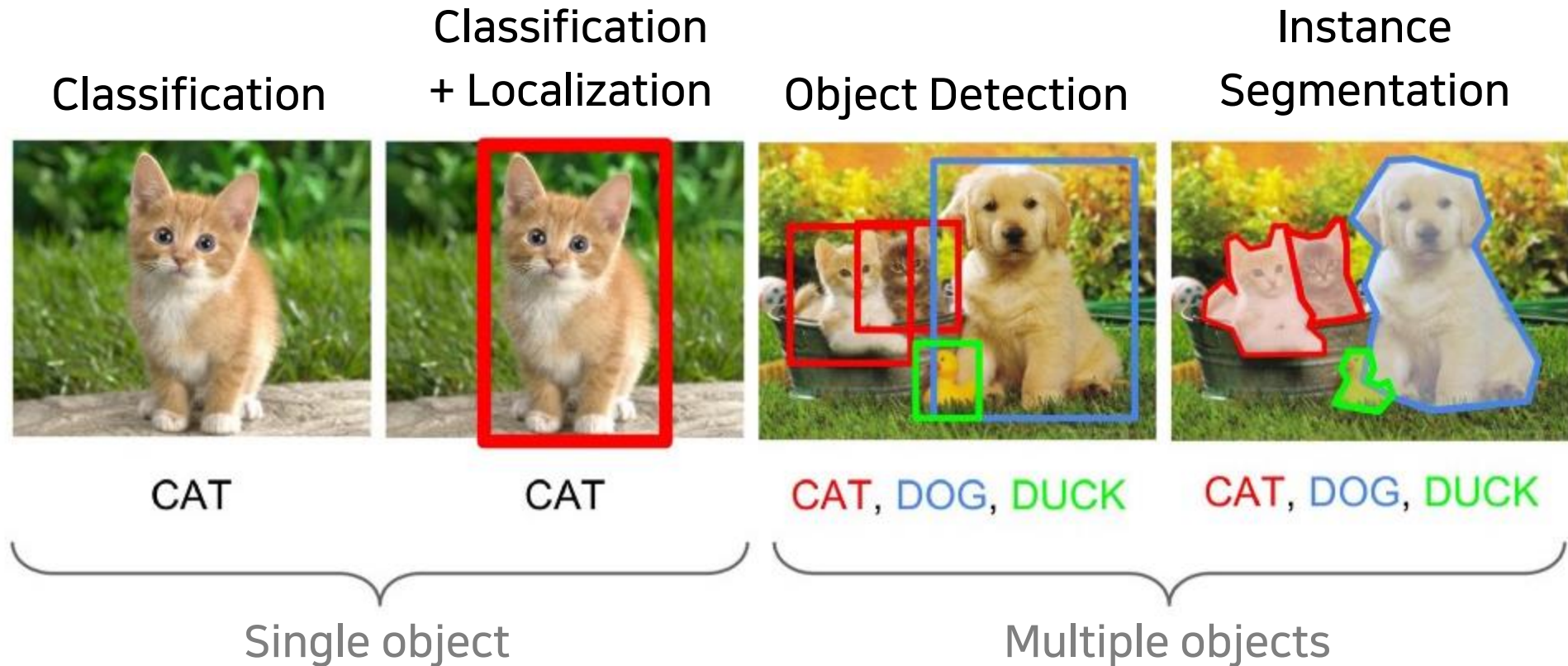


## 사물을 인식하는 다양한 문제 상황

- 이미지 내에서 사물 인식하는 방법에는 다양한 유형이 존재합니다.
  - Object Detection은 다수의 사물 존재하는 상황에서 각 사물의 위치와 클래스를 찾는 작업입니다.

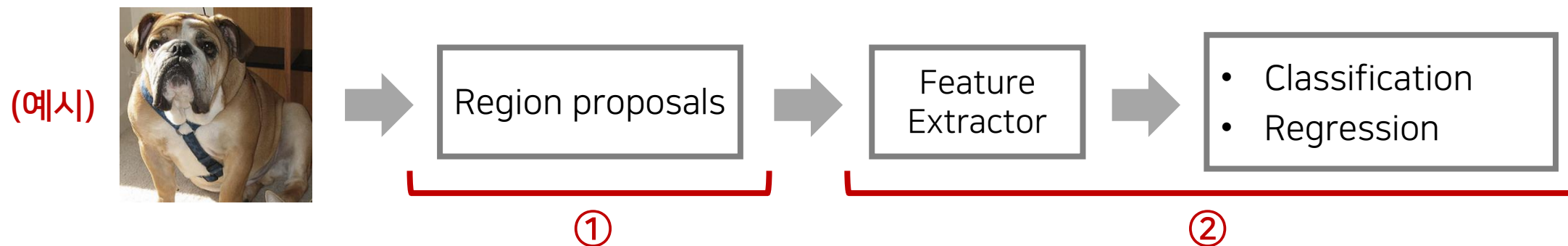


<https://medium.com/zylapp/review-of-deep-learning-algorithms-for-object-detection-c1f3d437b852>

# 객체 검출(Object Detection) 방식: 2-Stage 방식과 1-Stage 방식 비교하기

- 2-Stage Detector

- 물체의 ① 위치를 찾는 문제(localization)와 ② 분류(classification) 문제를 순차적으로 해결합니다.

