L_{reg} term of RPN loss function

$$L(\{p_i\},\{t_i\}) = \frac{1}{N_{cls}}\sum_i L_{cls}(p_i, p_i^*) + \lambda\frac{1}{N_{reg}}\sum_i p_i^* L_{reg}(t_i, t_i^*).$$

$$t_x = (x - x_a)/w_a, \quad t_y = (y - y_a)/h_a,$$
$$t_w = \log(w/w_a), \quad t_h = \log(h/h_a),$$
$$t_x^* = (x^* - x_a)/w_a, \quad t_y^* = (y^* - y_a)/h_a,$$
$$t_w^* = \log(w^*/w_a), \quad t_h^* = \log(h^*/h_a),$$

다른점 파라미터화(?)가 한 단계 거쳐서 진행

⇓

틀린말, 식 자체는 완전히 동일한 거였음.

regressor의 출력이 t_x 일때,

x_a, w_a 를 P_x, P_w 로 바꿔 보면 동일함을 알수 있음.

$$t_x = (x - P_x) / P_w$$
$$x = t_x \cdot P_w + P_x$$
$$\hat{G}_x = t_x \cdot P_w + P_x \Rightarrow \text{R-CNN 부터 쭉 그대로인 식.}$$

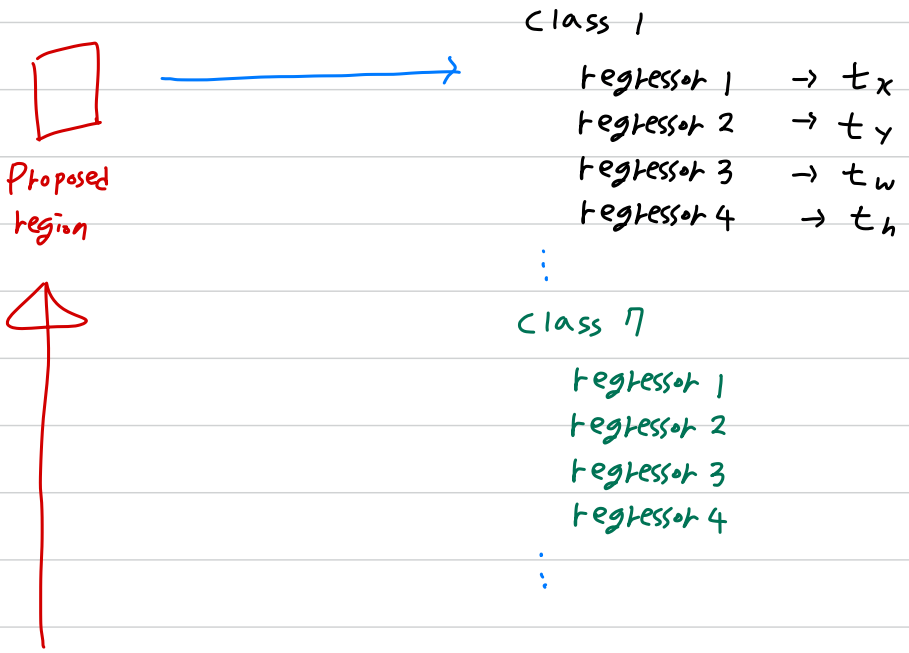
그럼 Anchor 별로 학습한다는 얘기는 어디서 ??

기존 R-CNN, fast R-CNN 의 regressor 를 살펴보면 클래스 별로

regressor 가 4개씩 존재함을 볼 때, 현재 Proposal 영역이

어떻게 생긴 모양이든 이에 대한 최적의 근사값을 회귀하도록 설계되어 있다.

