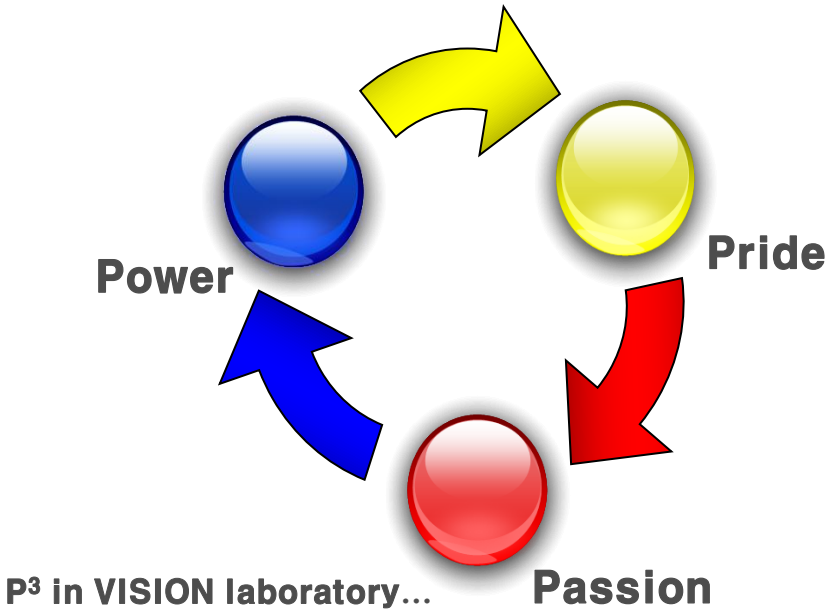


Fast R-CNN

Vision System Lab, Gyumin Park
yywnnaa@gmail.com
Jan 19, 2024

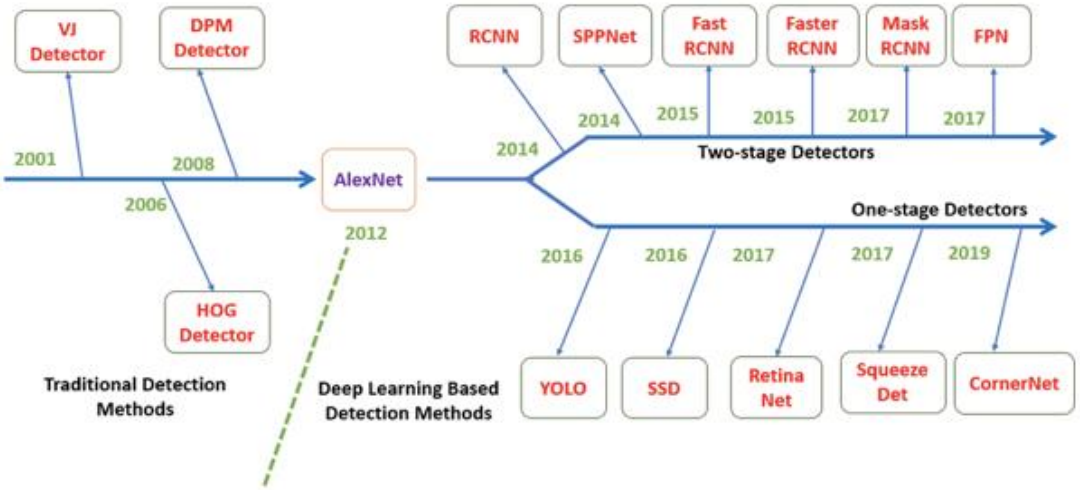
Soongsil University
VISION SYSTEM LABOARTORY

Object Detection

Object Detection이란 한 물체(single object)가 아닌 여러 물체(Multiple objects)에 대해 어떤 물체인지 클래스를 분류하는 **Classification** 문제와, 그 물체가 어디 있는지 박스를 (Bounding box) 통해 위치 정보를 나타내는 **Localization** 문제를 모두 포함

즉 Object Detection = Multiple Object에 대한 Multi-Labeled Classification + Bounding Box Regression(Localization) 라고 정리할 수 있다.

위쪽이 **2-stage Detector** 논문
물체를 식별하는 **Classification** 문제와, 물체의 위치를 찾는 **Localization** 문제
두 가지 task를 동시에 행하는 방법



R-CNN부터 Fast R-CNN,
Faster R-CNN같은 R-CNN 계열이 대표적

아래쪽은 **1-stage Detector** 논문
두 문제를 순차적으로 행하는 방법

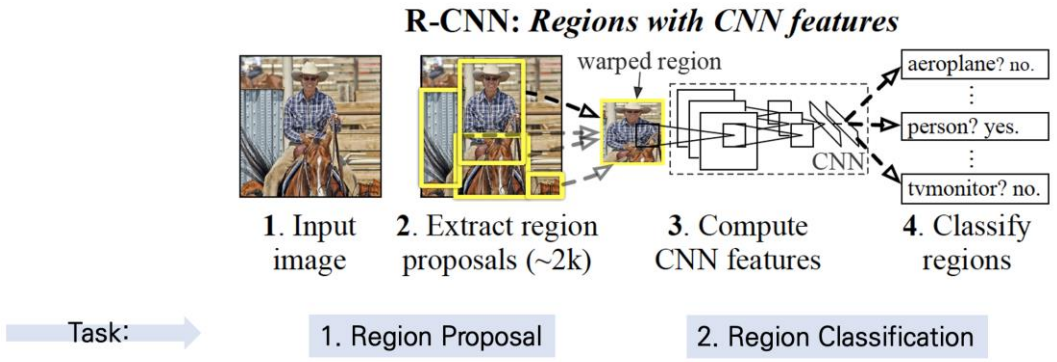
YOLO(You Only Look Once)계열과
SSD 계열 등이 포함

1-stage Detector : 비교적 빠르지만 정확도 낮음
2-stage Detector : 비교적 느리지만 정확도 높음

VISION SYSTEM LABORATORY

R-CNN

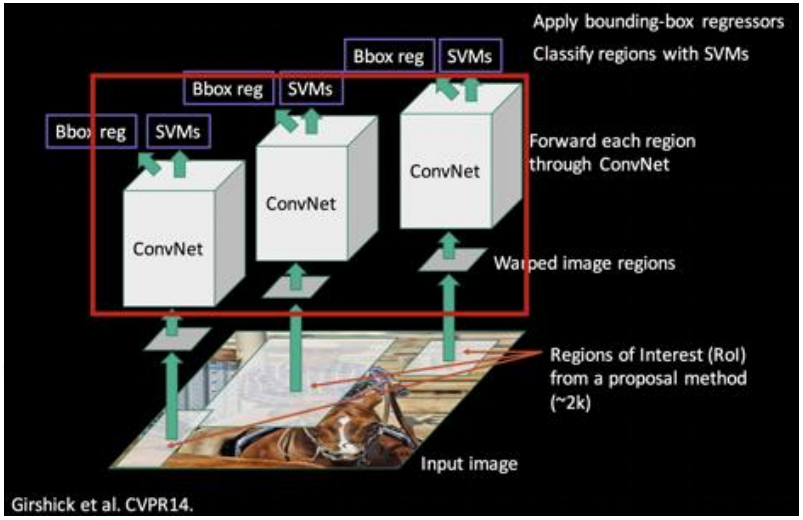
- R-CNN은 **'Regions with Convolutional Neural Networks features'**의 약자
- **설정한 Region**을 **CNN의 feature(입력값)로** 활용하여 **Object Detection**을 수행하는 **신경망**



CNN을 어떻게 활용할것인지 연구되는 상황

문제들을 어떻게 해결하였는지 집중

기본적인 구조는 2-stage Detector - 전체 task를 두 단계로 나눌 수 있음 :
물체의 위치를 찾는 Region Proposal, 물체를 분류하는 Region Classification
이 두 가지 task를 처리하기 위해 수행되는 R-CNN의 구조 :



1. 이미지에 있는 데이터와 레이블을 투입한 후 카테고리에 무관하게 물체의 영역을 찾는 Region Proposal
2. proposal 된 영역으로부터 고정된 크기의 Feature Vector를 warping/crop하여 CNN의 input으로 사용. 여기서 CNN은 ImageNet을 활용한 사전훈련된 네트워크를 사용 (CNN은 여기까지만 활용)
3. CNN을 통해 나온 feature map을 활용하여 선형 지도학습 모델인 SVM(Support Vector Machine)을 통한 분류 (이 당시 SVM이 더 성능이 좋았음)
4. Regressor를 통한 bounding box regression을 진행

mAP : Mean Average Precision

- 이미지 검색이나 객체 탐지 같은 문제에서 모델의 성능을 평가하는 데 널리 사용되는 지표

- mAP를 이해하려면 먼저 'Precision'과 'Recall'에 대해 알아야 함.
- 'Precision'은 모델이 True라고 분류한 것 중 실제로 True인 것의 비율
- 'Recall'은 실제 True인 것 중에서 모델이 True라고 예측한 것의 비율
- 객체 탐지에서는 여러 개의 예측 박스를 생성하게 되므로, 이 박스들 각각에 대한 Precision 값을 계산하고, 이를 평균내어 ‘Average Precision (AP)’을 구함