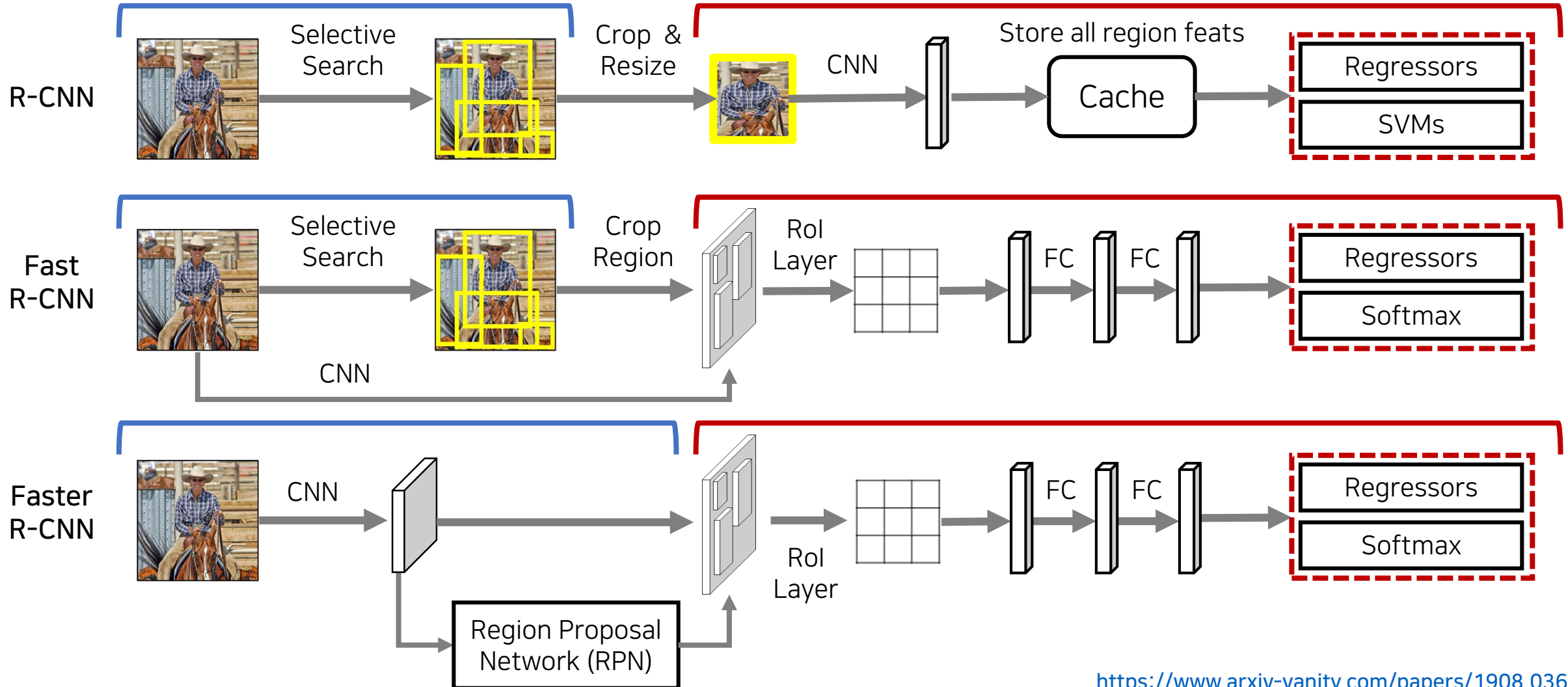


- 1-Stage Detector

- 물체의 위치를 찾는 문제(localization)와 분류(classification) 문제를 한 번에 해결합니다.



## 객체 검출 방식: 2-Stage 방식 예시



# R-CNN 계열 기법들의 발전 방향 및 장단점 분석

발전  
방향

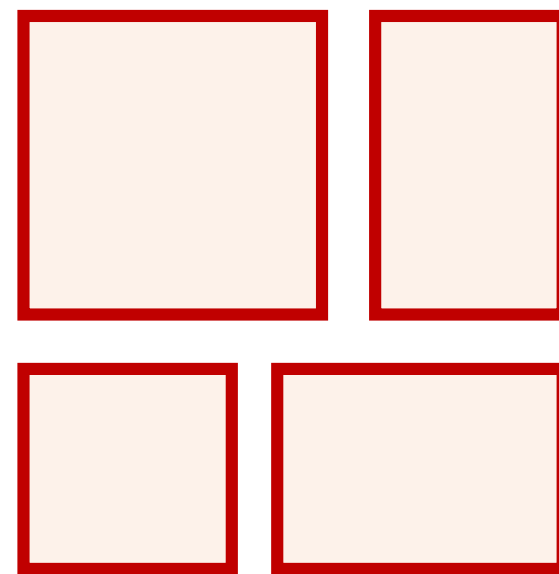
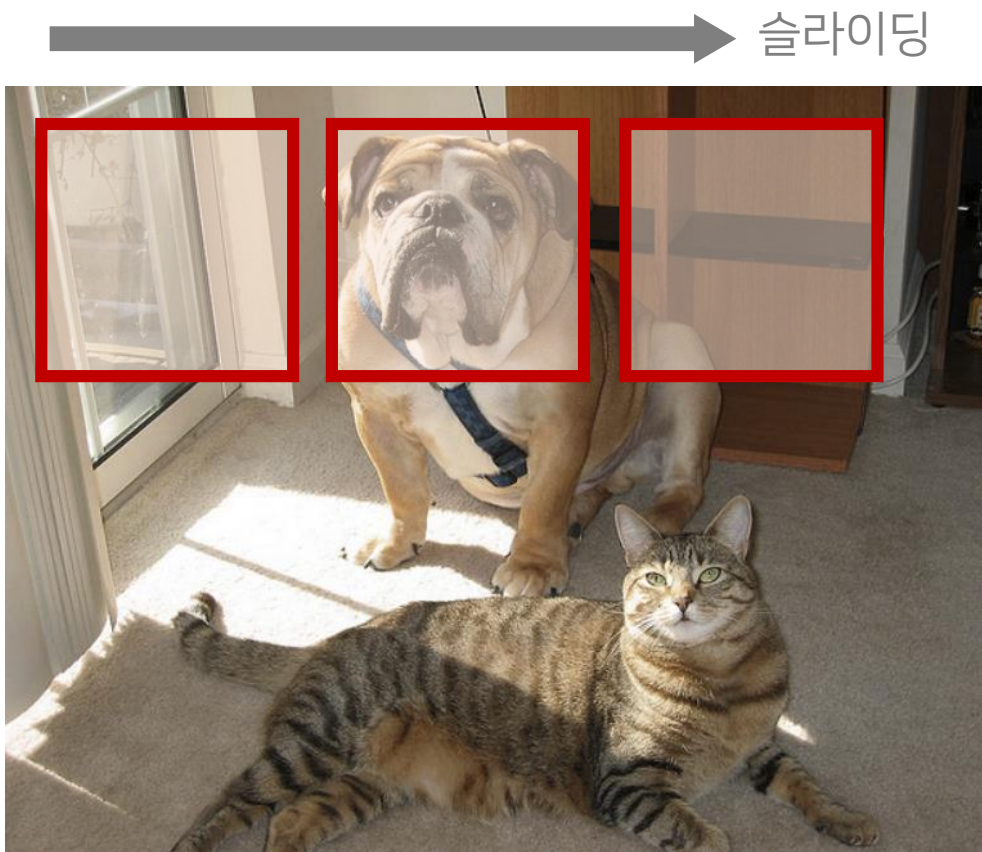


R-CNN (CVPR 2014)	장점	CNN을 이용해 각 Region의 클래스를 분류할 수 있습니다.
	단점	전체 프레임워크를 End-to-End 방식으로 학습할 수 없습니다. 따라서 Global Optimal Solution을 찾기 어렵습니다.
Fast R-CNN (ICCV 2015)	장점	Feature Extraction, RoI Pooling, Region Classification, Bounding Box Regression 단계(step)를 모두 End-to-End로 묶어서 학습할 수 있습니다.
	단점	여전히 첫 번째 Selective Search는 CPU에서 수행되므로 속도가 느립니다.
Faster R-CNN (NIPS 2015)	장점	RPN을 제안하여, 전체 프레임워크를 End-to-End로 학습할 수 있습니다.
	단점	여전히 많은 컴포넌트로 구성되며, Region Classification 단계에서 각 특징 벡터(feature vector)는 개별적으로 FC layer로 Forward 됩니다.

