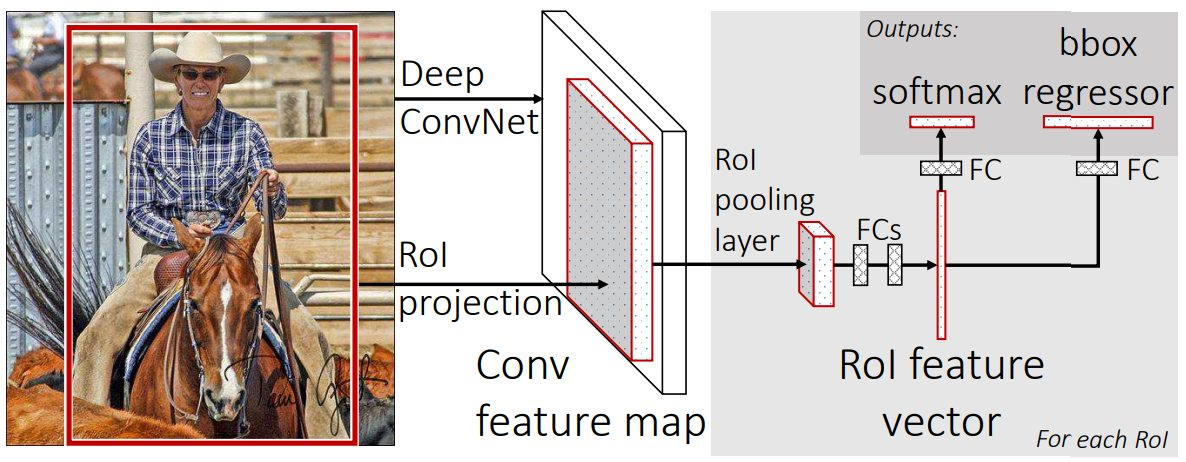
Fast R-CNN

论文引用：

Girshick R. Fast r-cnn[C]//Proceedings of the IEEE International Conference on Computer Vision(ICCV). 2015: 1440-1448.

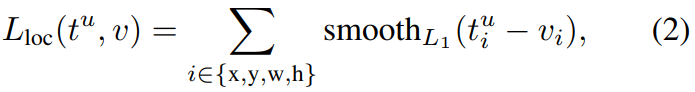
|  |  |
| --- | --- |
| ABSTRACT  This paper proposes a Fast Region-based Convolutional Network method (Fast R-CNN) for object detection. Fast R-CNN builds on previous work to efficiently classify object proposals using deep convolutional networks. Compared to previous work, Fast R-CNN employs several innovations to improve training and testing speed while also increasing detection accuracy. Fast R-CNN trains the very deep VGG16 network 9x faster than R-CNN, is 213x faster at test-time, and achieves a higher mAP on PASCAL VOC 2012. Compared to SPPnet, Fast R-CNN trains VGG16 3x faster, tests 10x faster, and is more accurate. Fast R-CNN is implemented in Python and C++ (using Caffe) and is available under the open-source MIT License at https://github.com/rbgirshick/fast-rcnn. | 摘要  本文提出了一种快速的基于区域的卷积网络方法（Fast R-CNN）用于目标检测。Fast R-CNN建立在以前使用的深卷积网络有效地分类目标的成果上。相比于之前的成果，Fast R-CNN采用了多项创新提高训练和测试速度同时提高检测精度。Fast R-CNN训练非常深的VGG16网络比R-CNN快9倍，测试时间快213倍，并在PASCAL VOC上得到更高的精度。与SPPnet相比，Fast R-CNN训练VGG16网络比他快3倍，测试速度快10倍，并且更准确。Fast R-CNN的Python和C ++（使用Caffe）实现以MIT开源许可证发布在：https://github.com/rbgirshick/fast-rcnn。 |
| 1. Introduction   Recently, deep ConvNets [14], [16] have significantly improved image classification [14] and object detection [9], [19] accuracy. Compared to image classification, object detection is a more challenging task that requires more complex methods to solve. Due to this complexity, current approaches (e.g., [9], [11], [19], [25]) train models in multi-stage pipelines that are slow and inelegant.  Complexity arises because detection requires the accurate localization of objects, creating two primary challenges. First, numerous candidate object locations (often called “proposals”) must be processed. Second, these candidates provide only rough localization that must be refined to achieve precise localization. Solutions to these problems often compromise speed, accuracy, or simplicity.  In this paper, we streamline the training process for state-of-the-art ConvNet-based object detectors [9], [11]. We propose a single-stage training algorithm that jointly learns to classify object proposals and refine their spatial locations.  The resulting method can train a very deep detection network (VGG 16 [20]) 9× faster than R-CNN [9] and 3× faster than SPPnet [11]. At runtime, the detection network processes images in 0.3s (excluding object proposal time) while achieving top accuracy on PASCAL VOC 2012 [7] with a mAP of 66% (vs. 62% for R-CNN).1   * 1. R-CNN and SPPnet   The Region-based Convolutional Network method (R-CNN) [9] achieves excellent object detection accuracy by using a deep ConvNet to classify object proposals. R-CNN, however, has notable drawbacks:   1. **Training is a Multi-Stage Pipeline**：RCNN first finetunes a ConvNet on object proposals using log loss. Then, it fits SVMs to ConvNet features. These SVMs act as object detectors, replacing the softmax classifier learnt by fine-tuning. In the third training stage, bounding-box regressors are learned. 