우리는 YOLO를 제시한다. 이것은 기존과 다른 물체 탐지 방법을 접근하였다. 이전의 물체 탐지 작업은 사용 목적에 맞게 물체를 분류하였다. 대신, 우리는 물체 탐지를 공간과 분리된 바운딩 박스, 혹은 관련되어있는 클래스에 대한 확률로 회기 문제로 프레임 한다. 단일 뉴런 네트워크는 바운딩 박스, 클래스의 확률을 직접 full 이미지에 대한 한번의 연산을 통해 예측합니다. 따라서 전체 예측에 대한 파이프 라인은 단일 network입니다. 이것은 끝에서 끝으로 직접적인 예측 성능을 최적화 시킬 수 있습니다.

우리의 통합 아키텍쳐는 매우 빠르다. 우리의 기본 YOLO model은 초당 45frame으로 실시간으로 이미지를 처리한다. Network의 작은 버전인 Fast Yolo는 다른 실시간 탐지기의 두배의 MAP을 달성하면서 초당 놀라운속도의 155 프레임을 처리한다. 최첨단(State-of-art) 탐지 시스템을 비교해보자. YOLO는 localization error를 더 만든다. But! Background에서 False Positive 예측의 가능성이 낮아진다. 마지막으로 YOLO는 물체의 일반적인 표현을 배운다?. 자연적인 이미지를 다른 영역의 예술작품으로 일반화할 때. 이것은 다른 탐지기 (DPM, R-CNN ..)에 비해 더 나은 성능을 낸다.

인간은 이미지를 휙휙 둘러보고 즉각적으로 어떤 이미지인지, 그들이 어디에 있는지, 그리고 어떻게 그들이 소통하는지 알 수 있다. 인간의 시각화 시스템은 매우 빠르고 정확해서 드라이브와 같은 복잡한 일을 조금의 의식적인 생각으로 할 수 있게 해준다. 물체 탐지를 위한 빠르고 정확한 알고리즘은 컴퓨터가 특별한 센서 없이 차를 운전하게 해줄 것이며, 보조적인 장치를 통해 사람 혹은 사용자에게 실시간으로 정보를 전달하는 것이 가능하며, 범용적이고, 반응적인 하는 로보틱 시스템의 잠재력을 열 수 있게 할 것이다.

현재의 물체 탐지 시스템은 목적에 맞게 분류를 수행한다. 물체 탐지를 하기 위해 이런 시스템은 테스트 이미지의 다양한 위치와 척도에서 계산된 분류를 진행한다. 변형 가능한 모델 (DPM) 과 같은 시스템은 분류기가 전체의 이미지의 균등한 간격 위치에서 실행되는 슬라이딩 윈도우 방식을 사용한다.

최신 접근방식의 R-CNN은 가능성이 있는 바운딩 박스를 이미지에 생성, 그 다음 분류기가 맞춰가는 과정의 사용 지역 제한 방식이다. 분류 후, 후처리 과정에서는 중복된 탐지를 제거하고, 배경 위에 다른 물체가 올려져 있는 박스의 점수를 다시 매긴다. 이러한 복잡한 파이프라인은 느리고 최적화 하기에 어렵다. 왜냐하면 각각 개인의 요소가 분리되어 훈련되기 떄문이다.

우리는 물체 탐지를 이미지 픽셀에서 직접 바운딩 박스를 조절하고 클래스의 확률을 하나의 회기 문제로 개조한다. 우리의 시스템을 이용하면, YOLO(you look only once) 이미지를 한번만 보고 이미지가 어디에 있든지 예측이 가능하다.

YOLO는 상쾌 통쾌하게 간단하다. 1. 물체를 본다. Single CNN은 동시에 여러 개의 바운딩 박스와, 박스에 대한 클래스의 확률을 예측한다. YOLO는 전체의 이미지를 학습하며 즉각적으로 탐지 성능을 최적화시킨다. 이 통일된 모델은 기존의 물체 탐지 방법에 비해 몇 가지 이득을 제공한다.

