소셜미디어의 로고 감지를 통한 맥주 브랜드별 노출도 파악

-일본불매운동 전후 비교-

목차#

- 문제 정의
- 연구 과정
- 데이터 정의
- 모델 정의
- 성능 평가
- 인사이트

01# 문제 정의

"일본 제품 불매운동 효과, SNS에서도 드러날까?"

사실상 '전멸' 日 맥주…"싸게 줄 테니 진열이라도"

김세진 | 기사입력 2019-11-05 19:49 | 최종수정 2019-11-05 19:50

일본 맥주 | 불매운동 | 편의점 | 수입맥주

【서울=뉴시스】박미영 기자 = '안사고, 안가고, 안 입는' 일본제품 불매운동이 100일을 맞았다.

일본 정부의 무역보복 조치로 촉발된 'NO 재팬' 운동은 현재 진행 중이다. 일부에서는 일본제품 구매가 다시 재개되는 움직임도 포착되고 있지만 여전히 '노 재팬' 운동은 국민들의 자발적인 동참으로 일본 제품이 좀처럼 끼어들 틈을 찾지 못하고 있다.

사실상 퇴출된 품목은 아사히 맥주 등 일본산 맥주다.
일본 맥주의 퇴출은 편의점과 슈퍼 등 소매채널이 앞장선 '안사고, 안파는' 불매 운동의 힘을 가장 잘 보여준 사례다.

8일 관세청에 따르면 지난 9월(잠정치) 일본 맥주 수입액은 6000달러(약 700만원)에 그쳤다. 이는 전년대비 99.9% 감소한 수치로, 일본 맥주가 사실상 수입 중단 수준으로 떨어진 것이다.

- 2019년 7월부터 일본 불매운동이 시작되며 일본 제품의 판매가 급격하게 감소했고, 그 중 **일본 맥주는 불매 운동의 영향을 가장 많이 받은 상품** 중 하나이다.
- 또한, 맥주는 **브랜드 로고 노출도가 높은 상품**이면서, 온라인 상에 인증하는 **젊은 세대의 특징을 가장 잘 담은 소비재**이다.
- 이에 따라 SNS 상의 일본 맥주 브랜드의 노출도 역시 줄었을 것이라 판단하여, **불매운동을 기점으로 각 맥주 브랜드 노출의 변화 양상과 특징을 분석**하고자 한다.

02# 연구 과정

03# 데이터 정의

날짜 기준

- **일본 무역 수출 규제에 대한 불매운동 시작일**을 기준으로, 금년 3/4분기와 작년 동기를 비교하기 위한 데이터 수집

브랜드 선정 기준

- 작년 4/4분기, 금년 3/4분기 **소매점 상위 매출 브랜드**[1] 및 **일본 유명 브랜드**를 합한 10개

수집 사이트

- **학습 데이터**
  : 구글, 네이버 (약 6600장)
- 모델 정확도 **검증용 데이터**
  : 인스타그램 (200장)
- **실제 테스트 데이터**
  : 페이스북 (약 4000장)
- **#맥주 #편맥** 사용하여 수집

1) FIS 식품산업통계정보 POS소매점 매출액 기준

03# 데이터 정의

금년 3/4분기와 작년 동기를
비교하기 위한 데이터 수집

일본 유명 브랜드를 합한 10개

- 모델 정확도 **검증용 데이터**
  : 인스타그램 (200장)
- **실제 테스트 데이터**
  : 페이스북 (약 4000장)
- **#맥주 #편맥** 사용하여 수집

1) FIS 식품산업통계정보 POS소매점 매출액 기준

03# 모델 정의

+

=

Classification

- 이미지에 class를 할당

Localization

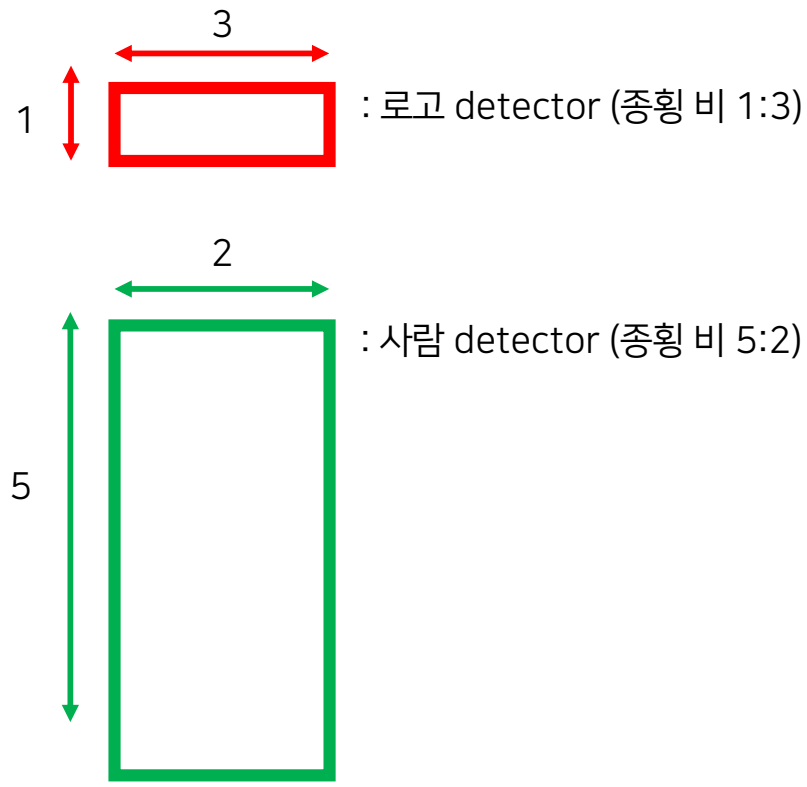
- Object가 이미지 안의 어느 위치에 있는지 **위치 정보를 출력**
- 주로 Bounding Box를 이용하여 출력

Object Detection

- **Classification + Localization**
- **1 stage network:** Anchor Box를 사용하여 전체 이미지에서 영역(region)에 대해 예측
- **2 stage network:** Region Proposal을 사용하는 방식

