Yolo 논문리뷰

1. introduction

사람의 시각 체계와 같이 빠르고 정확한 객체 검출 모델을 만들 수 있다면 자율주행차 기술도 급격히 발전할 것이다.

기존의 검출 모델은 분류기를 재정의하여 검출기로 사용하고 있다. 분류란 하나의 이미지를 보고 그것이 개인지 고양이인지 판단하는 것을 뜻한다. 하지만 객체 검출은 하나의 이미지 내에서 개는 어디에 위치해 있고, 고양이는 어디에 위치해 있는지 판단하는 것이다. 따라서 객체 검출은 분류 뿐만 아니라 위치 정보도 판단해야 한다. 기존의 객체 검출 모델로는 대표적으로 DPM고가 R-CNN이 있다.

DPM은 이미지 전체를 거쳐 슬라이딩 윈도 방식으로 객체 검출을 하는 모델이다. R-CNN은 이미지 안에서 BOUNDING BOX를 생성하기 위해 region prospal이라는 방법을 사용한다. 그렇게 제안된 bounding box에 classifier를 적용하여 분류한다. 분류한 뒤 bounding box를 조정하고, 중복된 검출을 제거하고, 객체에 따라 box의 점수를 재산정하기 위해 후처리를 합니다. 이런 복잡함 때문에 R-CNN은 느리다. 각 절차를 독립적으로 훈련시켜야 하므로 최적화 하기에도 힘들다.

그리하여 YOLO연구진은 객체 검출을 하나의 회귀 문제로 보고 절차를 개선했다. 이미지 픽셀로부터 bounding box의 위치 클래스 확률을 구하기까지의 일련의 절차를 하나의 회귀 문제로 재정의한 것이다. 이러한 시스템을 통해 yolo는 이미지 내에 어떤 물체가 있고 그 물체가 어디에 있는지를 하나의 파이프라인으로 빠르게 구해준다. 이미지를 한 번만 보면 객체를 검출 할 수 있다 하여 이름이 yolo이다.(you only look once)

Yolo는 단순하다. 하나의 컨볼루션 네트워크가 여러 bounding box와 bounding box의 클래스 확률을 동시에 계산해 준다. Yolo는 이미지 전체를 학습하여 곧바로 검출 성능을 최적화 한다. Yolo의 이런 통합된 모델은 기존의 객체 검출 모델이 비해 여러 가지 장점이 있다.

첫째 yolo는 굉장히 빠르다. 왜냐하면 yolo는 기존의 복잡한 객체 검출 프로세를 하나의 회귀 문제로 바꾸었기 때문이다. 그리하여 기존의 객체 검출 모델처럼 복잡한 파이프라인이 필요 없다. 단순히 테스트 단계에서 새로운 이미지를 yolo신경망에 넣어주기만 하면 쉽게 객체 검출을 할 수 있다. Yolo의 기본 네트워크는 yiyan x gpu에서 배치 처리 없이 1초에 45프레임을 처리한다. 빠른 버전의 yolo는 1초에 150프레임을 처리한다. 이는 동영상을 실시간으로 처리할 수 있다는 의미이다. 더욱이 yolo는 다른 실시간 객체 검 모델보다 2배 이상의 Map를 갖는다.

둘째, YOLO는 예측을 할 때 이미지 전체를 봅니다. 슬라이딩 윈도(sliding window)나 region proposal 방식과 달리, YOLO는 훈련과 테스트 단계에서 이미지 전체를 봅니다. 그리하여 클래스의 모양에 대한 정보뿐만 아니라 주변 정보까지 학습하여 처리합니다. 반면, YOLO 이전의 객체 검출 모델 중 가장 성능이 좋은 모델인 Fast R-CNN는 주변 정보까지는 처리하지 못합니다. 그래서 아무 물체가 없는 배경(background)에 반점이나 노이즈가 있으면 그것을 물체로 인식합니다. 이를 background error라고 합니다. YOLO는 이미지 전체를 처리하기 때문에 background error가 Fast R-CNN에 비해 훨씬 적습니다. (대략 1/2 가량)

셋째, YOLO는 물체의 일반적인 부분을 학습합니다. 일반적인 부분을 학습하기 때문에 자연 이미지를 학습하여 그림 이미지로 테스트할 때, YOLO의 성능은 DPM이나 R-CNN보다 월등히 뛰어납니다. 따라서 다른 모델에 비해 YOLO는 훈련 단계에서 보지 못한 새로운 이미지에 대해 더 강건(robust)합니다. 즉, 검출 정확도가 더 높다는 뜻입니다.

