# Fast R-CNN

## 1. Fast R-CNN Process

1.1. Selective Search를 통해 RoI(Region of Interest) 추출

1.2. 원본 이미지를 CNN에 통과시켜 feature map 추출

1.3. feature map에 RoI를 Projection

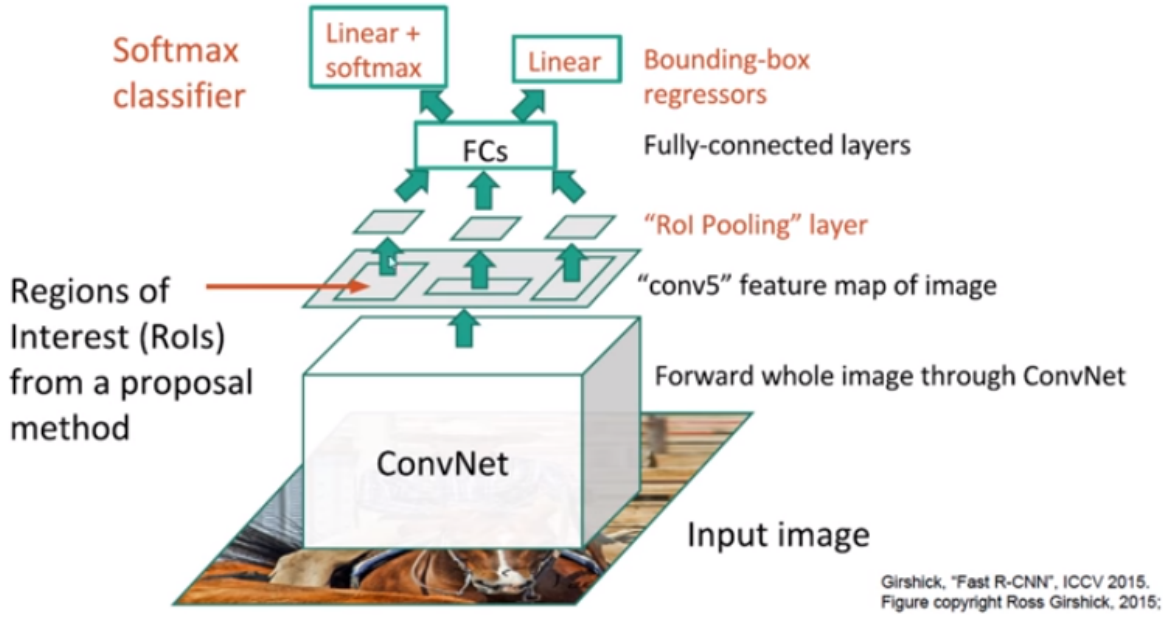
1.4. Projection시킨 feature map의 영역에 대해 RoI Pooling을 진행해 feature vector 추출

