# R-CNN

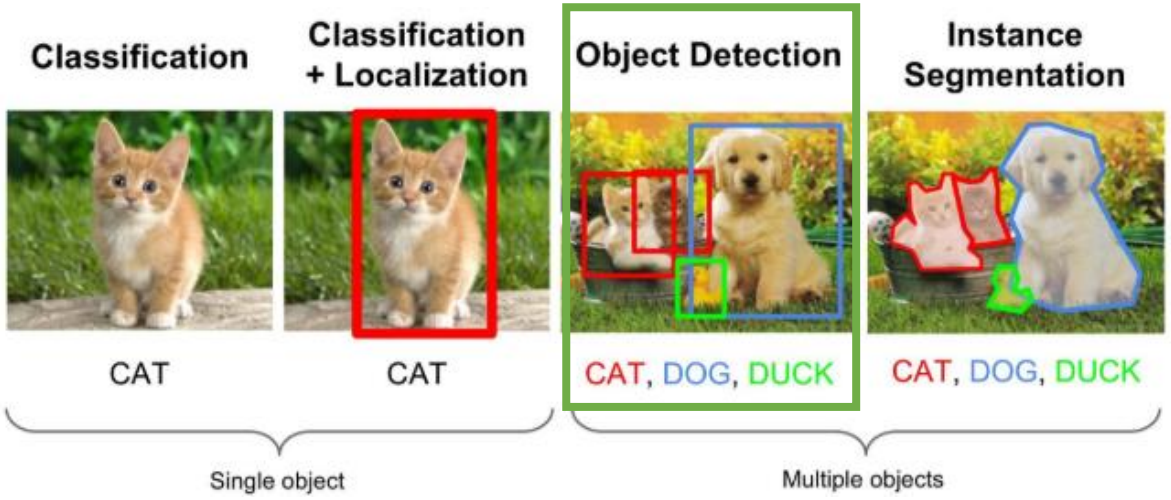
Rich feature hierarchies for accurate object detection and semantic segmentation