- 그런 다음, 데이터 세트에서 여러 카테고리에 대한 AP를 다시 평균내어 'Mean Average Precision (mAP) ' 를 구함.
- 즉, mAP는 모든 카테고리에 대한 AP의 평균값을 나타내는 지표

- 따라서, mAP가 높을수록 모델의 성능이 좋다고 판단할 수 있음

VISION SYSTEM LABORATORY

Abstract

- 물체 감지를 위한 빠른 지역 기반 합성곱 신경망(convolutional networks) 방법(Fast R-CNN)을 제안
- 이전 연구를 기반으로 깊은 합성곱 신경망을 사용하여 object proposals 을 효율적으로 분류하는 방식
- object proposals : 이미지에서 물체가 있을 것으로 예상되는 위치를 제안하는 후보 영역

- 이전 연구에 비해, Fast R-CNN은 훈련 및 테스트 속도를 향상시키는 몇 가지 혁신을 도입, detection accuracy또한 상승
- Fast R-CNN은 R-CNN보다 VGG16 네트워크를 9배 빠르게 훈련, 테스트 시간 213배 더 빠름
- PASCAL VOC 2012에서 높은 mAP를 달성 / PASCAL : 이미지 인식 및 분류를 위한 데이터 세트
- mAP(Mean Average Precision) : 물체 감지 모델의 성능을 측정하는 지표, 여러 클래스에 대한 평균 정밀도(average precision)의 평균값 / 정밀도 : 모델이 예측한 결과 중에서 실제로 정답인 비율
- SPPnet 보다 VGG16을 3배 빠르게 훈련하고, 10배 빠르게 테스트하며 더 높은 정확도

- Python 및 C++(Caffe를 사용)로 구현
- available under the open-source MIT License at https: //github.com/rbgirshick/fast-rcnn.

1. Introduction

- 이미지 분류와 비교하면, 물체 감지는 더 복잡한 방법이 필요한 더 어려운 작업
- 물체의 정확한 위치 지정이 필요하기 때문에 더 복잡
- 두 가지 주요 도전 과제 : object locations (often called "proposals") 를 처리

  이러한 후보는 정확한 위치 지정을 위해 세밀하게 보정되어야 하는 대략적인 위치만을 제공

  이러한 문제에 대한 solution들은 속도, 정확도, 단순함(simplicity) 중 하나를 타협(compromise)

- State-of-the-art(최신) ConvNet-based object detectors : 현재 가장 선두에 있는 Convolutional Neural Network(ConvNet) 기반 물체 탐지기 훈련 프로세스를 간소화(streamline)
- object proposals 을 분류하고 공간적 위치를 세밀하게 보정하는 것을 동시에 학습하는 단계별 훈련 알고리즘을 제안
- 결과적으로 이 방법은 R-CNN 대비 VGG16 탐지 네트워크를 9배, SPPnet 대비 3배 더 빠르게 훈련시킬 수 있음
- 실행 시에는 PASCAL VOC 2012에서 66%의 mAP(R-CNN 대비 62%)를 달성하면서 이미지를 0.3초 내에 처리 (객체 proposal 시간 제외)

1.1. R-CNN and SPPnet

R-CNN의 단점

1. Training이 multi-stage pipeline 이다 → 한 번에 학습이 안된다

log loss를 이용해 ConvNet을 fine tuning 하고 / feature vector를 얻은 뒤 SVM에 적용시킴

SVM들은 fine tuning으로 학습된 softmax 분류기를 대체하여 객체 탐지기로 작동

마지막 세 번째 training stage로 bounding box regressor가 학습이 됨

2. Training이 시공간적 비용이 비싸다

SVM과 bounding-box regressor 학습 시에 이미지마다 각각의 object proposal에 대해 feature를 추출하고 디스크에 기록해야 해서 시간과 저장 공간이 많이 든다 (feature를 저장해두고 다시 써야함)

3. Object detection is slow (test time)

Test time에서 각각의 object proposal에 대해 feature를 추출하므로 오래 걸린다

각각의 object proposal에 대한 forward pass 시에 연산을 공유하지 않아서 느림, 이를 개선한 것이 SPPnet

1.1. R-CNN and SPPnet(Spatial Pyramid Pooling network)

R-CNN에서 발전된 점:

1. 고정된 크기의 입력:

R-CNN은 입력 이미지에서 후보 영역(proposal region)을 추출하고, 각각을 신경망에 입력하기 전에 동일한 크기로 변환해야 했음

-> 계산적으로 비효율적, 시간이 많이 걸림, 이미지 정보 손실, 이미지 변형이 일어남

반면, SPP-Net은 Spatial Pyramid Pooling 레이어를 도입해 다양한 크기의 후보 영역을 처리할 수 있게 됨

이 덕분에 이미지를 한 번만 네트워크에 통과시키면 되어 처리 속도가 훨씬 빨라짐

2. 특징 공유:

SPP-Net은 이미지를 한 번만 Convolutional Network를 통과시켜 특징 맵(feature map)을 계산하고,

이를 이용해 각 후보 영역의 특징을 추출

이는 후보 영역마다 별도로 특징을 계산하는 R-CNN보다 훨씬 효율적

1.1. R-CNN and SPPnet(Spatial Pyramid Pooling network)

SPP-Net의 한계 :

1. 훈련의 복잡성 : R-CNN처럼 multi-stage pipeline을 사용한다는 단점 (후보 영역을 추출, fine-tuning, SVM 분류기 훈련 등 여러 단계를 거쳐야 함)
2. fine-tuning 알고리즘은 spatial pyramid pooling 이전의 convolution layer들에 대한 값을 update 할 수 없어 deep 해지면 좋은 결과를 낼 수 없다 (한계가 존재)

SPP-Net의 구조에서는 Spatial Pyramid Pooling(SPP) 레이어가 Convolution Layer들 이후에 위치,

SPP 레이어는 다양한 크기의 feature map을 고정된 길이의 벡터로 변환하는 역할

이러한 구조로 인해 SPP-Net을 전체적으로 end-to-end로 학습시키는 데 문제가 있음

Fine-tuning : 사전에 훈련된 모델을 새로운 데이터셋에 맞게 추가적으로 조정하는 과정, 이 과정에서 네트워크의 모든 파라미터가 업데이트될 수 있어야 하지만 **초기 SPP-Net의 구현에서는 SPP 레이어 이전에 있는 Convolution Layer들의 가중치를 업데이트하는 것이 제한적**

즉, SPP 레이어가 고정된 feature map을 생성하고, 이 feature map을 바탕으로 분류기가 훈련되는 구조

따라서, 네트워크가 깊어질수록, 즉 Convolution Layer들이 많아질수록, SPP-Net에서는 이 초기 레이어들의 가중치를 충분히 업데이트하지 못하는 문제가 발생 -> 네트워크의 성능에 제한을 두는 결과

네트워크가 깊어지면 깊어질수록, 초기 레이어들의 가중치를 적절히 조정하는 것이 중요해지는데, SPP-Net의 경우 이러한 조정이 충분히 이루어지지 않았다

1.2. Contributions

- R-CNN과 SPPnet의 단점을 해결하면서 속도와 정확도를 개선한 새로운 훈련 알고리즘을 제안
- 이 방법을 Fast R-CNN이라고 부르는데, 훈련과 테스트가 비교적 빠르기 때문

1. R-CNN, SPPnet보다 높은 mAP
2. 학습은 multi-task loss를 이용하여 single-stage로 이루어짐
3. 학습 시에 모든 network layer에 대한 update가 가능
4. feature caching을 위한 디스크 저장 공간이 필요하지 않음

2. Fast R-CNN architecture and training

Fast R-CNN

- input으로 이미지 전체와 object proposal set이 들어옴
- 여러 합성곱(conv) 및 max pooling 레이어를 사용 / 전체 이미지를 처리 / conv feature map 추출
- 각각의 object proposal마다 / ROI pooling layer가 / feature map으로부터 / 고정 길이 feature vector를 추출
- 각각 feature vector는 Fully-connected layers를 통과해 두 브랜치로 나뉜다.