## Region Proposals

## 물체가 있을 법한 위치 찾기(Region Proposal): Sliding Window

- 이미지에서 다양한 형태의 윈도우(window)를 슬라이딩(sliding)하며 물체가 존재하는지 확인합니다.
  - 너무 많은 영역에 대하여 확인해야 한다는 단점이 있습니다.



다양한 형태의 윈도우(window)



# 물체가 있을 법한 위치 찾기(Region Proposal): Selective Search

- 인접한 영역(region)끼리 유사성을 측정해 큰 영역으로 차례대로 통합해 나갑니다. (R-CNN / Fast R-CNN)

---

## Algorithm 1: Hierarchical Grouping Algorithm

---

**Input:** (colour) image

**Output:** Set of object location hypotheses  $L$

Obtain initial regions  $R = \{r_1, \dots, r_n\}$  using [13]

Initialise similarity set  $S = \emptyset$

**foreach** *Neighbouring region pair*  $(r_i, r_j)$  **do**

    Calculate similarity  $s(r_i, r_j)$

$S = S \cup s(r_i, r_j)$

**while**  $S \neq \emptyset$  **do**

    Get highest similarity  $s(r_i, r_j) = \max(S)$

    Merge corresponding regions  $r_t = r_i \cup r_j$

    Remove similarities regarding  $r_i : S = S \setminus s(r_i, r_*)$

    Remove similarities regarding  $r_j : S = S \setminus s(r_*, r_j)$

    Calculate similarity set  $S_t$  between  $r_t$  and its neighbours

$S = S \cup S_t$

$R = R \cup r_t$

Extract object location boxes  $L$  from all regions in  $R$

---

Selective Search 알고리즘



→ (처리 과정)

## 객체 검출(Object Detection) 정확도 측정 방법



## 성능 평가 지표: 정확도(Precision)와 재현율(Recall)

### 예측 결과 (Predict Result)

실제 정답 (Ground Truth)		예측 결과 (Predict Result)	
		Positive	Negative
	Positive	TP (True Positive) 있다고 올바르게 판단	FN (False Negative) 없다고 잘못 판단
	Negative	FP (False Positive) 있다고 잘못 판단	TN (True Negative) 없다고 올바르게 판단

- Precision (정확도)
  - 올바르게 탐지한 물체의 수(TP) / 모델이 탐지한 물체의 개수(TP + FP)
- Recall (재현율)
  - 올바르게 탐지한 물체의 수(TP) / 실제 정답 물체의 수(TP + FN)

## 성능 평가 지표: 정확도(Precision)와 재현율(Recall)

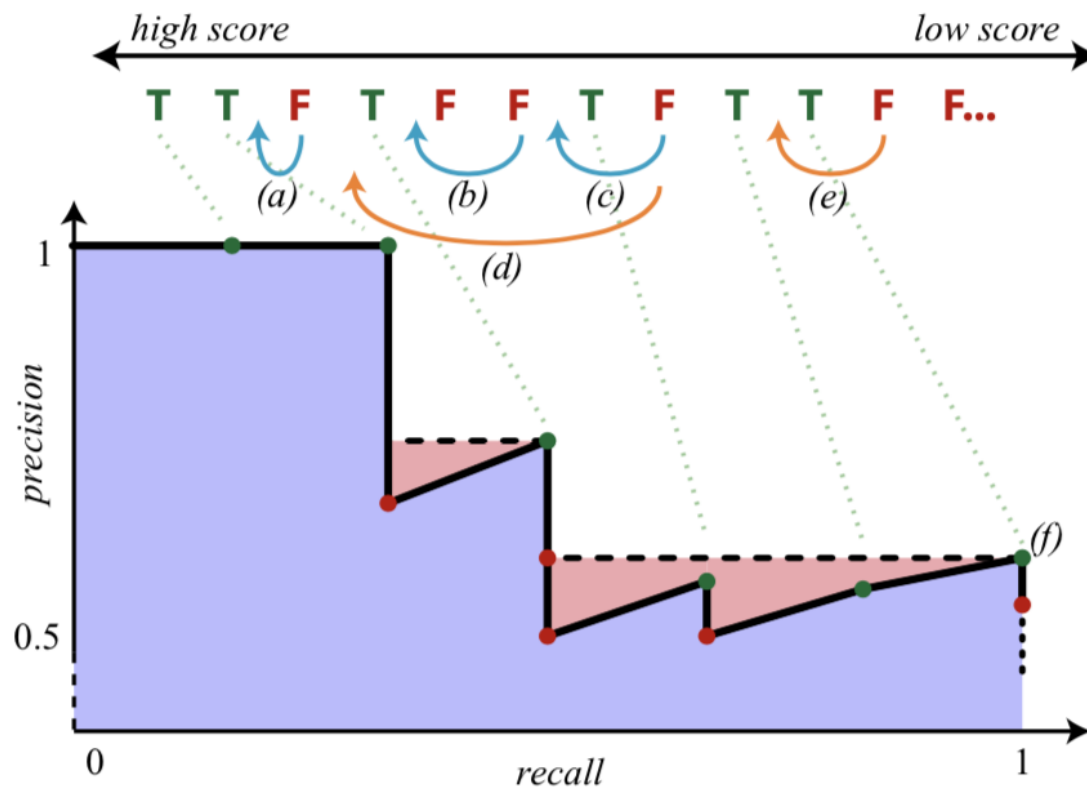
- 예시 1) 강아지가 20마리 존재할 때 모델이 10마리의 강아지를 검출하여 5마리는 정확히 맞힌 경우
  - 정확도(Precision) =  $5 / 10 = 50\%$ , 재현율(Recall) =  $5 / 20 = 25\%$
- 예시 2) 강아지가 10마리 존재할 때 모델이 20마리의 강아지를 검출하여 7마리는 정확히 맞힌 경우
  - 정확도(Precision) =  $7 / 20 = 35\%$ , 재현율(Recall) =  $7 / 10 = 70\%$



- 만약 모든 영역에 대하여 전부 사물이 존재한다고 판단을 해버리면 어떤 일이 벌어질까요?
  - 재현율은 높아지지만, 정확도는 떨어지게 됩니다.
- 만약 매우 확실할 때만(confidence가 높을 때만) 사물이 존재한다고 판단하면 어떤 일이 벌어질까요?
  - 정확도는 높아지지만, 재현율은 떨어지게 됩니다.

