机器学习 计算机视觉 神经网络 深度学习（Deep Learning）

关注者 2293 被浏览 110129

如何评价rcnn、fast-rcnn和faster-rcnn这一系列方法？

或者相关的检测方法如OverFeat、SPPNet、SSD和较新的YOLO、R-FCN。

4 条评论 分享 邀请回答 ···

关注问题 写回答

21 个回答 默认排序

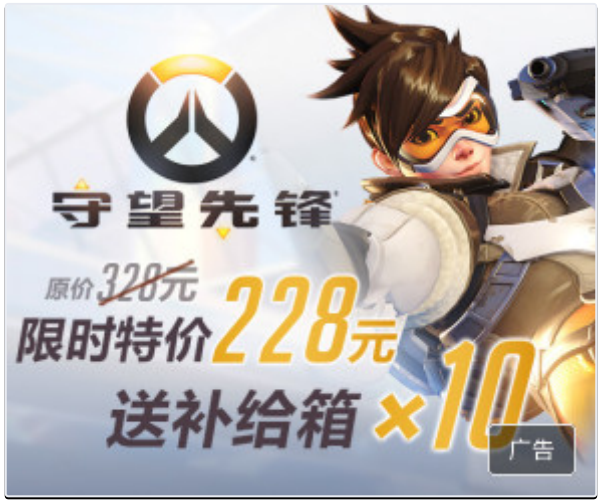
周博磊

深度学习（Deep Learning）、机器学习、人工智能 话题的优秀回答者

342 人赞同了该回答

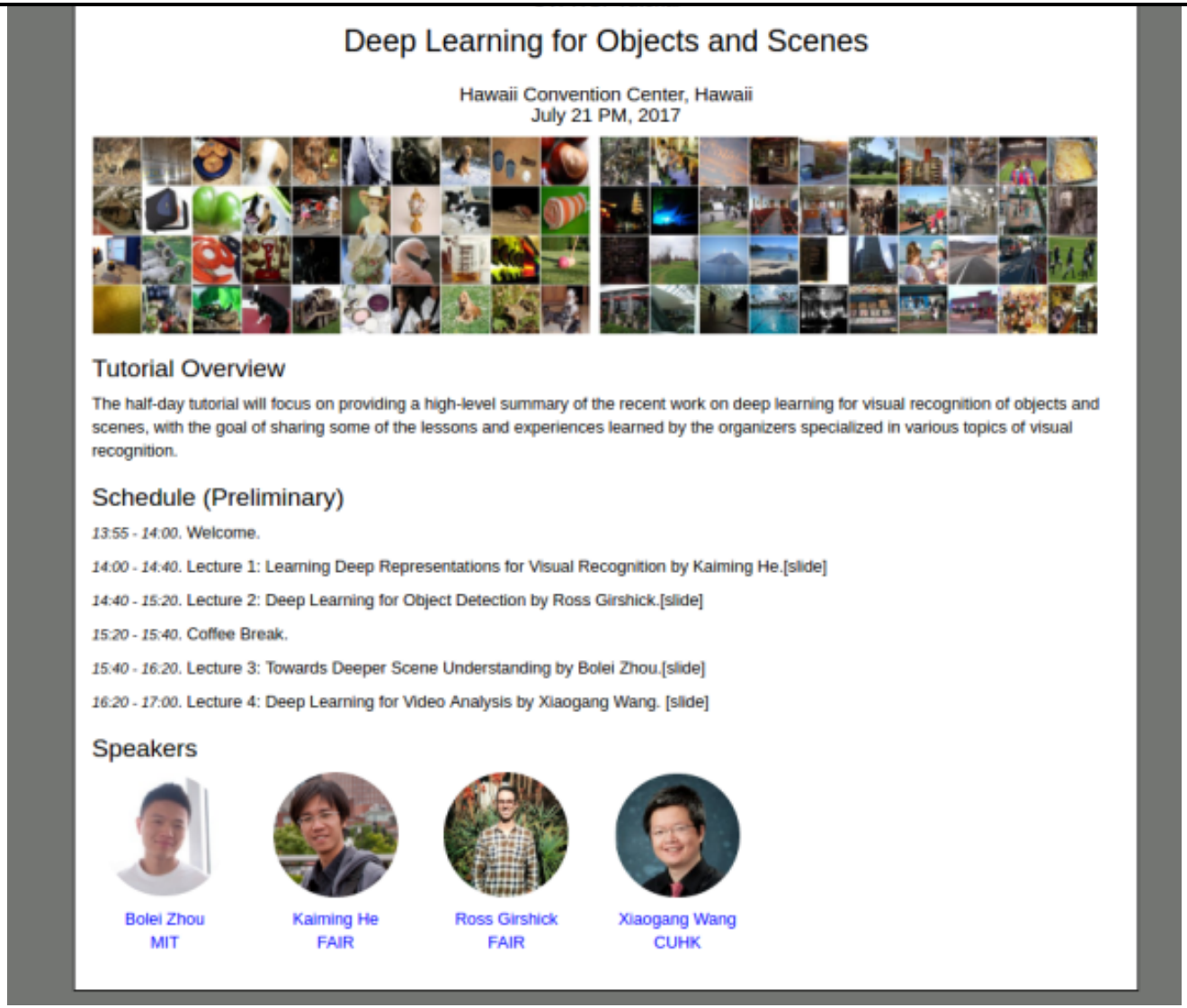
广告一发：今年CVPR'17，我们将组织一个Tutorial, 请到了你们的男神Ross Girshick和Kaiming He，外加Xiaogang Wang和我，将分别给个45分钟的讲座。Ross应该会将这几个object detection networks的关系理一遍，以及展望一下未来。Kaiming将梳理visual recognition网络结构的发展历史和未来，Xiaogang会介绍deep learning model在video detection上的应用，我会介绍deeper scene understanding && network interpretability 。敬请期待，各位朋友夏威夷见，预祝CVPR'17 submission顺利：）