  K개의 object class + background → K+1에 대한 softmax 확률이 나오는 곳

  K개의 object class 각각에 대한 4개의 실수값real-valued이 나오는 곳

  → bounding-box 위치와 regression을 위해 사용

  각각의 4개의 값 집합은 K개 클래스 중 하나에 대한 정제된(refined) bounding-box 위치를 인코딩

  refined bounding-box positions: 초기에 제안된 bounding-box의 위치를 더 정교하게 조정한 결과를 의미

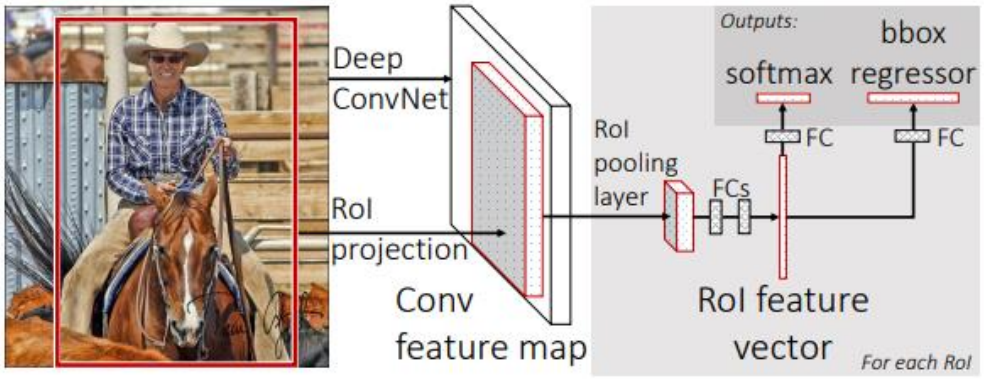
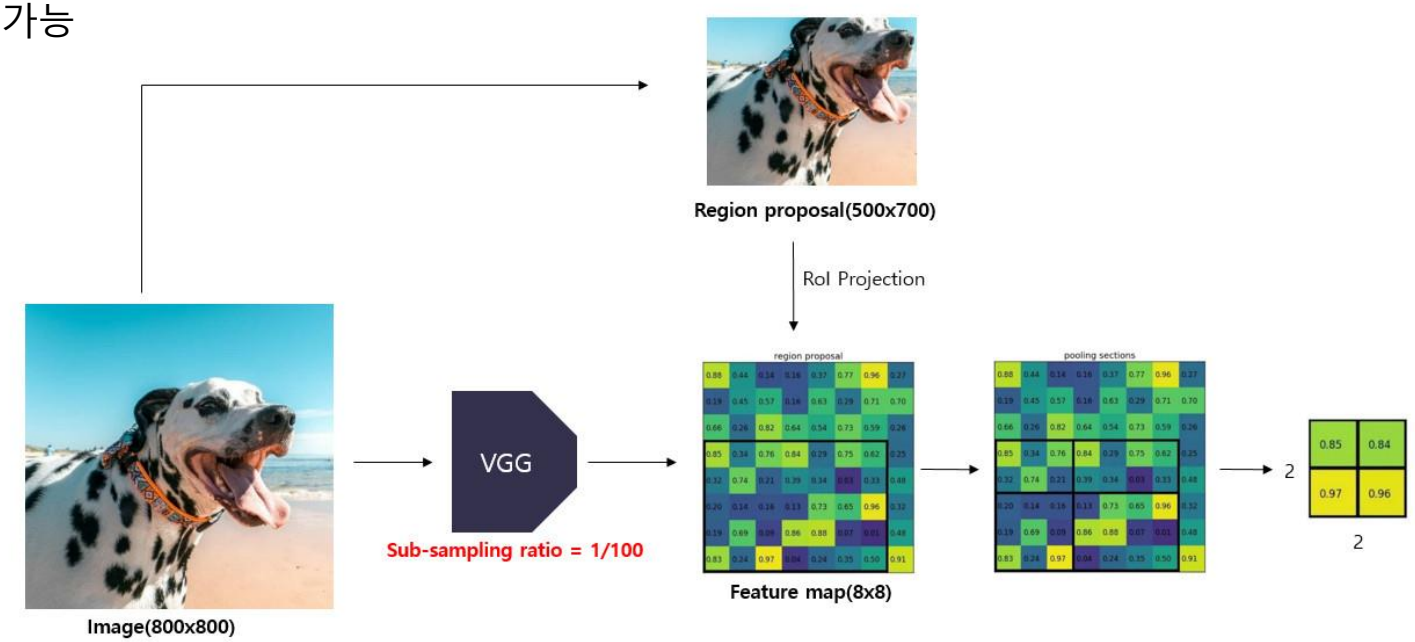


그림 1. Fast R-CNN 아키텍처.
입력 이미지와 여러 관심 영역(Regions of Interest, RoIs)이 완전 합성곱 네트워크에 입력됨. 각 RoI는 고정 크기의 피처 맵으로 풀링되고, 그런 다음 Fully Connected Layers(FCs)에 의해 피처 벡터로 매핑됨. 이 네트워크는 각 RoI 당 두 개의 출력 벡터를 가지고 있음: softmax확률과 클래스별 bounding-box regression (offsets). 이 아키텍처는 멀티태스크 손실과 함께 end-to-end로 훈련됨

2.1. The RoI pooling layer

- **RoI(Region of Interest) pooling**은 feature map에서 region proposals에 해당하는 **관심 영역(Region of Interest)**을 지정한 크기의 grid로 나눈 후 max pooling을 수행하는 방법
- 각 channel별로 독립적으로 수행하며, 이 같은 방법을 통해 **고정된 크기의 feature map을 출력**하는 것이 가능



1) 먼저 원본 이미지를 CNN 모델에 통과시켜 feature map을 얻음

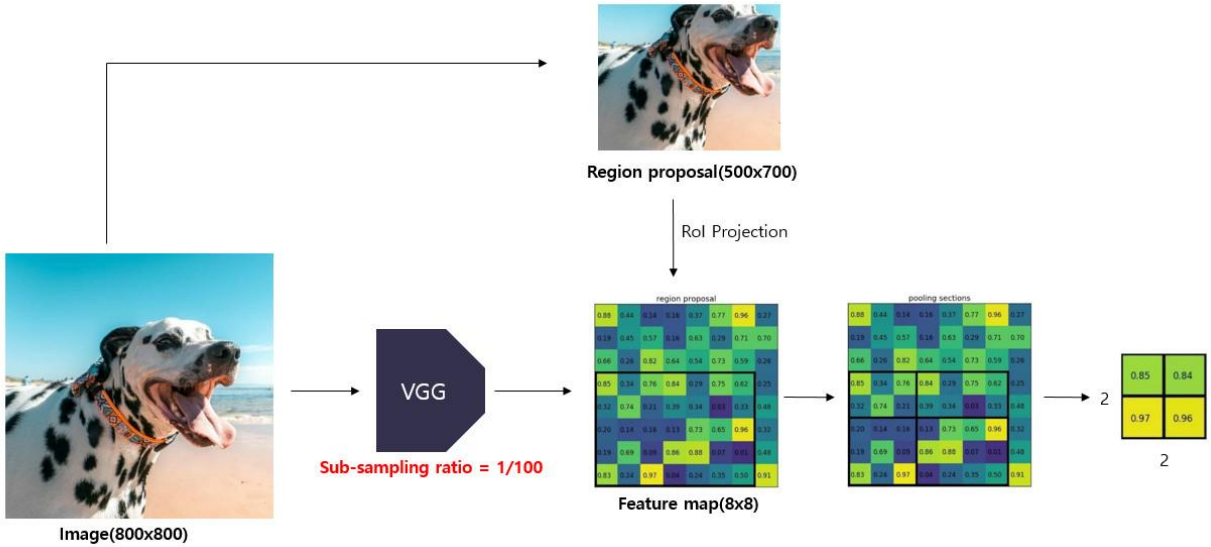
- 800x800 크기의 이미지를 VGG 모델에 입력하여 8x8 크기의 feature map을 얻음
- 이 때 sub-sampling ratio = 1/100이라고 할 수 있음(여기서 말하는 subsampling은 pooling을 거치는 과정을 의미)

2) 그리고 동시에 원본 이미지에 대하여 Selective search 알고리즘을 적용하여 region proposals를 얻음

- 원본 이미지에 Selective search 알고리즘을 적용하여 **500x700** 크기의 region proposal을 얻음

VISION SYSTEM LABORATORY

2.1. The RoI pooling layer



3) feature map에서 각 region proposals에 해당하는 영역을 추출 - **RoI Projection**을 통해 가능
Selective search를 통해 얻은 region proposals는 sub-sampling 과정을 거치지 않은 반면, 원본 이미지의 feature map은 sub-sampling 과정을 여러 번 거쳐 크기가 작아짐
작아진 feature map에서 region proposals이 encode(표현)하고 있는 부분을 찾기 위해 작아진 feature map에 맞게 region proposals를 투영해주는 과정이 필요,
이는 region proposal의 크기와 중심 좌표를 **sub sampling ratio**에 맞게 변경시켜줌으로써 가능
- Region proposal의 중심점 좌표, width, height와 sub-sampling ratio를 활용하여 feature map으로 투영시킴
- feature map에서 region proposal에 해당하는 **5x7** 영역을 추출

4) 추출한 RoI feature map을 지정한 **sub-window의 크기**에 맞게 grid로 나눔
- 추출한 5x7 크기의 영역을 지정한 **2x2** 크기에 맞게 grid를 나눔

5) grid의 각 셀에 대하여 max pooling을 수행하여 고정된 크기의 feature map을 얻음
- 각 grid 셀마다 max pooling을 수행하여 **2x2** 크기의 feature map을 얻음