03# 모델 정의 – YOLO

YOLO에서 Detection을 하는 방법 : anchor box를 이용

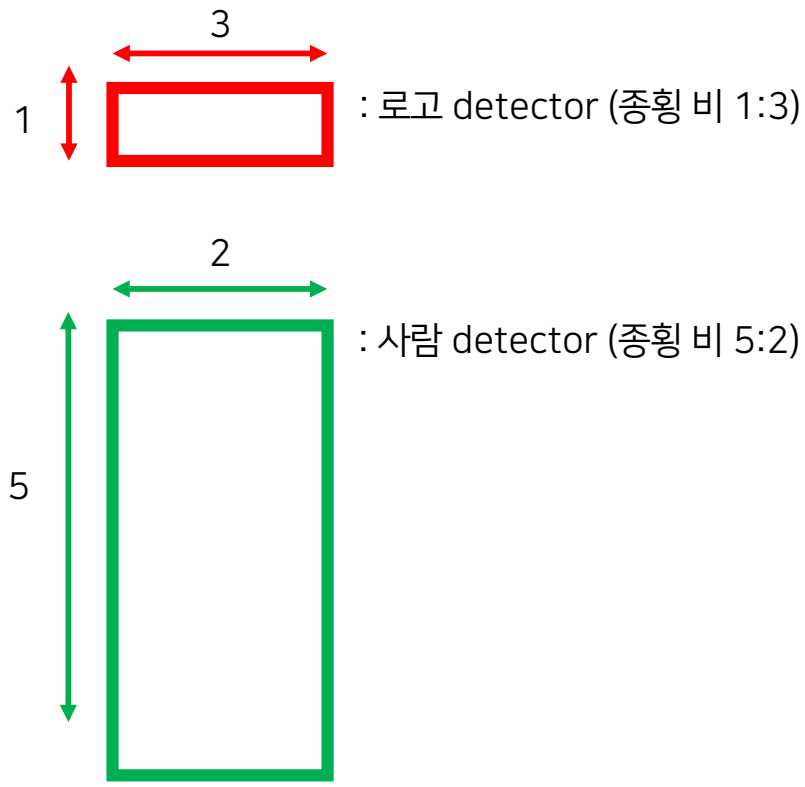


< Anchor Boxes : 찾을 객체 수와 일치>

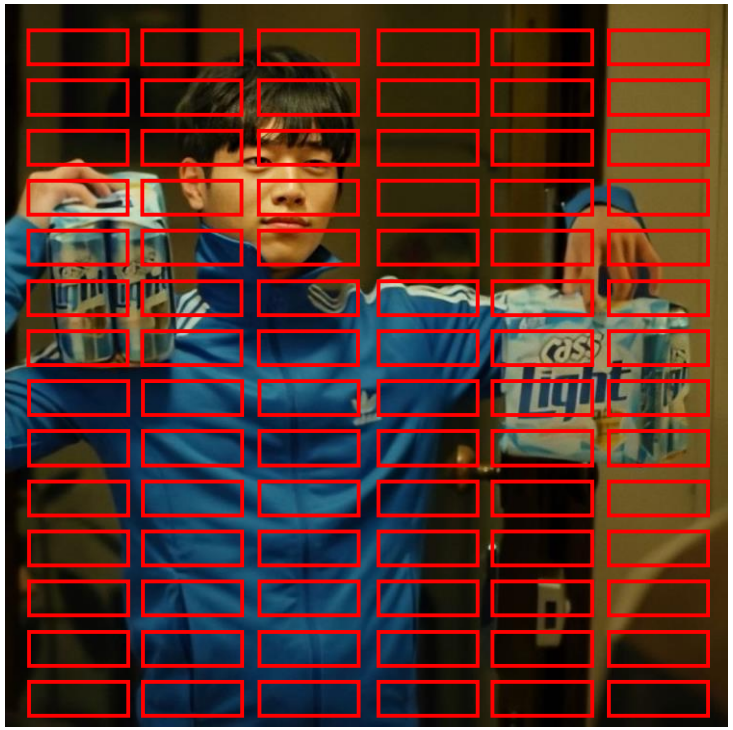
→ 학습시킨 객체들의 **box 모양을 기준으로**
각 객체의 **anchor box 비율을 정한다**

03# 모델 정의 – YOLO

YOLO에서 Detection을 하는 방법 : anchor box를 이용



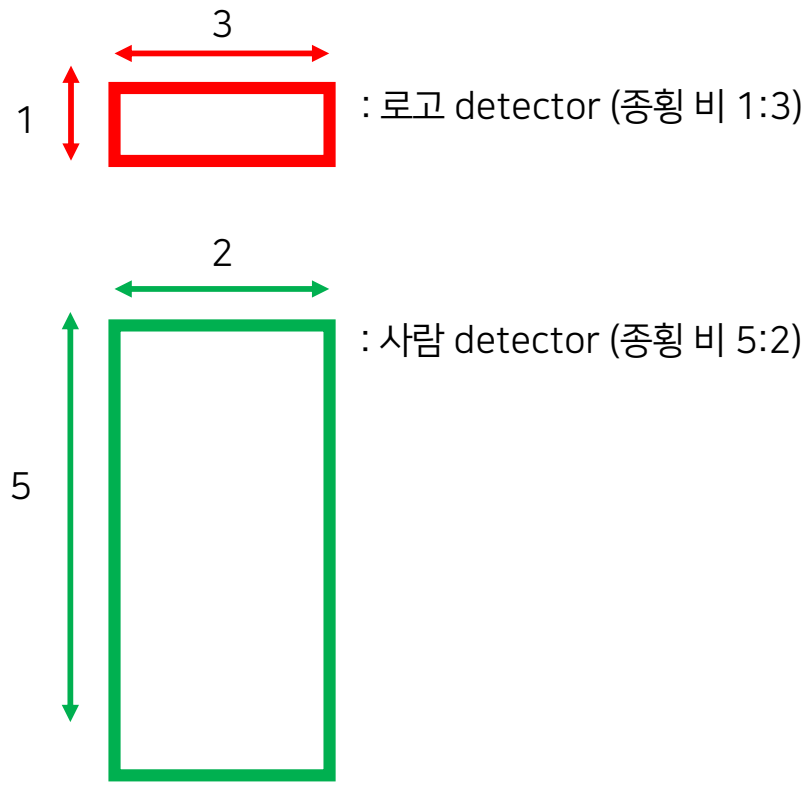
< Anchor Boxes : 찾을 객체 수와 일치>



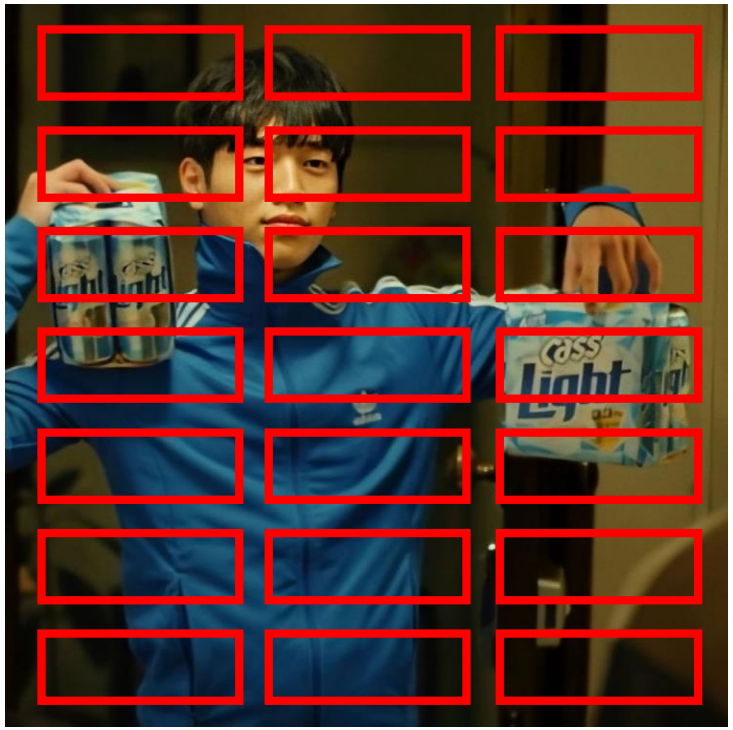
→ 학습시킨 객체들의 **box 모양을 기준으로**
각 객체의 **anchor box 비율을 정한다**