链接：CVPR'17 Tutorial: Deep Learning for Objects and Scenes



相关问题

可否通过编程为《秘密花园》填充出和谐的颜色？ 52 个回答



Deep Learning for Objects and Scenes

Hawaii Convention Center, Hawaii
July 21 PM, 2017

Tutorial Overview

The half-day tutorial will focus on providing a high-level summary of the recent work on deep learning for visual recognition of objects and scenes, with the goal of sharing some of the lessons and experiences learned by the organizers specialized in various topics of visual recognition.

Schedule (Preliminary)

13:55 - 14:00. Welcome.

14:00 - 14:40. Lecture 1: Learning Deep Representations for Visual Recognition by Kaiming He.[slide]

14:40 - 15:20. Lecture 2: Deep Learning for Object Detection by Ross Girshick.[slide]

15:20 - 15:40. Coffee Break.

15:40 - 16:20. Lecture 3: Towards Deeper Scene Understanding by Bolei Zhou.[slide]

16:20 - 17:00. Lecture 4: Deep Learning for Video Analysis by Xiaogang Wang. [slide]

Speakers

Bolei Zhou
MIT

Kaiming He
FAIR

Ross Girshick
FAIR

Xiaogang Wang
CUHK

编辑于 2017-02-16

239 人赞同了该回答

首先膜拜RBG（Ross B. Girshick）大神，不仅学术牛，工程也牛，代码健壮，文档详细，clone下来就能跑。

断断续续接触detection几个月，将自己所知做个大致梳理，业余级新手，理解不对的地方还请指正。

传统的detection主流方法是DPM(Deformable parts models)， 在VOC2007上能到43%的mAP，虽然DPM和CNN看起来差别很大，但RBG大神说“Deformable Part Models are Convolutional Neural Networks”（**arxiv.org/abs/1409.5403**）。

CNN流行之后，Szegedy做过将detection问题作为回归问题的尝试（Deep Neural Networks for Object Detection），但是效果差强人意，在VOC2007上mAP只有30.5%。

既然回归方法效果不好，而CNN在分类问题上效果很好，那么为什么不把detection问题转化为分类问题呢？RBG的RCNN使用region proposal（具体用的是Selective Search Koen van de Sande: Segmentation as Selective Search for Object Recognition）来得到有可能得到是object的若干（大概10^3量级）图像局部区域，然后把这些区域分别输入到CNN中，得到区域的feature，再在feature上加上分类器，判断feature对应的区域是属于具体某类object还是背景。当然，RBG还用了区域对应的feature做了针对boundingbox的回归，用来修正预测的boundingbox的位置。RCNN在VOC2007上的mAP是58%左右。