이처럼 미리 지정한 크기의 sub-window에서 max pooling을 수행하다보니
region proposal의 크기가 서로 달라도 고정된 크기의 feature map을 얻을 수 있음

VISION SYSTEM LABORATORY

2.2. Initializing from pre-trained networks

세 개의 사전 훈련된 ImageNet(이미지 분류 데이터셋) 네트워크를 실험(experiment)

각각 5개의 max pooling 레이어와 5~13개의 conv layer를 가지는 3개의 pre-trained 모델을

사전 훈련된 네트워크가 Fast R-CNN 네트워크를 초기화할 때 3가지 변환(transformations)을 거침

1 : 마지막 max pooling 레이어를 RoI 풀링 레이어로 변경

첫 FC layer와 호환(compatible)되도록 H와 W가 설정됨 (예: VGG16의 경우 H = W = 7로 설정)

2 : 마지막 FC layer와 softmax를 앞에서 설명한 두 개의 sibling 레이어로 변경

(K + 1 FC layer와 softmax, 카테고리별(category-specific) bounding-box regressors)

3 : (이미지 목록, 해당 이미지에서의 RoI 목록)을 input으로 사용

VISION SYSTEM LABORATORY

2.3. Fine-tuning for detection

Hierarchical계층적 **Sampling**

- R-CNN 모델은 학습 시 region proposal이 서로 다른 이미지에서 추출되고, 이로 인해 학습 시 연산을 공유할 수 없다는 단점
- 연산을 공유 share computation -> 무슨 의미?
- Fast R-CNN의 장점 : 모든 network에 대하여 backprop을 통해 update가 가능 -> end-to-end 학습 가능
- Fast R-CNN의 RoI가 너무 커서 기존 SPPnet, RNN에 적용된 방법을 사용하는 것은 비효율적
- 논문의 저자는 학습 시 **feature sharing**을 가능하게 하는 **Hierarchical sampling** 방법을 제시

  : SGD mini-batch를 구성할 때 N개의 이미지를 sampling하고, 총 R개의 region proposal을 사용한다고 할 때, 각각의 이미지에서 R/N개의 region proposals를 sampling하는 방법
- 같은 이미지에서 추출된 region proposals끼리는 forward, backward propogation 시, **연산과 메모리를 공유**
- 논문에서는 학습 시, N=2, R=128로 설정하여, 서로 다른 2장의 이미지에서 각각 64개의 region proposals를 sampling하여 mini-batch를 구성
- 각 이미지의 region proposals 중 25%(=16장)는 ground truth와의 IoU 값이 0.5 이상인 sample을 추출하고, 나머지(75%, 48장)에 대해서는 IoU 값이 0.1~0.5 사이의 sample을 추출
- 전자의 경우 positive sample로, 위에서 정의한 multi-task loss의 u=1이며, 후자는 u=0인 경우임

Multi-task loss

- Fast R-CNN 모델에서는 / feature vector를 / **multi-task loss**를 사용하여 / Classifier와 Bounding box regressior을 동시에 학습시킴 **Bounding box와 분류가 상관관계가 컸기 때문에 가능**
- 각각의 RoI(=region proposal)에 대하여 multi task loss를 사용하여 학습
- 이처럼 두 모델을 한번에 학습시키기 때문에, R-CNN 모델과 같이 각 모델을 독립적으로 학습시켜야 하는 번거로움이 없다는 장점
- multi-task loss : $L(p, u, t^u, v) = L_{cls}(p, u) + \lambda[u \geq 1]L_{loc}(t^u, v)$

$p = (p0, \ldots\ldots, p_k)$: (K+1)개의 class score

u : ground truth class score - u는 positive sample인 경우 1 / negative sample인 경우 0으로 설정되는 index parameter background는 u=0으로 라벨링 되어 loss 계산에서 무시

$t^u = (t^u_x, t^u_y, t^u_w, t^u_h)$: 예측한 bounding box 좌표를 조정하는 값

$v = (v_x, v_y, v_w, v_h)$: 실제 bounding box의 좌표값

λ : 두 loss 사이의 가중치를 조정하는 balancing hyperparamter 모든 실험은 λ=1을 사용

$L_{cls}(p, u) = -logp_u$: 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**)

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

예측 값과 라벨 값의 차가 1보다 작으면 0.5x^2로 L2 distance를 계산. 반면 1보다 클 경우 L1 distance를 계산. 이는 Object Detection task에 맞추어 loss function을 커스텀하는것으로 볼 수 있음, 저자들이 실험 과정에서 라벨 값과 지나치게 차이가 많이 나는 outlier 예측 값들이 발생했고, 이들을 그대로 L2 distance로 계산하여 적용할 경우 gradient가 explode 해버리는 현상을 관찰했음, 이를 방지하기 위해서 추가됨

- K개의 class를 분류한다고할 때, 배경을 포함한 (K+1)개의 class에 대하여 Classifier를 학습시켜줘야 함
- **L1 loss**는 / R-CNN, SPPnets에서 사용한 L2 loss에 비해 / **outlier에 덜 민감**하다는 장점
- multi task loss는 0.8~1.1% mAP를 상승시키는 효과

왜? -> 정리된 논문 2017년정도에 나옴 : **규제의 효과**

따로 할 경우 한가지 task에 대해서 과적합 발생 가능

Mini-batch sampling

- Fine-tuning 과정에서의 각 SGD(확률적 경사 하강법) 미니 배치는 임의로 선택된 N = 2개의 이미지로 구성됨.
- 통상적인 방법으로 데이터셋의 순서를 바꾸면서 반복
- 미니 배치의 크기는 R = 128로, 각 이미지에서 64개의 RoI(Region of Interest)를 샘플링
- R-CNN 논문에서 제시된 것처럼, 적어도 0.5 이상의 교차 합집합(IoU, Intersection over Union) 오버랩을 가진 실제 객체의 바운딩 박스와 겹치는 object proposal에서 25%의 RoI를 가져옴
- -> **IoU가 50% 이상의 RoI 25%를 사용, 50% 이하는 background로 취급**
- RoI들은 foreground object class로 라벨이 붙은 예시들, 즉 $u \geq 1$
- foreground는 바운딩 박스와 높은 IoU(Intersection over Union) 값을 가지는, 즉 실제 객체와 겹치는 영역을 가리키는 RoI(Region of Interest)를 의미

- 나머지 RoI는 [0.1, 0.5) 구간의 최대 IoU를 가지는 object proposal에서 샘플링되며, 이들은 배경 예시로 $u = 0$으로 라벨이 붙음.
- 0.1의 하한선은 어려운 예시를 채굴하기 위한 휴리스틱(경험적 규칙)으로 작용
- **Augmentation은 horizontal flip p=0.5 만 사용** (50%비율로 적용)

- 간단히 요약하면, fine-tuning 동안 미니 배치는 두 개의 이미지에서 샘플링된 128개의 RoI로 구성되며, 이 중 일부는 실제 객체와 높은 IoU를 갖는 전경 예시이고, 나머지는 낮은 IoU를 가진 배경 예시. 데이터 증강은 이미지의 수평 반전만 포함하며, 이는 훈련 과정에서 무작위로 적용됨

VISION SYSTEM LABORATORY

Back-propagation through RoI pooling layers

- 네트워크를 어디까지 학습시킬 것인가?
- SPP Net에서는 피쳐 맵을 뽑는 CNN 부분은 그대로 놔두고, SPP 이후의 FC들만 fine-tune
- 이 논문에서는 이럴 경우 이미지로부터 특징을 뽑는 가장 중요한 역할을 하는 CNN이 학습될 수 없기 때문에 성능 향상에 제약이 있다고 주장
- RoI Pooling 레이어 이전까지 back propagation을 전달할 수 있는지를 이론적으로 검증

$$\frac{\partial L}{\partial x_i} = \sum_r \sum_j [i = i^*(r,j)] \frac{\partial L}{\partial y_{rj}}. \qquad (4)$$

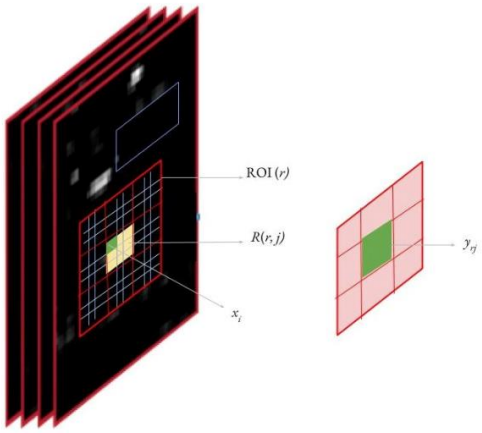
- xi : CNN을 통해 추출된 피쳐 맵에서 하나의 피쳐 값을 의미 / 실수
- 전체 Loss에 대해서 이 피쳐 값의 편미분 값을 구하면 그 값이 곧 xi에 대한 loss 값이 되며 역전파 알고리즘을 수행할 수 있음