하지만, YOLO는 최신(SOTA, state-of-the-art) 객체 검출 모델에 비해 정확도가 다소 떨어진다는 단점이 있습니다. 빠르게 객체를 검출할 수 있다는 장점은 있지만 정확성이 다소 떨어집니다. 특히 작은 물체에 대한 검출 정확도가 떨어집니다. 속도와 정확성은 trade-off 관계입니다. YOLO의 모든 코드는 오픈 소스이며, 사전 훈련된(pretrained) 모델도 다운받아 사용할 수 있습니다.

2. united Detection

YOLO는 객체 검출의 개별 요소를 단일 신경망(single neural network)으로 통합한 모델입니다. YOLO는 각각의 bounding box를 예측하기 위해 이미지 전체의 특징을 활용합니다. 이러한 YOLO의 디자인 덕분에 높은 정확성을 유지하면서 end-to-end 학습과 실시간 객체 검출이 가능합니다.

YOLO는 입력 이미지(input images)를 S x S 그리드(S x S grid)로 나눕니다. 만약 어떤 객체의 중심이 특정 그리드 셀(grid cell) 안에 위치한다면, 그 그리드 셀이 해당 객체를 검출해야 합니다. (If the center of an object falls into a grid cell, that grid cell is responsible for detecting that object.)

각각의 그리드 셀(grid cell)은 B개의 bounding box와 그 bounding box에 대한 confidence score를 예측합니다. confidence score는 bounding box가 객체를 포함한다는 것을 얼마나 믿을만한지, 그리고 예측한 bounding box가 얼마나 정확한지를 나타냅니다. confidence score는 다음과 같이 정의합니다.

2. yolo 객체 검출 모델

통합 신경망:

Yolo는 객체 검출의 모든 요소를 단일 신경망으로 통합하여, 이미지 전체의 특징을 활용해 높은 정확성을 유지하면서 end-to-end 학습과 실시간 객체 검출을 가능하게 합니다.

그리드 분할:

입력 이미지를 S x S 그리드로 나누고, 객체 중심이 속한 그리드 셀이 해당 객체를 검출합니다.

Bounding Box 예측:

각 그리드 셀은 B개의 bounding box와 해당하는 confidence score를 예측합니다. Confidence score는 bounding box가 객체를 포함하는 신뢰도를 나타내며, IOU(Intersection over Union)를 기준으로 정의됩니다.

Bounding Box 구성:

각 bounding box는 5개의 예측치 (x, y, w, h, confidence)로 구성됩니다. (x, y)는 그리드 셀 내의 상대적 위치, (w, h)는 이미지 전체 대비 상대적 너비와 높이를 나타냅니다. Confidence는 bounding box의 신뢰도를 나타냅니다.

클래스 확률 예측:

각 그리드 셀은 조건부 클래스 확률(C)을 예측합니다. 이는 객체가 해당 클래스일 확률을 나타냅니다.

테스트 단계:

조건부 클래스 확률과 bounding box의 confidence score를 곱해 class-specific confidence score를 계산합니다. 이는 해당 클래스가 bounding box에 나타날 확률과 예측된 bounding box가 객체에 얼마나 잘 맞는지를 나타냅니다.

파스칼 VOC 실험:

YOLO는 파스칼 VOC 데이터셋을 사용해 실험했습니다. 설정값은 S=7, B=2, C=20입니다. 입력 이미지는 7 x 7 그리드로 나뉘고, 각 그리드 셀은 2개의 bounding box를 예측합니다. 최종 예측 텐서는 (7 x 7 x 30)입니다.