RCNN存在着重复计算的问题（proposal的region有几千个，多数都是互相重叠，重叠部分会被多次重复提取feature），于是RBG借鉴Kaiming He的SPP-net的思路单枪匹马搞出了Fast-RCNN，跟RCNN最大区别就是Fast-RCNN将proposal的region映射到CNN的最后一层conv layer的feature map上，这样一张图片只需要提取一次feature，大大提高了速度，也由于流程的整合以及其他原因，在VOC2007上的mAP也提高到了68%。

netwrok实质是一个Fast-RCNN，这个Fast-RCNN输入的region proposal的是固定的（把一张图片划分成n*n个区域，每个区域给出9个不同ratio和scale的proposal），输出的是对输入的固定proposal是属于背景还是前景的判断和对齐位置的修正（regression）。Region proposal network的输出再输入第二个Fast-RCNN做更精细的分类和Boundingbox的位置修正。Fater-RCNN速度更快了，而且用VGG net作为feature extractor时在VOC2007上mAP能到73%。

个人觉得制约RCNN框架内的方法精度提升的瓶颈是将dectection问题转化成了对图片局部区域的分类问题后，不能充分利用图片局部object在整个图片中的context信息。可能RBG也意识到了这一点，所以**他最新的一篇文章YOLO**（arxiv.org/abs/1506.0264...）又回到了regression的方法下，这个方法效果很好，在VOC2007上mAP能到63.4%，而且速度非常快，能达到对视频的实时处理（油管视频：youtube.com/channel/UC7...），虽然不如Fast-RCNN，但是比传统的实时方法精度提升了太多，而且我觉得还有提升空间。

感谢有RGB这样的牛人们不断推动detection的进步&期待YOLO代码的公布

编辑于 2015-11-20

孔涛
深度学习（Deep Learning）话题的优秀回答者

258 人赞同了该回答

泻药，终于看到符合自己胃口的问题啦！怒答一枚。RCNN和Fast-RCNN简直是引领了最近两年目标检测的潮流！

提到这两个工作，不得不提到RBG大神rbg's home page，该大神在读博士的时候就因为dpm获得过pascal voc 的终身成就奖。博士后期间更是不断发力，RCNN和Fast-RCNN就是他的典型作品。

的kaiming组的SPPNET做了相应的加速。

Fast-RCNN：RCNN的加速版本，在我看来，这不仅仅是一个加速版本，其优点还包括：
(a) 首先，它提供了在caffe的框架下，如何定义自己的层/参数/结构的范例，这个范例的一个重要的应用是python layer的应用，我在这里支持多label的caffe，有比较好的实现吗？ - 孔涛的回答也提到了。
(2) training and testing end-to-end 这一点很重要，为了达到这一点其定义了ROIPooling层，因为有了这个，使得训练效果提升不少。
(3) 速度上的提升，因为有了Fast-RCNN，这种基于CNN的 real-time 的目标检测方法看到了希望，在工程上的实践也有了可能，后续也出现了诸如Faster-RCNN/YOLO等相关工作。

这个领域的脉络是：RCNN -> SPPNET -> Fast-RCNN -> Faster-RCNN。关于具体的细节，建议题主还是阅读相关文献吧。

这使我看到了目标检测领域的希望。起码有这么一部分人，他们不仅仅是为了几个百分点的提升，而是切实踏实在做贡献，相信不久这个领域会有新的工作出来。

以上纯属个人观点，欢迎批评指正。

参考：
[1] R-CNN: Girshick R, Donahue J, Darrell T, et al. Rich feature hierarchies for accurate object detection and semantic segmentation[C], CVPR, 2014.
[2] SPPNET: He K, Zhang X, Ren S, et al. Spatial pyramid pooling in deep convolutional networks for visual recognition[C], ECCV, 2014.
[3] Fast-RCNN: Girshick R. Fast R-CNN[C]. ICCV, 2015.
[4] Fater-RCNN: Ren S, He K, Girshick R, et al. Faster r-cnn: Towards real-time object detection with region proposal networks[C]. NIPS, 2015.
[5] YOLO: Redmon J, Divvala S, Girshick R, et al. You Only Look Once: Unified, Real-Time Object Detection[J]. arXiv preprint arXiv:1506.02640, 2015.

