Pusan National University

Computer Science and Engineering

Technical Report 2021-19

Low Quality CCTV Processing System

저자 1: 박찬호

저자 2: 조현덕

저자 3: 강병민

지도교수: 차의영 교수님

목 차

1. 서론

1.1. 연구 배경

저가의 카메라로 촬영한 저해상도의 사진을 소프트웨어적으로 업그레이드해 해상도를 크게 높일 수 있게 만들고자 한다. 카메라가 흔들리거나 물체가 움직여 발생하는 영상 흔들림(blurring)문제도 해결해 깨끗한 영상을 복원할 수 있다. 영상 초해상도 기술은 영상의 질을 높여야 하는 곳이면 모두 사용할 수 있다. 우선 CCTV에서 멀리 떨어진 자동차를 구별하기 힘들 때도 영상을 확대해 세밀하게 볼 수 있다. 야간에는 차량 번호판을 확인하기 어려운데 크게 확대하면 가능하다. 차량 블랙박스 영상의 화질도 개선할 수 있다. 범죄 과학수사에도 요긴하기 때문에 국립 과학 수사 연구원이나 검찰, 경찰 등에서도 활용할 수 있다. 도로나 건물 내부 등에 CCTV가 많지만 10~20m만 떨어져도 피상체의 크기가 작아지고 해상도가 높지 않은 카메라의 경우 식별하기 어려움이 있다. 비싼 고해상도 카메라로 바꾸는 것이 쉽지 않은데 딥러닝 기반 알고리즘 소프트웨어를 활용하여 고해상도 카메라 운용 비용을 절감할 수 있게 된다. 압축, 손상 및 왜곡된 영상을 복원하는 학습 모델을 통해서 영상을 원본에 가깝게 복원할 수 있다.

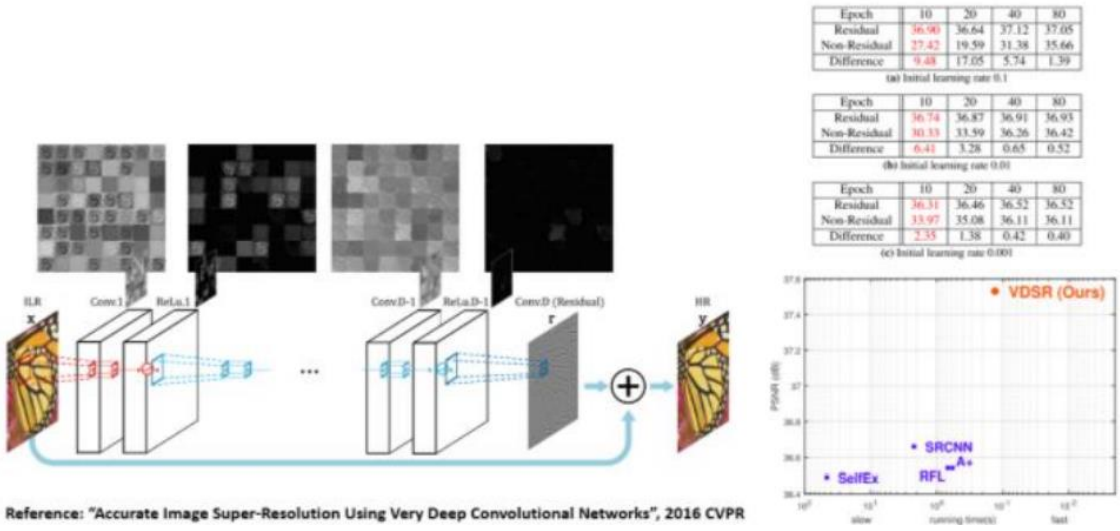


그림 1.

자기공명영상(MRI)과 컴퓨터단층촬영(CT)등 의료용 영상에도 적용 가능해 정밀 판별을 할 수 있다. 초고화질 TV(UHD TV)의 고해상도 영상 변환이나 스마트폰 화질 개선, 반도체와 부품 결 함 검사 등에도 유용하다. 위성영상에 작게 보이는 자동차도 확대할 수 있다. 4차 산업혁명의 핵심분야로 꼽히는 드론, 지능형로봇, 자율주행차의 영상 분석에도

필수적인 기술로 부상할 것으로 전망된다.

1.2. 기존 문제점

도로나 건물 내부 등에 CCTV가 많지만 10~20m만 떨어져도 피상체의 크기가 작아지고 해상도가 높지 않은 카메라의 경우 식별하는데 어려움이 있다. 비싼 고해상도 카메라로 바꾸는 것이 쉽지 않은데 딥러닝 기반 알고리즘 소프트웨어를 활용하여 고해상도 카메라 운용 비용을 절감할 수 있게 된다.

1.3. 연구 목표

CCTV 영상 처리를 통한 화질 개선 및 원본 추측, 그리고 빅데이터 수집을 위한 컴퓨터 비전 & 딥러닝 모델 개발에 목표를 둔다.

2. 연구 배경

2.1. 연구 개발 환경

CPU	Intel(R) Core(TM) i7-6700 @ 3.40GHz
HDD	500GB
RAM	32GB
GRAPHIC	NVDIA GeForce GTX 1080
OS	Window 10 Pro

2.2. GAN (Generative Adversarial Nets)

GAN(적대적 생성 신경망)은 두 개의 신경망 모델이 서로 경쟁하면서 더 나은 결과를 만들어내는 학습이다. Generator(생성자)와 Discriminator(판별자)로 구성된다. 생성자 G는 실제 데이터의 분포를 흉내내는 훈련을 반복하고, 판별자 D는 G가 생성한 것과 원래 데이터를 구별한다. G는 이미지를 잘 생성해서 속일 확률을 높이고 D는 구분할 확률을 높이는 것을 목표로 한다. 이 과정이 계속 반복되면 특정 시점에서 G가 x의 분포를 정확하게 모사하는 수준에 이르게 되고 D는 원래 데이터와 모사본을 구분하지 못하게 된다.