1.5. feature vector가 FC layer(Fully Connected layer)를 통과

1.6 FC layer를 통과한 벡터가 softmax classifier, bounding box regressor를 각각 통과

1.7 softmax classifier에서 class에 대한 classification

1.8 bounding box regregressor에서 bounding box에 대한 regression



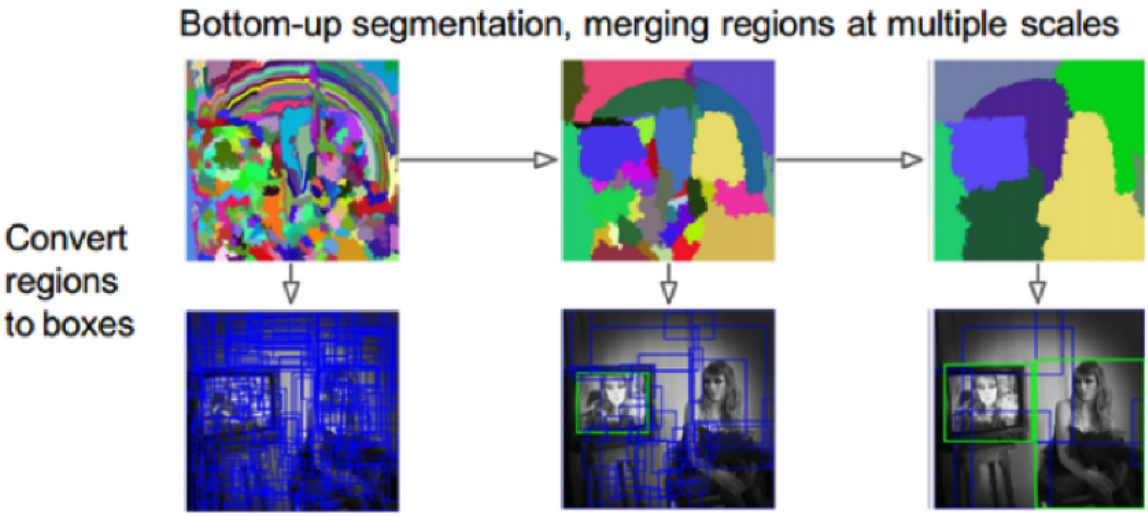

<Fast R-CNN Process>

## 2. R-CNN과의 주요한 차이점

| R-CNN | Fast R-CNN |
| --- | --- |
| 이미지의 RoI마다 CNN 연산 | 하나의 이미지에 한번의 CNN연산 |
| 모델을 한번에 학습 시키지 못함 | CNN 특징 추출부터 classification, bounding box regression까지 하나의 모델에서 학습 |

## 3. Selective Search를 통해 RoI(Region of Interest) 추출

- RoI를 추출하는 방법은 R-CNN과 동일
- Selective Search 알고리즘은 원본 이미지의 segmented area를 통해 RoI를 추출하는 기법임



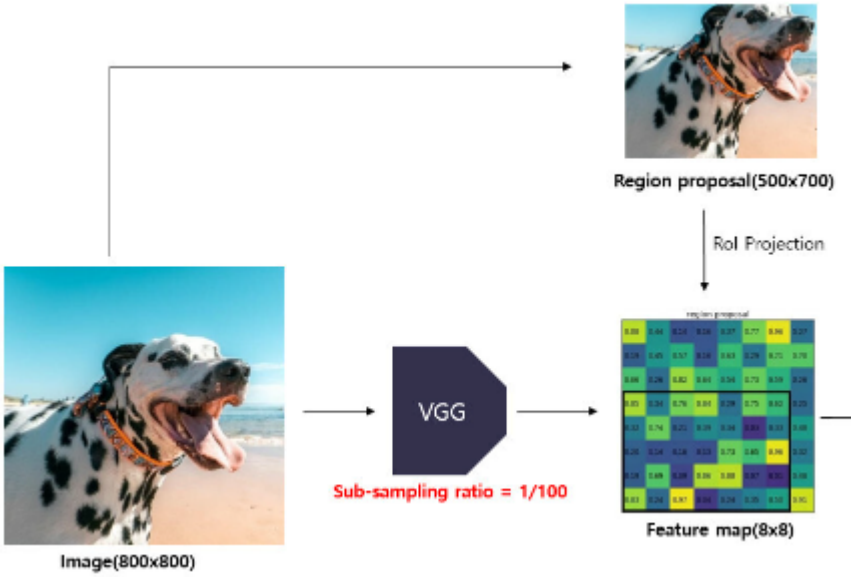

<Selective Search algorithm>

## 4. 원본 이미지를 CNN에 통과시켜 feature map 추출

- Fast R-CNN에서는 feature map 추출을 위한 CNN으로 **VGG16**을 사용

## 5. feature map에 RoI를 Projection

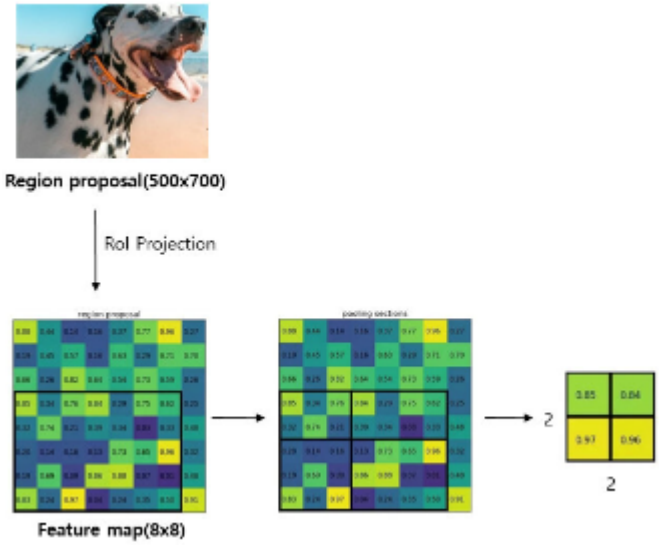
- 원본 이미지를 VGG16에 통과시켜 얻은 feature map에 RoI를 Projection
- feature map에서 원본 이미지의 RoI 영역에 해당하는 영역을 찾는다고 생각하면 쉬움
- 어짜피 RoI는 원본 이미지의 일부분이니 RoI마다 CNN 연산을 하지 말고 원본 이미지만 CNN연산을 해서 전체 feature map에서 RoI의 feature map에 해당하는 영역을 찾자는 아이디어
- 이를 통해 R-CNN의 **2000번 CNN연산**을 **한번의 CNN연산**으로 줄이고 속도를 대폭 높임