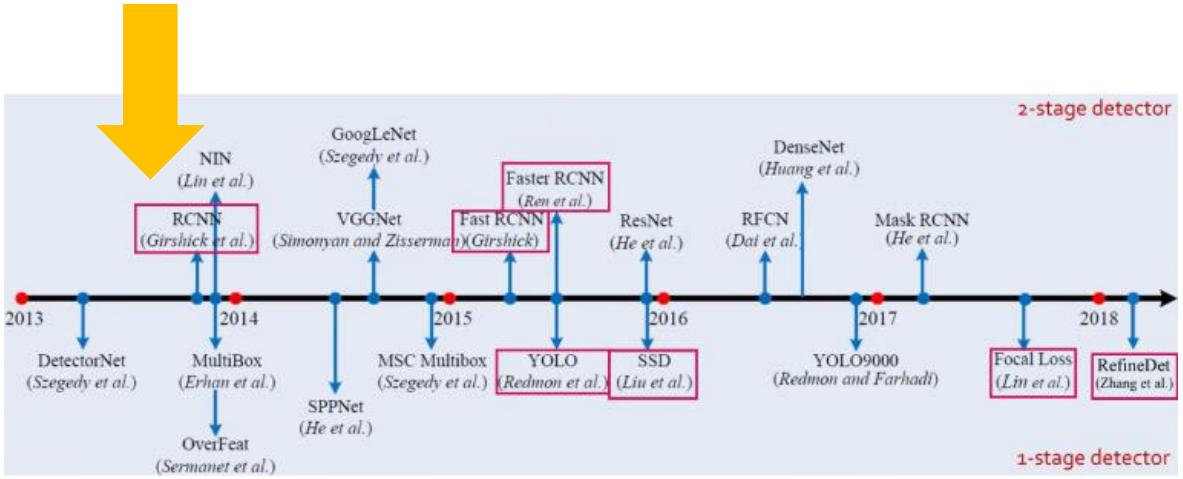
---

BOAZ 16기 박은지

2021.04.01

# 0. 들어가기에 앞서



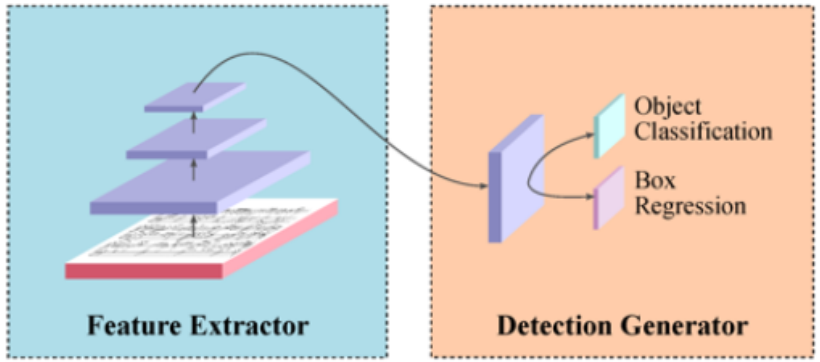

## Object Detection

Multiple objects에서 각각의 object에 대해 Classification + Localization을 수행하는 것

**Detection은** 크게2가지방식(One-Stage Method, Two-Stage Method)

# 0. 들어가기에 앞서

## • One-Stage Detectors

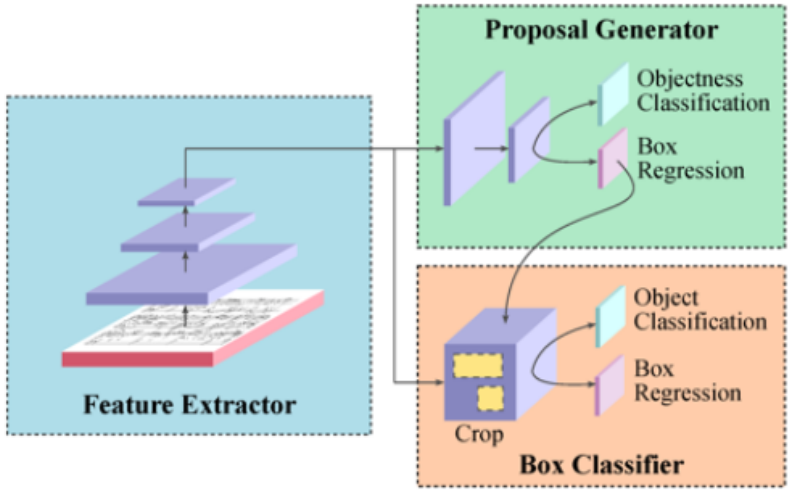



(a) Basic architecture of a one-stage detector.

빠름

**YOLO, SSD**

## • Two-Stage Detectors




(b) Basic architecture of a two-stage detector.

정확도가 좋음

**R-CNN,** Fast R-CNN, Faster R-CNN

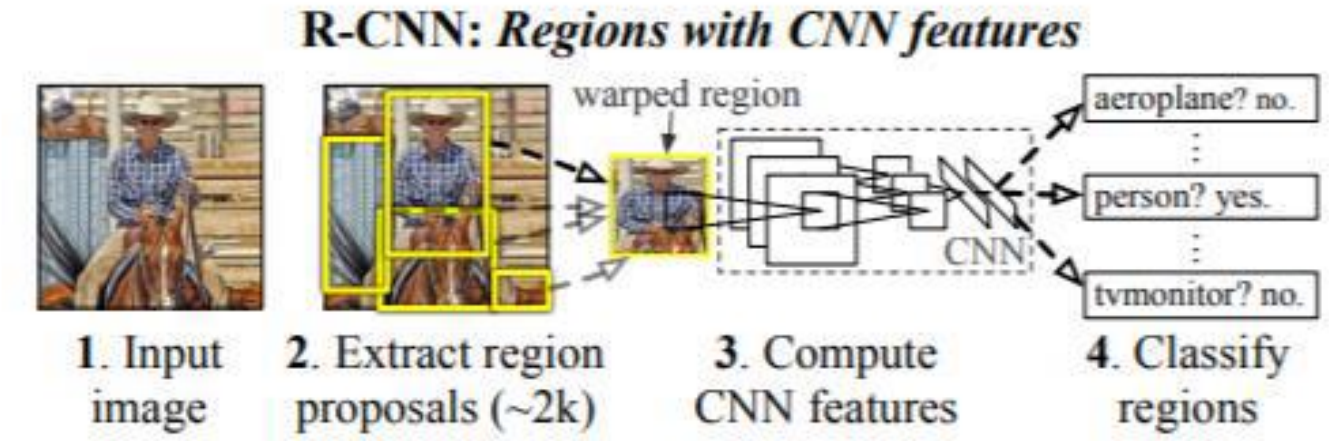
**RoI**(Region of Interest) : **object가 있을만한 영역**