첫번째로 YOLO는 매우 빠르다. Frame 탐지가 회기문제로서 우리는 복잡한 파이프라인이 필요하지 않았다. 우리는 간단하게 새로운 이미지에 대해 우리의 NN을 실행하여 탐지를 예측한다. 우리의 기초 네트워크는 배치 처리 없이 TITAN X GPU에서 초당 45frame의 속도로 실행하며 빠른 버전은 150fps의 속도로 실행이 가능하다. 이것은 생방송 비디오를 초당 25 ms 보다 적은 대기 시간으로 분석이 가능하다는 뜻입니다. 추가적으로 YOLO는 다른 실시간 탐지 시스템의 2배 이상 평균 정밀도를 달성하였습니다. 다음 페이지에서는 웹 캠에서 실시간으로 작동하는 우리의 데모 버전을 볼 수 있다. Pjreddie.com/yolo

두번째, YOLO는 이미지를 예측할 때 이미지에 대해 전 세계적으로 추론한다. 지역 제한 기술 및 슬라이딩 윈도우 기술과는 다르게, YOLO는 학습 및 테스트 시간 동안 전체의 이미지를 본다 따라서 클래스의 모양 뿐만 아니라 클래스에 대한 상황정보를 암시적으로 인코딩합니다.

Fast R-CNN, 최고의 탐지 방법, 이미지의 배경 부분을 object로 잘못 지정하게 된다 왜냐하면 그것은 많은 부분을 확인할 수 없기 때문이다. YOLO는 Fast R-CNN에 비교해 배경 오류에 관한 숫자를 절반 넘게 줄였습니다.

세번째로 YOLO는 물체의 일반화된 표현을 배웁니다. 우리가 자연의 이미지를 훈련해서 미술작품에 대해 평가할 때 YOLO는 DPM과 Fast R-CNN이 최고의 탐지 방법을 큰 차이로 능가합니다. 때문에 YOLO는 매우 일반화 되어 있어 예기치 않은 입력 혹은 새로운 도메인이 들어왔을 때 고장날 확률이 적습니다.   
YOLO는 여전히 최신 물체 탐지 기술에 비해 정확도가 떨어집니다. 욜로는 이미지에서 빠르게 물체를 식별할 수 있지만, 특히 어떤 물체, 작은 물체들을 정확하게 물체를 위치시키기 위해 몸부림칩니다. 우리는 우리의 실험에서 tradeoff를 더 자세히 검사합니다

우리의 훈련 및 테스트 코드는 오픈소스입니다. 다양한 사전학습 모델은 다운로드가 가능합니다.

**2. Unified Detection**

우리는 단일 NN 안에 물체 탐지에서 분해된 요소를 통합한다. 우리의 Network는 특징을 사용해 전체의 이미지에서 각각의 바운딩 박스를 예측합니다. 또한 이것은 모든 바운딩 박스를 거쳐 모든 클래스에서 이미지를 즉각적으로 예측합니다. 이는 전체 이미지와 이미지의 모든 객체에 대한 네트워크의 이유를 의미합니다. YOLO는 끝에서 끝으로 훈련이 가능하고 실시간 속도를 높은 평균 정확도를 유지할 수 있게 디자인 되었다.

우리의 시스템은 입력 이미지를 SxS의 Grid로 나눈다. 만약 물체의 중심이 Grid cell안에 들어올 경우 그 Grid cell은 탐지된 물체를 담당한다.

각각의 Grid cell은 바운딩 박스와 박스에 대한 신뢰도 B를 예측한다. 이 신뢰도는 물체가 포함된 박스와 또한 상자가 얼마나 정확하게 예측하는지 얼마나 이 모델이 확신하는지를 반영한다. 일반적으로 우리는 신뢰도를 Pr(Object) \* 로 정의한다. 만약 물체가 Cell 안에 존재하지 않을 경우 신뢰도는 0이다. 반대로 우리는 예측 결과의 박스와 실제 ground truth(실제 사진) 사이의 intersection over union (IOU)와 동일한 신뢰도를 원한다.

각각의 바운딩 박스는 5개의 예측이 존재한다. X, y, w, h 그리고 신뢰도이다. (x,y) 좌표는 grid cell의 경계와 박스의 중앙이 관련이 있는지 표현하다. Width와 height는 전체의 이미지에서 관련된 부분을 예측한다. 마침내 IOU는 Ground truth(실제 사진)과 예측된 박스 사이의 신뢰 예측을 나타낸다.