# Average Precision

- 일반적으로 정확도(Precision)와 재현율(Recall)은 반비례 관계를 가집니다.
- 따라서 Average Precision으로 성능을 평가하는 경우가 많습니다. (단조 감소 그래프로 넓이를 계산)



<https://www.catalyzex.com/paper/arxiv:1607.03476>

# Intersection over Union (IoU)

- IoU란 두 바운딩 박스가 겹치는 비율을 의미합니다.
  - 성능 평가 예시: mAP@0.5는 정답과 예측의 IoU가 50% 이상일 때 정답으로 판정하겠다는 의미입니다.
  - NMS 계산 예시: 같은 클래스(class)끼리 IoU가 50% 이상일 때 낮은 confidence의 box를 제거합니다.

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}}$$


(교집합)

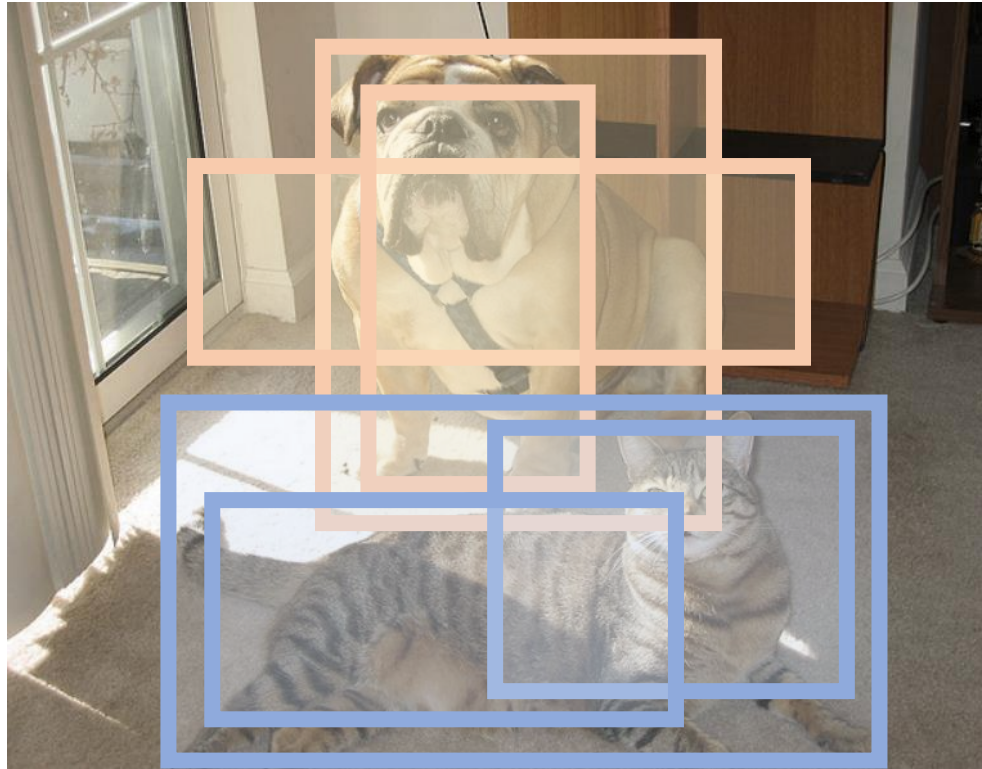
(합집합)

<https://www.pyimagesearch.com/2016/11/07/intersection-over-union-iou-for-object-detection/>

# NMS (Non Maximum Suppression)

- 객체 검출(object detection)에서는 하나의 인스턴스(instance)에 하나의 bounding box가 적용되어야 합니다.
  - 따라서 여러 개의 bounding box가 겹쳐 있는 경우에 하나로 합치는 방법이 필요합니다.