# 1. Abstract

1. Object Detection 성능을 평가하는 고전적 Dataset인 PASCAL VOC 데이터,
   이 논문에서는 기존의 방식보다 mAP*이 **30% 향상**된 detection 알고리즘 제안
   * mean Average Precision(mAP) : Object Detection 성능 평가 지표

2. R-CNN은 2가지 예상되는 문제점 제시, 해결
1) **Localization** : 객체를 localize and segment 하기 위해  bottom-up 방식의 region proposal 에 CNN을 적용
2) **Scarce of labeled data** : supervised pre-trained CNN model과 fine-tuning으로 성능 향상

# 2. Introduction




1. Input 이미지로부터 2,000개의 독립적인 **region proposal**을 생성

2. CNN을 통해 각 proposal마다 고정된 길이의 **feature vector를 추출**
   CNN 적용 시 서로 다른 region shape에 영향을 받지 않기 위해 fixed-size로 **input 이미지를 변경 (warp)**

3. 각 region 마다 category-specific linear **SVM**을 적용하여 classification을 수행

Region proposals → Feature extraction → SVM

# 3. Object detection with R-CNN

Region proposals → Feature extraction → SVM → Training → Rusults

Sliding window방식 (비효율적)

Selective search 알고리즘

- Selective Search의 프로세스

1. 이미지의 초기 세그먼트를 정하고 수많은 region 후보 생성
2. Greedy 알고리즘을 이용해 각 region을 기준으로 주변의 유사 영역을 결합

   * 후보들 간의 color, texture, size, fill 바탕으로 유사도 계산, 유사도 높은 순서대로 결합
3. 결합되어 커진 region을 최종 region proposal로 제안

# 3. Object detection with R-CNN

Region proposals → **Feature extraction** → SVM → Training → Rusults

각각의 region로부터 Warp된 동일한 size의 input을 CNN에 통과시켜 **4096차원의 feature vector 추출**

- Features는 5개의 convolutional layer와 2개의 fully connected layer로 전파되는데
  이때 CNN의 Input으로 사용되기 위해 각 region은 227x227 RGB의 **고정된 사이즈로 변환이 필요함**
- 사이즈나 종횡비에 상관없이 **Warpping하여 Input 으로 사용했음**
  * 이때 16 pixel의 padding을 가한 후의 warping이 가장 성능이 좋았음

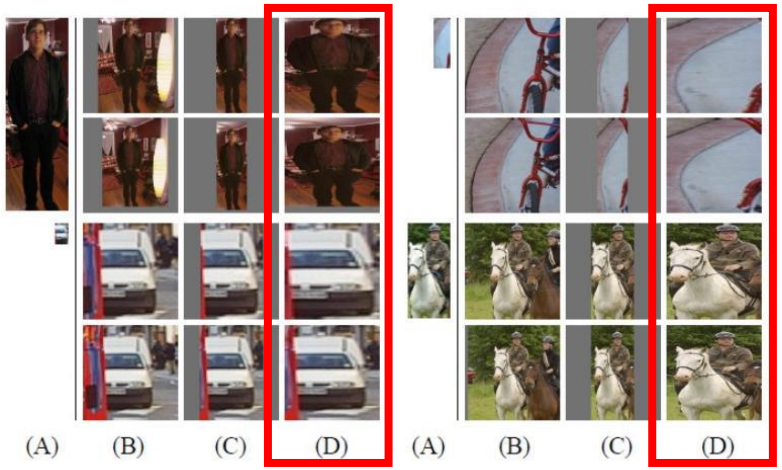



**Figure 7: Different object proposal transformations.** (A) the

# 3. Object detection with R-CNN

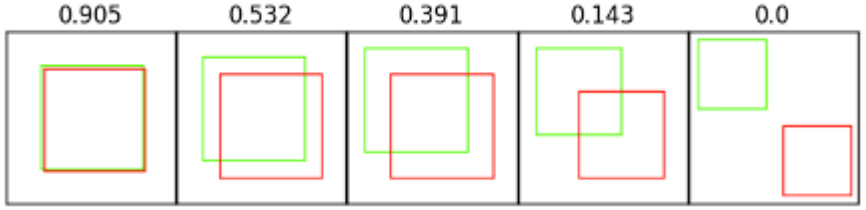
Region proposals → Feature extraction → SVM → Training → Rusults