2.1 Network Design

Yolo 모델은 하나의 CNN 구조로 디자인되었습니다. yolo연구진은 파스칼 VOC 데이터셋에 대해 모델링을 수행했다. 이 CNN의 앞단은 컨볼루션 계층이고, 이어서 전결합 계층으로 구성되어 있다. 컨볼루션 계층은 이미지로부터 특징을 추출하고, 전결합 계층은 클래스 확률과 bounding box의 좌표를 예측한다.

YOLO의 신경망 구조는 이미지 분류(image classification)에 사용되는 GoogLeNet에서 따왔습니다. YOLO는 총 24개의 컨볼루션 계층(convolutional layers)과 2개의 전결합 계층(fully connected layers)으로 구성되어 있습니다. GoogLeNet의 인셉션 구조 대신 YOLO는 1 x 1 축소 계층(reduction layer)과 3 x 3 컨볼루션 계층의 결합을 사용했습니다. 1 x 1 축소 계층(reduction layer)과 3 x 3 컨볼루션 계층의 결합이 인셉션 구조를 대신한다고 합니다. YOLO 모델의 전체 구조는 다음과 같습니다. 이 모델의 최종 아웃풋은 7 x 7 x 30의 예측 텐서(prediction tensors)입니다.

좀 더 빠른 객체 인식 속도를 위해 YOLO보다 더 적은 컨볼루션 계층(24개 대신 9개)과 필터를 사용하는 Fast YOLO라는 것도 있습니다. 크기만 다를 뿐이고 훈련 및 테스트 시 사용하는 나머지 파라미터는 YOLO와 모두 동일합니다.

2.2 Training

사전훈련:

YOLO의 컨볼루션 계층은 ImageNet 데이터셋을 이용해 사전 훈련되었습니다. 24개의 컨볼루션 계층 중 첫 20개를 사용했고, 전결합 계층을 추가하여 약 1주간 훈련한 결과, ImageNet 2012 검증 데이터셋에서 88%의 정확도를 기록했습니다. 이 과정은 Darknet 프레임워크를 사용했습니다.

모델 변환 및 확장:

사전 훈련된 20개 컨볼루션 계층 뒤에 4개의 컨볼루션 계층과 2개의 전결합 계층을 추가했습니다. 이 추가된 계층의 가중치는 임의로 초기화되었으며, 입력 이미지의 해상도는 224x224에서 448x448로 증가시켰습니다. 최종 출력은 클래스 확률과 bounding box 위치 정보(x, y, w, h)로, 이들은 모두 0~1 사이의 값으로 정규화되었습니다. 마지막 계층에는 선형 활성화 함수를, 나머지 계층에는 leaky ReLU를 적용했습니다.

Loss Function:

YOLO의 손실 함수는 SSE(sum-squared error)를 기반으로 합니다. Localization loss와 classification loss가 있으며, 객체가 없는 그리드 셀의 loss 가중치는 줄이고, 객체가 있는 셀의 loss 가중치는 늘렸습니다. 이를 위해 λ\_coord와 λ\_noobj 파라미터를 사용하여 각각 5와 0.5로 설정했습니다. Bounding box의 너비와 높이에는 제곱근을 취해 작은 객체에 대한 민감도를 높였습니다.

훈련 과정:

파스칼 VOC 2007, 2012 데이터셋을 이용해 135 epochs 동안 훈련했습니다. Batch size는 64, momentum은 0.9, decay는 0.0005로 설정했습니다. 초기 학습률은 0.001에서 시작해 0.01까지 증가시키고, 이후 75 epochs 동안 0.01, 30 epochs 동안 0.001, 마지막 30 epochs 동안은 0.0001로 설정했습니다. 과적합을 막기 위해 드롭아웃(0.5)과 data augmentation을 적용했습니다. Data augmentation은 원본 이미지의 20%까지 랜덤 스케일링과 랜덤 이동을 포함했습니다.

2.3 inference

추론 단계:

YOLO는 테스트 이미지로부터 객체를 검출할 때, 단 하나의 신경망 계산만 필요합니다. 파스칼 VOC 데이터셋에 대해 이미지당 98개의 bounding box와 각 bounding box의 클래스 확률을 예측합니다. 이는 YOLO가 R-CNN 등과 비교하여 매우 빠르게 동작하는 이유입니다.