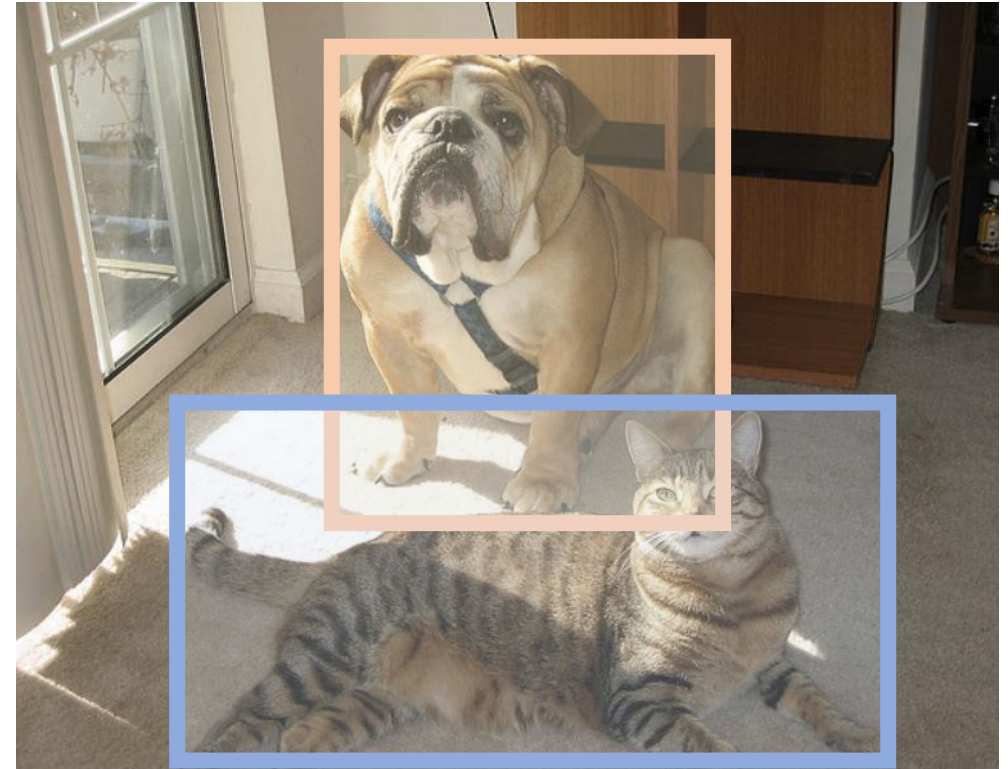
■ : dog ■ : cat



NMS



IoU가 특정 임계점(threshold) 이상인 중복 box 제거

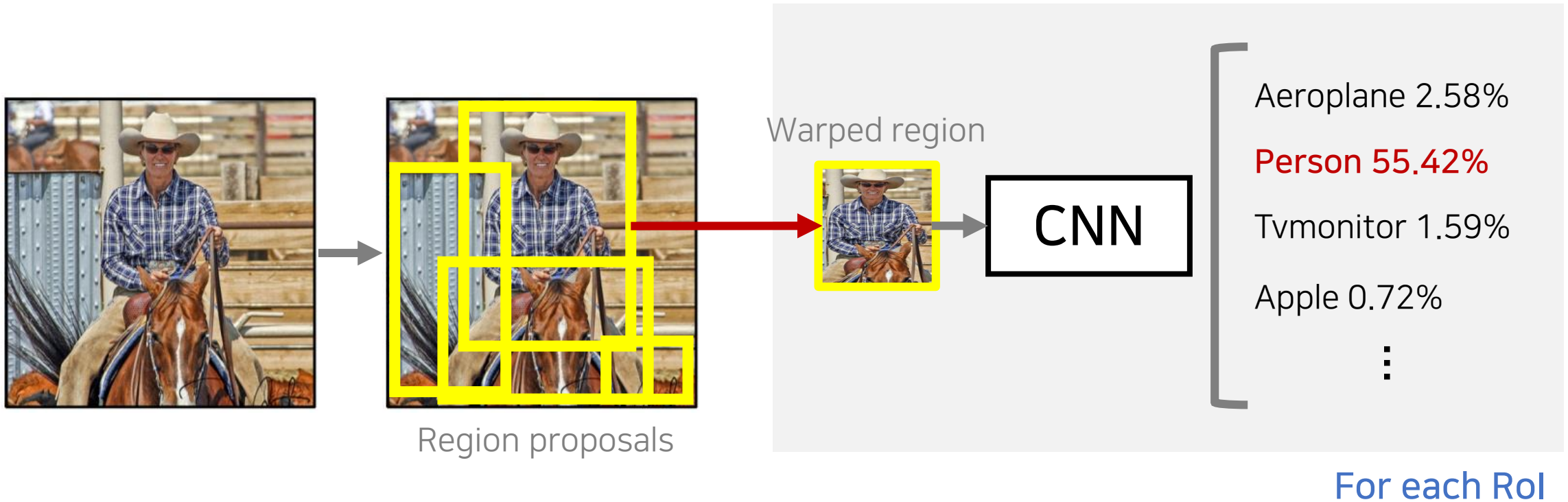


R-CNN (CVPR 2014)

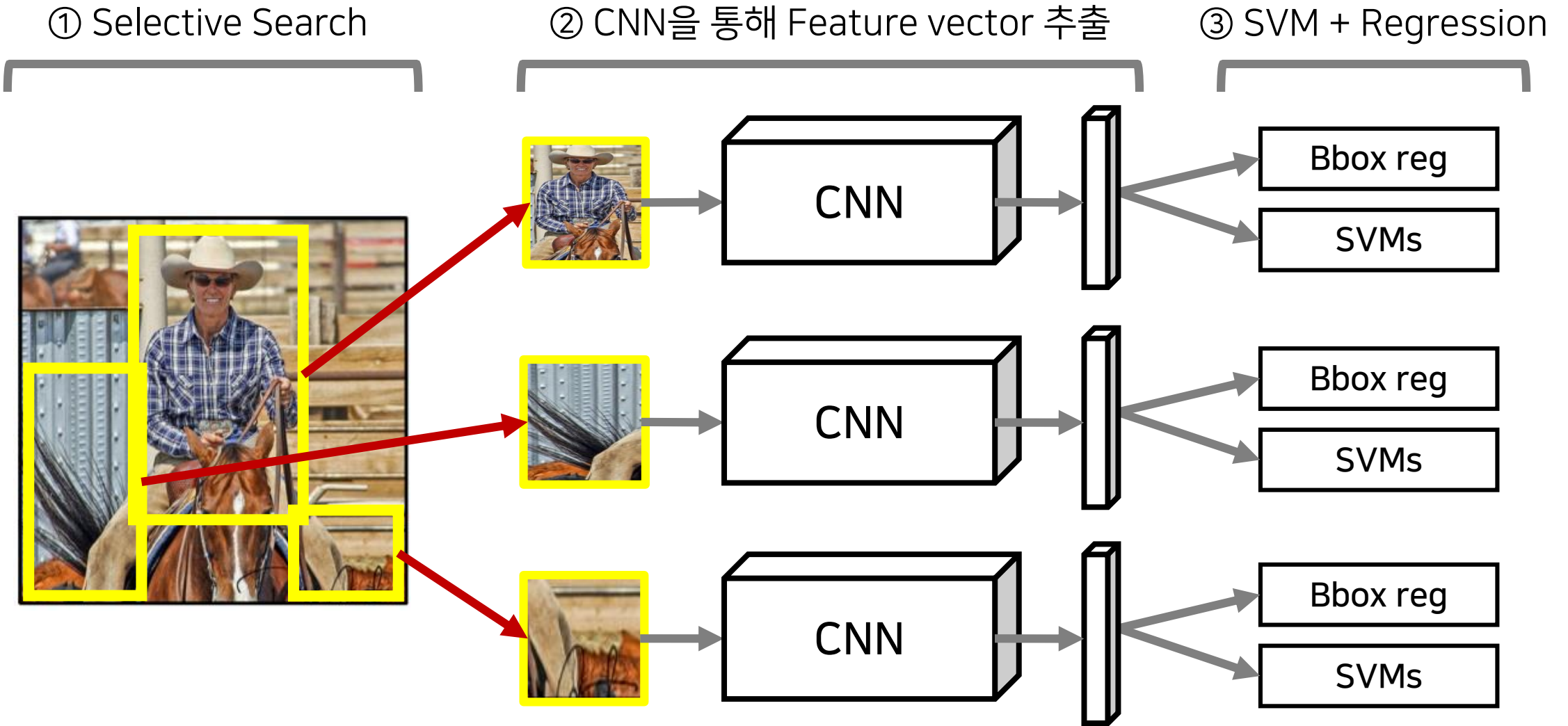


## R-CNN: Regions with CNN features (CVPR 2014)

- R-CNN 원본 논문에서는 먼저 Selective Search를 이용해 2,000개의 Region Proposal을 생성합니다.
  - 각 Region Proposal을 일일이 CNN에 넣어서(forward) 결과를 계산합니다.