각각의 Grid cell은 C라는 현재 클래스의 조건부확률을 예측하며 로 표기한다. 이러한 확률은 grid cell이 물체를 포함하는 조건이다. 우리는 박스의 수 B에 상관하지 않고 단지 하나의 grid cell에 대한 하나의 클래스에 대한 확률만 계산합니다.

Test 시간에서 우리는 조건부 클래스의 확률과 각 박스의 신뢰 예측을 곱하여 각 박스에 대해 클래스 별 신뢰 점수를 제공한다. 이 점수는 박스안의 클래스가 나타나는 확률과 얼마나 박스를 물체에 맞게 예측하였는지 모두 encode 한다.

Figure 2 : The model. 우리 시스템 모델은 회기문제로서 탐지한다. 이것은 이미지를 SxS의 Grid로 나누고, 각 Grid cell의 대해 B : 바운딩 박스들 예측, 그 박스의 신뢰도, 그리고 클래스의 확률 C. 이러한 예측은 S x S x (B \* 5) + C 의 텐서 형태로 이루어진다.

**2.1 Network Design**

우리는 Convolutional Neural Network로서 모델을 향상시키고, Pascal VOC에서 dataset 탐지 연산을 한다. 초기 네트워크의 Convolutional layer는 Fully Connected layers가 출력 확률과 좌표를 예측하는 동안 이미지에서 특징을 추출한다.

우리의 네트워크 구조는 이미지 분류모델 GoogLeNet에서 영감을 받았다. 우리의 네트워크는 24개의 Convolutional 층을 가지며 뒤이어 2개의 fully connected layers가 따른다. GoogLeNet에서 사용하는 inception model 대신 우리는 간단한 1 x 1 축소된 레이어 뒤에 3 x 3의 Convolutional layers가 따른다. (??? Let et al ). 이 전체 네트워크는 Figure 3에 표시되어 있다.

또한 우리는 빠른 물체 탐지의 경계를 넘기도록 설계된 빠른 버전의 Fast YOLO를 훈련시킨다. Fast YOLO는 더 적은 Convolutional layers (24 대신 9)와 각 층의 더 적은 필터가 Neural Network에서 사용되었다. 이외에 네트워크 사이즈, 모든 훈련 및 테스트 파라메타가 YOLO와 Fast YOLO가 동일하다.

**2.1 Training**

우리의 Convolutional layer들과 ImageNet 안에 1000개 클래스 경쟁 dataset을 사전훈련 시켰다.사전 학습 전, Figure 3에서 우리는 먼저 20개의 Convolutional 레이어만 사용하며 뒤이어 average-polling layer와 fully connected layer를 사용한다. 우리는 대략 일주일 정도 네트워크를 훈련하고, 2012년 ImageNet의 검증 셋으로 GoogLeNet의 Caffe’s Model Zoo모델과 비슷한 single crop 88퍼의 정확도, top-5를 달성하였다. 우리는 DarkNet framework를 사용하여 모두가 학습과 추론이 가능하다.

우리는 모델이 탐지를 수행하기 위해 변환해야 한다. (Ren et al). Convolutional과 connected layers 모두 더하는 것 사전학습 성능을 향상시키는 것을 보여준다. 예제를 따라오면 우리는 4개의 Convolutional 레이어와 2개의 fully connected layer를 랜덤하게 가중치를 섞어서 더한다. 탐지는 때때로 Fine-grained 시각 정보를 요구하기에 우리는 224x224를 448x448로 네트워크의 입력 차원을 증가시킵니다.

우리의 출력층은 클래스의 확률과 박스의 좌표 모두 예측합니다. 우리는 바운딩 박스의 넓이와 높이를 이미지의 넓이와 높이로 일반화하여 바운딩 박스의 넓이와 높이를 0과 1 사이로 바꿔줍니다. 우리는 바운딩 박스 x,y 좌표를 상쇄시킬 특별한 grid cell 위치 매개변수를 설정하여 마찬가지로 0과 1사이의 bounded 되도록 합니다.

Coarse-grain or find-grain -> 낱알의 개념을 공학에 적용하여 프로세스를 잘게 쪼개느냐 혹은 크게 쪼개느냐 차이이다.

Ex) 이미지 분류에서 자동차인지, 사람인지, 동물인지 구분 -> Coarse-grain 분류 : ImageNet

자동차의 모델 분류, 꽃의 종 분류 -> Fine-Grain 분류 : Stanford Dogs, Stanford Cars