- 명확성(clarity)를 위해 미니 배치당 하나의 이미지(N = 1)만 있다고 가정하지만, 실제로는 모든 이미지가 독립적으로 처리되므로 N > 1로 확장하는 것은 간단(straightforward)
- xi∈R을 RoI 풀링 레이어에 대한 i번째 활성화 입력으로, yrj를 r번째 RoI에서 레이어의 j번째 출력으로 설정
- R(r, j)는 출력 단위 yrj가 최대 풀링하는 서브 윈도우 안의 입력 인덱스 집합.
- 하나의 xi는 여러 다른 출력 yrj에 할당될 수 있음

VISION SYSTEM LABORATORY

Back-propagation through RoI pooling layers

- 이제 피쳐 맵에서 RoI를 찾고 RoI Pooling을 적용하기 위해서 H x W 크기의 grid로 나눔
- 이 그리드들을 sub-window라고 부르며, 위 수식에서 j란 몇 번째 sub-window 인지를 나타내는 인덱스
- yrj란 이 RoI pooling을 통과하여 최종적으로 얻어진 output의 값 / 실수
- 이를 그림으로 나타내면 아래와 같음



$$\frac{\partial L}{\partial x_i} = \sum_r \sum_j [i = i^*(r,j)] \frac{\partial L}{\partial y_{rj}}. \qquad (4)$$

- 즉, 각 미니 배치 RoI r 및 각 풀링 출력 단위 yrj에 대해 i가 최대 풀링에 의해 yrj에 대해 선택된 argmax 이면 편미분 ∂L/∂yrj가 누적
- 역전파에서 편미분 ∂L/∂yrj는 RoI 풀링 레이어 위에있는 레이어의 backwards 함수에 의해 이미 계산됨

VISION SYSTEM LABORATORY

SGD hyper-parameters

- softmax 분류 및 bounding-box regression 에 사용되는 fully connected layers 는 각각 표준 편차가 0.01, 0.001 / 평균이 0인 가우시안 분포로 초기화
- 편향(bias)은 0으로 초기화
- 모든 레이어는 가중치에 레이어당 학습률 1, 편향에 레이어당 학습률 2, 전역 학습률 0.001사용
- VOC07 또는 VOC12 trainval에서 훈련할 때 SGD를 30,000개의 미니배치 반복 동안 실행한 다음 학습률을 0.0001로 낮추어 추가로 10,000개의 반복을 수행
- 모멘텀 0.9
- 가중치와 편향에 대한 매개 변수 감소 (weight decay) 0.0005

2.4. Scale invariance

Scale invariance 하게 detection 하는 방법 2가지

(1) brute force 학습을 통한 방법 : train 및 test에서 미리 정의된 size로 맞춰서 진행

네트워크는 훈련 데이터에서 직접적으로 크기에 불변한 물체 검출을 학습

(2) image pyramids 사용을 통한 방법 : 다중 스케일(**multi-scale**) 접근 방식

이미지 피라미드를 통해 네트워크에 대략적인 크기 불변성(approximate scale-invariance)을 제공

테스트 시에 이미지 피라미드는 각 object proposal을 대략적으로 크기 정규화하는 데 사용됨

다중 스케일 훈련 중에는 이미지를 샘플링할 때마다 피라미드 스케일을 무작위로 샘플링

: **여러 scale로 random 하게 진행**

GPU 메모리 제한으로 인해 작은 네트워크에 대해서만 다중 스케일 훈련을 실험

이미지 피라미드 : 동일한 이미지를 서로 다른 해상도로 표현한 이미지 집합
원본 이미지에서 시작하여 점차적으로 해상도를 줄여가며 여러 단계의 이미지를 생성,
이렇게 생성된 각 단계의 이미지는 이전 단계의 이미지보다 해상도가 낮음

3. Fast R-CNN detection

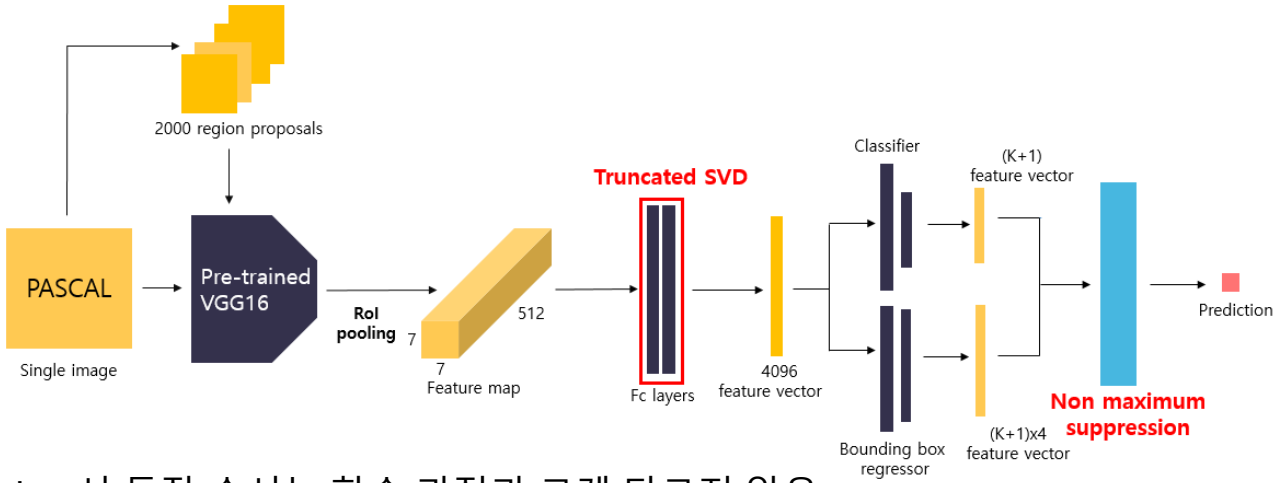
- Fast R-CNN 네트워크가 fine-tuning 되면, object proposals이 미리 계산된다고 가정하면,
  detection amounts to little more than running a forward pass 전방향 통과
  : 입력 데이터를 받아서 출력을 낸 그 자체 / backward 안해도됨,
  훈련을 했으면 그대로 forward해서 내보내면 된다
  amount to : ~에 이르다, ~와 마찬가지이다

- 네트워크는 이미지(또는 이미지 목록으로 인코딩된 이미지 피라미드)와 점수를 매길 R개의 object proposals 목록을 입력으로 받음
- 테스트 시간에는 R이 일반적으로 약 2000 정도이지만, 이보다 큰 경우(약 45k)도 고려하게 됨
- 이미지 피라미드를 사용할 때, 각 RoI(Region of Interest)는 224x224 픽셀 면적에 가장 근접하도록 스케일에 할당됨

- 각 테스트 RoI에 대해, forward pass는 클래스 후방 확률 분포와 r에 상대적인 예측된 bounding-box 오프셋을 출력(각 K 클래스에 대해 자체적으로 정제된 bounding-box 예측이 있음)
- k 클래스에 대한 r의 탐지 확신도를 예측된 확률 $Pr(class = k \mid r) = p_k$를 사용하여 할당
- R-CNN에서 사용된 알고리즘과 설정을 사용하여 각 클래스에 대해 독립적으로 최대치 억제(non-maximum suppression)를 수행

- 간단히 요약하자면, Fast R-CNN을 통한 객체 탐지는 이미지와 object proposals 목록을 네트워크에 주고, 각각의 RoI에 대해 클래스별 확률과 bounding-box 조정을 출력하며, 이렇게 계산된 확률을 기반으로 각 클래스별로 최대치 억제 과정을 거쳐 최종 탐지 결과를 도출하는 과정을 말함

3. Fast R-CNN detection



- Detection 시 동작 순서는 학습 과정과 크게 다르지 않음
- 4096 크기의 feature vector를 출력하는 fc layer에 **Truncated SVD**를 적용한다는 점에서 차이가 있음
- 또한 예측한 bounding box에 대하여 Non maximum suppression 알고리즘이 추가되어 최적의 bounding box만을 출력하게 됨

- Non-maximum suppression (NMS)은 객체 탐지(object detection)에서 중복된 탐지 결과를 줄이기 위해 사용되는 기술
- 탐지 과정에서 하나의 객체에 대해 여러 개의 바운딩 박스(bounding boxes)가 생성될 수 있는데, NMS는 이 중 가장 정확하다고 판단되는 단일 박스를 선택하는 데 도움을 줌
- NMS의 기본적인 단계:

1. 탐지된 모든 바운딩 박스에 대해, 각각의 객체에 대한 탐지 확률(또는 점수)를 기준으로 정렬
2. 가장 높은 점수를 받은 바운딩 박스를 선택하고, 이를 최종 탐지 목록에 추가
3. 선택된 박스와 나머지 모든 박스 간의 IoU(Intersection over Union)를 계산
4. IoU가 특정 임계값(threshold) 이상인 박스들을 제거합니다. 이 임계값은 일반적으로 0.3에서 0.5 사이로 설정되며, 이는 실험적으로 결정되거나 특정 작업에 최적화될 수 있음
5. 남아 있는 박스 중에서, 다시 가장 높은 점수를 가진 박스를 선택하고 2-4단계를 반복