2.2.1. SRGAN

기존 SR문제에서 정확도와 속도 측면에서 많은 발전을 이루었지만 질감의 디테일이 떨어지는 문제가 있었다. 기존의 SR모델은 MSE(Mean Square Error) loss를 사용했지만 SRGAN에선 adversarial loss와 content loss로 구성된 perceptual loss를 이용해서 사람이

보기에 더 좋아 보이는 이미지를 생성하게 된다.[1]

2.2.2. ESRGAN

ESRGAN (Enhanced Super-Resolution Generative Adversarial Networks)은 SRGAN을 3가지 측면에서 개선했다. 첫째로 배치 정규화 (batch normalization layers) 없이 Residual-in-Residual Dense Block (RRDB)을 사용해서 더 심층적인 모델을 사용한다. 둘째로 Relativistic average GAN을 사용한다. 셋째로 activation전에 features를 이용해서 perceptual loss를 개선한다.[2]

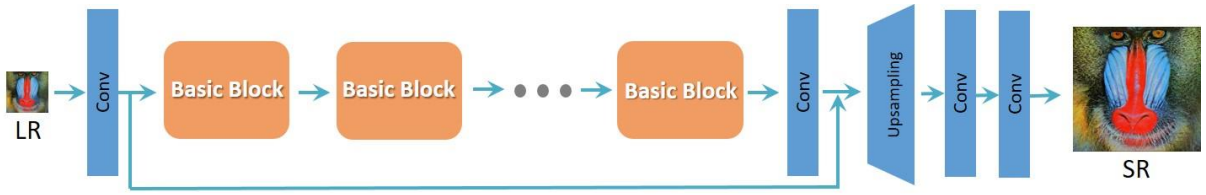


그림 2. 기본 아키텍처

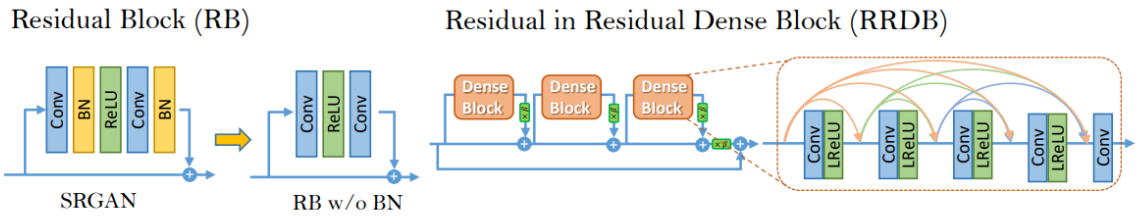


그림 3. ESRGAN에서 RRDB가 basic block으로 사용된다

2.3. YOLO v4

최신 Neural Networks들은 높은 정확도를 가지지만, 낮은 FPS와 너무나 큰 mini-batch-size로 인해 학습하는데 많은 수의 GPU들이 필요하다는 단점이 있다. 이러한 문제를 해결하기 위해, YOLO v4는 다음과 같은 기여를 제공한다.

1) 일반적인 학습환경에서도 높은 정확도와 빠른 object detector를 학습시킬 수 있다.

2) detector를 학습하는 동안, 최신 BOF, BOS기법이 성능에 미치는 영향을 증명한다.

3) CBN, PAN, SAM을 포함한 기법을 활용하여 single GPU training에 효과적이다.

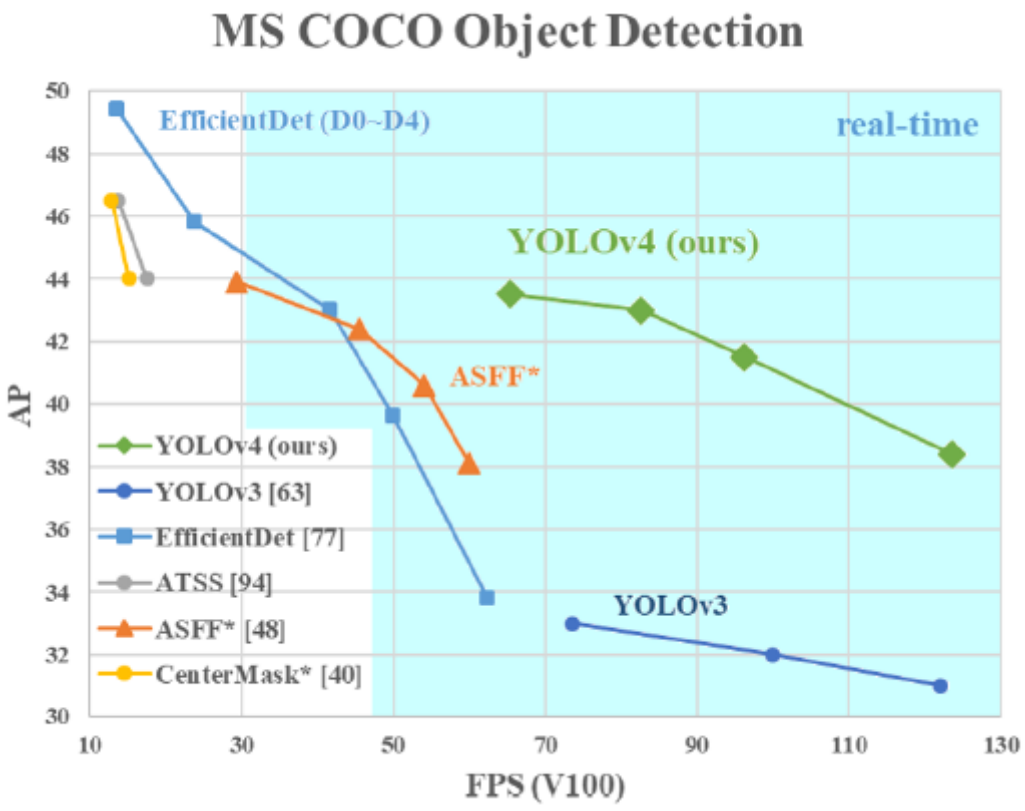


그림 4.