<feature extraction & RoI Projection>

## 6. RoI Pooling, feature vector 추출

- RoI를 Projection한 영역에서 feature vector를 추출하는 단계
- RoI마다 각기 다른 크기의 feature map 영역에서 **동일한 크기의 feature vector를 추출**하기 위해서 **RoI Pooling**이 사용됨
- 아래 예시에서 RoI Projection된 5×7 영역이 있는데 해당 영역을 적절하게 나눈 뒤 2×2 크기가 되도록 max pooling한 모습임
- 다른 RoI에서 RoI Projection된 영역의 크기가 5×7가 아닌 6×7이라 하더라도 RoI Pooling을 통해 동일한 2×2 출력을 얻을 수 있음
- 이 2×2 출력(RoI feature map)을 flatten하여 feature vector 추출

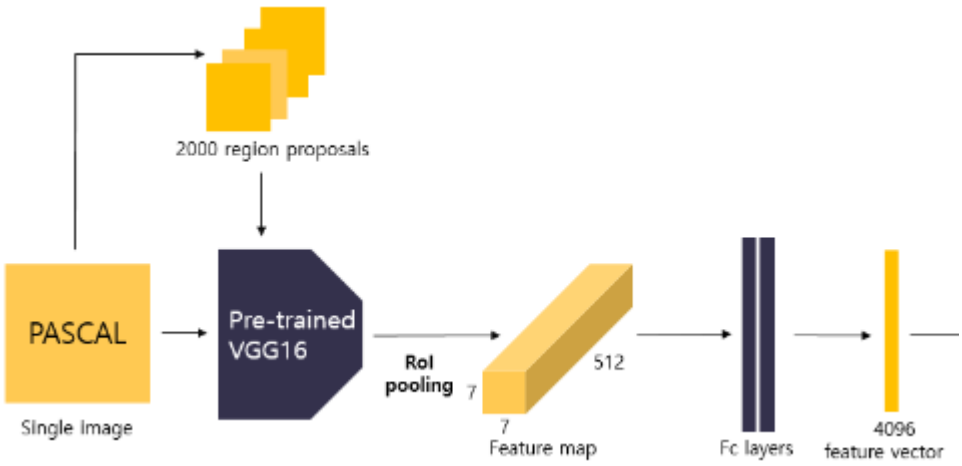



<RoI Pooling>

- 예시에서는 8×8, 2×2이지만 실제 Fast R-CNN에서는 14×14, 7×7임
- R-CNN에서는 동일한 크기의 feature vector를 얻기 위해서 RoI를 warp 시켰음
- 하지만 CNN의 입력 크기가 동일해야 하는 것이 아니라 FC layer의 입력 크기가 동일해야 하는 것
- Fast R-CNN은 RoI Pooling을 통해 FC layer의 입력 크기를 동일하게 맞춰주었으며 RoI를 warp 시키면서 가지는 패널티를 해결함