# R-CNN: Regions with CNN features (CVPR 2014)

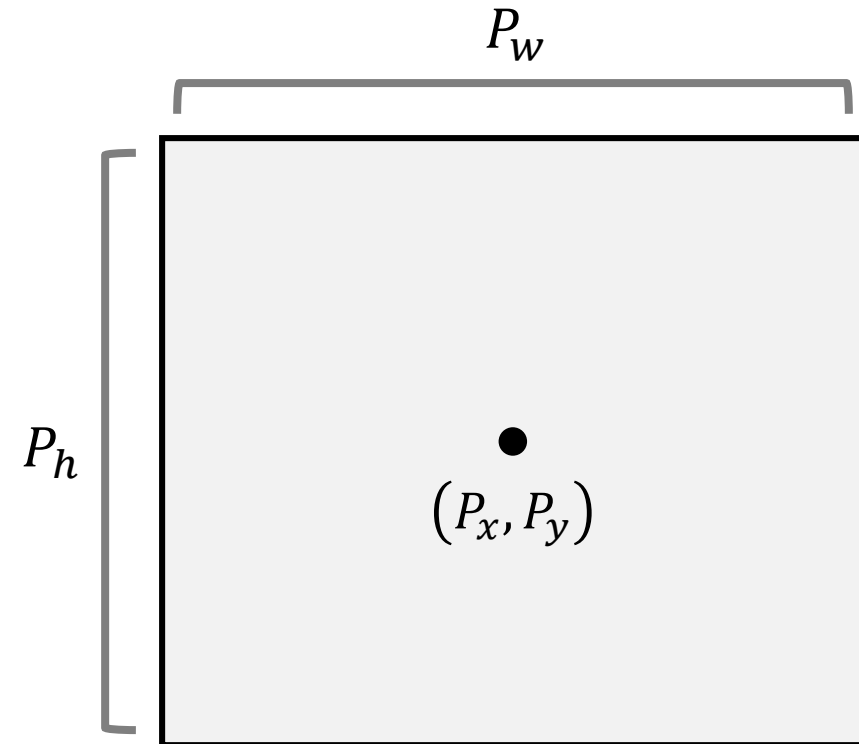


## R-CNN의 동작 과정

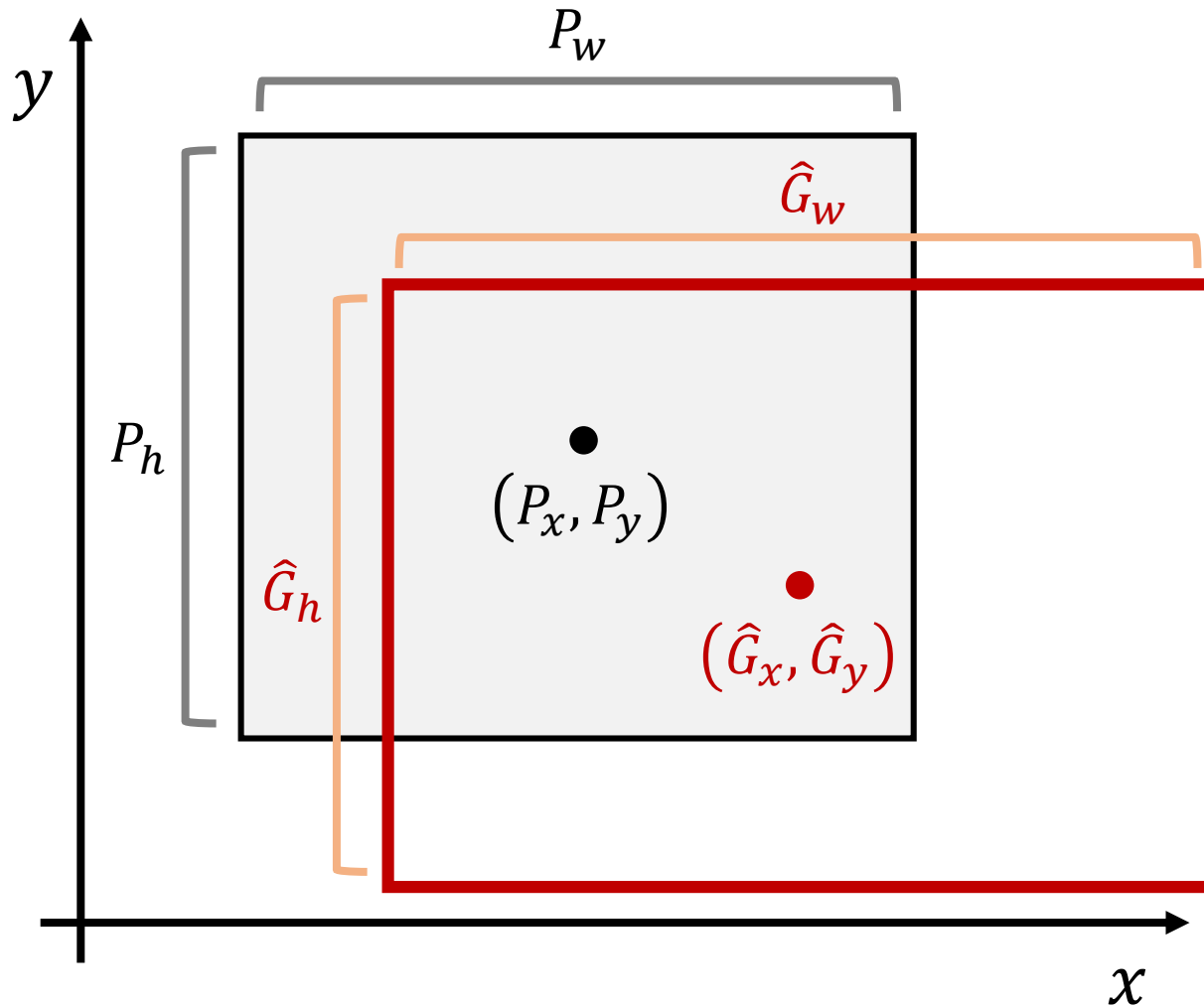
1. Selective search를 이용해 2,000개의 RoI(Region of Interest)를 추출합니다.
2. 각 RoI에 대하여 warping을 수행하여 동일한 크기의 입력 이미지로 변경합니다.
3. Warped image를 CNN에 넣어서(forward) 이미지 feature를 추출합니다.
4. 해당 feature를 SVM에 넣어 클래스(class) 분류 결과를 얻습니다.
  - 이때 각 클래스에 대해 독립적으로 훈련된 이진(binary) SVM을 사용합니다.
5. 해당 feature를 regressor에 넣어 위치(bounding box)를 예측합니다.