최신 detector는 주로 백본(Backbone)과 헤드(Head)라는 두 부분으로 구성된다. 백본은 입력 이미지를 feature map으로 변형시켜주는 부분이다. ImageNet 데이터셋으로 pre-trained 시킨 VGG16, ResNet-50등이 대표적인 백본이다. 헤드는 백본에서 추출한 feature map의 location작업을 수행하는 부분이다. 헤드에서 predict classes와 bounding boxes작업이 수행된다.

헤드는 크게 Dense Prediction, Sparse Prediction 으로 나뉘는데, 이는 Object Detection 의 종류인 1-stage 인지 2-stage 인지와 직결된다. Sparse Prediction 헤드를 사용하는 Two-Stage Detector 는 대표적으로 Faster R-CNN, R-FCN 등이 있다. Predict Classes 와 Bounding Box Regression 부분이 분리되어 있는 것이 특징이다. Dense Prediction 헤드를 사용하는 One-Stage Detector 는 대표적으로 YOLO, SSD 등이 있다. Two-Stage Detector 과 다르게, One-Stage Detector 는 Predict Classes 와 Bounding Box Regression 이 통합되어 있는 것이 특징이다. [3]

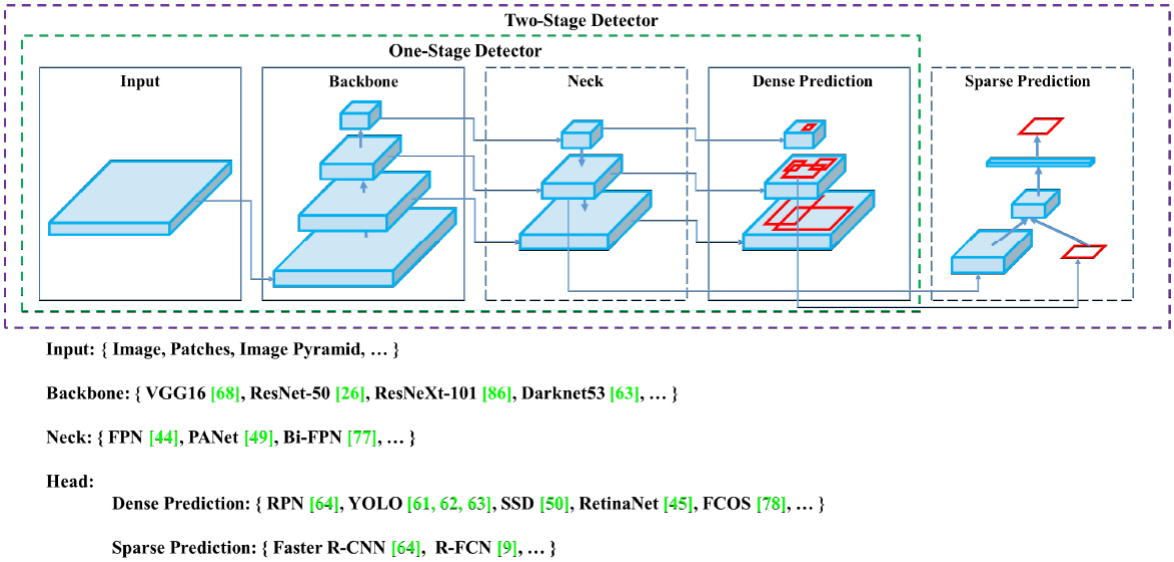


그림 5.

2.4. Tesseract OCR

Tesseract는 다양한 운영체제를 위한 광학 문자 인식 엔진이다. 이 소프트웨어는 Apache License, 버전 2.0에 따라 배포되는 무료 소프트웨어이며 2006년부터 Google에서 개발을 후원했다.

버전 3.00부터 Tesseract는 출력 텍스트 형식, hOCR위치 정보 및 페이지 레이아웃 분석을 지원했다. Leptonica 라이브러리를 사용하여 여러가지 새로운 이미지 형식에 대한 지원이 추가되었다. Tesseract는 텍스트가 단일 간격인지 또는 비례 간격인지 감지할 수 있다.

또한 Tesseract는 백엔드로 사용하기에 적합하며 OCRopus와 같은 프론트 엔드를 사용하여 레이아웃 분석을 포함하여 보다 복잡한 OCR작업에 사용할 수 있다. 입력 이미지가 이미지에 맞게 사전 처리되지 않은 경우 출력 품질이 매우 떨어진다. 이미지의 크기를 조정하고 회전 또는 기울어짐을 수정해야 한다. [4]

2.5. Open Image V6

Open Images는 컴퓨터 비전 작업을 위한 최신 심층 컨볼루션 신경망(CNN) 훈련에 사용하기 위해 여러 측면에서 가장 큰 주석이 달린 이미지 데이터셋이다. 19년 5월 버전 5가 소개되면서 Open Images 데이터셋에는 36M 이미지 레벨 레이블이 붙은 9M 이미지, 15.8M 경계 상자, 2.8M 인스턴스 세그먼트 및 391k 시각적 관계가 포함된다. 데이터셋 자체와 함께 관련 Open Images Challenges는 물체감지, 인스턴스 분할 및 시각적 관계

감지의 최신 기술의 발전을 촉진시켰다. [5]