- 이 과정을 모든 박스가 검토될 때까지 계속하게 되면, 각 객체에 대해 하나의 최적화된 박스만이 결과로 남게 되어 객체 탐지의 정확도를 높이게 됨. NMS는 다수의 탐지 결과 중에서 가장 높은 신뢰도를 가진 탐지만을 남기므로 객체 탐지 모델의 성능을 개선하는 데 매우 중요한 역할

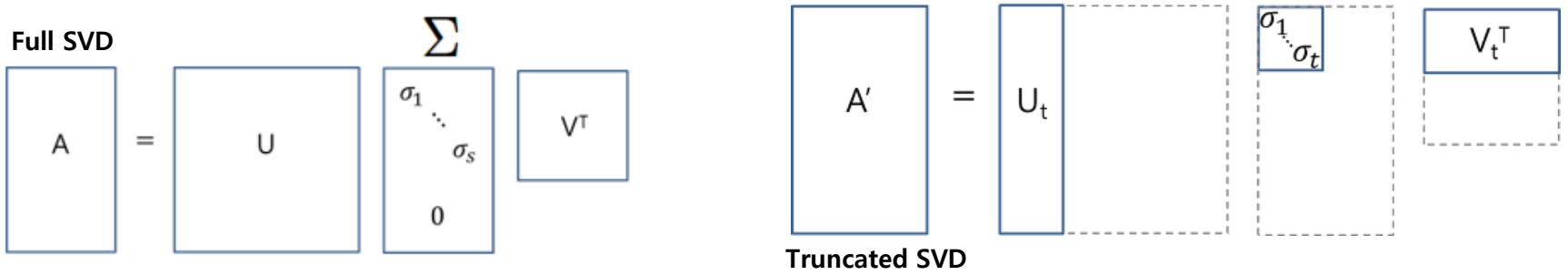
3.1. Truncated SVD for faster detection

Fast R-CNN 모델은 detection 시, RoI를 처리할 때 fc layer에서 많은 시간 소요

논문에서는 detection 시간을 감소시키기 위해 Truncated SVD(Singular Vector Decomposition)를 통해 FC layer를 압축하는 방법을 제시

Full SVD (Singular Vector Decomposition) : 행렬 A를 m×m크기인 U, m×n크기인 Σ, n×n크기인 V^T 로 특이값 분해(SVD)하는 것

Full SVD보다는 Truncated SVD와 같이 분해된 행렬 중 일부분만을 활용하는 reduced SVD를 일반적으로 많이 사용



Truncated SVD는 Σ의 비대각 부분과 대각 원소 중 특이값이 0인 부분을 모두 제거하고, 제거된 Σ에 대응되는 U, V 원소도 함께 제거하여 차원을 줄인 형태

Ut의 크기는 m×t, Σt의 크기는 t×t, V^t의 크기는 t×n / 이렇게 행렬 A를 상당히 근사하는 것이 가능

fc layer의 가중치 행렬이 W(=u×v)라고 할 때, Truncated SVD를 통해 위와 같이 근사하는 것이 가능

이를 통해 파라미터 수를 u×v에서 t(u+v)로 감소시키는 것이 가능

Truncated SVD를 FC layer의 가중치 행렬 W에 적용하면, FC layer는 두 개의 fc layer로 나눠짐

첫 번째 FC layer는 ΣtV^T가중치 행렬, 두 번째 FC layer는 U가중치 행렬

이를 통해 네트워크를 효율적으로 압축하는 것이 가능하며, 논문에서 **Truncated SVD를 통해 detection 시간이 30% 정도 감소**되었다고 말함

VISION SYSTEM LABORATORY

3.1. Truncated SVD for faster detection

- 전체 이미지 분류에서는 FC layers의 계산 시간이 컨볼루션 계층(conv layers)에 비해 상대적으로 작음
- 객체 탐지의 경우 처리해야 할 관심 영역(RoIs)의 수가 많기 때문에, 전방향 패스(forward pass) 시간의 거의 절반 가까이가 전체 연결 계층의 계산에 소요
- 전체 연결 계층은 truncated SVD를 사용하여 압축함으로써 쉽게 가속화될 수 있음

$$W \approx U\Sigma_t V^T$$

- u × v 가중치 행렬 W가 / SVD를 사용하여 UΣtV^T로 approximately(거의, ~에 가깝게) 분해
- 이 분해에서 U는 W의 처음 t개의 왼쪽 특이 벡터를 포함하는 u × t 행렬, Σt는 W의 상위 t개의 특이값을 포함하는 t × t 대각행렬, V는 W의 처음 t개의 오른쪽 특이 벡터를 포함하는 v × t 행렬

- truncated SVD는 매개변수 수를 uv에서 t(u + v)로 줄여주며, t가 min(u, v)보다 훨씬 작다면 이는 상당한 차이를 만들어냄
- 네트워크를 압축(compress)하기 위해, W에 해당하는 single FC layer은 중간에 비선형성이 없이 두 개의 FC layers으로 대체
- 첫 번째 계층은 가중치 행렬 ΣtV^T(편향 없음)을 사용하고, 두 번째 계층은 원래 W와 연결된 편향을 가진 U를 사용. 이 간단한 압축 방법은 RoI의 수가 많을 때 좋은 속도 향상을 제공

VISION SYSTEM LABORATORY

4. Main results

Three main results support this paper's contributions기여:

1. State-of-the-art (sota, 현존 최고 수준) mAP on VOC07, 2010, and 2012
2. Fast training and testing compared to R-CNN, SPPnet
3. VGG16의 conv layer를 fine-tuning시 mAP 상승

4.1. Experimental setup

3가지 pre-trained 모델 사용

1. R-CNN의 CaffeNet (본질적으로 AlexNet) - S (small)
2. VGG CNN M 1024 : S와 같은 깊이를 가지지만 더 넓음 - M (medium)
3. 매우 깊은 VGG16 - L (large)

single-scale로 train, test 진행

4.2. VOC 2010 and 2012 results

- Fast R-CNN이 VOC12에서 mAP 65.7%(추가 데이터 사용 시 68.4%)로 최고의 결과를 달성

  "느린" R-CNN 파이프라인을 기반으로 하는 다른 방법들보다 훨씬 빠름

- VOC10에서는 SegDeepM이 Fast R-CNN보다 높은 mAP를 달성했지만,

  Fast R-CNN은 R-CNN 대신 SegDeepM에 도입되어 더 나은 결과를 가져올 수 있음

  또한, 확대된 07++12 학습 세트를 사용하면 Fast R-CNN의 mAP가 68.8%로 증가하여 SegDeepM을 능가

- Fast R-CNN의 높은 효율성과 성능을 잘 보여주는 결과임

method	train set	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	persn	plant	sheep	sofa	train	tv	mAP
BabyLearning	Prop.	77.7	73.8	62.3	48.8	45.4	67.3	67.0	80.3	41.3	70.8	49.7	79.5	74.7	78.6	64.5	36.0	69.9	55.7	70.4	61.7	63.8
R-CNN BB [10]	12	79.3	72.4	63.1	44.0	44.4	64.6	66.3	84.9	38.8	67.3	48.4	82.3	75.0	76.7	65.7	35.8	66.2	54.8	69.1	58.8	62.9
SegDeepM	12+seg	**82.3**	75.2	67.1	50.7	**49.8**	71.1	69.6	88.2	42.5	71.2	50.0	85.7	76.6	81.8	69.3	**41.5**	**71.9**	62.2	73.2	**64.6**	67.2
FRCN [ours]	12	80.1	74.4	67.7	49.4	41.4	74.2	68.8	87.8	41.9	70.1	50.2	86.1	77.3	81.1	70.4	33.3	67.0	63.3	77.2	60.0	66.1
FRCN [ours]	07++12	82.0	**77.8**	**71.6**	**55.3**	42.4	**77.3**	**71.7**	**89.3**	**44.5**	**72.1**	**53.7**	**87.7**	**80.0**	**82.5**	**72.7**	36.6	68.7	**65.4**	**81.1**	62.7	**68.8**

표 2. VOC 2010 테스트 탐지 평균 정밀도 (%). BabyLearning은 [17]에 기반한 네트워크를 사용. 다른 모든 방법은 VGG16을 사용. 훈련 세트 키 : 12 : VOC12 trainval, Prop.: 독점 데이터 세트, 12+seg: 분할 주석이 있는 12, 07++12: VOC07 결합 trainval, VOC07 테스트 및 VOC12 trainval.