Oh233
Intuition and Insight

115 人赞同了该回答

是这样的，如果都用一句话来描述

RCNN 解决的是，“为什么不用CNN做classification呢？”
（但是这个方法相当于过一遍network出bounding box，再过另一个出label，原文写的很不“elegant”

Fast-RCNN 解决的是，“为什么不一起输出bounding box和label呢？”
（但是这个时候用selective search generate regional proposal的时间实在太长了

Faster-RCNN 解决的是，“为什么还要用selective search呢？”

于是就达到了real-time。开山之作确实是开山之作，但是也是顺应了“Deep learning 搞一切vision”这一潮流吧。

发布于 2015-11-19

▲ 115 ▼ 10 条评论 分享 收藏 感谢

iker peng
炒蒜苗都没有了味道

52 人赞同了该回答

直接画一张图吧：不过 YOLO出了第二版了，亲测很牛逼！地址：YOLO: Real-Time Object Detection

个人觉得，分析比较Faster Yolo SSD这几种算法，有一个问题要先回答，Yolo SSD为什么快？

最主要的原因还是提proposal（最后输出将全连接换成全卷积也是一点）。其实总结起来我认为有两种方式：1.RPN，2. 暴力划分。RPN的设计相当于是一个sliding window 对最后的特征图每一个位置都进行了估计，由此找出anchor上面不同变换的proposal，设计非常经典，代价就是sliding window的代价。相比较 yolo比较暴力 ，直接划为7*7的网格，估计以网格为中心两个位置也就是总共98个”proposal“。快的很明显，精度和格子的大小有关。SSD则是结合：不同layer输出的输出的不同尺度的 Feature Map提出来，划格子，多种尺度的格子，在格子上提“anchor”。结果显而易见。

还需要说明一个核心： 目前虽然已经有更多的RCNN，但是Faster RCNN当中的RPN仍然是一个经典的设计。下面来说一下RPN：（当然你也可以将YOLO和SSD看作是一种RPN的设计）

在Faster RCNN当中，一张大小为224*224的图片经过前面的5个卷积层，输出256张大小为13*13的 特征图（你也可以理解为一张13*13*256大小的特征图，256表示通道数）。接下来将其输入到RPN网络，输出可能存在目标的reign WHk个（其中WH是特征图的大小，k是anchor的个数）。

实际上，**这个RPN由两部分构成：一个卷积层，一对全连接层**分别输出分类结果（cls layer）以及坐标回归结果（reg layer）。卷积层：stride为1，卷积核大小为3*3，输出256张特征图（这一层实际参数为3*3*256*256）。相当于一个sliding window 探索输入特征图的每一个3*3的区域位置。当这个13*13*256特征图输入到RPN网络以后，**通过卷积层得到13*13个 256特征图。也就是169个256维的特征向量，每一个对应一个3*3的区域位置，每一个位置提供9个anchor**。于是，对于每一个256维的特征，经过**一对** 全连接网络（也可以是1*1的卷积核的卷积网络），一个输出 前景还是

编辑于 2017-02-17

晓雷

67 人赞同了该回答

在个人知乎专栏里写了一个系列：

- RCNN-将CNN引入目标检测的开山之作
- SPPNet-引入空间金字塔池化改进RCNN
- Fast R-CNN
- Faster R-CNN
- 图解YOLO
- SSD
- YOLO2

晓雷机器学习笔记

发布于 2017-02-22

周新一

27 人赞同了该回答

RCNN的一个出发点在文章第一句话：features matter。传统的物体检测使用hand-engineered的特征。如HOG特征可以表示为filter(convolutional)+gating(nonlinear)+pooling+normalization(LRN)，很多传统特征可以用图片经过CNN后得到的feature map表示。并且，由于卷积操作具有平移不变形，feature map里不仅包含了物体的what信息，还包含着物体的where信息。因此可以利用CNN特征来代替传统特征，对特征用SVM得到分类结果，同时可以对特征作回归得到更精确的位置。