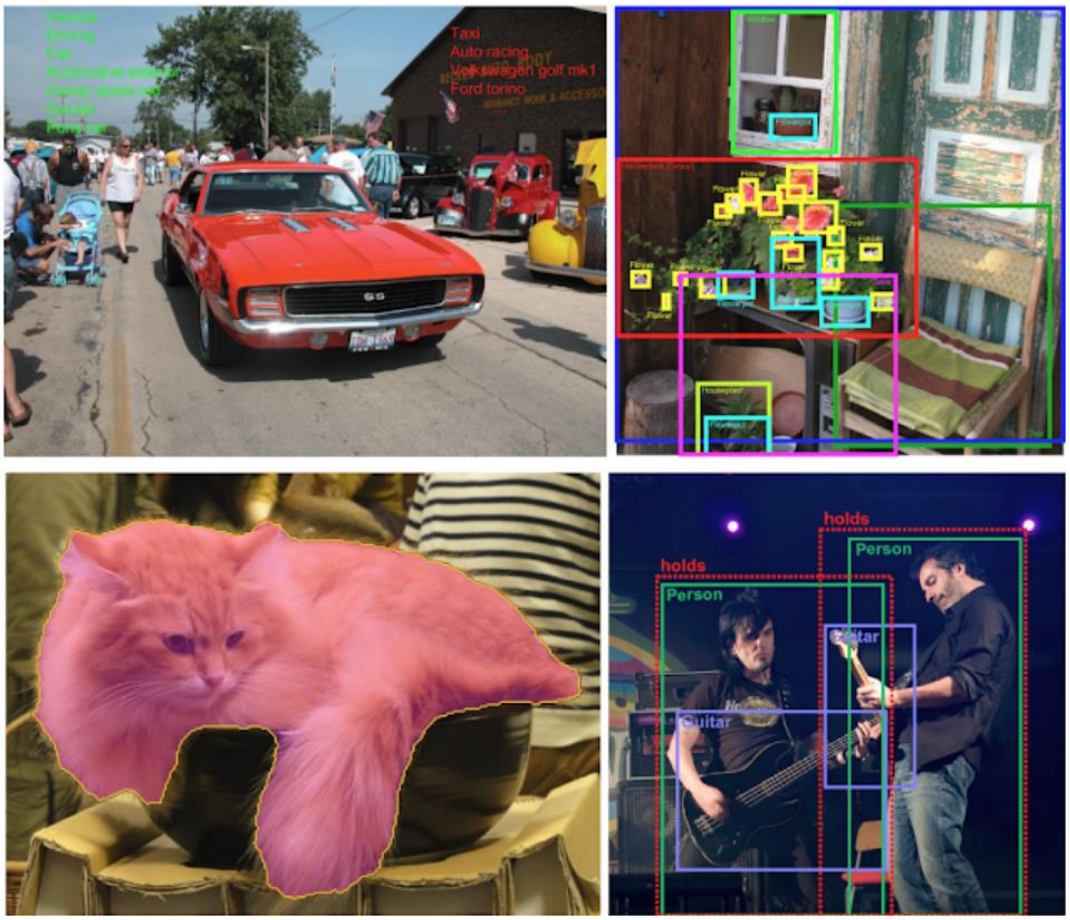


그림 6.

3. 연구 내용

3.1. 번호판 초해상화

번호판의 화질을 개선하기 위해 ESRGAN을 이용한다. 여러 모델을 시험해본 결과 ESRGAN이 성능이 가장 좋았다.

3.1.1. 학습 데이터셋 구하기

초해상화(Super Resolution)를 학습하기 위한 고해상도의 한국 번호판 데이터 셋이 필요하다. 마땅한 데이터셋이 없었기 때문에 한국 번호판 생성 코드를 이용했다. 번호판 생성 코드는 숫자 0~9, 번호판에 쓰이는 한글 중 39개, 지역 이름 16개의 이미지를 조합해서 번호판을 생성한다. 출력 이미지는 흑백이다.

그림 7. 생성된 번호판

3.1.2. 학습 진행

원활한 학습진행을 위해 크기가 큰 이미지들을 100x100으로 잘라 32400개의 이미지를 만들었다.

그림 8. 잘린 번호판

학습을 진행하면 5000번의 iteration마다 모델을 저장한다. 학습이 진행되면서 나오는 모델들로 실제로 어떻게 결과가 나오는지 확인을 해보면서 학습을 하였다.

그림 9. 차례대로 원본, 5000, 50000, 125000 iteration 모델로 SR한 결과

학습이 진행될수록 번호판의 숫자와 문자가 개선되는 것을 확인할 수 있다.

3.2. YOLO 객체인식

3.2.1. YOLO v3 기반 오픈소스 사용

초기 yolo v3 기반 오픈소스 weigth 파일 사용하여 차량의 번호판 부분 추출

- 객체 탐지 결과

그림 10. [원본영상화질: 1080p] [4 배 확대]

발견된 YOLO v3 기반 weight 의 문제점

(1) 차량 크기가 작을 경우 인식률 급격히 저하, 동일한 영상이라도 확대했을 때 차량을 인식하는 것을 확인할 수 있었다. 학습된 이미지 데이터의 크기가 대체로 컸었고 학습 데이터의 이미지 크기가 절대적으로 영향을 미친다는 것을 확인했다. 따라서 영상에서 원하는 크기의 번호판을 인식하려면 새로운 데이터로 학습시키거나 적절한 weight 를 찾아야 한다.

(2) 차량 번호판 인식 성공률 높지 않음. 학습된 데이터 셋 대부분이 외국 차량의 외국 번호판이었기 때문에 수십 대의 차량에서 한 두 대의 번호판만 인식할 수 있는 문제가 있었다.

(3) CCTV 영상 각도에서 인식률 낮음

-> training data 의 객체 촬영 구도도 탐지에 큰 영향이 있었다. CCTV 의 각도와 유사한 각도로 촬영 된 data 를 충분히 학습 시킨 yolo 모델 필요하다.

3.2.2. 학습용 CCTV 데이터 마크

환경에 맞는 YOLO weight 학습이 필요하다고 판단하여 사용하는 CCTV 와 유사한 영상을 확보, 0.5 초 단위로 이미지 파일 저장하여 태그 및 학습을 시도하였다.