2. **Training is Expensive in Space and Time**：For SVM and bounding-box regressor training, features are extracted from each object proposal in each image and written to disk. With very deep networks, such as VGG 16, this process takes 2.5 GPU-days for the 5k images of the VOC07 trainval set. These features require hundreds of gigabytes of storage. 3. **Object Detection is Slow**：At test-time, features are extracted from each object proposal in each test image. Detection with VGG16 takes 47s / image (on a GPU).   R-CNN is slow because it performs a ConvNet forward pass for each object proposal, without sharing computation. Spatial pyramid pooling networks (SPPnets) [11] were proposed to speed up R-CNN by sharing computation. The SPPnet method computes a convolutional feature map for the entire input image and then classifies each object proposal using a feature vector extracted from the shared feature map. Features are extracted for a proposal by max-pooling the portion of the feature map inside the proposal into a fixed-size output (e.g., 6×6). Multiple output sizes are pooled and then concatenated as in spatial pyramid pooling [15]. SPPnet accelerates R-CNN by 10 to 100× at test time. Training time is also reduced by 3× due to faster proposal feature extraction. SPPnet also has notable drawbacks. Like R-CNN, training is a multi-stage pipeline that involves extracting features, fine-tuning a network with log loss, training SVMs, and finally fitting bounding-box regressors. Features are also written to disk. But unlike R-CNN, the fine-tuning algorithm proposed in [11] cannot update the convolutional layers that precede the spatial pyramid pooling. Unsurprisingly, this limitation (fixed convolutional layers) limits the accuracy of very deep networks.   * 1. Contributions   We propose a new training algorithm that fixes the disadvantages of R-CNN and SPPnet, while improving on their speed and accuracy. We call this method Fast R-CNN because it's comparatively fast to train and test. The Fast R-CNN method has several advantages:  1. Higher detection quality (mAP) than R-CNN, SPPnet  2. Training is single-stage, using a multi-task loss  3. Training can update all network layers  4. No disk storage is required for feature caching  Fast R-CNN is written in Python and C++ (Caffe [13]) and is available under the open-source MIT License at <https://github.com/rbgirshick/fast-rCNN>. | 1. 简介   最近，深度卷积网络[14], [16]已经显著提高了图像分类[14]和目标检测[9], [19]的准确性。与图像分类相比，目标检测是一个更具挑战性的任务，需要更复杂的方法来解决。由于这种复杂性，当前的方法（例如，[9], [11], [19], [25]）采用多级流水线的方式训练模型，既慢且精度不高。  复杂性的产生是因为检测需要目标的精确定位，这就导致两个主要的难点。首先，必须处理大量候选目标位置（通常称为“提案”）。 