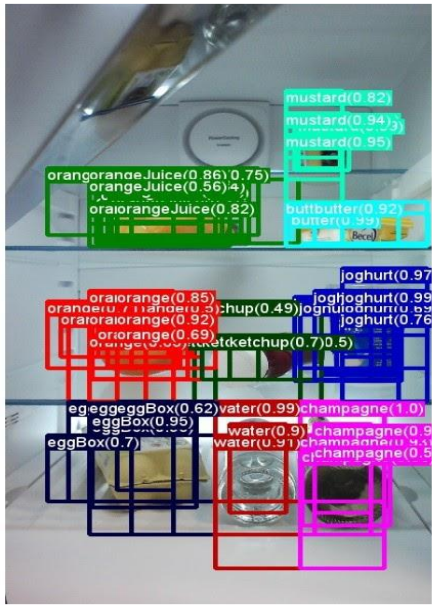
1. 각 class에 대해, 추출된 feature vector를 input 하여 **SVM으로 score 계산**
2. 이미지에 대해 모든 region이 점수가 매겨지면, **NMS *** 를 이용하여 class score가 높은 region과 IoU가 threshold(0.5)보다 큰 region들을 제거

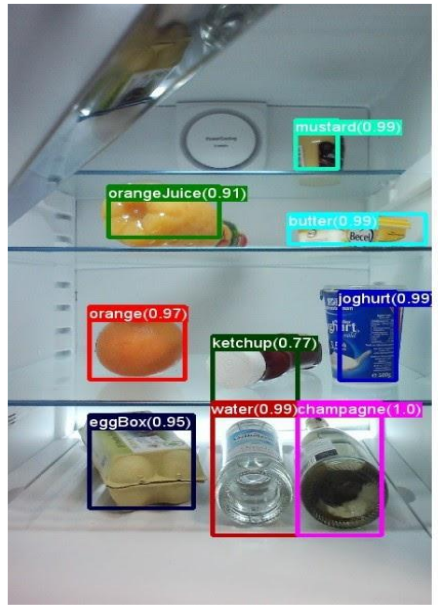
*** NMS, Non-Maximum Suppression**

1) 예측한 bounding box들의 예측 점수를 내림차순으로 정렬
2) 높은 점수의 박스부터 시작하여 나머지 박스들 간의 IoU를 계산
3) **IoU값**이 지정한 threshold(논문에서는 0.5)보다 높은 박스를 제거
4) 최적의 박스만 남을 때 까지 위 과정을 반복

*** IoU** : Area of Overlap(교집합) / Area of Union(합집합)



# 3. Object detection with R-CNN

Region proposals → Feature extraction → SVM → **Training** → Rusults

CNN 모델은 ILSVRC 2012 데이터 셋으로 미리 학습된 pre-trained CNN(AlexNet)을 사용

**Supervised pre-training.**
큰 보조데이터 셋인 ILSVRC2012 분류 데이터셋의 image-level annotation을 활용하여 pre-training

**Domain-specific fine-tuning.**
- 새로운 task(detection)와 새로운 도메인(wraped proposal windows)에 CNN을 적용하기 위해 CNN parameter를 SGD로 학습시킴
- AlexNet 마지막 1000 classification을 N+1(N개의 클래스, 1개의 background) classification으로만 수정

- positive sample : IoU가 0.5이상
- negative sample : 나머지 (background)
- SGD는 learning rate를 0.001에서 시작, 각 iteration마다 mini-batch 128 구성 (32개의 positive sample를, 96개의 background(negative sample)사용)

# 3. Object detection with R-CNN


**Object category classifier(SVM)**

- SVM을 따로 학습한 이유는 당시에 mAP가 softmax (50.9%)한 것이 SVM(54.2%)보다 낮아서

- pos/neg를 정하는 것은 mAP에 직결되며 논문에선 grid search를 통해 threshold를 정함.
- positive sample : ground-truth boxes로 정의 (정답 data / IoU 1)
- negative sample : threshold 0.3 미만인 것.