03# 모델 정의 – YOLO

YOLO에서 Detection을 하는 방법 : anchor box를 이용



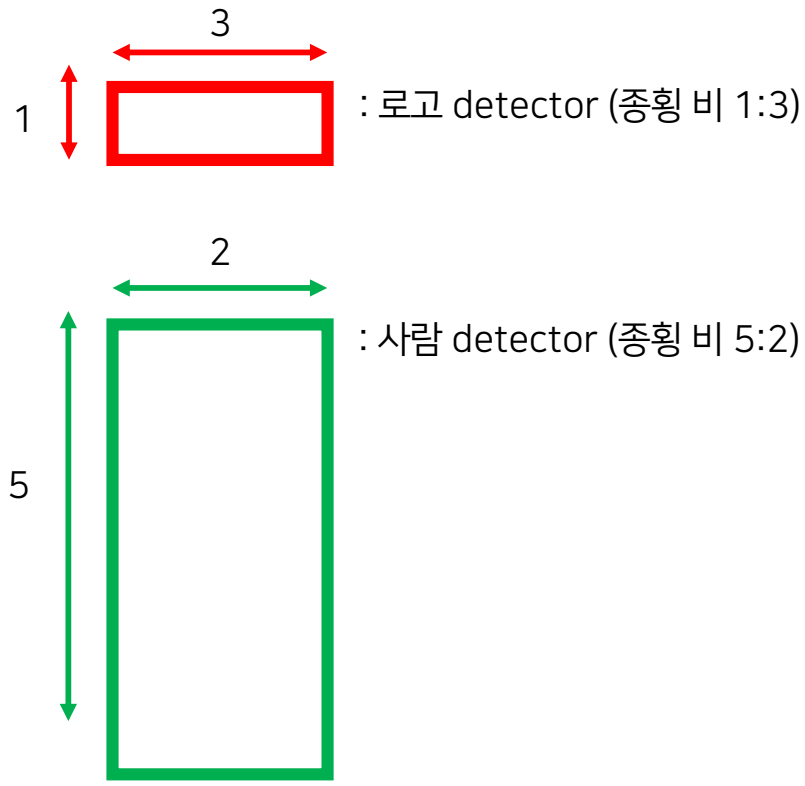
< Anchor Boxes : 찾을 객체 수와 일치>



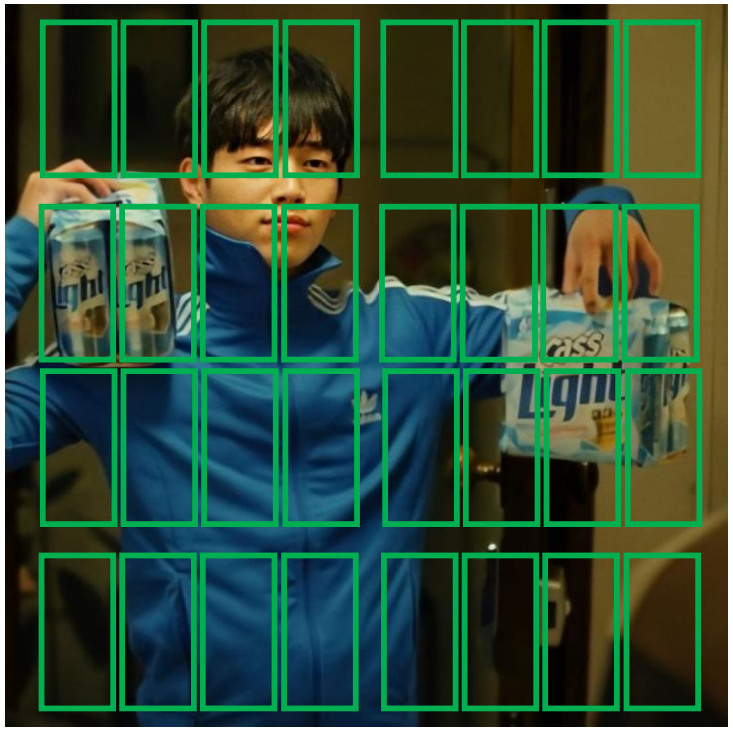
→ 학습시킨 객체들의 **box 모양을 기준으로**
각 객체의 **anchor box 비율을 정한다**

03# 모델 정의 – YOLO

YOLO에서 Detection을 하는 방법 : anchor box를 이용



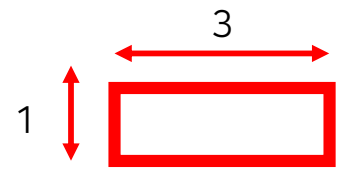
< Anchor Boxes : 찾을 객체 수와 일치>



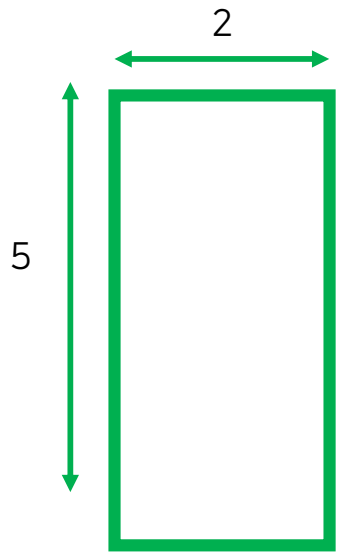
→ 학습시킨 객체들의 **box 모양을 기준으로** 각 객체의 **anchor box 비율을 정한다**

03# 모델 정의 – YOLO

YOLO에서 Detection을 하는 방법 : anchor box를 이용

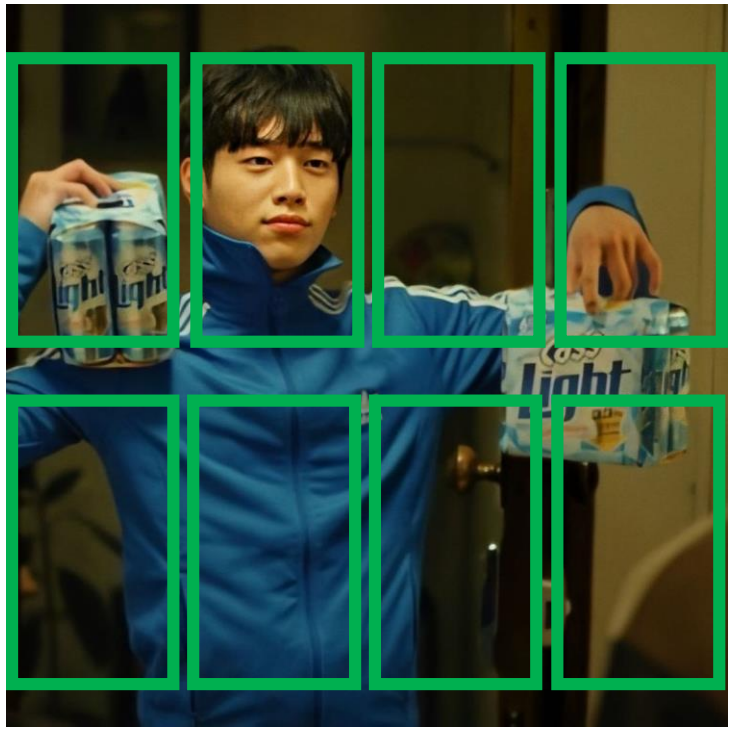


: 로고 detector (종횡 비 1:3)



: 사람 detector (종횡 비 5:2)

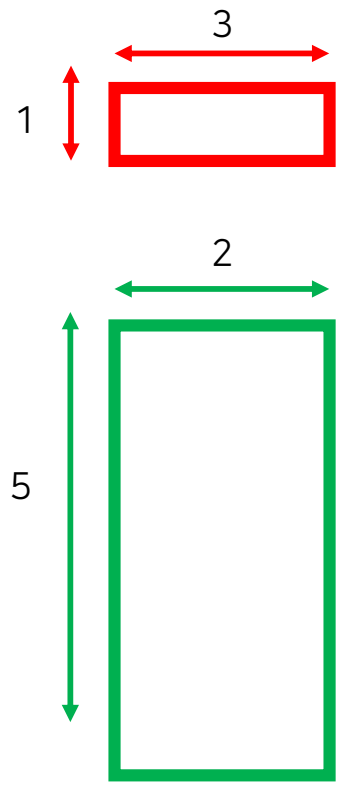
< Anchor Boxes : 찾을 객체 수와 일치>



→ 학습시킨 객체들의 **box 모양을 기준으로** 각 객체의 **anchor box 비율을 정한다**

03# 모델 정의 – YOLO

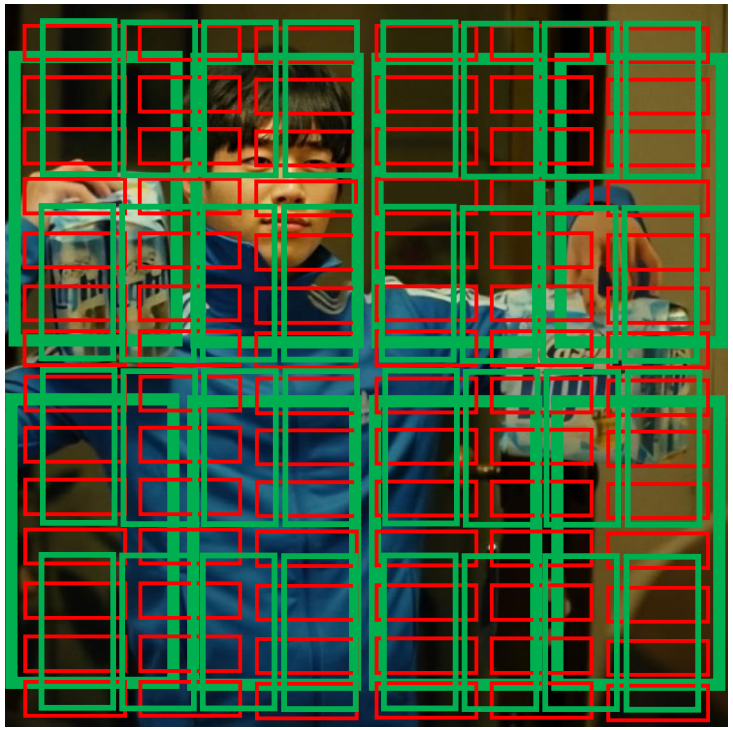
YOLO에서 Detection을 하는 방법 : anchor box를 이용