다중 검출 문제:

YOLO의 그리드 디자인에는 객체가 여러 그리드 셀에 걸쳐있을 경우, 여러 그리드 셀이 동일한 객체를 검출하는 문제가 있습니다. 특히 객체가 크거나 그리드 셀 경계에 인접해 있는 경우 이러한 다중 검출이 발생할 수 있습니다.

비 최대 억제(NMS):

이 문제를 해결하기 위해 YOLO는 비 최대 억제(non-maximal suppression) 기법을 사용합니다. 이 방법을 통해 다중 검출 문제를 줄이고, 평균 정확도(mAP)를 2~3%가량 향상시켰습니다.

2.4 Limitation of YOLO

YOLO는 하나의 그리드 셀마다 두 개의 bounding box를 예측합니다. 그리고 하나의 그리드 셀마다 오직 하나의 객체만 검출할 수 있습니다. 이는 공간적 제약(spatial constraints)을 야기합니다. 공간적 제약이란 '하나의 그리드 셀은 오직 하나의 객체만 검출하므로 하나의 그리드 셀에 두 개 이상의 객체가 붙어있다면 이를 잘 검출하지 못하는 문제'를 뜻합니다. 예를 들어, 새 떼와 같이 작은 물체가 몰려 있는 경우 공간적 제약 때문에 객체 검출이 제한적일 수밖에 없습니다. 하나의 그리드 셀은 오직 하나의 객체만 검출하는데 여러 객체가 몰려있으면 검출하지 못하는 객체도 존재하는 것이죠.

그리고 YOLO 모델은 데이터로부터 bounding box를 예측하는 것을 학습하기 때문에 훈련 단계에서 학습하지 못했던 새로운 종횡비(aspect ratio, 가로 세로 비율)를 마주하면 고전할 수밖에 없습니다.

마지막으로 YOLO 모델은 큰 bounding box와 작은 bounding box의 loss에 대해 동일한 가중치를 둔다는 단점이 있습니다. 크기가 큰 bounding box는 위치가 약간 달라져도 비교적 성능에 별 영향을 주지 않는데, 크기가 작은 bounding box는 위치가 조금만 달라져도 성능에 큰 영향을 줄 수 있습니다. 큰 bounding box에 비해 작은 bounding box가 위치 변화에 따른 IOU 변화가 더 심하기 때문입니다. 이를 부정확한 localization 문제라고 부릅니다.

3. comparison to other Detection Systems

Deformable Parts Models (DPM):

방식: 슬라이딩 윈도(sliding window) 방식 사용.

구조: 독립적인 파이프라인으로 구성 (특징 추출, 위치 파악, bounding box 예측 등).

특징: 각 단계가 분리되어 있어, 처리 속도가 느리며 통합되지 않은 구조.

Region-based Convolutional Neural Networks (R-CNN):

방식: Region proposal 방식을 사용.

구조: 여러 단계로 구성된 복잡한 파이프라인 (selective search로 bounding box 생성, 컨볼루션 신경망으로 특징 추출, SVM으로 점수 측정, 선형 모델로 bounding box 조정, 비 최대 억제로 중복 검출 제거).

특징: 높은 정확성에도 불구하고, 각 단계별로 독립적으로 튜닝해야 하므로 속도가 매우 느려 실시간 객체 검출에 한계 (이미지 처리 시간 40초 이상).

You Only Look Once (YOLO):

방식: 단일 컨볼루션 신경망 사용.

구조: 통합된 신경망으로 특징 추출, bounding box 예측, 비 최대 억제 등을 한 번에 처리.

특징: DPM보다 빠르고 정확하며, R-CNN보다 예측 bounding box 개수가 적고 속도가 매우 빠름 (이미지당 98개의 bounding box 예측).

비교 요약:

DPM: 느리고 분리된 파이프라인으로 구성.

R-CNN: 높은 정확성, 매우 느린 속도, 복잡한 파이프라인.

YOLO: 빠르고 통합된 구조, 실시간 객체 검출에 적합, 다중 검출 문제를 비 최대 억제로 해결.

4. Experiments

먼저, YOLO를 다른 실시간(real-time) 객체 검출 모델과 비교해보겠습니다. YOLO와 Fast R-CNN의 성능의 차이를 비교하기 위해 파스칼 VOC 2007 데이터 셋에서의 에러를 구해봤습니다. Fast R-CNN은 이 논문이 나온 시점을 기준으로 성능이 가장 좋은 R-CNN 계열의 모델이었습니다.

