# 4. YOLOv1

## Introduction

- Object Detection 관련 연구에서는 Detection을 위해 classifier를 사용하는 반면, YOLO는 Bounding Box, Class Probabilities에 대한 regression problem으로 접근
  - 한 번의 evaluation만으로 전체 이미지에서 Bounding Boxes와 Class probabilites를 예측
  - 전체 Pipeline이 Single Neural Network이기 때문에, end - to - end 로써 최적
- YOLO
  - YOLO : 1초에 45frame
  - fast YOLO : 1초에 155 frame
    - 다른 Detector에 비해 mAP 2배
  - YOLO는 localization error가 높긴 하지만 배경을 잘못 판단하는 경우는 더 적음
  - YOLO는 물체에 대한 general representations를 학습

- Detection 방식 차이
  - 기존(R-CNN, Fast R-CNN ...)
    - 이미지에서 다양한 위치와 크기에 대해 classifier를 적용하여 물체를 검출
    - R-CNN은 우선 potential bounding boxes를 먼저 만들고 이들에 대해 classifier를 적용,
      이 후, bounding boxes를 수정, 중복되는 검출 제거, 다른 물체들을 근거로 박스 재평가 하는 post-processing이 진행
    - 이런 복잡한 pipeline은 각 요소들이 개별적으로 학습되어야 하기 때문에 느리고 최적화하기 어려움
  - YOLO
    - 이미지 픽셀에서 bounding box 좌표(coordinate) 및 class probabilites에 이르기까지 object detection을 single regression problem으로 재구성
    - 이미지를 한 번만 보더라도 그 안에 어떤 물체들이 있는지, 어디에 있는지 파악 가능

- YOLO는 single convolutional network가 이미지 전체에서 다수의 bounding box를 예측하고, 동시에 박스에서 class probability를 계산하는 통합된 모델을 사용

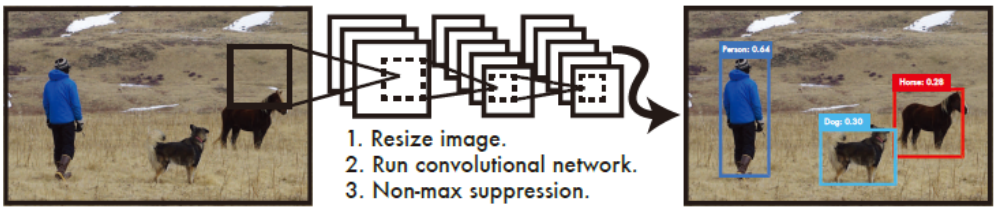


**Figure 1: The YOLO Detection System.** Processing images with YOLO is simple and straightforward. Our system (1) resizes the input image to 448 × 448, (2) runs a single convolutional network on the image, and (3) thresholds the resulting detections by the model's confidence.

- 장점
  - YOLO is Fast
    - Detection을 regression 문제로 바꿔 생각했기 때문에 complex pipeline이 필요 없음
    - test image에서 검출을 위해 신경망을 작동시키기만 하면 됨
  - YOLO는 이미지 전체를 두고 판단함
    - 학습과 평가에서 전체 이미지를 보기 때문에 생김새 뿐만 아니라 contextual information를 부호화(encode)함
      - Fast R-CNN은 큰 맥락을 보지 못하기 때문에 배경부분을 물체로 인식
      - YOLO는 이에 비해 배경 에러가 절반 정도
  - YOLO는 물체의 일반적 표현을 학습
    - 자연 이미지에서 학습하고 예술작품으로 평가했을 때, YOLO는 DPM과 R-CNN의 성능을 큰 차이로 뛰어넘어 YOLO는 보편성이 높아 새로운 영역이나 예상치 못한 입력에도 망가지는 경우가 적음
- 단점
  - YOLO는 state - of - the - art - system에 비해 정확도가 떨어짐

        - 빠르게 물체를 감지하는 만큼, 물체의 위치를 파악하는데 어려움이 있음, 특히 물체가 작을수록 더 어려움이 발생함

## Unified Detection

- YOLO는 object Detection의 여러가지 요소들을 하나의 신경망으로 합침
    - 신경망은 이미지 전체에서 나온 feature를 사용하며 모든 bounding box에서 모든 class에 대한 예측을 동시에 진행 → 이 때문에 end - to - end 학습과 실시간 작동이 가능
        - 모든 class에 대해 바로 예측하는게 아닌, class에 대해 확률을 통해 계산
- 입력 이미지를 S * S 개의 격자로 분할, 만약 물체의 중심이 셀격자(grid cell)안에 있다면, 셀격자는 그 물체를 검출하고 각 셀격자는 bounding boxes B개에 대해 예측하고 신뢰성 점수(confidence score)를 매김
    - 신뢰성 점수는 이 시스템이 물체를 포함한다는 예측을 얼마나 확신하는지, 박스에 대한 예측이 얼마나 정확할지를 의미함

