OCR의 정확도가 50% 이상이 된다면 결국 가장 OCR에서 빈도가 높게 나온 값을 최종 출력으로 정하는 시스템에서 정확도 100%의 OCR을 수행해 낼 수 있다고 결론을 내릴 수 있었다.

2-6. 비즈니스 모델 구현

비즈니스 모델 구현의 핵심기술은 Object detection과 OCR 결과의 정확도라고 생각되어 이 두가지에 초점을 두어 프로젝트를 진행하였으며 최종적으로 10~30km/h의 속력을 가진 차량의 번호판을 인식하는 모델을 구현하였다. 비즈니스 모델을 구현하고 실제로 모델이 사용될 환경에 대한 조건으로 아래의 두가지 필요조건과 설치조건을 가정하였다.

<필요조건>

- 버스를 가지고 지속적인 실험이 어려우므로 승용차를 통해 실험
- 버스가 정류장으로 들어올 때의 속도를 기준으로 30km/h 이하의 속도에서 적용 가능한 모델 구현

<설치조건>

- 버스가 들어올 때 속도가 느려지는 구간(ex. 과속방지턱 주변)에 설치

또한 Model 적용시 2가지의 조건문을 통해 OCR 결과로 출력되는 Text의 OCR의 신뢰도를 높여주었다. 첫번째 조건으로 Text에 할당되는 byte의 크기에 제한을 두어 한글 차량번호판에서 주로 인식되는 형태일 때 받아지는 값들로 조건을 두었다. 한글은 3byte, 숫자는 1byte로 보통 single line 차량 번호판일 경우 '00한0000' or '000글0000' 형식이므로 9, 10 byte로 조건을 만들어 주었다. 두번째 조건으로는 출력되는 Text의 글자수에 대한 조건으로 보통 single line 차량 번호판일 경우 '00한0000' or '000글0000' 형식이므로 7~8개로 이루어진 결과만을 출력되도록 하는 조건을 만들어 주어 OCR 결과값 출력시 올바른 결과 값의 형식을 갖추지 않은 출력값들을 1차적으로 제거해 주어 OCR의 신뢰도를 높여주었다.

```
length = len(result_chars.encode('utf-8'))
if len(result_chars) == 8 or len(result_chars) == 7:
  if length == 9 or length == 10:
    cv2_imshow(img_result)
    print(result_chars,img_result.shape) # 한 프레임 내 OCR로 인식된 번호들의 집합
    plate_chars.append(result_chars)
```

Figure10. 비즈니스 모델 적용과 OCR결과 신뢰도를 높이기 위한 조건문 code

Figure11. 비즈니스 모델 LPR 결과 (QHD 2560x1440 (30FPS)로 촬영)

3. 프로젝트 수행 과정에서 배운 점

이번 프로젝트를 진행하면서 LPR을 수행하는데 주로 사용되었던 YOLO와 OCR의 원리에 대해서 자세히 알아볼 수 있는 시간이었다. YOLO의 원리인 Grid 생성을 통한 Single convolutional network 구조로 이전까지 사용되었던 R-CNN보다 훨씬 빨리 물체를 인식하는 것을 알게 되었다. 또한 OCR을 수행하는데 있어서 원본 이미지를 바로 사용하지 않고 이미지 전처리 과정들을 먼저 수행하는 것을 통해 OCR이 원활하게 이루어지도록 하는 원리에 대해서 알게 되었다.

하나의 프로젝트를 진행하면서 처음 설정한 목표에 도달하기까지 전체 시스템의 방향성과 시스템을 이해하기 위해 먼저 배워야 할 것들을 단계적으로 구성하고 계획하여 성취해 나가며 목표에 점점 가까워지는 방법이 프로젝트 진행과 구현하는 과정에 있어서 꽤나 중요하다는 것을 경험할 수 있었고, 뚜렷한 기준과 목표 설정이 프로젝트 진행을 좀 더 뚜렷하고 원활히 진행될 수 있게 해준다는 것을 깨달을 수 있었다.

진행했던 LPR 모델 프로젝트를 비즈니스 모델로 적용하면서 특정 기술을 실제에서 적용할 때 적용하려는 환경과 조건을 먼저 고려하여 그에 맞추어 필요로 되는 device의 성능 수준을 결정하고 원하는 결과값을 출력해 내기위해 맞춤형 모델을 설계하는 것을 통해서 불필요하게 초과되는 성능과 경제적으로 모델링하는 것의 필요성을 느낄 수 있었다.

4. 수행 프로젝트 관련하여 추가로 수행을 제안하는 과제 (Future Work)

이번 프로젝트에서는 LPR 모델을 가지고 가장 대중적인 single text line의 한글 차량 번호판을 읽는 것이 주된 목표였으며, 필요로 되는 목적에 따른 모델 구현에 있어서 추가로 수행이 필요로 될 수 있는 과제들은 아래와 같다. 아래 적힌 Future work 과제들은 data 수집과 모델 test 등 여건상 모델을 테스트 해보기 어려웠던 과제들로 다른 상황에서 LPR 시스템을 사용할 때에 인식해야 할 대상으로 사용될 가능성이 높은 것들을 제시하였다.

1. 구형 번호판 인식

- 서울 경기 등 지역명이 포함된 구 번호판
- Double line 형식의 구 번호판

2. 신형번호판 인식

- 한글 앞 숫자가 세자리('000가0000' 형식)인 번호판

3. 다양한 번호판 인식 실험

- 다양한 나라의 번호판
- 전기차 인식(다른 번호판 색)
- 차량과 번호판을 동시에 인식 (차종+번호판 출력)