## R-CNN: Bounding Box Regression

- 지역화(localization) 성능을 높이기 위해 bounding-box regression을 사용합니다.
- 학습 데이터셋:  $\{(P^i, G^i)\}_{i=1, \dots, N}$ 
  - 예측 위치:  $P^i = (P_x^i, P_y^i, P_w^i, P_h^i)$
  - 실제 위치:  $G^i = (G_x^i, G_y^i, G_w^i, G_h^i)$



## R-CNN: Bounding Box Regression



- 학습할 4개의 파라미터(parameter)
  - $d_x(P), d_y(P), d_w(P), d_h(P)$
  - Linear regression을 진행합니다.

$$\hat{G}_x = P_w d_x(P) + P_x$$

$$\hat{G}_y = P_h d_y(P) + P_y$$

$$\hat{G}_w = P_w \exp(d_w(P))$$

$$\hat{G}_h = P_h \exp(d_h(P))$$

<https://lilianweng.github.io/lil-log/2017/12/31/object-recognition-for-dummies-part-3.html>

## R-CNN의 한계점

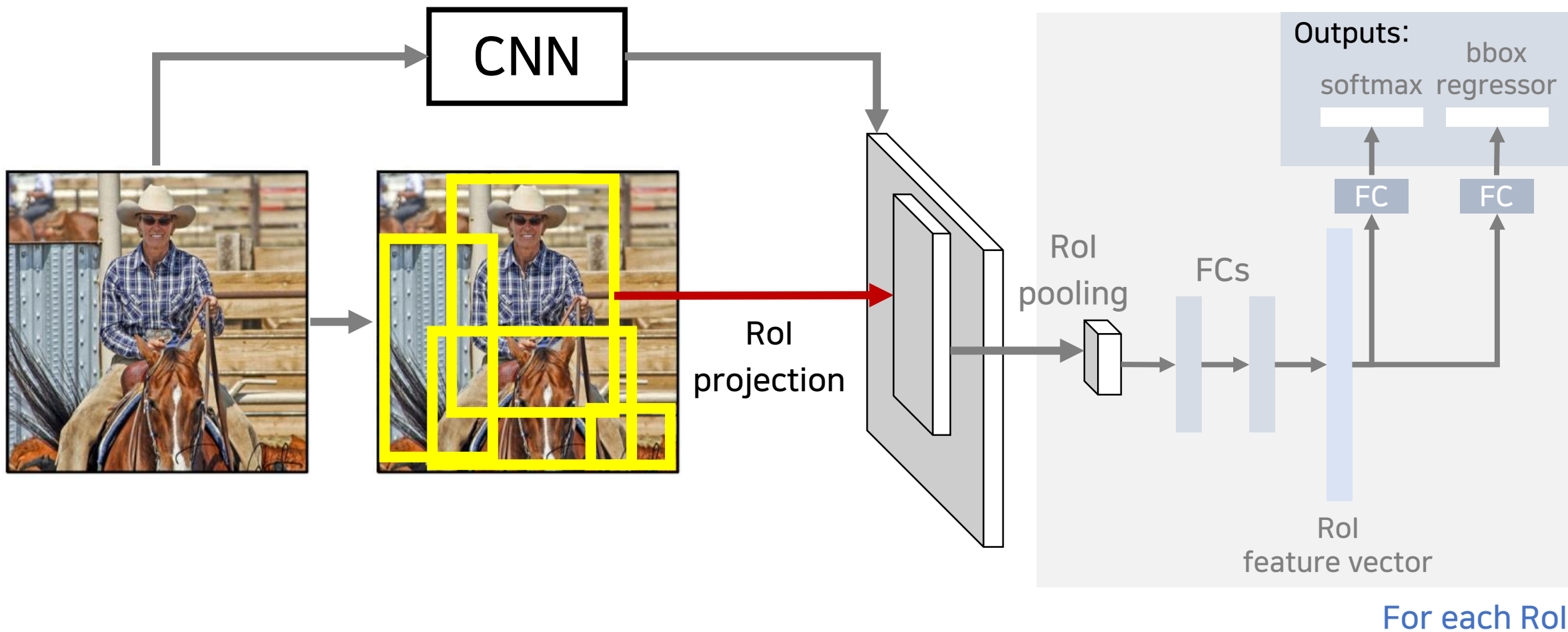
- 입력 이미지에 대하여 CPU 기반의 Selective Search를 진행해야 하므로 많은 시간이 소요됩니다.
- 전체 아키텍처에서 SVM, Regressor 모듈이 CNN과 분리되어 있습니다.
  - CNN은 고정되므로 SVM과 Bounding Box Regression 결과로 CNN을 업데이트할 수 없습니다.
  - 다시 말해 end-to-end 방식으로 학습할 수 없습니다.
- 모든 RoI(Region of Interest)를 CNN에 넣어야 하기 때문에 2,000번의 CNN 연산이 필요합니다.
  - 학습(training)과 평가(testing) 과정에서 많은 시간이 필요합니다.