SPP首先解决传统的CNN输入需要图片长宽固定的问题（这个问题在基于区域的输入上更为明显），把原来的“从原图上截取区域”转化为“从feature map上截取区域”，有了这个操作，同一张图片上不同区域的物体检测可以共享同一个原图特征。

然而SPP的区域特征提取是在卷积层参数固定后才进行的，并且要额外进行SVM分类和区域框回归。Fast RCNN解决了两个技术问题：1）如何让区域的选择可导（提出ROI Pooling操作，区域特征选择像Max Pooling一样可导），2）如何让SGD高效（把同一张图的不同区域放在同一个batch里），并且把原来的区域框回归作为一个Multitask接在网络里（Smooth L1 norm），这样除了区域推荐外的全部任务可以在一个网络完成。

Faster RCNN把区域推荐也放在网络里完成（RPN）。这样整个框架都可以在一个网络里运行。

然而，Faster RCNN的训练为了保证卷积特征一致性需要分4步训练。RBG在Tutorial上表示已有让区域推荐可导（类似Spatial Transform Network中的sampler操作）的joint training。

编辑于 2015-12-22

知乎用户

R-CNN：针对区域提取做CNN的object detction。

SPPNet：针对不同尺寸输入图片，在CNN之后的Feature Map上进行相同维度的区域分割并Pooling，转化成相同尺度的向量。但是分类用的是SVM。

Fast R-CNN：区域提取转移到Feature Map之后做，这样不用对所有的区域进行单独的CNN Forward步骤。同时最终一起回归bounding box和类别。

1.为什么做Faster-rcnn

a. SPPnet 和 Fast R-CNN 已经减少了detection步骤的执行时间，只剩下region proposal成为瓶颈

b. 因此提出了 Region Proposal Network(RPN) 用来提取检测区域，并且和整个检测网络共享卷积部分的特征。RPN同时训练区域的边界和objectness score(理解为是否可信存在oject)。

c. 最终把RPN和Fast-RCNN合并在一起，用了“attention” mechanisms(其实就是说共享这事)。在VOC数据集上可以做到每张图只提300个proposals(Fast-RCNN用selective search是2000个)。

2.继续阐述现存问题

Fast-RCNN已经做到了Region Proposal在卷积以后，这样大大减少了卷积作用的次数，全图的Feature Map提取同样只是做一次。目前最大的问题就是 Selective Search是耗时的瓶颈(CPU上大概2s一张)。尤其是这个方法没有应用在GPU上，优化比较难。RPN共享网络，可以做到每张10ms，并且一起在GPU上进行了。

3.具体网络构建

实现就是在RPN和Fast-RCNN之间选择性切换，实际各训练了两次。最终结果，即使选择了VGG里最复杂的网络，仍然可以做到5fps的速度。

实现的时候，前面feature map之前的网络是shareable convolutional layers(VGG有13层，ZF有5层，不同的模型不一样)。RPN的具体实现就不说了，3x3尺寸的filter，卷积一层，后面几个FC，假如每个点上proposal k种不同的尺度和比例(论文里定义k=9)，输出就有4*k的尺度信息(位置2维，长宽各一维)和2*k的判断信息(是否是object)。RPN的一个重大优点在于translation invariant，因为是针对各个点周围3X3进行独立卷积的，后面的 RPN整个网络的权重又是共享的。而如果MultiBox方法的话，核心k-means很难保有这种独立性，毕竟object的位置在图上都是随机的。另外现有的Region Proposal方法基本上是构建image pyramid或者是filter pyramid的基础上，这里就是真的实现了Single filter，Single scale。

4.LOSS定义和训练过程

Loss Function定义包括几个部分，RPN训练的尺度信息的边框(bounding box)部分，每个点对应的k个anchors里和ground truth IoU的overlap最大的标记为正，或者与任何IoU的overlap超过0.7也标记为正。每一个ground truth的box可能会标记给多个anchor。如果anchor与IoU的overlap小于0.3将

区域各进行k个不同权重的scale和size的训练。 RPN训练用的是每个batch来自单张图的proposal。但是各个图占主要部分的negative samples会产生很大的bias。于是每次每张图选择256个随机的anchors，并且保证postive和negative的anchors比例为1:1。RPN本身的部分高斯随机初始化而来，共享部分来自ImageNet之前的训练结果。后面decay，learning rate什么的就不详细说了。训练过程就是固定一部分，训练一部分，RPN和Fast-RCNN交叉着来，定义了很多种，Caffe里实现的，也不具体说了。