: 로고 detector (종횡 비 1:3)

: 사람 detector (종횡 비 5:2)

< Anchor Boxes : 찾을 객체 수와 일치>

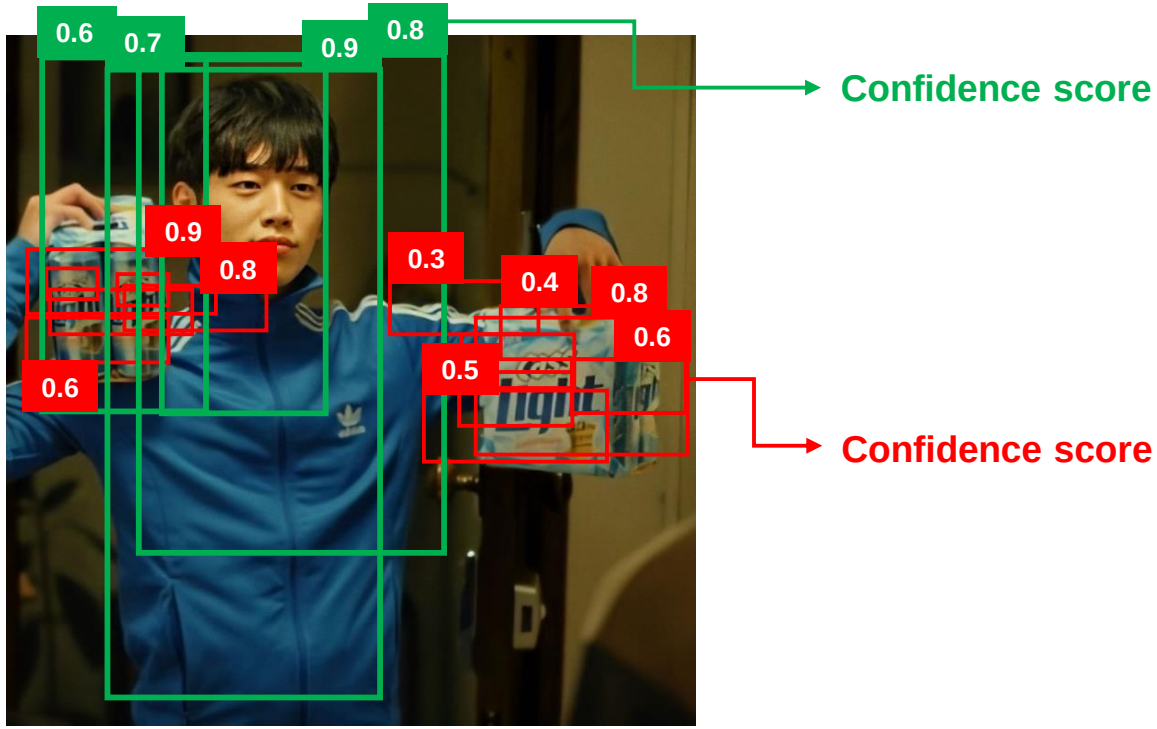


→ 학습시킨 객체들의 **box 모양을 기준으로**
각 객체의 **anchor box 비율을 정한다**

03# 모델 정의 – YOLO

중복된 Bounding Box를 제거하는 방법 : Non Max Suppression

- 하나의 객체 당 여러 개의 경계박스들이 중복되는 경우, **최대값을 갖는 하나만 남기고** 나머지는 지운다.

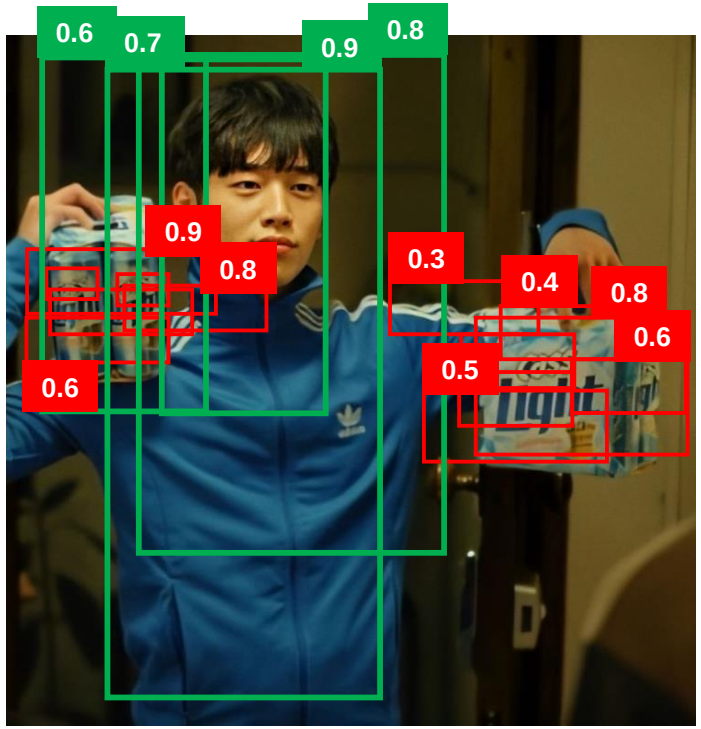


→ 각 box마다 객체가 있을 확률(confidence score)을 출력

03# 모델 정의 – YOLO

중복된 Bounding Box를 제거하는 방법 : Non Max Suppression

- 하나의 객체 당 여러 개의 경계박스들이 중복되는 경우, **최대값을 갖는 하나만 남기고** 나머지는 지운다.