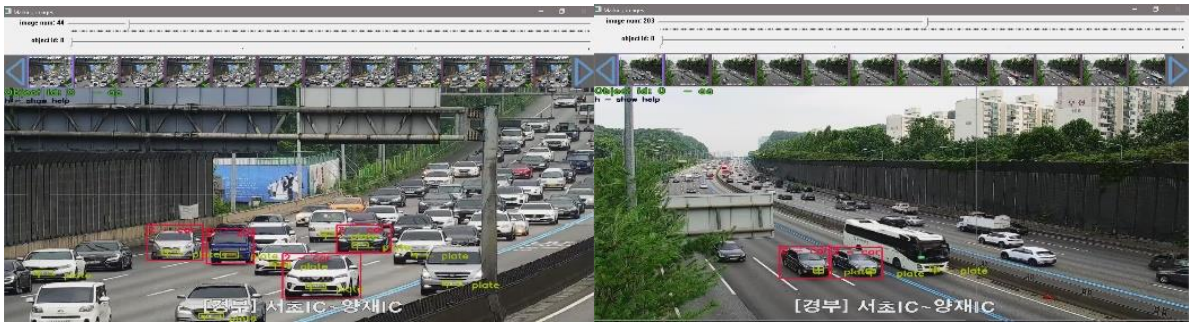


그림 11.

-데이터셋 출처: 서울 교통정보 시스템 TOPIS,

총 4 개의 장소에서 CCTV 영상을 추출하였고, 각 영상들을 1 초단위로 분할하여 이미지로 만든 뒤에 캡처한 이미지에 Yolo Mark 툴을 이용하여 Yolo 학습용 데이터를 만들었다.

-총 400 장의 Yolo Mark 로 태그한 학습데이터로 weight 학습하였다.

-기대한 성능이 나오지 않았고 객체를 충분히 탐지해내지 못했다. 산학 협력 기업 중간멘토링에서 학습데이터가 너무 적은 점과, Yolo version 3 의 한계점을 지적해주어서 연구의 방향을 바꾸었다.

3.2.3. YOLO v4로 weigh model 학습

(1) Yolov4 제공 weight model

그림 12.

Yolov4 에서 제공하는 기본 모델을 사용하여 CCTV 이미지를 스캔한 결과이다. 탐지되는 클래스가 많고, 정확도가 매우 높으나 번호판은 기본 인식 객체에 없기 때문에 커스터마이징이 필요했다.

(2) 번호판 인식 Yolo v4 사용

이미 학습 완료된 yolo v4 모델

https://github.com/theAIGuysCode/yolov4-custom-functions 의 custom.weight) 사용

인식된 번호판들을 따로 저장하여 데이터 셋을 만드는 함수 작성

그림 13. [원본 영상 인식률] **[X2 영상 인식률]**

이미지 크기에 비해 작은 이미지에 대한 인식률이 높음

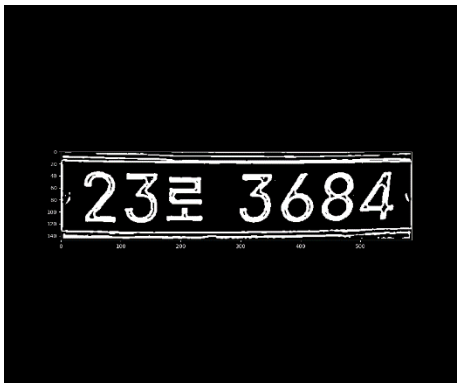
-> 인식률이 높고, 인식한 번호판 영역의 정확도가 높기 때문에 이 모델을 사용하여 연구를 진행하기로 결정했다.

3.3. Tesseract OCR

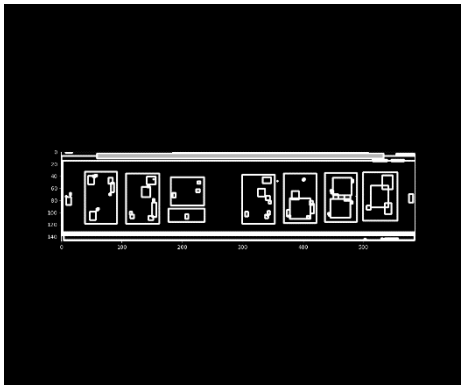
3.3.1. 번호판 이미지 전처리 과정

그림 14. [탐지된 번호판]

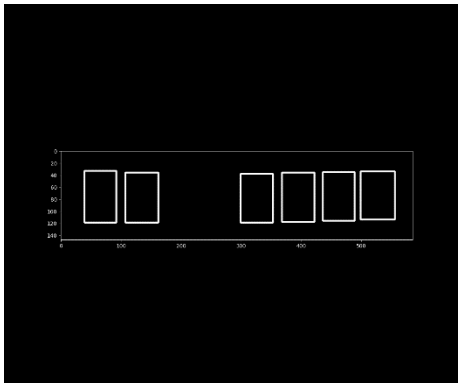
탐지된 번호판을 Tesseract OCR 로 인식을 시도할 시에 인식 에러 발생확률 높다. 사용결과 OCR 의 성능이 그렇게 정교하지 않아서 text 의 크기, 회전각도, 노이즈, 색깔 등에 민감하게 영향을 받는 것을 확인하였다. 따라서 탐지된 번호판의 text 를 인식하기 이전에 전처리 과정은 필수적이다.