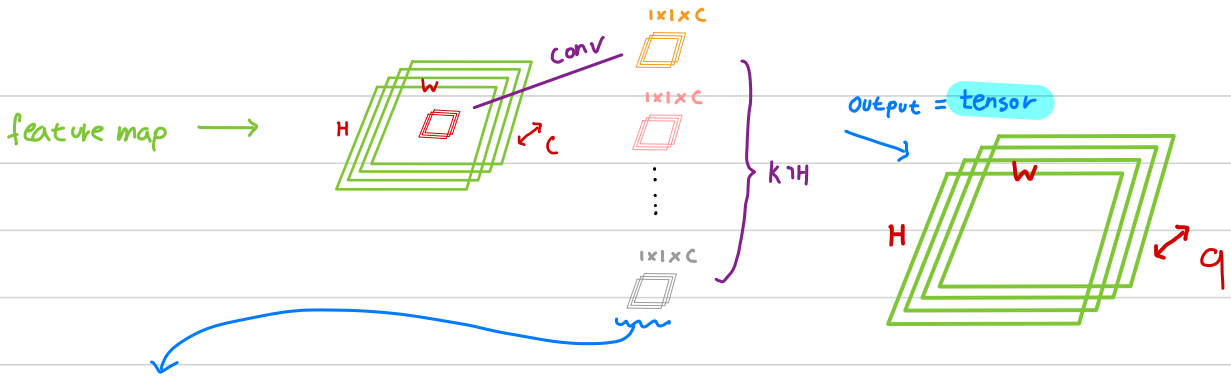
R-CNN, Fast R-CNN



이 영역이 어떤 생김새든 구분을 짓지 않음.

⇒ 이를 바꿔 표현하면 다양한 생김새의 Proposal 에 대해 가중치를 공유하며 학습한다고 할 수 있음!

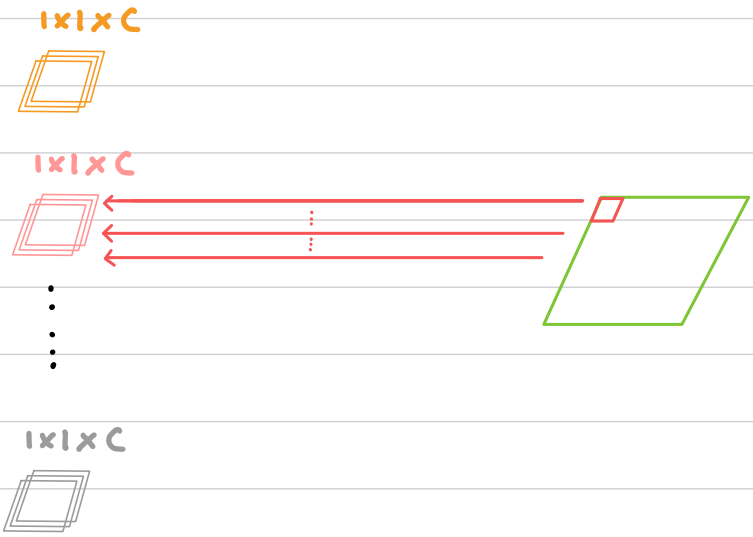
하지만 RPN 의 regressor 는 별개 학습이 가능하도록 설계 되어 있음 !!



RPN 에서는 이 1x1 filter 하나하나가 regressor !!

fully-connected 설계였다면, 한 요소에 전달된 loss 가 모든 weight 에 back-prop 되지만

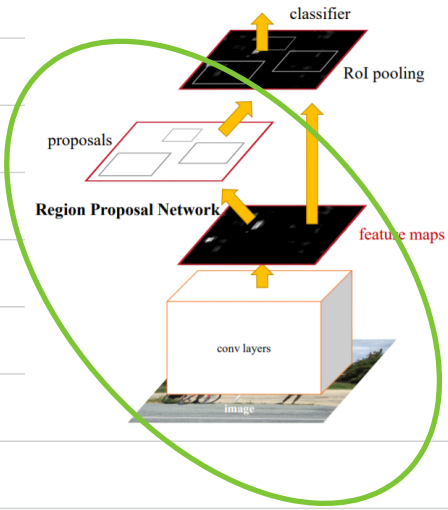
fully-convolutional 이기에 해당 필터에만 손실이 전달.



How RPN and Detector share feature maps ?

4-step alternating training

step 1.



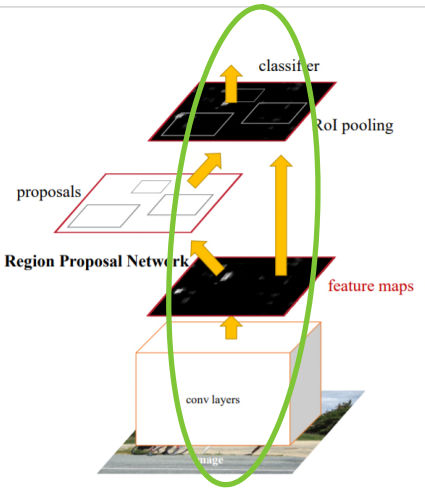
RPN 1차 학습단계.

pretrained Imagenet network 로 초기화 하여 (VGG 16, ZF)

RPN 을 end-to-end 학습

이때 학습된 backbone 을 M_0 라 하자.

step. 2



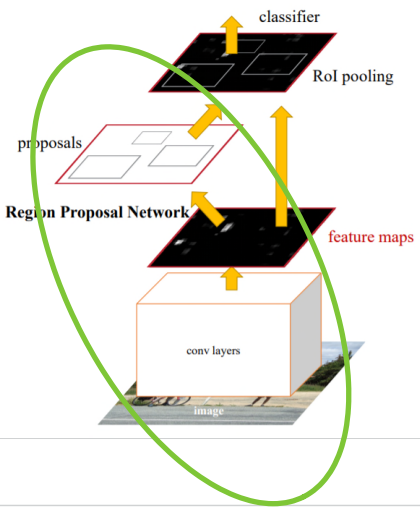
Detector 1차 학습단계.