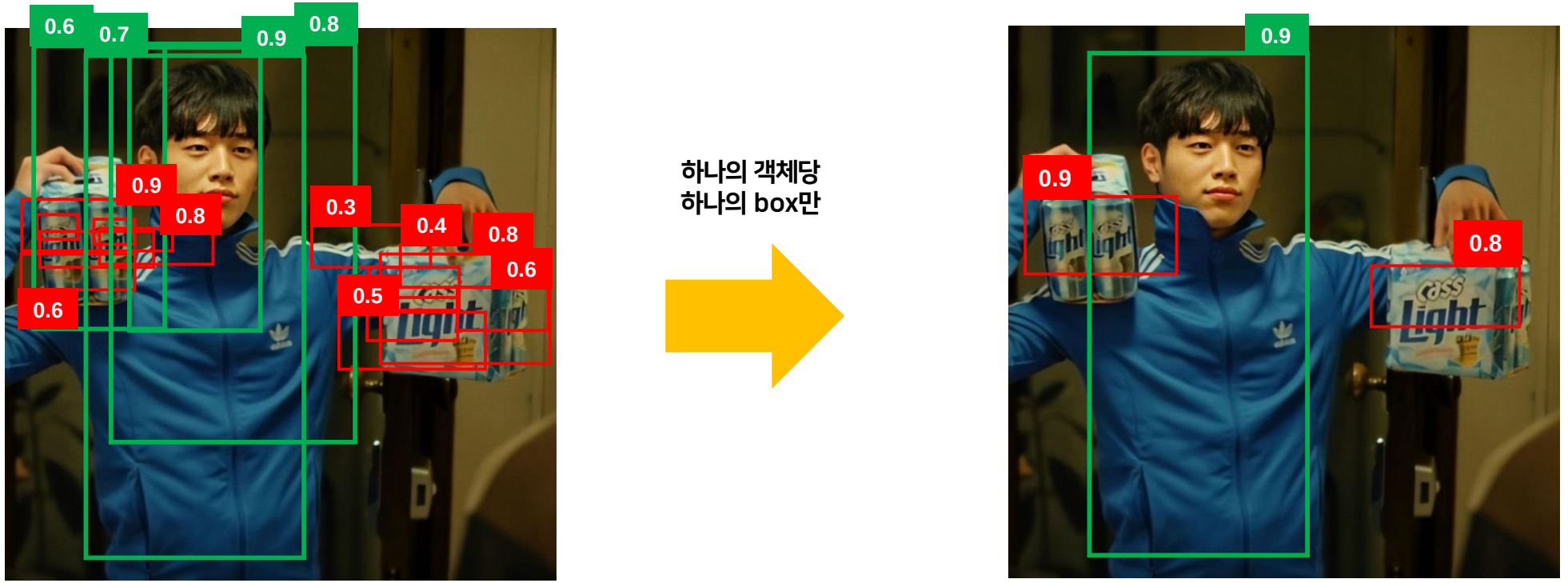
하나의 객체당
하나의 box만

→ 각 box마다 객체가 있을 확률(confidence score)을 출력

03# 모델 정의 – YOLO

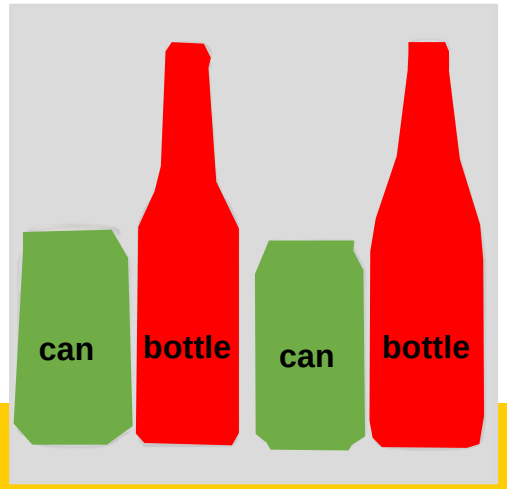
중복된 Bounding Box를 제거하는 방법 : Non Max Suppression

- 하나의 객체 당 여러 개의 경계박스들이 중복되는 경우, **최대값을 갖는 하나만 남기고** 나머지는 지운다.



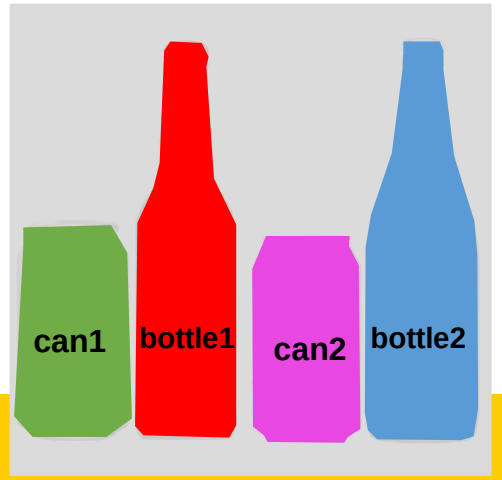
→ 각 box마다 객체가 있을 확률(confidence score)을 출력

03# 모델 정의 - Mask R-CNN



+

=



Semantic Segmentation

- 이미지를 pixel 단위로 분할한 뒤, **같은 class인 object들은 같은 색**으로 표시
- 즉, 이미지 내의 물체들을 **의미 있는 단위로 분할**해내는 것

Object Detection

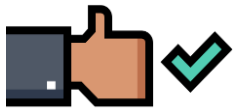
- 각 객체의 **Bounding Box**를 구한다.
- 구한 Bounding Box에 대해서 class를 할당한다.

Instance Segmentation

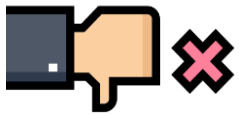
- **같은 class내의 개별 객체들을 구별**한다.
- 의미 단위로 분할된 픽셀을 서로 다른 객체로 분할하면 된다.

03# 모델 정의 - 비교

YOLO v3

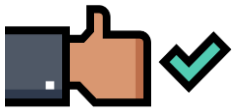


- 한 이미지에서 **여러 사물들을 효과적으로 탐지**할 수 있다.
- 이미지 전체로부터 학습을 하기 때문에 Background Error(배경을 물체로 인식)가 적다.
- 1 stage로 처리해 속도가 빠르기 때문에 **동영상 객체인식에서 유용**하다.



- **겹쳐진 사물의 구분**이 어렵다.
(Anchor box의 중심점이 겹치는 경우)

MASK R - CNN



- **세밀한이미지의 경계**를 알 수 있다.
- 정확도가 높다.

- 2 stage로 처리해 YOLO에 비해 **속도가 느리다.**
- 트레이닝 시 세밀한 경계값을 넣어줘야 한다.

03# 모델 정의

Classification (CNN)

- 이미지에 **class를 할당**

Object Localization

- Object가 이미지 안의 어느 위치에 있는지 **위치 정보를 출력**
- 주로 Bounding box를 이용하여 출력

Semantic Segmentation

- 이미지를 pixel 단위로 분할한 뒤, **같은 class인 object들은 같은 색**으로 표시

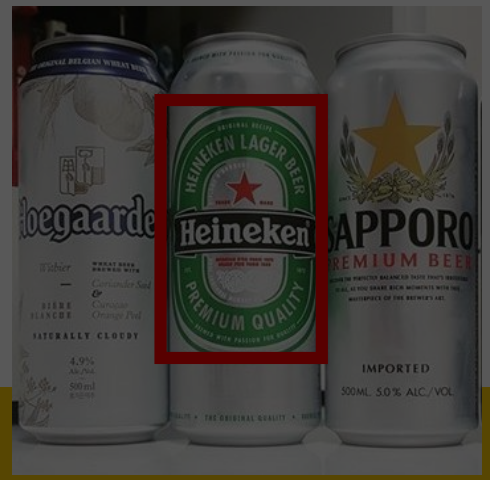
Instance Segmentation

- semantic segmentation에서 한발 더 나아가서, **같은 class이더라도 서로 다른 색**을 주어 구분

03# 모델 정의

Classification (CNN)

- 이미지에 **class를 할당**

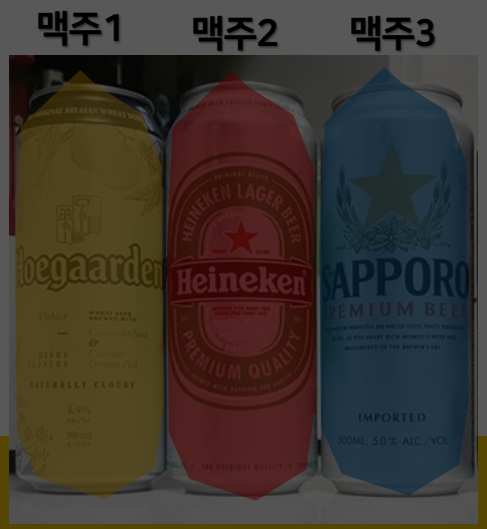


Object Localization

- Object가 이미지 안의 어느 위치에 있는지 **위치 정보를 출력**
- 주로 Bounding box를 이용하여 출력

Semantic Segmentation

- 이미지를 pixel 단위로 분할한 뒤, **같은 class인 object들은 같은 색**으로 표시



Instance Segmentation

- semantic segmentation에서 한발 더 나아가서, **같은 class이더라도 서로 다른 색**을 주어 구분

03# 모델 정의

Classification (CNN)

- 이미지에 **class를 할당**

Object Localization

- Object가 이미지 안의 어느 위치에 있는지 **위치 정보를 출력**
- 주로 Bounding box를 이용하여 출력

맥주 맥주 맥주

Semantic Segmentation

- 이미지를 pixel 단위로 분할한 뒤, **같은 class인 object들은 같은 색**으로 표시

맥주1 맥주2 맥주3

Instance Segmentation

- semantic segmentation에서 한발 더 나아가서, **같은 class이더라도 서로 다른 색**을 주어 구분

Object Detection
1. YOLO

03# 모델 정의

Object Detection
2. MASK R - CNN

Classification (CNN)