4. 1. Comparison to Other Real-Time Systems

속도와 정확도:

Fast YOLO: 파스칼 데이터셋 기준 mAP 52.7%, 30Hz/100Hz에서 DPM보다 2배 높은 정확도.

YOLO: 실시간 성능 유지하며 mAP 63.4%.

YOLO with VGG-16: 높은 mAP, 하지만 속도 느림, 실시간 검출에는 적합하지 않음.

특징: 단일 신경망으로 빠르고 효율적, 실시간 객체 검출에 가장 적합.

DPM (Deformable Parts Models):

속도와 정확도:

Fast DPM: 약간의 mAP 하락, 속도 향상, 그러나 실시간 검출에는 여전히 느림.

특징: 독립된 파이프라인으로 구성, 신경망 기반 모델보다 정확도 낮음.

R-CNN (Region-based Convolutional Neural Networks):

속도와 정확도:

R-CNN minus R: selective search 대신 정적 bounding box proposal 사용, 빠르지만 실시간 검출에는 부족.

Fast R-CNN: 높은 정확도, 초당 0.5 프레임 처리, 실시간 검출에는 역부족.

Faster R-CNN: selective search 대신 신경망 사용으로 속도 향상, 그러나 YOLO보다 여전히 느림.

특징: 복잡한 파이프라인, 실시간 검출에는 부적합.

성능 비교표 (예시):

Fast YOLO: mAP 52.7%, FPS > 30

YOLO: mAP 63.4%, FPS > 30

YOLO with VGG-16: 높은 mAP, 낮은 FPS

Fast DPM: 낮은 mAP, 낮은 FPS

R-CNN minus R: 높은 mAP, 낮은 FPS

Fast R-CNN: 높은 mAP, FPS 0.5

Faster R-CNN: 높은 mAP, 낮은 FPS

결론:

YOLO: 높은 속도와 적당한 정확도로 실시간 객체 검출에 가장 적합.

DPM 및 R-CNN 계열: 높은 정확도에도 불구하고 속도가 느려 실시간 검출에는 부적합.

이 요약은 YOLO 연구진이 다양한 객체 검출 모델을 비교한 결과를 기반으로 합니다. YOLO는 실시간 성능과 정확도를 모두 만족시키며, 다양한 응용 분야에서 실시간 객체 검출에 최적화된 모델로 평가받고 있습니다.

4. 2. VOC 2007 Error Analysis

YOLO와 기존 객체 검출 모델을 더 비교해보겠습니다. 우선, 파스칼 VOC 2007 데이터 셋에 대해 YOLO와 Fast R-CNN의 성능을 비교해보았습니다. 이를 위해 Diagnosing Error in Object Detectors 논문에 소개된 에러 측정 방법론을 사용했습니다. 다음과 같은 기준으로 객체 검출이 정확한지, 틀렸다면 어떤 error type인지를 구분했습니다.

Correct : class가 정확하며 IOU > 0.5 인 경우

Localization : class가 정확하고, 0.1 < IOU < 0.5 인 경우

Similar : class가 유사하고 IOU > 0.1 인 경우

Other : class는 틀렸으나, IOU > 0.1 인 경우

Background : 어떤 Object라도 IOU < 0.1 인 경우

YOLO는 localization error 가 상대적으로 큽니다. localization error는 19.0%로 나머지 error를 모두 합한 15.5%(6.75%+4.0%+4.75%) 보다 큽니다. Fast R-CNN은 YOLO에 비해 localization error가 작습니다. 반면, background error가 상대적으로 큽니다. backgound error는 배경에 아무 물체가 없는데 물체가 있다고 판단하는 false positive error입니다. Fast R-CNN은 YOLO에 비해 background error가 3배나 더 큽니다.