1차로 학습된 RPN 으로부터 이미지를 M_0 에 넣어 출력된 regional-proposal 만 추출

Detector 또한 마찬가지로 pretrained weight 로 초기화하여 위에서 추출한 ROI 를 참고하여 end-to-end 로 학습.

Step 2. 까지는 feature map 을 공유하지 않으며 이때 학습된 backbone (feature-extractor) 을 M_1 이라 하자.

Step 3.



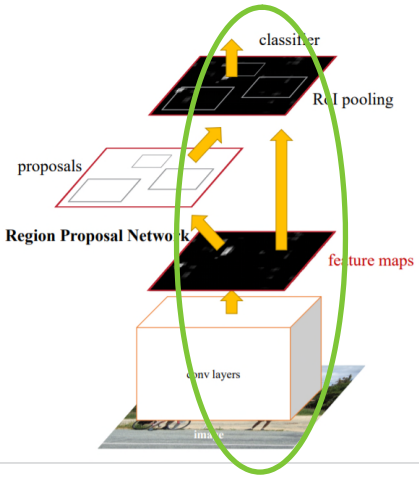
RPN 2차 학습 단계.

이번에는 RPN 의 feature-extractor 를 M1 으로 초기화.

마찬가지로 end-to-end training 을 진행하는데 backbone (M1) 은 고정시키고 3x3 conv 와 1x1 conv filter 들만 갱신한다.

이 단계부터 feature map 을 공유하게 된다.

Step 4.

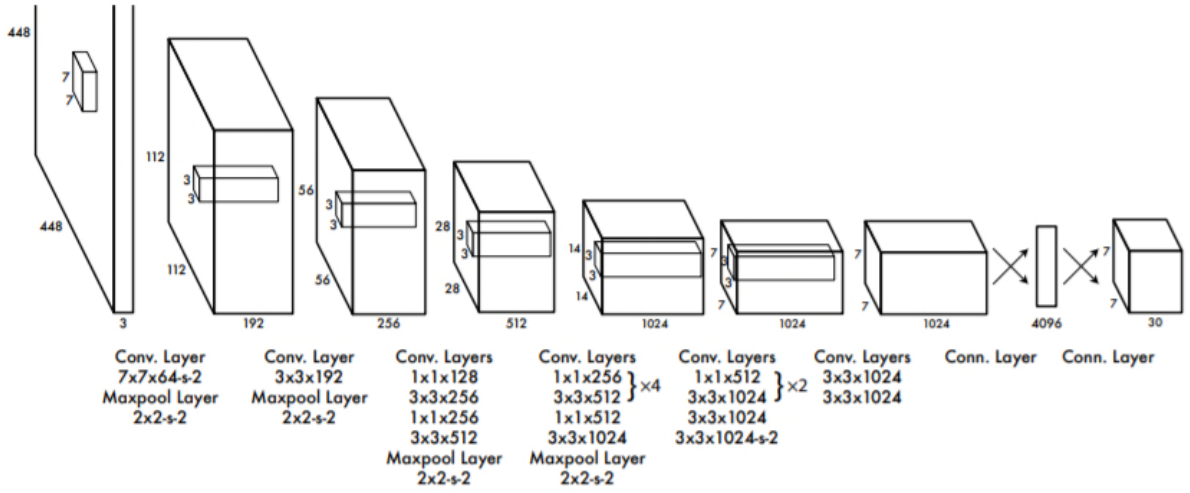


Detector 2차 학습 단계이자 마지막 단계

Step 3 의 결과 가중치 그대로 가져와 Detector 를 end-to-end 학습한다.

이때 backbone 이나 RPN 은 고정시키고 오직 Detector 만 튜닝한다.

Yolo v1 맛보기



생각보다 복잡하고, 생각보다 간단하다.

기존의 R- 패밀리들은 "object dection" task 를

feature map → ROI → classification
→ regression

처럼 multi-task 로 접근했지만

Yolo 의 경우

feature map → regression

이라는 single-regression 문제로 접근 하였다.

↳