- 이미지에 **class를 할당**

Object Localization

- Object가 이미지 안의 어느 위치에 있는지 **위치 정보를 출력**
- 주로 Bounding box를 이용하여 출력

Semantic Segmentation

- 이미지를 pixel 단위로 분할한 뒤, **같은 class인 object들은 같은 색**으로 표시

Instance Segmentation

- semantic segmentation에서 한발 더 나아가서, **같은 class이더라도 서로 다른 색**을 주어 구분

04# 학습 과정 – 라벨링

YOLO

- 각 이미지에 대한 .txt 파일 생성, **box 형태로 마킹.**
- **Label, x, y, width, height**
  4 0.332031 0.818056 0.220313 0.358333
  4 0.632813 0.467361 0.221875 0.437500

MASK R – CNN

- 각 이미지에 대한 .json 파일 생성, **polygon 형태로 마킹.**
- google_terra5.jpg14476“
  : {**"filename"**: "google_terra5.jpg“, **"size"**: 14476,
  "regions": [{"shape_attributes": {"name": "polygon", "all_points_x": [169, 227, 227, 172, 168], "all_points_y": [72, 44, 111, 91, 79]}, "region_attributes": {"logo": "4"}}, {"shape_attributes": {"name": "polygon", "all_points_x": [64, 138, 138, 129, 110, 92, 65], "all_points_y": [109, 107, 117, 135, 161, 161, 112]}, region_attributes": {"logo": "4"}}],
  "file_attributes": {}}

04# 학습 과정 - Tool

Darknet

- C 언어로 작성된 **물체 인식 오픈 소스 신경망**
- YOLO 알고리즘 소스코드가 들어있는 프로젝트를 "Darknet"이라고 부른다.
- 리눅스 명령어를 사용해야 한다.
- **YOLOv3를 사용하기 위해** Darknet을 사용했다.
- 원래 YOLOv3는 한번에 한 이미지만 예측 가능하지만, **여러 이미지를 한꺼번에 예측할 수 있는 패키지 "VG_AlexeyAB_darknet"**을 사용했다.

Colab

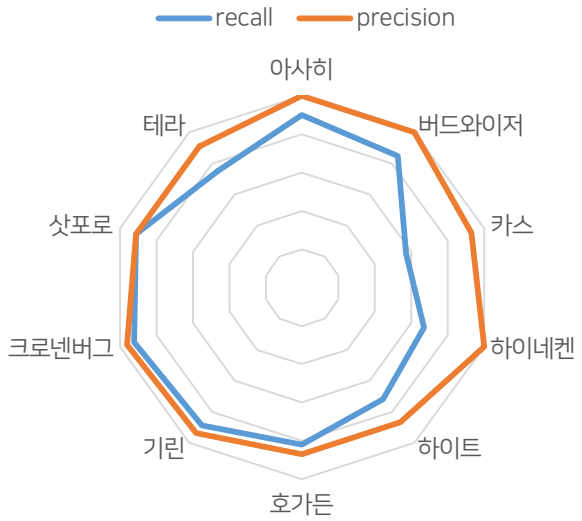
colab

- 클라우드 기반의 google의 무료 Jupyter 노트북 개발 환경이며, **GPU를 빌려 쓸 수 있다.**
- 여러 명이 동시에 수정할 수 있고, 언제 어디서든 접속 가능하지만, 최대 세션 유지시간은 12시간이다.
- Google Drive와 연동하여 데이터를 보관할 수 있다.
- **GPU 의 한계와 각종 프로그램 설치의 어려움으로 COLAB 사용**했다.

05# 성능 평가 - YOLO

1. 정확도

YOLO 브랜드별 정확도[1] (0~100%)



- Recall: 사진에 존재하는 로고를 **빠트리지 않고 검출해낸 비율**
- Precision: 검출된 **결과가 얼마나 정확한 지**에 대한 비율

카스, 하이네켄, 하이트: 테스트 이미지에 존재하는 로고들은 많이 못 찾아냈지만, 결과값으로 찾아낸 로고는 거의 맞음

1) 인스타그램 데이터(200개) 사용

카스: 다른 글자를 cass로 인식함

- cass light와 cass fresh를 같은 라벨로 지정하여 **cass라는 로고를 학습하지 못했기 때문**으로 보임

하이트: extra cold만 구분함

- **extra cold까지 포함한 마킹**을 했기 때문으로 보임

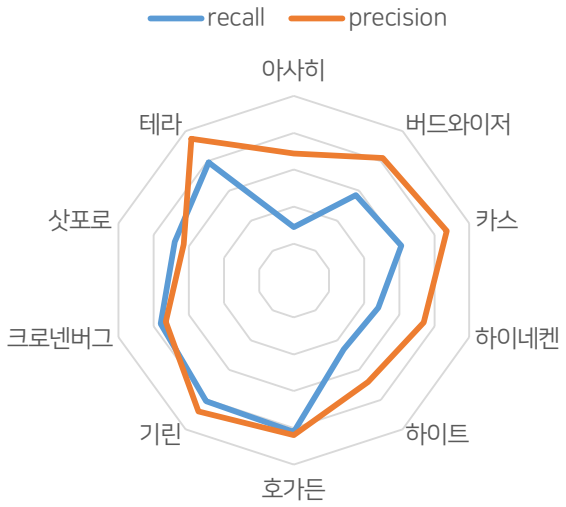
하이네켄: 다양한 로고 디자인

- 스페셜 에디션 등 **로고 디자인의 변화가 있는 경우**를 학습하지 못함

05# 성능 평가 - MASK R CNN

1. 정확도

Mask R-CNN 브랜드별 정확도[1] (0~100%)



- Recall: 사진에 존재하는 로고를 **빠트리지 않고 검출해낸 비율**
- Precision: 검출된 **결과가 얼마나 정확한 지**에 대한 비율

아사히, 버드와이저, 카스, 하이네켄, 하이트: 테스트 이미지에 존재하는 로고들은 많이 못 찾아냈지만, 결과값으로 찾아낸 로고는 거의 맞음

아사히: 인식률 자체가 매우 낮아 재현률이 떨어짐

- 모델이 스스로 생각한 **확신도가 90% 이상일 경우에만 결과로 도출해내기 때문**인 것으로 보임

호가든: 여러가지 모양에서도 상대적으로 인식이 잘 되는 편

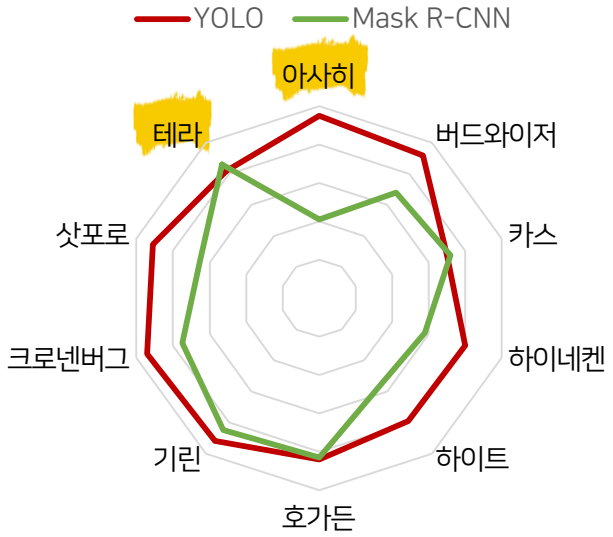
- **학습 데이터와 테스트 이미지가 비슷한 경우가 많았기 때문**으로 보임

1) 인스타그램 데이터(200개) 사용

05# 성능 평가 – MASK R CNN

1. 정확도

알고리즘 성능 비교[1] (정확도 f1값 기준)