4. 3. Combining Fast R-CNN and YOLO

YOLO는 Fast R-CNN에 비해 background error가 훨씬 적습니다. 따라서 Fast R-CNN에 YOLO를 결합하여 background error를 줄인다면 굉장히 높은 성능을 낼 수 있을 것입니다. R-CNN이 예측하는 모든 boudning box에 대해 YOLO도 유사하게 예측하는지를 체크하면 됩니다. 만약 R-CNN이 예측한 bounding box와 YOLO가 예측한 bounding box가 유사하다면 두 bounding box가 겹치는 부분을 bounding box로 잡으면 됩니다.

파스칼 VOC 2007 데이터 셋에 대해 가장 성능이 좋은 Fast R-CNN 모델은 71.8%의 mAP를 기록했습니다. Fast R-CNN과 YOLO를 결합하면 mAP가 3.2% 올라 75.0%가 됩니다. Fast R-CNN과 다른 모델과도 앙상블을 해봤지만 mAP 향상은 0.3%, 0.6%로 미미했습니다.

물론 Fast R-CNN과 YOLO를 결합한 모델은 YOLO에 비해 느립니다. 왜냐하면 Fast R-CNN과 YOLO를 독립적으로 돌려 결과를 앙상블 하는 방식이기 때문입니다. 그렇지만 YOLO가 워낙 빨라 Fast R-CNN을 단독으로 돌리는 것과 앙상블 모델을 돌리는 것의 속도는 거의 유사합니다. 그 말은 Fast R-CNN을 사용하는 것보다는 Fast R-CNN과 YOLO를 결합한 모델을 사용하는 것이 더 낫다는 뜻입니다.

4. 4. VOC 2012 Results

파스칼 VOC 2012 데이터 셋에서 YOLO는 57.9%의 mAP를 달성했습니다. 이는 VGG-16을 사용한 R-CNN의 mAP와 비슷합니다.

속도 측면에서는 YOLO가 빠르고, 정확도 측면에서는 Fast R-CNN과 YOLO를 결합한 모델이 가장 좋습니다.

4. 5. Generalizability: Person Detection in Artwork

객체 검출 연구를 위해 사용하는 데이터 셋은 훈련 데이터 셋과 테스트 데이터 셋이 동일한 분포를 지닙니다. 하지만 실제 이미지 데이터는 훈련 데이터 셋과 테스트 데이터 셋의 분포가 다를 수 있습니다. YOLO 연구진은 훈련 데이터 셋과 다른 분포는 지닌 테스트 데이터 셋(즉, 훈련 데이터 셋에서 보지 못한 새로운 데이터 셋)을 활용하여 테스트해봤습니다. 여기서는 피카소 데이터 셋과 일반 예술 작품을 사용했습니다. 훈련 단계에서는 실제 이미지로 학습했지만 테스트 단계에서는 예술 작품을 활용해 테스트해보는 것입니다.

아래 표는 YOLO과 다른 객체 검출 모델의 성능을 측정한 것입니다. 파스칼 VOC 2007에서 학습한 YOLO, R-CNN, DPM 등의 성능을 서로 비교했습니다. R-CNN은 VOC 2007에서는 높은 정확도를 보이지만 예술작품에 대해서는 굉장히 낮은 정확도를 보입니다. DPM은 예술 작품에 대해서도 정확도가 크게 떨어지지는 않았습니다. 다만 VOC 2007에서의 정확도도 그리 높은 편은 아닙니다. 반면 YOLO는 VOC 2007에서도 가장 높은 정확도를 보였고, 예술 작품에 대해서도 정확도가 크게 떨어지지 않았습니다.

5. Real-Time Detection In The Wild

YOLO는 컴퓨터 비전 애플리케이션에 활용할 수 있는 빠르고 정확한 객체 검출 모델입니다. 연구진은 YOLO를 웹캠과 연결하여 실시간(real-time) 객체 검출을 수행했습니다. 이는 웹사이트를 통해 확인할 수 있습니다.

6. Conclusion

지금까지 객체 검출을 위한 통합 모델인 YOLO에 대해 소개했습니다. YOLO는 단순하면서도 빠르고 정확합니다. 또한 YOLO는 훈련 단계에서 보지 못한 새로운 이미지에 대해서도 객체를 잘 검출합니다. 즉, 새로운 이미지에 대해서도 강건하여 애플리케이션에서도 충분히 활용할만한 가치가 있습니다.