首先膜拜RBG大神，这周以前我是不知道这个人的，之前我一直想用移动机器人带摄像头做一些实时object detection的任务，其实自动驾驶上很多人在做了，行人检测什么的。但是我对CNN的作用还一直停留在全图做image classification上。万万没想到，RBG一个人把DPM，以及后来整个object detection的所有任务链全部用CNN实现了(也不能说一个人，还有MSRA的几个华人大牛)。

同时之前我对Caffe的认识也一直停留在C++和命令行接口调用上，今天看了一下py-faster-rcnn的源代码，简直对Caffe有了全新的认识(RBG毕竟也参与了Caffe的开发)。希望早日把源代码搞透，然后试着把Faster-RCNN用在检测其他特定物体的任务上。(源代码学习，个人Blog缓慢更新中Faster R-CNN 论文笔记 · Tai Lei Home Page，坑太大，一时半会填不满)。

编辑于 2016-10-16

午后阳光
理想是养一只叫柯西的柯基，然后和他数月亮看星星

10 人赞同了该回答

fast rcnn借鉴了spp net，只把整张图过一遍cnn，然后在roi layer把proposal对应到feature map上，速度提升非常大

faster rcnn主要是认为selective search只能跑在cpu，不能跑在gpu上，所以提出了rpn来得到proposal。一般selective search会产生2000个proposal，但rpn就几百个。但是个人经验faster rcnn训练时间长。

小白一枚，希望大神指正。宿舍网sb了，手机打的，将就看吧

编辑于 2015-11-20

李蓉
高校教师

8 人赞同了该回答

其实R-CNN的思路就是我们大家的思路，先做proposal，然后对每个proposal进行分类时利用CNN来分类，我们做某个特定目标的检测时，也是这个思路；fast R-CNN巧妙之处在于它回避了R-CNN要对每个proposal都使用一遍CNN，这也是造成速度慢的重要原因，而CNN在此主要用来提特征，因此fast R-CNN直接对全图进行卷积，获得特征图，再将proposal的位置映射到feature map中从而得到每个proposal所对应的特征，由于没有对原图的尺寸进行限制，所以又添加了ROI pooling层对feature维数归一化，再用这些特征训练几个linear SVM做判别；但是这个方法由于也还是需要事先提取proposal，而所有后续的操作也都是基于这些proposal的，所以要求proposal算法要快而且召回率尽可能高；faster R-CNN针对fast R-CNN的局限又进行了改进，主要就在于proposal上面，faster算法采用一个全卷积网络用作为region proposal网络，用该网络在feature map上滑动，并在每个位置引入多尺度多长宽比的anchor窗口来捕捉未知尺寸和ratio rate的目标；RPN网络的训练目标也是一个多任务的目标，同时学习分类和回归目标的包围盒；同时，还提出了一个四步的训练方法，使得RPN网络和fast-RCNN网络能够共享卷积层，这个四步的方法很巧妙，开始时分别训练RPN和fast-RCNN网络，然后在第三步，利用精调的方法，使用检测网络fast-RCNN来初始化RPN

同层的参数，这样，最终所训练得到的两个网络，是共享卷积层参数的。

发布于 2016-09-25

TTKK

right now

仅仅做个补充，最近Yolo V2出了，效果至少mAP上来说非常好：

YOLO: Real-Time Object Detection

关于YOLO V1, Fast(er) RCNN, Region proposal之类的资料，CS231n Lecture 8 - Localization and Detection大概讲了下基本思路和各个算法的优势和劣势。

编辑于 2017-02-16

开膛手水货

judge me before you even know me?