- **f1-score(f1값):** recall(재현률) 값과 precision(정밀도) 값을 조화평균한 것

YOLO의 정확도가 전반적으로 **더 우수함**
→ 이후의 **페이스북** 로고 노출도 분석에서는 **YOLO를 사용함**

1) 인스타그램 데이터(200개) 사용

Mask R-CNN

YOLO

05# 성능 평가

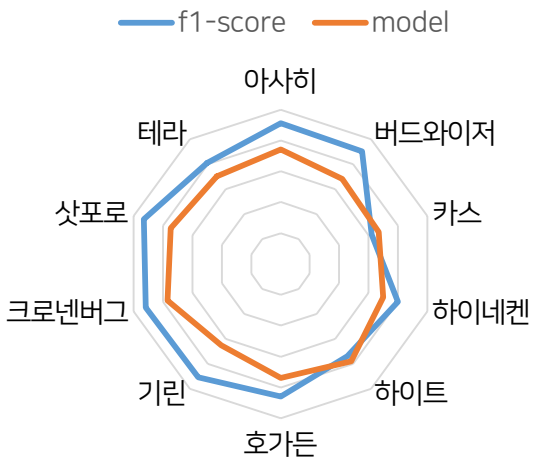
2. 속도: 초 당 처리 이미지 개수(GPU 기준)

- **YOLO:** 사진 한 장당 0.019초
- **MASK R – CNN:** 사진 한 장당 약 0.4초

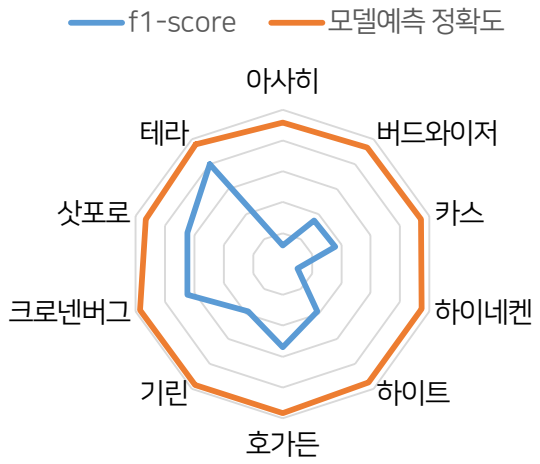
3. 예측 확신 정도

- YOLO는 학습 횟수가 증가 할수록 모델의 예측 확신 정도와 실제 정확도가 일치하므로, **충분히 신뢰할만한 지표가 됨**
- MASK R – CNN은 모델이 예측한 정확도는 매우 높았으나, **실제로는 정확도가 매우 낮았음**

YOLO: 9600회 학습(0~100%)



MASK R – CNN: 6000회 학습(0~100%)



05# 성능 평가

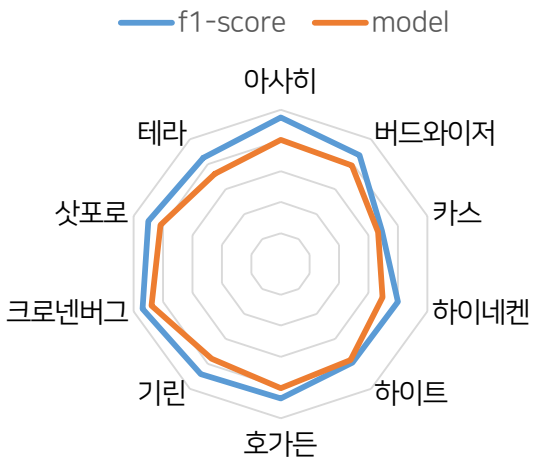
2. 속도: 초 당 처리 이미지 개수(GPU 기준)

- **YOLO:** 사진 한 장당 0.019초
- **MASK R – CNN:** 사진 한 장당 약 0.4초

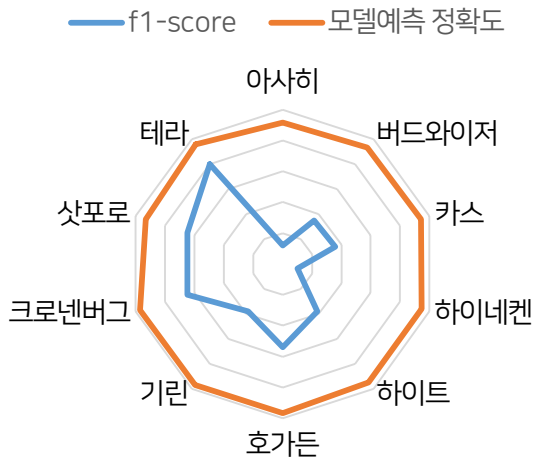
3. 예측 확신 정도

- YOLO는 학습 횟수가 증가 할수록 모델의 예측 확신 정도와 실제 정확도가 일치하므로, **충분히 신뢰할만한 지표가 됨**
- MASK R – CNN은 모델이 예측한 정확도는 매우 높았으나, **실제로는 정확도가 매우 낮았음**

YOLO: 14700회 학습(0~100%)



MASK R – CNN: 6000회 학습(0~100%)



05# 성능 평가

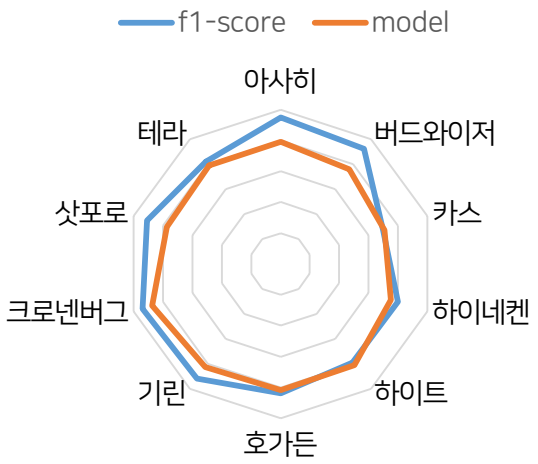
2. 속도: 초 당 처리 이미지 개수(GPU 기준)

- **YOLO:** 사진 한 장당 0.019초
- **MASK R – CNN:** 사진 한 장당 약 0.4초

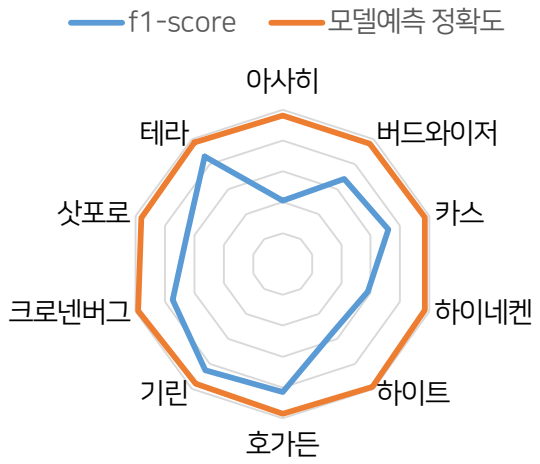
3. 예측 확신 정도

- YOLO는 학습 횟수가 증가 할수록 모델의 예측 확신 정도와 실제 정확도가 일치하므로, **충분히 신뢰할만한 지표가 됨**
- MASK R – CNN은 모델이 예측한 정확도는 매우 높았으나, **실제로는 정확도가 매우 낮았음**

YOLO: 20000회 학습(0~100%)



MASK R-CNN: 20000회 학습(0~100%)