# 3. Object detection with R-CNN

Region proposals → Feature extraction → SVM → Training → Rusults

| VOC 2010 test | aero | bike | bird | boat | bottle | bus | car | cat | chair | cow | table | dog | horse | mbike | person | plant | sheep | sofa | train | tv | mAP |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DPM v5 [20]† | 49.2 | 53.8 | 13.1 | 15.3 | 35.5 | 53.4 | 49.7 | 27.0 | 17.2 | 28.8 | 14.7 | 17.8 | 46.4 | 51.2 | 47.7 | 10.8 | 34.2 | 20.7 | 43.8 | 38.3 | 33.4 |
| UVA [39] | 56.2 | 42.4 | 15.3 | 12.6 | 21.8 | 49.3 | 36.8 | 46.1 | 12.9 | 32.1 | 30.0 | 36.5 | 43.5 | 52.9 | 32.9 | 15.3 | 41.1 | 31.8 | 47.0 | 44.8 | 35.1 |
| Regionlets [41] | 65.0 | 48.9 | 25.9 | 24.6 | 24.5 | 56.1 | 54.5 | 51.2 | 17.0 | 28.9 | 30.2 | 35.8 | 40.2 | 55.7 | 43.5 | 14.3 | 43.9 | 32.6 | 54.0 | 45.9 | 39.7 |
| SegDPM [18]† | 61.4 | 53.4 | 25.6 | 25.2 | 35.5 | 51.7 | 50.6 | 50.8 | 19.3 | 33.8 | 26.8 | 40.4 | 48.3 | 54.4 | 47.1 | 14.8 | 38.7 | 35.0 | 52.8 | 43.1 | 40.4 |
| R-CNN | 67.1 | 64.1 | 46.7 | 32.0 | 30.5 | 56.4 | 57.2 | 65.9 | 27.0 | 47.3 | 40.9 | 66.6 | 57.8 | 65.9 | 53.6 | 26.7 | 56.5 | 38.1 | 52.8 | 50.2 | 50.2 |
| R-CNN BB | **71.8** | **65.8** | **53.0** | **36.8** | **35.9** | **59.7** | **60.0** | **69.9** | **27.9** | **50.6** | **41.4** | **70.0** | **62.0** | **69.0** | **58.1** | **29.5** | **59.4** | **39.3** | **61.2** | **52.4** | **53.7** |

**Table 1: Detection average precision (%) on VOC 2010 test.** R-CNN is most directly comparable to UVA and Regionlets since all methods use selective search region proposals. Bounding-box regression (BB) is described in Section C. At publication time, SegDPM was the top-performer on the PASCAL VOC leaderboard. †DPM and SegDPM use context rescoring not used by the other methods.

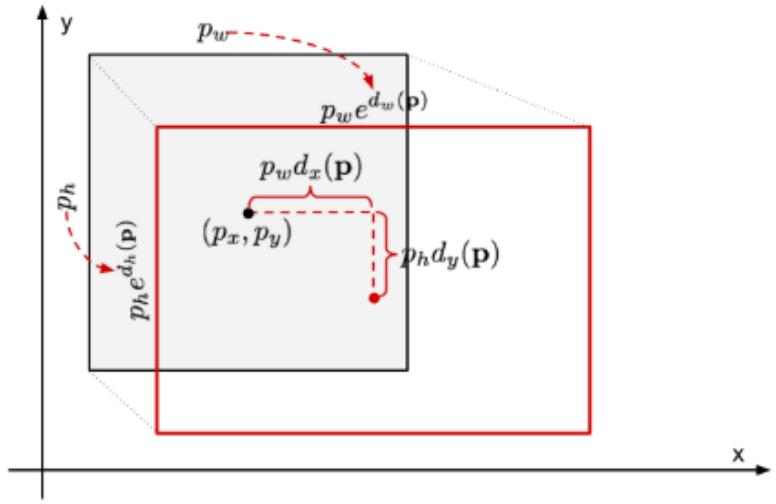
R-CNN이 좋은 성능을 냄. BB(Boundin-box regression)를 이용한 R-CNN이 더욱 좋은 성능을 냄.
→ CNN, **BB의 효과**를 볼 수 있음