method	train set	aero	bike	bird	boat	bottle	bus	car	cat	chair	cow	table	dog	horse	mbike	persn	plant	sheep	sofa	train	tv	mAP
BabyLearning	Prop.	78.0	74.2	61.3	45.7	42.7	68.2	66.8	80.2	40.6	70.0	49.8	79.0	74.5	77.9	64.0	35.3	67.9	55.7	68.7	62.6	63.2
NUS_NIN_c2000	Unk.	80.2	73.8	61.9	43.7	**43.0**	70.3	67.6	80.7	41.9	69.7	51.7	78.2	75.2	76.9	65.1	**38.6**	**68.3**	58.0	68.7	63.3	63.8
R-CNN BB [10]	12	79.6	72.7	61.9	41.2	41.9	65.9	66.4	84.6	38.5	67.2	46.7	82.0	74.8	76.0	65.2	35.6	65.4	54.2	67.4	60.3	62.4
FRCN [ours]	12	80.3	74.7	66.9	46.9	37.7	73.9	68.6	87.7	41.7	71.1	51.1	86.0	77.8	79.8	69.8	32.1	65.5	63.8	76.4	61.7	65.7
FRCN [ours]	07++12	**82.3**	**78.4**	**70.8**	**52.3**	38.7	**77.8**	**71.6**	**89.3**	**44.2**	**73.0**	**55.0**	**87.5**	**80.5**	**80.8**	**72.0**	35.1	**68.3**	**65.7**	**80.4**	**64.2**	**68.4**

표 3. VOC 2012 테스트 탐지 평균 정밀도(%). BabyLearning과 NUS_NIN_c2000은 [17]에 기반한 네트워크를 사용. 다른 모든 방법은 VGG16을 사용. 훈련 세트 키 : 표 2 참조, Unk .: 알 수 없음

4.3. VOC 2007 results

- VOC07에서는 Fast R-CNN을 R-CNN 및 SPPnet과 비교
- 모든 방법은 동일한 pre-trained VGG16 네트워크에서 시작하며 bounding-box regression을 사용
- (VGG16 SPPnet 결과는 [11]의 저자에 의해 계산됨)
- [11] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In ECCV, 2014.

- SPPnet은 훈련 및 테스트 중에 5가지 척도를 사용
- Fast R-CNN이 conv 레이어를 미세 조정하면 mAP가 크게 향상된다는 것을 확인함 (63.1%에서 66.9%로) (반면 R-CNN은 66.0%의 mAP를 달성)

- 사소한 점으로(As a minor point), SPPnet은 PASCAL에서 "어려움"으로 표시된 예제 없이 훈련됨.
- 이러한 사례를 제거하면 Fast R-CNN mAP가 68.1%로 향상됨.

- 다른 모든 실험은 "어려운" 사례를 포함하여 진행됨

VISION SYSTEM LABORATORY

4.4. Training and testing time

	Fast R-CNN			R-CNN			SPPnet
	S	M	L	S	M	L	†L
train time (h)	**1.2**	2.0	9.5	22	28	84	25
train speedup	**18.3**×	14.0×	8.8×	1×	1×	1×	3.4×
test rate (s/im)	0.10	0.15	0.32	9.8	12.1	47.0	2.3
▷ with SVD	**0.06**	0.08	0.22	-	-	-	-
test speedup	98×	80×	146×	1×	1×	1×	20×
▷ with SVD	169×	150×	**213**×	-	-	-	-
VOC07 mAP	57.1	59.2	**66.9**	58.5	60.2	66.0	63.1
▷ with SVD	56.5	58.7	66.6	-	-	-	-

표 4. Fast RCNN, R-CNN 및 SPPnet에서 동일한 모델 간의 런타임 비교
Fast R-CNN은 단일 스케일 모드를 사용
SPPnet은 [11]에 명시된 5가지 스케일을 사용
Timing은 [11]의 저자에 의해 제공됨
시간은 Nvidia K40 GPU에서 측정됨

빠른 학습 및 테스트 시간

표 4는 Fast RCNN, R-CNN 및 SPPnet 간의 VOC07에 대한 훈련 시간, 테스트 속도 (이미지 당 초) 및 mAP를 비교

R-CNN보다 VGG16을 146배 (Truncated SVD 없이) / 213배 (with Truncated SVD) 빠르게 이미지를 처리
훈련 시간이 84시간에서 9.5시간으로 9배 단축

SPPnet보다 VGG16을 2.7배 빠르게 (in 9.5 vs. 25.5 hours)훈련, 테스트는 7배 (Truncated SVD 없이) / 10배 (with Truncated SVD) 더 빠름

Fast R-CNN은 기능을 캐시하지 않기 때문에 수백 기가 바이트의 디스크 스토리지를 제거

Truncated SVD를 사용해 훨씬 빨라졌다

feature caching을 하지 않아 별도 저장공간이 필요 없다

VISION SYSTEM LABORATORY

4.5. Which layers to fine-tune?

	layers that are fine-tuned in model L			SPPnet L
	≥ fc6	≥ conv3_1	≥ conv2_1	≥ fc6
VOC07 mAP	61.4	66.9	**67.2**	63.1
test rate (s/im)	**0.32**	**0.32**	**0.32**	2.3

표 5. VGG16에 대해 미세 조정되는 레이어를 제한하는 효과. fc6 이상의 미세 조정은 SPPnet 훈련 알고리즘 [11]을 에뮬레이트하지만 단일 척도를 사용합. SPPnet L 결과는 상당한 (7배) 속도 비용으로 5개의 스케일을 사용하여 얻었음

SPPnet 논문 : less deep 네트워크에 대해서, fully connected layers만 fine-tuning 하는 것이 높은 정확도에 충분(sufficient)하다는 결과

이 논문 : 이 결과가 매우 깊은 네트워크에는 적용되지 않을(would not hold)것이라고 가정(hypothesized)

VGG16에서 conv 레이어를 fine-tuning하는 것이 중요하다는 것을 확인하기 위해

Fast R-CNN을 사용하여 fine-tuning하되, 13개의 conv 레이어를 고정시켜서, fully connected layers만 학습하도록 함

이 ablation(분리)은 single-scale SPPnet 훈련을 모방(emulate)하며, mAP를 66.9%에서 61.4%로 감소시킴 (표 5).

이 실험은 가설을 검증; RoI pooling layer를 통한 훈련은 very deep nets에 중요함

이것이 **모든 conv 계층이 fine-tuning 되어야 한다는 것을 의미? -> 아님**

더 작은 네트워크(S와 M)에서는 conv1이 일반적(generic)이고 작업에 독립적(task independent)임

conv1을 학습시키는 것이 mAP에 무의미한 영향

VGG16에서는 conv3 1 계층 이상(13개의 conv 계층 중 9개)을 업데이트하는 것이 필요,

이 관찰은 실용적: (1) conv2 1부터 업데이트하면 conv3 1에서 학습하는 것에 비해 훈련 속도가 1.3배 느려짐(12.5시간 대 9.5시간). (2) conv1 1부터 업데이트하면 GPU 메모리가 넘침

요약 : **VGG16은 9번째 이전의 conv layer는 freeze 하고 그 이후 layer만 학습**

(훈련 속도 빠름, GPU memory over-run 방지)

이 논문에서 VGG16을 사용한 모든 Fast R-CNN 결과는 conv3 1 계층 이상을 fine-tuning

모델 S와 M을 사용한 모든 실험은 conv2 계층 이상을 fine-tuning

VISION SYSTEM LABORATORY

5. Design evaluation

- Fast R-CNN이 R-CNN과 SPPnet에 비해 어떻게 비교되는지, 그리고 디자인 결정(design decisions)을 평가하기 위해 실험을 수행했음(conducted)

- 모범 사례를 따라(Following best practices), 이러한 실험을 PASCAL VOC07 데이터셋에서 수행(performed)했음

5.1. Does multi-task training help?

- multi-task 훈련은 연속적으로 훈련된 작업의 파이프라인을 관리하는 것을 피할 수 있어 편리하며,
- **공유된 표현(여기서는 ConvNet)을 통해 작업들이 서로에게 영향을 미치므로 결과를 개선**할 수 있는 잠재력을 가지고 있음
- (한 작업에서 학습된 정보가 다른 작업의 학습에 도움을 줄 수 있다)
- classification loss, Lcls만을 사용하여 기본 네트워크를 훈련시킨 결과를 통해 이를 테스트함
- 이러한 기본 모델들은 bounding-box regressors를 갖고 있지 않음

- 다음으로 multi-task loss로 훈련된 네트워크를 취하되, 테스트 시에는 bounding-box regressor를 비활성화
- 이로써 네트워크의 분류 정확도를 분리하고 기본 네트워크와의 비교를 가능하게 함

- 세 가지 네트워크 모두에서 **multi-task 훈련이 순수 분류 정확도를 향상시키는 것을 확인**
- 향상 폭은 +0.8에서 +1.1 mAP 포인트로, multi-task 학습으로부터 일관되게 긍정적인 효과를 보여줌

- 마지막으로, classification loss만을 사용하여 훈련된 기본 모델에 bounding-box regressor 계층을 추가하고, 모든 다른 네트워크 매개변수를 고정한 상태에서 Lloc을 사용하여 훈련
- 이 단계별 훈련 방식의 결과는 mAP가 향상되었지만, 다중 작업 훈련에 비해 떨어진 성능을 보였음