$$Confidence = Pr(Object) * IOU_{pred}^{truth}$$

    - S * S개의 격자가 각각 B개의 Bounding Box에 대해 예측하고 평가하므로 B*S*S 개의 bounding box가 생김(B와 S는 사용자가 정하는 값)
    - IOU는 두 영역이 겹쳐져 있을 때 두 영역의 교집합을 합집합으로 나눈것 →(교집합 넓이)/(전체 넓이)
        - 두 영역이 완전히 겹쳐있을 때 IOU = 1
        - 어떤 물체에 대해 시스템이 예측하는 bounding box가 있는 반면, 미리 정답으로 주어진 bounding box가 있음 → 둘이 서로 비슷하게 만들고 싶다면 IOU가 1에 가까워지도록 목표
    - Pr(Object)는 셀격자 내에 물체가 존재할 확률, 물체가 존재하지 않으면 confidence = 0 * IOU = 0
        - 이 0 값이 나중에 큰 오류로 발생함. 시스템은 Pr(Object)를 1에 가깝게 만들고 싶어하며 그럴 경우 confidence = IOU가돼 confidence에 대한 오차 수정이 곧 IOU에 대한 오차 수정으로 됨
- Bounding box는 5가지 예측값을 가짐 : x, y, w, h and confidence → (x , y)좌표는 셀격자의 경계에 대한 상대적인 box 중심좌표 , width와 height는 전체 이미지에 비례, confidence는 예측 box와 실제 정답 사이의 IOU를 의미

- 각 셀격자는 C(conditional class probabilities) = Pr(Class_i|Object)를 계산 → 이 확률은 물체를 포함한 셀격자에 한정. B와 상관없이 셀격자마다 하나의 C set만을 예측
    - 셀 격자는 (B*5+C)크기의 vector를 가짐, 이미지 전체는 (S*S*(B*5+C))크기의 tensor를 가짐
    - 5*B : B개의 bounding box 각각에 대해 x, y, w, h 그리고 confidence까지의 5개 변수
    - C : class 수 - 셀격자에 포함된 물체가 각 Class에 해당될 확률들의 벡터의 크기 - 각 클래스일 확률 .... - 확률 벡터의 크기는 class의 갯수와 같음
- Conditional class probabilities와 각 박스의 class-specific confidence score를 주는 confidence prediction을 곱함 → 이 점수는 class가 박스안에 존재하는지와 박스가 물체에 얼마나 적합한가를 모두 포함

$$Pr(Class_i|Object) * Pr(Object) * IOU_{pred}^{truth} = Pr(Class_i) * IOU_{pred}^{truth}$$

- YOLO를 PASCAL VOC로 평가할 때, S = 7, B = 2
- PASCAL VOC는 20개 라벨의 Class가 있으므로 C = 20, 따라서 최종 예측결과는 7*7*30 tensor