14 人赞同了该回答

RCNN：用SS去选框，CNN提特征，SVM分类。BB盒回归。

fast: 吐槽上面的stage各自玩各自的，就统一起来(实际上只统一了后面三个步骤)，最大的改进在"在训练过程中，SGD的mini-batch选取是有“层次的”，同一张图片上的ROI在BP的时候会使用相同的原始图像。举个例子，当N=2，R=128的时候，相当于只BP了2张图像（但实际上是128个ROI），这比传统的RCNN、SPPnet直接使用128个ROI 要快上64倍。感觉这个改进思路很简单也

Faster: 吐槽SS太慢，丫的，也用CNN给你整进去，这样就更快了。

编辑于 2015-11-26

董文博
云计算，机器学习

3 人赞同了该回答

基于深度学习的目标检测研究进展_深度学习大讲堂_传送门

来发个链接

编辑于 2017-01-23

Shrine Chen
keep digging

6 人赞同了该回答

对于输入图像，RCNN 方法先用区域生成方法生成 2000 个左右的候选区域，几乎涵盖了图像中物体可能出现所有位置，减少了需要搜索的空间，之后用训练好的 CNN 模型在这些区域上提取特征，最后用线性 SVM 进行分类，所使用的 CNN 模型可以根据分类效果进一步调节参数

编辑于 2016-02-27

MasterPa
人工智障行业纺织工

4 人赞同了该回答

从《river flows in you》到《一人我饮酒醉》就是 rcnn 到 fast-rcnn 再到 faster-rcnn。

By @时冰蓝

发布于 2017-02-20

苑斌斌

1 人赞同了该回答

只知道当时写作业用的是caffe，并且用了fast cnn的库。
就感觉fast cnn特征提取要快不少，当然一次也要提取不少特征。

发布于 2016-12-14

6 人赞同了该回答

瞄到这题…顺道贴个当年的阅读笔记…飘走～（为何知乎不支持markdown.....

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

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

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

Paper：cs.berkeley.edu/~rbg/#...
Tech report: arxiv.org/pdf/1311.2524...
Project：github.com/rbgirshick/r...
Slides: cs.berkeley.edu/~rbg/sl...

Referrence: a [blog](zhangliliang.com/2014/0...)

object detection system

Three modules:
1. Generate region proposals (~2k/image)
2. Compute CNN features

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)

- 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: arxiv.org/pdf/1504.0808...
Project: github.com/rbgirshick/f...

Referrence: [blog](zhangliliang.com/2015/0...)

Motivation

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)

- 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 $$L_{loc}(t, t^*) = \sum_{i \in \{x,y,w,h\}} \text{smooth}_{L_1}(t_i, t_i^*)$$ $$\text{smooth}_{L_1}(x) = \left\{ \begin{array}{cc} \text{0.5}x^2 & \text{if}|x|<1\\|x| - 0.5 & \text{otherwise}\end{array}\right.$$ 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

- __BP through RoI Pooling Layer__
$\displaystyle \frac{\partial L}{\partial x} = \sum_{r \in R} \sum_{y\in r}[y ~ \text{polled}~ x] \frac{\partial L}{\partial y}$ (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
$W \sim U\Sigma_tV^T$ (U : u*t, Sigma_t: t*t, V: v*t)
Compression : ($Wx+b$) fc -> ($\Sigma_tV^T x$) fc + ($Ux+b$) fc

Faster R-CNN

Paper: arxiv.org/abs/1506.0149...

Reference: [blog1](blog.csdn.net/anxiaoxi4...) [blog2](blog.cvmarcher.com/post...)

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)

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

编辑于 2017-02-15

Jason Zhang

代码的世界里匍匐前进

4 人赞同了该回答

这些都是近几年较新且效果显著的DL算法，通吃cv的图像检测，rcnn是加入了selective search 解决传统滑动窗口的时间复杂度问题，spp-net就是改进region proposal 的提取，fast rcnn 和faster rcnn 就是进一步希望做到实时性，端到端训练，包括加入ROI pooling layer和anchor机制，相对来说rcnn系列的这些算法精度都还可以，YOLO基于回归，简单且速度很快，SSD就是综合faster rcnn 的anchor和YOLO的回归。

发布于 2016-12-09

赵丽丽

深度学习 图像处理 计算机视觉 视频编解码

发布于 2017-02-16

bill pattern

densebox 有点像 faster rcnn 的另类进化，把proposal 也进化没了，直接给出。缺点是分类不如原来了。

发布于 2015-12-18

徐然
Research Scientist

1 人赞同了该回答

重要的是有开源的code。
开创性比pff的的dpm和alexnet都差不少。

发布于 2015-11-25

1 个回答被折叠（为什么？）