5.2. Scale invariance: to brute force or finesse?

- 스케일 불변scale-invariant 객체 감지object detection를 달성하기 위한 두 가지 전략: brute-force 학습(단일 스케일)(single scale)과 이미지 피라미드(다중 스케일)(multi-scale)를 비교
- 두 경우 모두 이미지의 스케일 s는 그 짧은 변의 길이로 정의
- single scale 실험에서는 s = 600 픽셀을 사용하였으며, 이미지의 가장 긴 변을 1000 픽셀로 제한하고 이미지의 종횡비를 유지하므로 s가 600 미만일 수도 있음
- multi-scale 설정에서는 SPPnet과 비교하기 위해 동일한 5개의 스케일을 사용

  그러나, GPU 메모리를 초과하지 않도록 가장 긴 변은 2000 픽셀로 제한

- 결과적으로, **single-scale 탐지가 multi-scale 탐지와 거의 비슷한 성능**
- multi-scale – 큰 계산 시간에 비해 성능 향상이 작음
- 속도와 정확도 사이의 최적의 균형을 제공하는 **single-scale 처리를 사용**하여 모든 실험을 수행

VISION SYSTEM LABORATORY

5.3. Do we need more training data?

- 좋은 객체 탐지기는 **더 많은 훈련 데이터가 제공될 때 성능이 향상**되어야 한다(should)
- 이 연구에서는 VOC07 trainval 세트에 VOC12 trainval 세트를 추가하여 이미지 수를 대략 3배인 16.5k로 늘려 Fast R-CNN을 평가
- training 세트를 확대하면 VOC07 테스트의 mAP가 66.9%에서 70.0%로 향상됨
- 데이터 세트에서 훈련할 때 40k 대신 60k 미니 배치 반복을 사용
- 비슷한 실험을 VOC10 및 2012에 대해 수행하였고, 이를 위해 VOC07 trainval, 테스트, 및 VOC12 trainval의 합집합에서 21.5k 이미지의 데이터셋을 구성
- 이 데이터셋에서 훈련할 때는 100k의 SGD 반복을 사용하고, 학습률은 40k 반복마다 0.1×씩 낮춤
- 결과적으로, VOC10과 2012에 대해 mAP는 각각 66.1%에서 68.8%, 65.7%에서 68.4%로 향상

5.4. Do SVMs outperform softmax?

- Fast R-CNN은 R-CNN 및 SPPnet에서 수행된 것처럼 사후에 일대다(one-vs-rest) 선형 SVM을 학습하는 대신, fine-tuning 과정에서 학습된 softmax 분류기를 사용
- 이 선택의 impact를 이해하기 위해, Fast R-CNN에서 hard negative mining을 통한 post-hoc SVM 학습을 구현
- R-CNN에서 사용된 것과 동일한 학습 알고리즘과 하이퍼파라미터를 사용

method	classifier	S	M	L
R-CNN [9, 10]	SVM	**58.5**	**60.2**	66.0
FRCN [ours]	SVM	56.3	58.7	66.8
FRCN [ours]	softmax	57.1	59.2	**66.9**

표 8. 소프트 맥스 vs. SVM (VOC07 mAP)을 사용한 Fast R-CNN.

- 표 8은 모든 세 네트워크에 대해 **softmax가 SVM을 약간 능가**하며, mAP 점수가 +0.1에서 +0.8 상승하는 것을 보여줌
- 이 효과는 작지만, 이전의 다단계 학습 접근법에 비해 **'one-shot' fine-tuning이 충분**(sufficient)하다는 것을 보여줌
- 미리 학습된 모델을 가져와서 특정 작업에 대한 추가 데이터를 사용하여 모델을 더 조정하는 것을 의미, 적은 양의 추가 데이터만 사용되므로 일반적인 fine tuning보다는 데이터 부족 상황에서도 효과적으로 모델을 개선할 수 있는 장점
- softmax는 일대다 SVM과 달리, RoI를 점수화할 때 클래스 간 경쟁을 도입

5.5. Are more proposals always better?

- 객체 탐지기는 크게 두 가지 유형:
  희소한(sparse) object proposal을 사용하는 탐지기(예: 선택적 검색 selective search),
  밀집된(dense) object proposal을 사용하는 탐지기(예: DPM)
- 희소한 proposal을 분류하는 것은 제안 메커니즘이 먼저 대량vast의 후보candidates를 거부하고 분류기가 평가할 작은 집합을 남기는 종류의 cascade, 이런 방식은 탐지 정확도를 향상
- 선택적 검색의 품질 모드를 사용하여 이미지 당 proposal 수를 늘려보았을 때, mAP(평균 정밀도)가 증가했다가 약간 감소하는 것을 발견
- 즉, **더 많은 proposal로 deep classifier를 범람**(swamping)**시키는 것은 도움이 되지 않으며, 심지어 정확도를 약간 해치는**(slightly hurts) **것을 확인**

- 이미지 당 제안 수에 따른 AR(평균 회수)와 mAP 간의 상관관계를 확인했을 때, AR이 mAP와 잘 상관관계를 보이지 않는다는 것을 확인
- AR은 신중하게 사용, 더 많은 proposal로 인한 높은 AR은 mAP가 증가한다는 것을 의미하지 않음

- 밀집된 박스를 사용하여 실험을 진행했을 때, 선택적 검색 박스 각각이 가장 가까운 밀집 박스로 대체될 때 mAP가 1점만 떨어지는 것을 확인
- 하지만 밀집 박스를 더 추가하면 mAP가 더 강하게 떨어졌으며, 밀집 박스만을 사용하여 Fast R-CNN을 학습하고 테스트했을 때는 mAP가 52.9%로 나타남
- 마지막으로, 밀집 박스 분포에 대처하기 위해 어려운 부정 샘플링이 있는 SVM이 필요한지 확인했는데, 그 결과 SVM은 더 나빠져서 49.3%로 나타남

VISION SYSTEM LABORATORY

5.6. Preliminary예비 MS COCO results

- Fast R-CNN (VGG16을 사용)을 MS COCO 데이터셋에 적용하여 초기 베이스라인을 설정
- 80,000개의 이미지 훈련 세트에서 240,000 번의 반복을 통해 훈련, "test-dev" 세트에서 평가를 수행
- PASCAL 형식의 mAP는 35.9%
- IoU 임계값을 초과하는 평균이기도 한 새로운 COCO 스타일 AP는 19.7%

- COCO-style Average Precision (AP)는 COCO (Common Objects in Context) 데이터셋을 사용하여 객체 탐지 모델의 성능을 평가할 때 사용되는 지표
- COCO-style AP는 PASCAL VOC에서 사용되는 mAP와 비슷하지만, 몇 가지 중요한 차이점 있음
- COCO-style AP는 여러 IoU (Intersection over Union) 임계값에서의 성능을 평균내는 반면, PASCAL VOC의 mAP는 단일 IoU 임계값에서만 성능을 평가
- IoU는 예측된 바운딩 박스와 실제 바운딩 박스 사이의 겹치는 영역을 측정하는 지표
- COCO-style AP는 IoU가 0.5에서 0.95까지 0.05씩 증가하는 10개의 임계값에서 평균 AP를 계산
- 따라서, 모델이 다양한 정밀도에서 얼마나 잘 작동하는지를 평가할 수 있음

VISION SYSTEM LABORATORY

6. Conclusion

- 본 논문에서는 R-CNN 및 SPPnet을 깔끔하고 빠르게 업데이트한 Fast R-CNN을 제안
- 최신의 검출 결과를 보고하는 것 외에도 새로운 통찰력을 제공하는 상세한 실험을 제시
- 특히(Of particular note) sparse희소 object proposals이 검출기 품질을 향상
- 이 문제는 이전에는 시간이 많이 소요되어 조사하기 어려웠지만, Fast R-CNN을 사용하면 실용적(practical)
- 물론, 밀도 있는 박스dense boxes가 희소한 제안sparse proposals만큼 잘 수행되게 하는 아직 발견되지 않은 기술이 존재할 수 있음
- 이러한 방법이 개발되면 물체 검출 속도를 더 높이는 데 도움이 될 것

한계

R-CNN이나 SPP-net에 비하면 뛰어난 성능을 보이고
1.2와 같은 Contribution들을 가지고 있으며,
앞의 논문들이 가지고 있었던 많은 문제들을 해결

하지만 이미지 한 장당 2.3초의 Test time이 걸리는 알고리즘은 Real-time Object Detector로는 역부족
게다가 Region Proposal에 걸리는 총 2.3초 중 2초의 시간 때문에 병목이 생기는 구조

이 문제들을 해결하기 위해
비슷하지만 다른 구조의 Faster R-CNN 논문이 나옴

참고 자료 출처

- R. Girshick, “Fast R-CNN,” in IEEE International Conference onComputer Vision (ICCV), 2015.
- https://velog.io/@whiteamericano/R-CNN-을-알아보자
- https://herbwood.tistory.com/8
- https://yeomko.tistory.com/15
- https://nuggy875.tistory.com/33
- https://talktato.tistory.com/9