[关闭]
@zhenni94 2015-09-07 21:35 字数 7562 阅读 2031

R-CNN & Fast R-CNN & Faster R-CNN

R-CNN: Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation

Paper：http://www.cs.berkeley.edu/~rbg/#girshick2014rcnn
Tech report: http://arxiv.org/pdf/1311.2524v5.pdf
Project：https://github.com/rbgirshick/rcnn
Slides: http://www.cs.berkeley.edu/~rbg/slides/rcnn-cvpr14-slides.pdf

Referrence: a blog

object detection system

Three modules:
1. Generate region proposals (~2k/image)
2. Compute CNN features
3. Classify regions using linear SVM

R-CNN at test time

- **Region proposals**
  Proposal-method agnostic, many choices:
  - **Selective Search** (2k/image "fast mode") [van de Sande, Uijlings et al.] (Used in this work)(Enable a controlled comparison with prior detection work)
  - Objectness [Alexe et al.]
  - Category independent object proposals [Endres & Hoiem]
  - CPMC [Carreira & Sminchisescu] - segmentation
  - BING [Ming et al.] – fast
  - MCG [Arbelaez et al.] – high-quality segmentation
- **Feature extraction with CNN**
  - Dilate the proposal (At the warped size there are exactly p=16 pixels warped image context around the original box)

    - Crop and scale to 227*227(anisotropic)
    - Forward propagate in AlexNet (5conv & 2fc). Get fc_7 layer features.
- **Classify regions by SVM**
    - linear SVM per class
      (With the sofmax classifier from fine-tuning mAP decreases from 54% to 51%)
    - greedy NMS(non-maximum suppression) per class : rejects a region if it has an intersection-overunion (IoU) overlap with a higher scoring selected region larger than a learned threshold.
- **Object proposal refinement**
    - Linear bounding-box regression on CNN features (pool_5 feature: mAP ~4% up)
    - (in Appendix C)

Training R-CNN

- Bounding-box labeled detection data is scarce

- Use supervised pre-training on a data-rich auxiliary task and transfer to detection

- **Supervised pre-training**
  Pre-train CNN on ILSVRC2012(1.2 million 1000-way image classification) using image-level annotations only

- **Domain-specific fine-tuing**
  Adapt to new task(detection) and new domain(warped proposal)
    - random initialize (N+1)-way classification layer (N classes + background)
    - Positives: $\geq$0.5 IoU overlap with a ground-truth box. Negative: o.w.
    - SGD: learning rate: 0.001 (1/10 of original) mini-batch: 32 pos & 96 neg
- **Train binary SVM**
    - IoU overlap threshold: grid search over {0, 0.1, ... 0.5}
      IoU = 0.5 : mAP ~5% down
      IoU = 0.0 : mAP ~4% down

Fast R-CNN

Paper: http://arxiv.org/pdf/1504.08083v1.pdf
Project: https://github.com/rbgirshick/fast-rcnn

Referrence: [blog]()

Motivation

Drawback of R-CNN and the modification:
1. Training is a multi-stage pipeline. -> End-to-end joint training.
2. Training is expensive in space and time. -> Convolutional layer sharing. Classification in memory.
For SVM and regressor training, features are extracted from each warped object proposal in each image and written to disk.(VGG16, 5k VOC07 trainval images : 2.5 GPU days). Hundreds of gigabytes of storage.
3. Test-time detection is slow. -> Single scale testing, SVD fc layer.
At test-time, features are extracted from each warped proposal in each img. (VGG16: 47s / image).

Contributions:
1. Higher detection quality (mAP) than R-CNN
2. Training is single-stage, using a multi-task loss
3. All network layers can be updated during training
4. No disk storage is required for feature caching

Fast R-CNN training

- **RoI pooling layer**
  - Find the patch in feature map corresponding to the RoI; Get fixed-length feature using SPPnet to feed in fc layer
  - A simplified version of the spatial pyramid pooling used in SPPnet, in which "pyramid" has only one level
  - Input :
    N feature maps (last conv layer H*W*C),
    a list of R RoI(tuple [n, r, c, h, w] n: index of a feature map, (r,c): top-left loc) (R $\ll$ N)
  - Output: max-pooled feature maps(H'*W'*C) (H'$\leq$H, W'$\leq$W)
- **Use pre-trained Networks**
  Tree transformations:(VGG 16)
  - last pooling layer -> RoI pooling layer (H'*W' compatibale to fc layer)
  - final fc and softmax layer -> two sibling layers: fc + (K+1)-softmax and fc + bounding box regressor (K is the number of the classes)
  - Modified to take two data inputs: N feature maps and a list of RoI
- **Fine-tuning for detection**
  - Back propogation through SPP layer.
  - BP through conv: Image-centric sampling. mini-batch sample hierachically: images -> RoI
    Same image shares computation and memory
  - Joint optimaize a softmax classifier and bounding-box regressors
  - **Multi-task Loss**
    - Two sibling output layers:
      1. fc + (K+1)-softmax: Discrete probability distribution per RoI $p = (p_0, .., p_K)$
      2. fc + bbox regressor: bbox regression offsets $t^k = (t^k_x, t^k_y, t^k_w, t^k_h)$, t^k: a scale -invariant translation and log-space height-width shift relative to an object proposal
    - Multi-task loss