# 4. Appendix(C)

## Bounding Box Regression

Selective search로 만든 bounding box는 정확하지 않음
물체를 좀 더 **정확히 감싸도록 조정**해주는 bounding box regression(선형회귀 모델)이 도움이 됨

$$P^i = (P^i_x, P^i_y, P^i_w, P^i_h)$$
$$G = (G_x, G_y, G_w, G_h).$$




- P를 이동시키는 함수의 식

$$\hat{G}_x = P_w d_x(P) + P_x \quad (1)$$
$$\hat{G}_y = P_h d_y(P) + P_y \quad (2)$$
$$\hat{G}_w = P_w \exp(d_w(P)) \quad (3)$$
$$\hat{G}_h = P_h \exp(d_h(P)). \quad (4)$$

- P를 G로 이동시키기 위해서 필요한 이동량

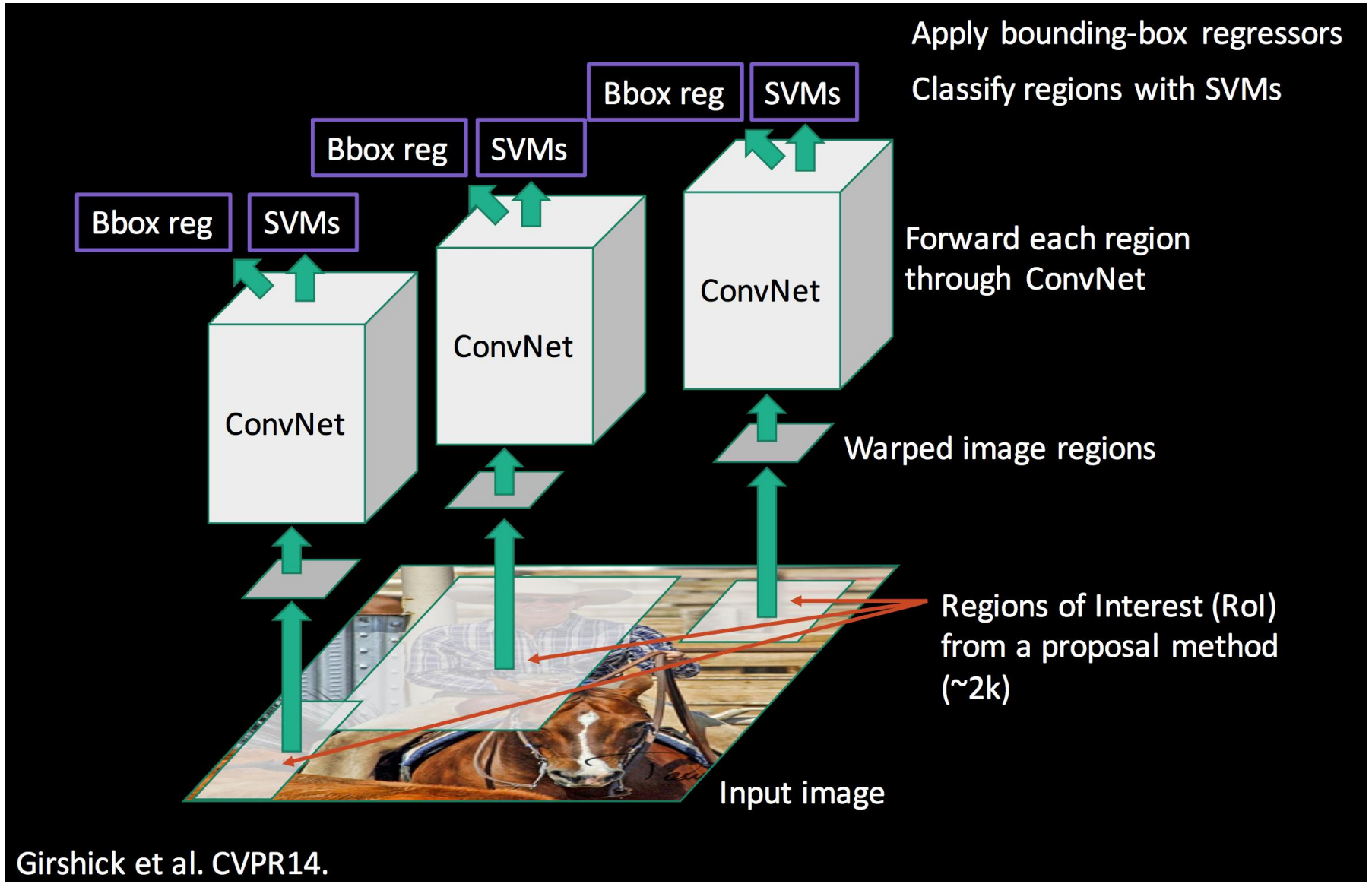
$$t_x = (G_x - P_x)/P_w \quad (6)$$
$$t_y = (G_y - P_y)/P_h \quad (7)$$
$$t_w = \log(G_w/P_w) \quad (8)$$
$$t_h = \log(G_h/P_h). \quad (9)$$