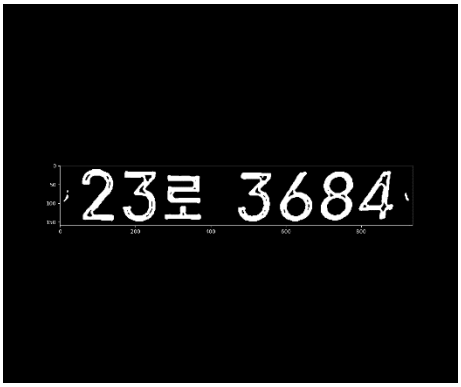
[threshold 적용]



[contour 추출]



[후보 추출]



[OCR Input data]

그림 15. 전처리 과정

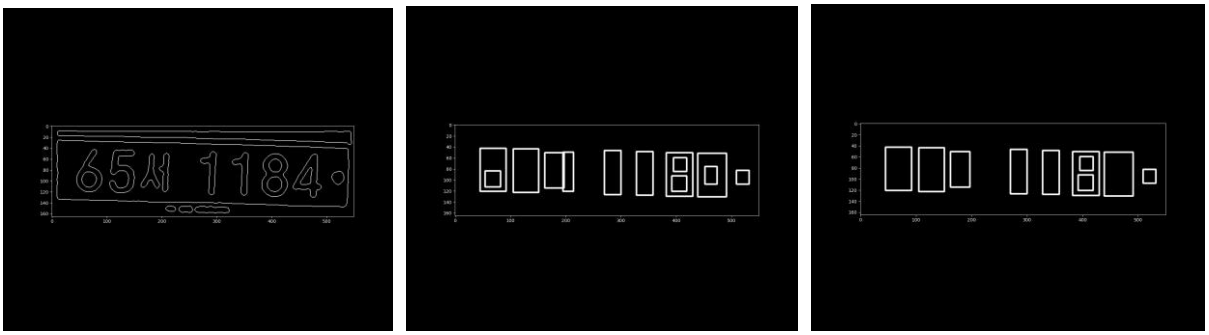
전처리 이후 라이브러리 pytesseract 의 image_to_string 로 번호판의 text 를 인식하였다.

전처리 과정은 크게 7 가지 과정을 거친다.

(1) 먼저 입력 받은 이미지가 색깔의 영향을 받지 않도록 gray scale 로 변환하고 노이즈에 영향을 받지 않도록 가우시안 블러링을 수행한다.

(2) 일정 밝기 이상의 픽셀은 더 밝게, 이하는 더 어둡게 하여 텍스트와 판의 경계를 뚜렷하게 한다.

(3) 미리 정해둔 thresh hold 값으로 이하의 밝기를 가진 픽셀은 0, 이상의 밝기를 1 로 설정한다.

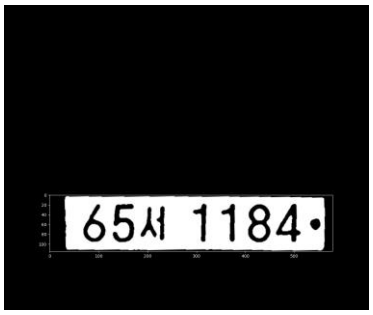


(4) 이미지의 모든 contour 들을 찾아서 위치를 저장한다.

(5) contour 중에서 가로 세로의 비율이 text 에 가까운 contour 만 남긴다.

(6) 남은 contour 중에서 다른 contour 와 정렬이 되어있는 contour 들이 최종 후보이다.

(7) 해당 영역을 자른 뒤 OCR 을 수행하도록 넘긴다.