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

where k^* is the true class label

1. $L_{cls}(p, k^*) = -\log p_{k^*}$: standard cross entropy/log loss
2. L_{loc} : $t^* = (t^*_x, t^*_y, t^*_w, t^*_h)$ true bbox regression target $t = (t_x, t_y, t_w, t_h)$ predicted tuple for class k

$$L_{loc}(t, t^*) = \sum_{i \in \{x,y,w,h\}} \text{smooth}_{L_1}(t_i, t^*_i)$$

$$\text{smooth}_{L_1}(x) = \begin{cases} 0.5x^2 & \text{if} |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases}$$

   smoothed L_1 loss : less sensitive to outliers (R-CNN L2 loss: requires significant tuning of learning rate, prevent exploding gradients)
3. hyper-parameter: λ (=1) normalize t^* to zero mean and unit variance

- **Mini-batch Sampling**
  128: 2 randomly sampled images with 64 PoI sampled from each image
  25% positive: IoU > 0.5
  75% background:IoU ∈ [0.1, 0.5)
  horizontally flipped with prob = 0.5
- **BP through RoI Pooling Layer**
  $\frac{\partial L}{\partial x} = \sum_{r \in R} \sum_{y \in r} [y \text{ polled } x] \frac{\partial L}{\partial y}$ (if x was argmax assigned to y during the pool)
- **SGD hyper-parameter**
  new fc for softmax is initialized by N(0, 0.01)
  new fc for bbox-reg is initilized by N(0. 0.001)
  base_lr: 0.001 weight_lr: 1 bias_lr: 2
  VOC07 VOC12: 30k-iter -> lr = 0.0001 10k-iter (larger dataset: momentum term 0.9 weight decay 0.0005)

- **Scale Invariance**
  scale invariance object detection : brute-force learning; using image pyramids [followed SPP]

Fast R-CNN detection

- R ~ 2k, Forward pass, assign detection confidence $\Pr(\text{calss} = k|r) = p_k$, ans NMS
- **Truncated SVD for faster detection**
  mAP ~ 0.3% down; speed ~ 30% up

number of RoI for detection is large -> time spent on fc
$W \sim U\Sigma_t V^T$ (U : u*t, Sigma_t: t*t, V: v*t)
Compression : $(Wx+b)$ fc -> $(\Sigma_t V^T x)$ fc + $(Ux+b)$ fc

Faster R-CNN

Paper: http://arxiv.org/abs/1506.01497
Caffe Project: https://github.com/ShaoqingRen/caffe

Reference: [blog1]() [blog2]()

Region Proposal Networks

RPN input: image of any size, output: rectangular object proposals with objectness score

- **Fully convolutional network**
  share computation with Fast R-CNN detection network(share conv layer)
- Slide on n*n conv feature map output by last shared conv layer(ZF 5conv, VGG 13conv)
  Sliding window mapped to a lower-dim vector(256-d ZF ， 512-d VGG) (n = 3 large recpt field)
  Fed into two sibling fc layers(1*1 conv): bbox-reg layer + box-cls layer
- **Translation-Invariant Anchors**
  At each sliding window loc, pridict k proposal: 4k outputs for reg layer, 2k outputs for cls layer (binary softmax).
  Anchor: centered at sliding window with scale and aspect ratio: ($128^2, 256^2, 512^2$; 1:2, 2:1, 1:1)
  For a conv feature map: $W * H * k$ (k=9 anchors) (2+4)*9 output layer
- **Loss function for Learning Region Proposal**
  positive label: the anchor has highest IoU with a gt-box or has an IoU>0.7 with any gt-box
  negative label: IoU<0.3 for all gt-box
  Objective function with multi-task loss: Similar to Fast R-CNN.

$$L(p_i, t_i) = L_{cls(p_i, p_i^*)} + \lambda p_i^* L_{reg}(t_i, t_i^*)$$

  where p_i^* is 1 if the anchor is labeled positive, and is 0 if the anchor is negative.
  $\lambda = 10$ bias towards better box location
- Optimization
  fcn trained by end-to-end by bp and sgd
  image-centric sampling strategy, sample 256 anchors in an image(Pos:neg = 1:1)
  new layer initialization ~ N(0, 0.01)
  tune ZFnet and conv3_1 and up for VGGnet, lr=0.001 for 60k batches, 0.0001 for 20k on PASCAL

- **Share Convolutional Features for Region Proposal and Objection Detection**
  Four-step training algorithm:
  1. Train RPN, initialized with ImageNet pre-trained model
  2. Train a separate detection network by Fast R-CNN using proposals generated by step-1 RPN, initialized by ImageNet pre-trained model
  3. Fix conv layer, fine-tune unique layers to RPN, initialized by detector network in Step2
  4. Fix conv layer, fine-tune fc-layers of Fast R-CNN

- **内容目录**

- 报告问题，建议
- 联系我们

添加新批注

保存 取消

在作者公开此批注前，只有你和作者可见。

保存 取消

修改 保存 取消 删除

- 私有
- 公开
- 删除

查看更早的 5 条回复

回复批注

×

通知

取消 确认