## Fast R-CNN (ICCV 2015)

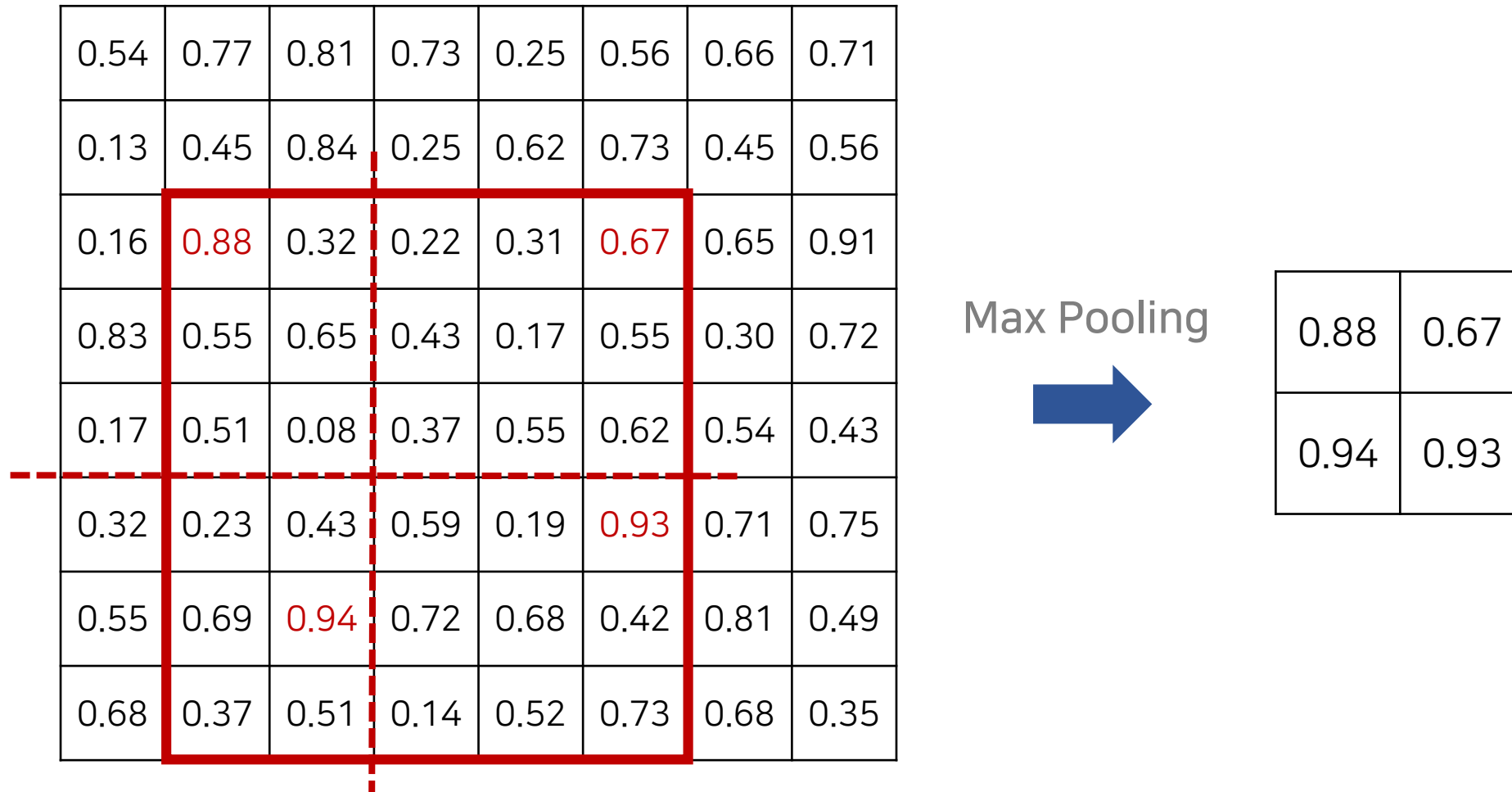
# Fast R-CNN (ICCV 2015)

- 동일한 Region proposal을 이용하되 이미지를 **한 번만 CNN에 넣어** Feature Map을 생성합니다.



## Fast R-CNN: RoI Pooling Layer

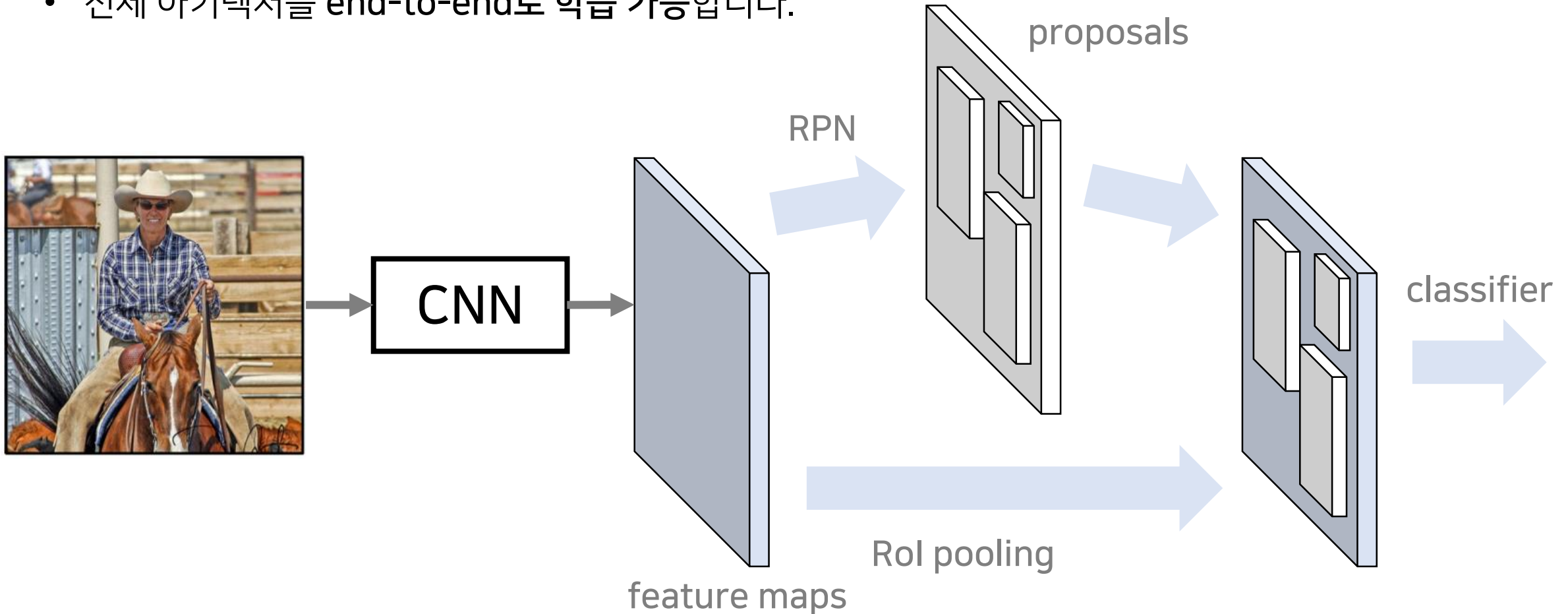
- 각 RoI 영역에 대하여 맥스 풀링(max pooling)을 이용해 고정된 크기의 벡터(vector)를 생성합니다.



## Faster R-CNN (NIPS 2015)

# Faster R-CNN (NIPS 2015)

- 병목(bottleneck)에 해당하던 Region Proposal 작업을 GPU 장치에서 수행하도록 합니다. (RPN 적용)
  - 전체 아키텍처를 end-to-end로 학습 가능합니다.



# Faster R-CNN: Region Proposal Networks (RPN)

- RPN 네트워크는 feature map이 주어졌을 때 물체가 있을 법한 위치를 예측합니다.
  - k개의 앵커 박스(anchor box)를 이용합니다.
  - 슬라이딩 윈도우(sliding window)을 거쳐 각 위치에 대해 Regression과 Classification을 수행합니다.