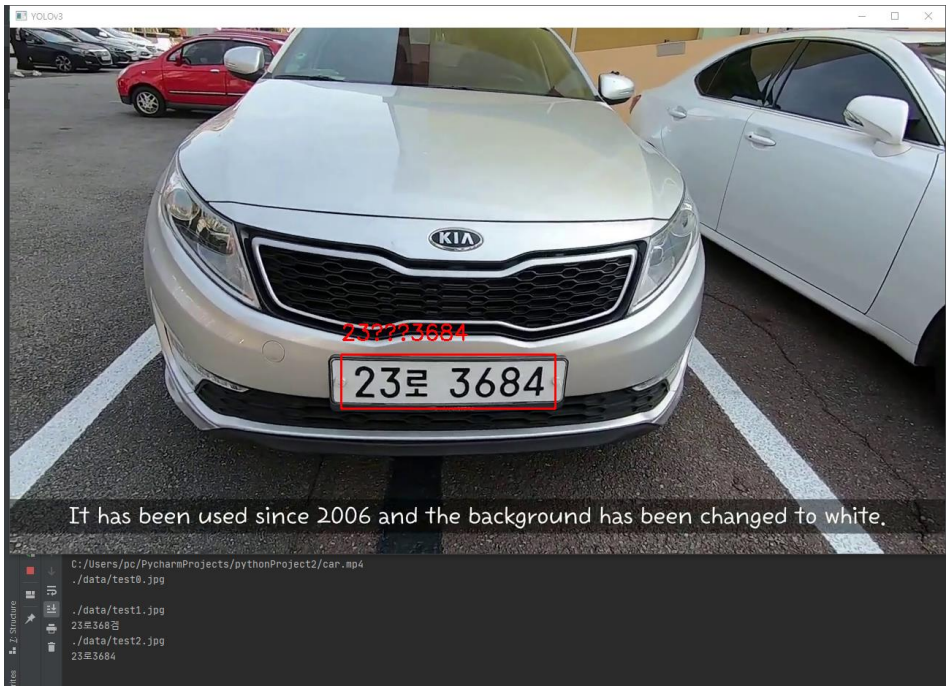


그림 16. 최종 처리 화면

3.3.2. 저화질 번호판

그림 17. 탐지된 저화질 번호판

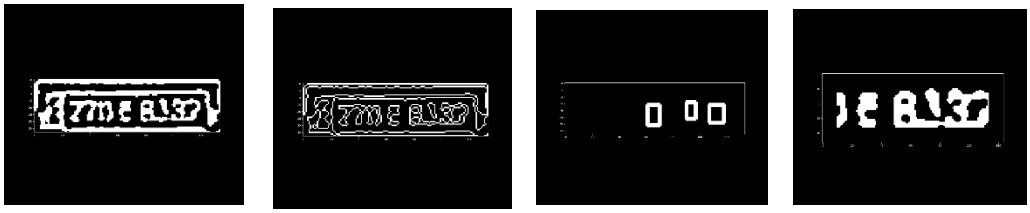


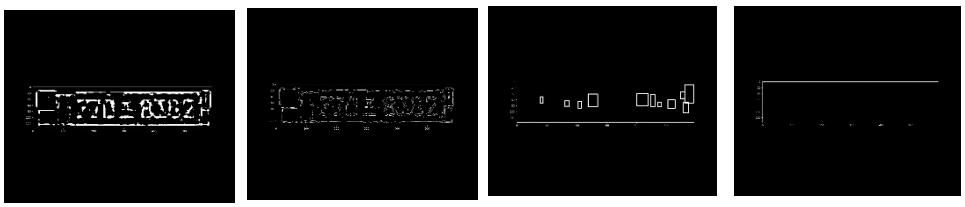
그림 18. 저화질 번호판 이미지 전처리 과정

저화질 번호판의 인식과정은 생각보다 까다롭다. 그 이유는 이전의 전처리 과정으로 처리했을 때 저화질로 인해 뭉개져서 연결되는 부분이 하나의 contour 가 되어서 개별의 문자로 인식되어야 할 것들이 하나의 contour 가 되어 문자 후보에서 빠지기 때문이다. 따라서 이전의 처리과정을 적용했을 때에 그림처럼 엉뚱한 영역을 번호영역으로 추출하거나 아예 text 후보를 찾지 못해서 번호영역이 없다고 판단하게 된다.

Super Resolution 으로 화질 개선 이후 번호판 인식 과정

[개선 전] [개선 후]

육안으로는 인식률이 상승한 듯 보인다.



[개선된 번호판 이미지 전처리 과정]

Super Resolution 으로 복원한 뒤 오히려 인식률이 낮아지는 문제가 있었다.

-> 인식한 번호판 영역의 text 들이 Super Resolution 과정에서 모두 하나의 contour 로 이어져버렸기 때문에 다른 방법을 찾아야 했다.

3.3.3. 끊어서 인식시키는 방법

그렇다면 문자를 모두 끊어서 인식시키는 방법을 고려해보아야 했다. 이 경우 일반적 번호판의 글자 비율을 짐작해서 글자를 끊어야 했기 때문에 정확도가 낮았고 전처리 과정이 매우 길어졌다. 그리고 단일 문자를 OCR 로 인식한다고 해도 인식률 굉장히 낮았다. OCR 은 조금의 노이즈나 회전에도 민감하게 반응하기 때문에 글자를 아예 인식하지 못하거나 숫자를 한글로 인식하는 문제가 있었다.

하지만 이 문제는 Super Resolution 의 방식을 변경하면서 해결되었다.

4. 연구 결과 분석 및 평가

4.1. 성능분석

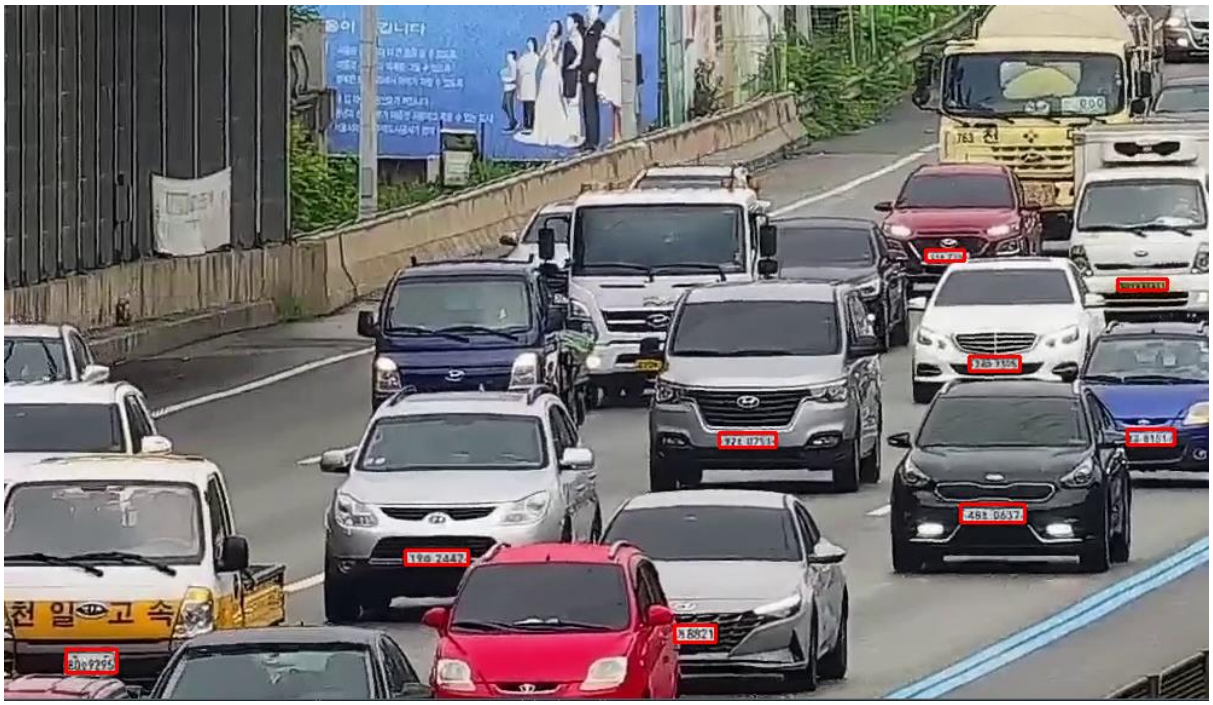


그림 19. 인식된 저화질 번호판 영역

구현한 객체 인식 모델이 위의 그림과 같이 CCTV 영상에서의 번호판 영역을 거의 대부분 잡아낸다. 이렇게 잡아낸 번호판들은 먼저 Super Resolution의 과정을 거친다.