## 7. feature vector가 FC layer(Fully Connected layer)를 통과

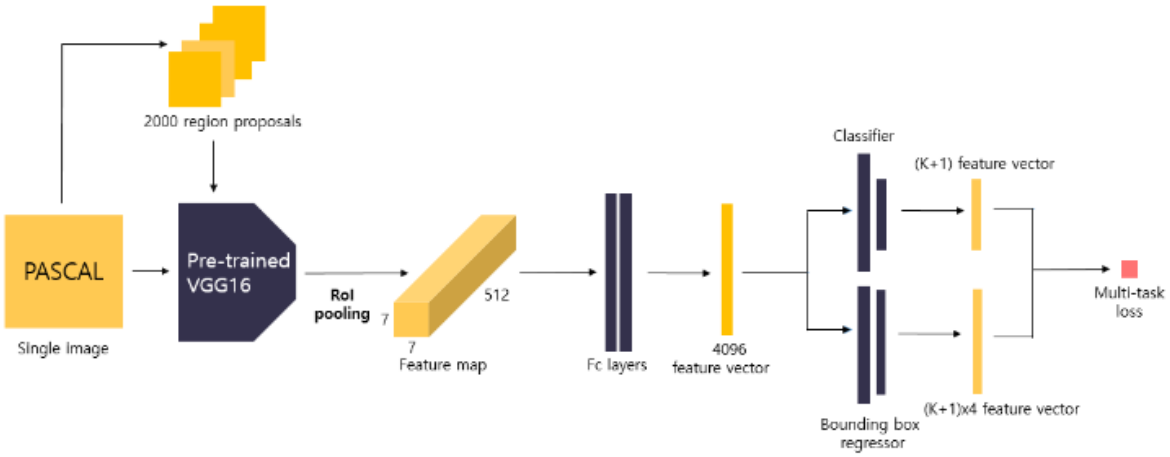
- RoI의 feature vector가 FC layer를 통과함
- FC layer의 입력 노드는 25088개(7×7×512), 출력 노드는 4096개임




<~FC layer>

## 8. softmax classifier, bounding box regressor를 각각 통과

- FC layer의 출력 벡터가 softmax classifier와 bounding box regressor를 각각 통과
- softmax classifier에서는 RoI의 class에 대한 classification이 진행
- softmax classifier의 출력 노드는 K+1개임 (K개의 class + 배경)
- bounding box regressor에서는 RoI의 bounding box에 대한 regression이 진행
- bounding box regressor의 출력 노드는 (K+1)×4개임 (각 class의 bounding box를 표현하는 4개의 값)




<~classification, regression>

## 9. Multi-task loss

- Multi-task loss는 RoI의 class에 대한 분류와 bounding box에 대한 회귀가 잘 되었는지 **종합적으로** 판단할 수 있는 지표임

$$L(p, u, t^u, v) = L_{cls}(p, u) + \lambda[u \geq 1]L_{loc}(t^u, v)$$

$p = (p0, \ldots\ldots, p_k)$ : (K+1)개의 class score
$u$ : ground truth class score
$t^u = (t^u_x, t^u_y, t^u_w, t^u_h)$ : 예측한 bounding box 좌표를 조정하는 값
$v = (v_x, v_y, v_w, v_h)$ : 실제 bounding box의 좌표값

$L_{cls}(p, u) = -logp_u$ : classification loss(**Log loss**)
$L_loc(t^u, v) = \sum_{i \in \{x,y,w,h\}} smooth_{L_1}(t^u_i - v_i)$ : regression loss(**Smooth L1 loss**)

$$smooth_{L_1}(t^u_i - v_i) = \begin{cases} 0.5x^2, if|x| < 1 \\ |x| - 0.5, otherwise \end{cases}$$

$\lambda$ : 두 loss 사이의 가중치를 조정하는 balancing hyperparamter

<Multi-task loss>

- 간단하게 $L_{cls}(p, u)$는 classification에 대한 Loss, $[u \geq 1]L_{loc}(t^u, v)$는 regression에 대한 Loss, $\lambda$는 두 Loss에 대한 가중치임
- CNN(VGG16), softmax classifier, bounding box regressor는 Multi-task loss를 줄인다는 공통된 목표를 가지고 학습 됨
- Fast R-CNN은 Multi-task loss를 통해 CNN(VGG16), softmax classifier, bounding box regressor가 한번에 end to end로 학습될 수 있음

## 10. 결론

- Fast R-CNN의 main idea는 **RoI Pooling**과 **Multi-task loss**임
- RoI Pooling을 통해 RoI 마다 CNN 연산을 하지 않아도 되고 warp를 통해 원본 이미지가 훼손되는 패널티 또한 피할 수 있음

- R-CNN은 AlexNet, linear SVM, Bounding box regressor를 따로 학습 시켜 속도가 느리고 각 모델이 greedy해질 수 있다는 단점을 가지고 있었지만 Fast R-CNN은 Multi-task loss를 통해 CNN(VGG16), softmax classifier, bounding box regressor를 end to end로 학습 시키면서 속도와 성능을 둘 다 향상 시킴
- 하지만 Selective Search를 통한 RoI 추출은 여전히 CNN 외부에서 진행되므로 이 부분이 속도를 상당히 느리게 함