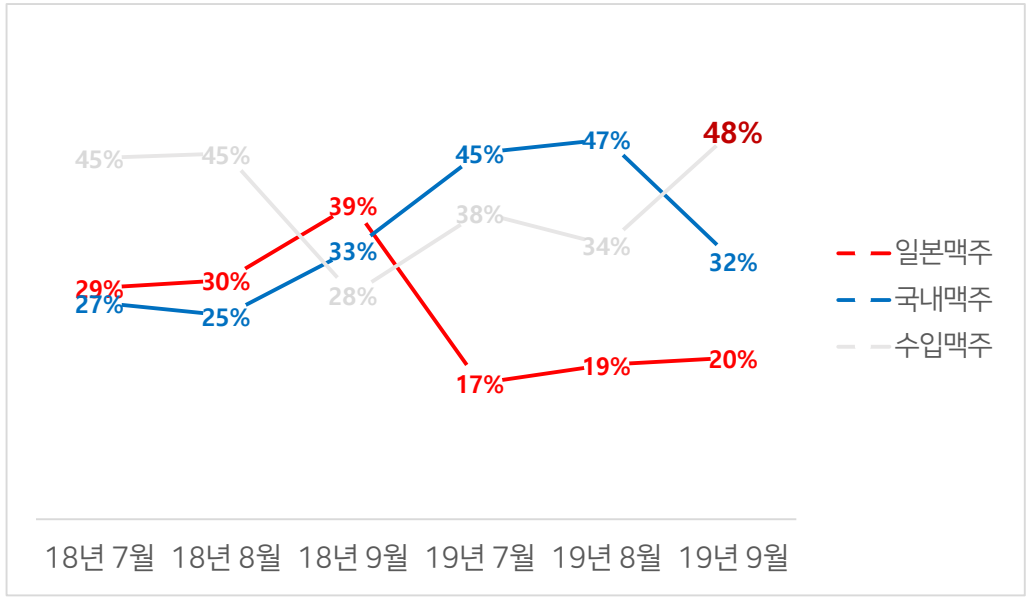


06# 인사이트

Facebook 데이터 분석

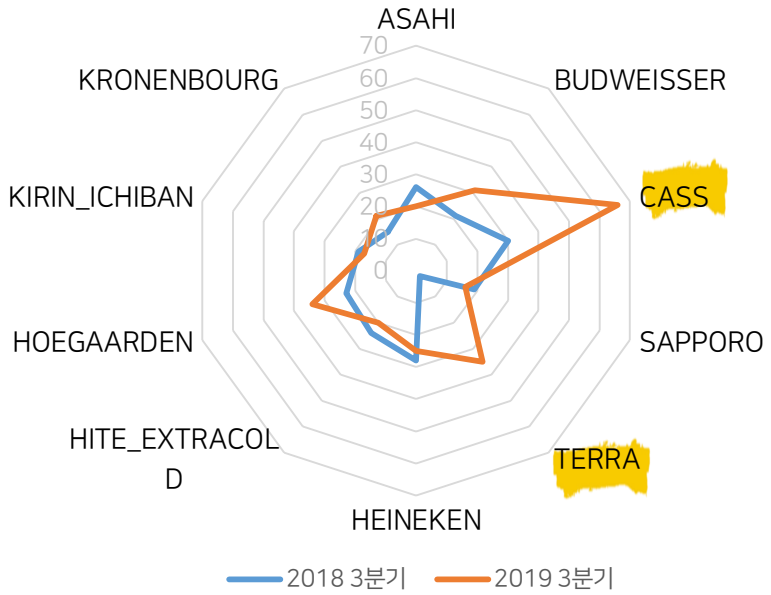
- YOLOv3로 약 4000장의 페이스북 데이터에 대한 로고 디텍팅 수행

월별 노출도 그래프



- 불매운동 전, 일본맥주와 국내맥주의 변화 양상은 비슷하다.
- 불매운동 후, **일본맥주의 노출도는 하락한 반면 국내맥주는 상승**하였다.
- 2019년 9월에는 불매운동이 잠잠해지면서 **국내맥주와 수입맥주의 노출도가 반전**되었다.

각 브랜드의 페이스북 노출도 (단위:개)



- 브랜드 별로 비교했을 때,
  카스와 테라의 노출도가 급격하게 상승하였다.

06# 인사이트 - 향후 과제

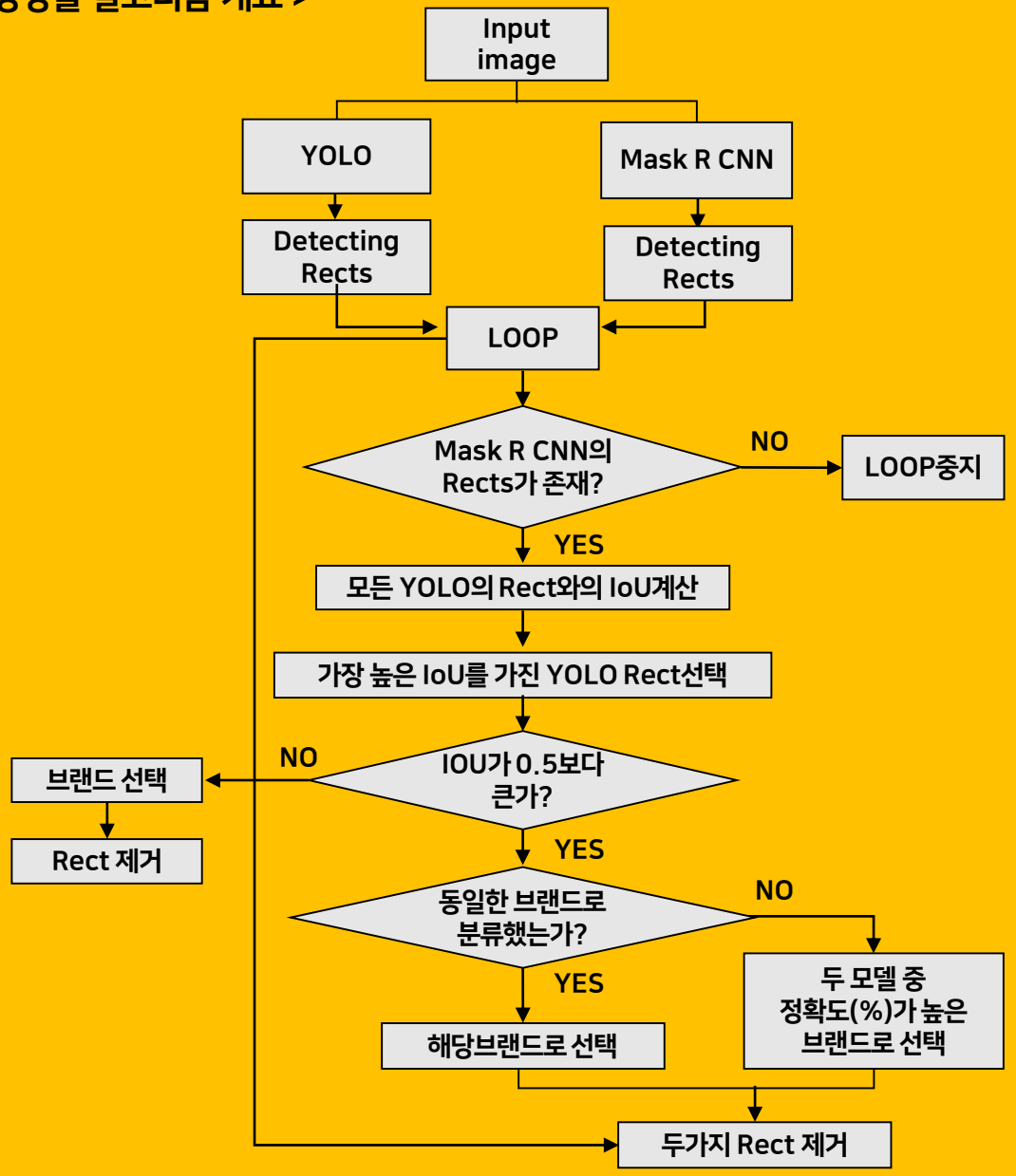
학습 데이터 정제

- 카스, 하이트의 **로고만 학습**시켜 모델의 정확도 높이기

알고리즘 개선

- 너무 어두운 사진은 detecting 시작 전에 **자동으로 밝기 조절**
- 맥주 캔을 먼저 인식 후, 로고 탐지 방식
- YOLO와 Mask R - CNN을 합한 **앙상블 모델**
- **영상 탐지**

< 앙상블 알고리즘 개요 >



감사합니다

윤서진 김태현 권휘국 천희우 김하연 허우진