그림 20. CCTV에서 인식된 한 번호판의 영역

다음 그림과 같이 저화질 CCTV에서 추출된 번호판이 Super Resolution의 과정을 거치지 않고 OCR로 text 인식을 한다면 결과는 다음과 같다.

```
C:\Users\pc\PycharmProjects\pythonProjects\venv\Scripts\python.exe
Detected characters in input2.jpg: None of character is recognized

Process finished with exit code 0
```

그림 21. 결과 프롬프트 창

당연히 어떠한 문자도 인식하지 못한다.

그림 22. Super Resolution 수행 후

다음은 Super Resolution을 거친 동일한 이미지이다. 가로 세로 4배가 확대된 원본 추측 이미지를 얻을 수 있다. 이 이미지로 OCR 문자 인식을 수행한다면

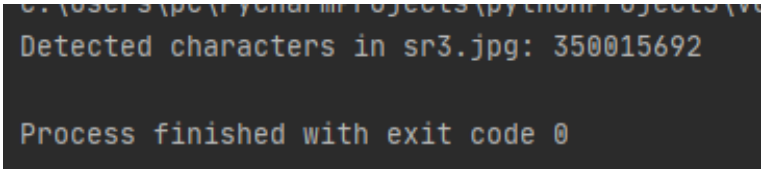


그림 23. 결과 프롬포트 창

다음과 같이 인식된 결과를 확인할 수 있다. 숫자들을 다 정확히 인식 했으나 중앙의 한글인 '이'를 '01'로 인식하는 문제가 있었다. 아무 것도 인식하지 못했던 것보다는 나은 결과이나 OCR 학습 데이터와 복원된 한글이 모양이 달라서 인식한 최선의 결과가 '01' 이었다.

그림 24. Super Resolution 복원 전 후의 이미지

다음의 복원 결과도 높은 성능을 보여준다. 매우 흐릿한 이미지였으나 학습된 번호판 데이터를 바탕으로 뚜련한 text를 복원하였고 전기차 마크까지 또렷히 복원해냈다.

4.2. 유의 사항

같은 번호판에 대해 입력 크기에 따라 다른 결과가 출력된다. 이유는 Super Resolution 모델이 학습된 이미지 크기에 민감하기 때문이다.

그림 25. 저화질 번호판 '770조 8387'

SR 모델의 학습데이터의 크기는 대략 높이 150 픽셀 정도이다. 같은 번호판 이미지를 각각 다른 사이즈로 입력했을 때 나오는 결과물을 비교하면 다음과 같다.

그림 26. 학습데이터 크기보다 작은 입력에 대한 결과

그림 27. 학습데이터 크기와 유사한 입력에 대한 결과

그림 28. 학습데이터 크기보다 큰 입력에 대한 결과

다음과 같이 같은 번호판이라도 입력되는 이미지의 크기에 따라 다른 결과를 보인다. 학습데이터보다 작은 경우 픽셀이 부족해서 제대로 복원 되지 않고 뭉개지는 모습을 볼 수 있다. 반면 학습데이터보다 입력 이미지의 크기가 클 경우 중간의 '8' 부분에서 윗 쪽과 아래 쪽에 '8'을 두 개 유추했듯이, 하나의 글자 안에서도 여러가지의 글자를 추측하는 모습을 볼 수 있다. 학습데이터와 유사한 크기일 경우 복원 성능이 가장 높았다. 그렇기 때문에 객체 인식으로 탐지한 번호판의 크기를 학습데이터와 유사한 크기로 resize한 뒤에 Super Resolution을 진행하여야 최고의 성능을 기대할 수 있다.

5. 결론 및 향후 연구 방향

본 과제에서는 ESRGAN을 활용하여 저화질 CCTV영상의 화질을 개선시키는 실험을 진행하였다. CCTV의 영상을 입력받고 YOLO v4 모델을 이용하여 번호판을 객체로 인식받는다. 번호판을 Super Resolution 모델인 ESRGAN을 활용하여 화질을 개선한 후에 Tesseract OCR을 통해 문자를 인식한다. 실험결과 Super Resolution을 적용한 모델이 적용이 되지 않은 모델보다 문자를 인식하는 정확도가 높음을 보였다. 이는 실생활에서도 저화질 CCTV영상을 고화질로 개선할 수 있음을 의미한다. 범죄나 재난 예방 등 다방면으로 사용될 수도 있으며, 오래된 영상 복구에도 사용될 수 있을 것으로 보인다. 또한 정확한 영상 데이터를 제공함으로써 도움을 줄 수 있다.

향후 연구 과제로는 데이터 전처리 과정이 보다 간단하게 이루어질 수 있도록 기법을 개선하는 것이 필요하다. 또한 본 논문에서 제안한 기법의 프로세싱 속도를 높일 수 있도록 보완하여 실시간으로 화질을 개선하는 방안을 연구하는 것이 향후 과제가 될 수 있다. 한편 제안한 기법의 외부적인 한계점으로 컴퓨터가 고성능이 아니라면 기법을 적용하기 어렵다는 점을 들 수 있고, 딥러닝을 시키기 위한 전처리 과정이 수작업으로 진행된다는 점이 보완돼야 할 부분이다.

6. 참고 문헌

[1] Christian Ledig, “Photo-Realistic Single Image Super-Resolution Using a Generative Adversarial Network” CVPR 2017

[2] Xintao Wang, “ESRGAN: Enhanced Super-Resolution Generative Adversarial Networks” ECCV 2018

[3] Alexey Bochkovskiy, "YOLOv4: Optimal Speed and Accuracy of Object Detection" 2020

[4] Ray Smith, Hewlett-Packard, [Online]. Available: https://ko.wikipedia.org/wiki/%ED%85%8C%EC%84%9C%EB%9E%99%ED%8A%B8

[5] Jordi Pont-Tuset, [Online]. Available: https://ai.googleblog.com/2020/02/open-images-v6-now-featuring-localized.html