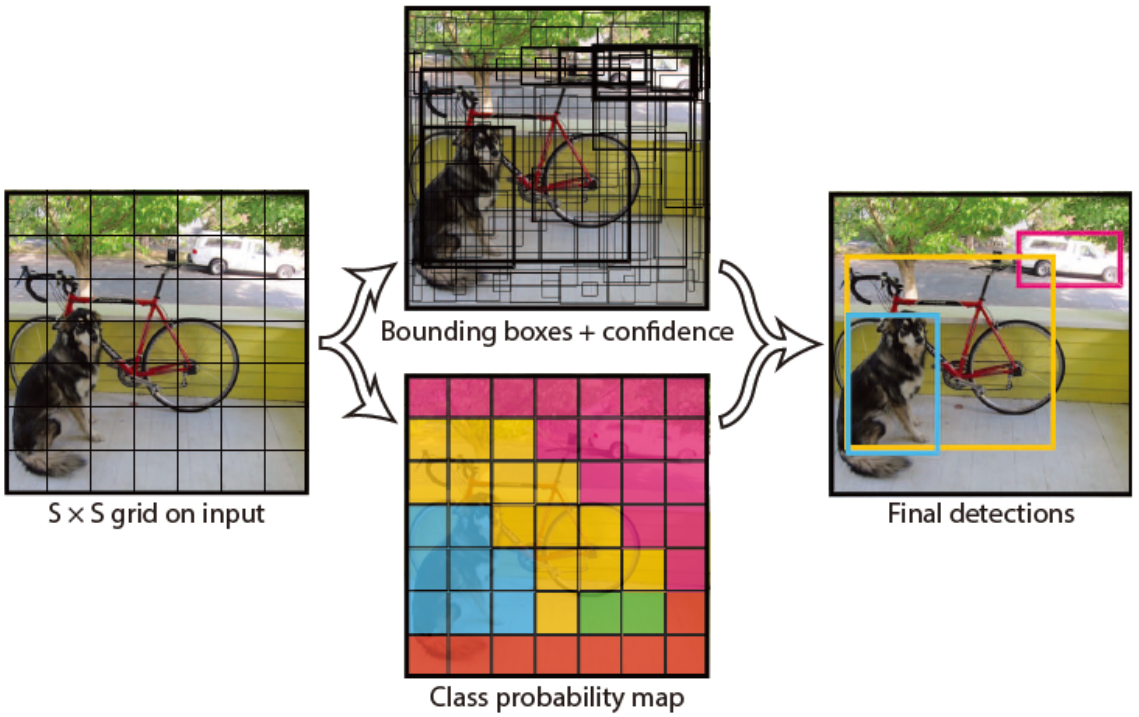


**Figure 2: The Model.** Our system models detection as a regression problem. It divides the image into an $S \times S$ grid and for each grid cell predicts $B$ bounding boxes, confidence for those boxes, and $C$ class probabilities. These predictions are encoded as an $S \times S \times (B * 5 + C)$ tensor.

## Network Design

- GoogLeNet을 참고해 Classifier를 먼저 만듬
- CNN으로 모델을 구현하고 PASCAL VOC Detection Dataset으로 평가
  - 초기의 합성곱층은 이미지로부터 feature를 추출하고, 전결합층은 확률과 좌표를 예측
  - 24개의 합성곱층과 2개의 전결합층으로 이루어짐
  - GoogLeNet에서 사용된 인셉션 모듈을 대신하여 단순한 1 * 1 reduction layer, 3 * 3 합성곱층 사용
  - Fast YOLO는 9개의 합성곱층만을 이용하였으며 필터의 갯수는 더 적음

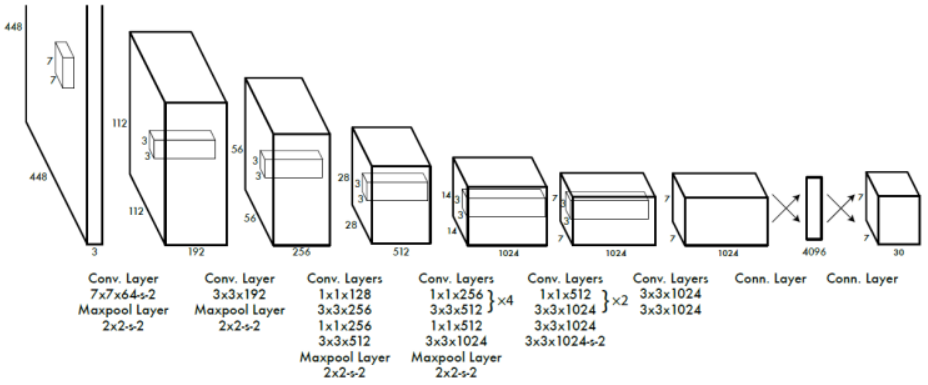


**Figure 3: The Architecture.** Our detection network has 24 convolutional layers followed by 2 fully connected layers. Alternating $1 \times 1$ convolutional layers reduce the features space from preceding layers. We pretrain the convolutional layers on the ImageNet classification task at half the resolution ($224 \times 224$ input image) and then double the resolution for detection.

## Training

- ImageNet의 1000-class competition dataset으로 사전 훈련 진행
  - 20개의 합성곱층을 사용했고 average-pooling layer와 전결합층을 사용
  - 성능 향상을 위해 Layer 수와 해상도를 높임
    - 4개의 합성곱층과 2개의 전결합층을 더함
  - Detector를 만들기 위해 class들에 대한 확률뿐만 아닌 Bounding Box 위치를 예측
    - 마지막 Layer는 class probabilities와 bounding box coordinates 모두 예측
    - 마지막 layer에서 선형 활성화 함수를 사용 그 외에 모든 layer에서는 leaky rectified linear activation을 사용

$$\phi(x) = \begin{cases} x, & \text{if } x > 0 \\ 0.1x, & \text{otherwise} \end{cases}$$