- 웨이트를 학습시킬 Loss Function ( MSE 에러 함수에 L2 norm을 추가한 형태)

$$\mathbf{w}_\star = \underset{\hat{\mathbf{w}}_\star}{\operatorname{argmin}} \sum_i^N (t^i_\star - \hat{\mathbf{w}}^{\mathrm{T}}_\star \boldsymbol{\phi}_5(P^i))^2 + \lambda \|\hat{\mathbf{w}}_\star\|^2. \quad (5)$$

$$d_\star(P) = \mathbf{w}^{\mathrm{T}}_\star \boldsymbol{\phi}_5(P)$$

# 5. Conclusion : 처음부터 다시 정리하면 !



# 5. Conclusion

PASCAL VOC2012에서 가장 좋은 결과를 냈던 과거 연구보다

**30% 향상된 성능을 보인 간단하면서도 가변적인 object detection 알고리즘**

이것은 두가지 인사이트를 통해서 가능했으며 두가지는 아래와 같다.

1. The first is to apply high-capacity convolutional neural networks to bottom-up region proposals in order to localize and segment objects. **(CNN의 적용)**
2. The second is a paradigm for training large CNNs when labeled training data is scarce.
   **(supervised pretraining, domain specific fine tuning을 적용)**

# 6. 후속연구

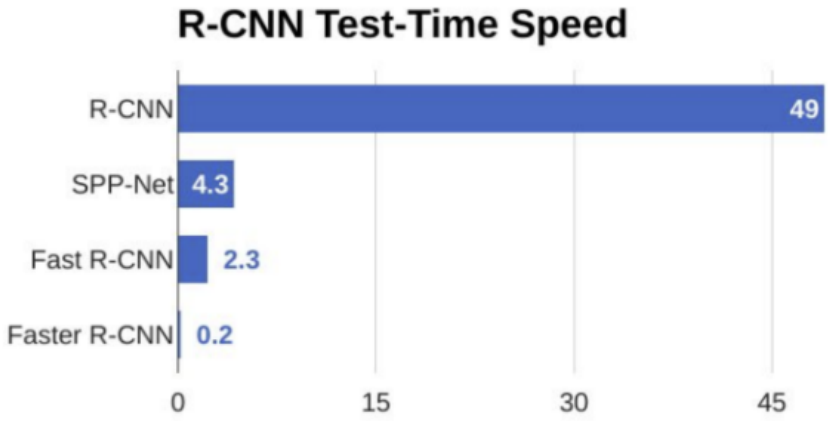
딥러닝을 이용한 Object Detection의 포문을 연 R-CNN !
초기 모델이라서 전통적인 비전 알고리즘들도 함께 사용해 구조가 복잡함

*과도한 연산량과 시간…….*

1) selective search 로 2000개의 region을 뽑고, 각 영역마다 CNN연산을 수행하므로 **시간이 매우느림**

2) CNN, SVM, Bounding Box Regression 총 세가지의 모델이 multi-stage pipelines으로 한 번에 학습되지 않음

-〉 이러한 문제를 해결한 Fast R-CNN 등이  나오게 됨 !  * 다음주에 배우게 됩니다 ☺



# 감사합니다 !

궁금한 점이 있으신가요 ?