第二，这些候选框仅提供粗略定位，其必须被精细化以实现精确定位。 这些问题的解决方案经常会影响速度，准确性或简单性。  在本文中，我们简化了最先进的基于卷积网络的目标检测器的训练过程[9], [11]。我们提出一个单阶段训练算法，联合学习候选框分类和修正他们的空间位置。  所得到的方法用来训练非常深的检测网络（例如VGG16） 比R-CNN快9倍，比SPPnet快3倍。在运行时，检测网络在PASCAL VOC 2012数据集上实现最高准确度，其中mAP为66％（R-CNN为62％），每张图像处理时间为0.3秒，不包括候选框的生成。  1.1 R-CNN and SPPnet  基于区域的卷积网络方法（RCNN）通过使用深度卷积网络来分类目标候选框，获得了很高的目标检测精度。然而，R-CNN具有显着的缺点：   1. **训练过程是多级流水线**：R-CNN首先使用目标候选框对卷积神经网络使用log损失进行微调。然后，它将卷积神经网络得到的特征送入SVM。这些SVM作为目标检测器，替代通过微调学习的softmax分类器。在第三个训练阶段，学习检测框回归。 2. **训练在时间和空间上是的开销很大**：于SVM和检测框回归训练，从每个图像中的每个目标候选框提取特征，并写入磁盘。对于非常深的网络，如VGG16，这个过程在单个GPU上需要2.5天（VOC07 trainval上的5k个图像）。这些特征需要数百GB的存储空间。 3. **目标检测速度很慢**：在测试时，从每个测试图像中的每个目标候选框提取特征。用VGG16网络检测目标每个图像需要47秒（在GPU上）。   R-CNN很慢是因为它为每个目标候选框进行卷积神经网络正向传递，而不共享计算。SPPnet5通过共享计算加速R-CNN。SPPnet5计算整个输入图像的卷积特征图，然后使用从共享特征图提取的特征向量来对每个候选框进行分类。通过最大池化将候选框内的特征图转化为固定大小的输出（例如，6×6）来提取针对候选框的特征。多个输出被池化，然后连接成空间金字塔池[15]。SPPnet在测试时将R-CNN加速10到100倍。由于更快的候选框特征提取训练时间也减少3倍。SPP网络也有显著的缺点。像R-CNN一样，训练过程是一个多级流水线，涉及提取特征，使用log损失对网络进行微调，训练SVM分类器，最后拟合检测框回归。特征也写入磁盘。但与R-CNN不同，在[11]中提出的微调算法不能更新在空间金字塔池之前的卷积层。不出所料，这种限制（固定的卷积层）限制了深层网络的精度。  1.2 贡献  我们提出一种新的训练算法，修正R-CNN和SPPnet的缺点，同时提高其速度和准确性。因为它能比较快地进行训练和测试，我们称之为Fast R-CNN。Fast RCNN方法有以下几个优点：   1. 比R-CNN和SPPnet具有更高的检测精度(mAP) 2. 训练是使用多任务损失的单阶段训练 3. 训练可以更新所有网络层参数 4. 不需要磁盘空间缓存特征   Fast R-CNN使用Python和C++(Caffe8)语言编写，以MIT开源许可证发布在：https://github.com/rbgirshick/fast-rcnn。 |
| 1. Fast R-CNN Architecture and Training   Fig. 1 illustrates the Fast R-CNN architecture. A Fast R-CNN network takes as input an entire image and a set of object proposals. The network first processes the whole image with several convolutional (conv) and max pooling layers to produce a conv feature map. Then, for each object proposal a region of interest (RoI) pooling layer extracts a fixed-length feature vector from the feature map. Each feature vector is fed into a sequence of fully connected (fc) layers that finally branch into two sibling output layers: one that produces softmax probability estimates over K object classes plus a catch-all “background” class and another layer that outputs four real-valued numbers for each of the K object classes. Each set of 4 values encodes refined bounding-box positions for one of the K classes. | 1. **Fast R-CNN架构与训练**   Fig. 1显示了Fast R-CNN 的网络架构。Fast R-CNN网络将整个图像和一组候选框作为输入。网络首先使用几个卷积层（conv）和最大池化层来处理整个图像，以产生卷积特征图。然后，对于每个候选框，RoI池化层从特征图中提取固定长度的特征向量。每个特征向量被送入一系列全连接（fc）层中，其最终分支成两个同级输出层 ：一个输出K个目标类别加上1个背景类别的Softmax概率估计，另一个为K个目标类别的每一个类别输出四个实数值。每组4个值表示K个类别的一个类别的检测框位置的修正。 |



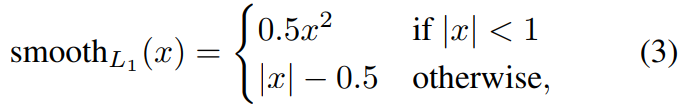
|  |  |
| --- | --- |
| *Figure 1. Fast R-CNN architecture. An input image and multiple regions of interest (rois) are input into a fully convolutional network. Each roi is pooled into a fixed-size feature map and then mapped to a feature vector by fully connected layers (fcs). The network has two output vectors per roi: softmax probabilities and per-class bounding-box regression offsets. The architecture is trained end-to-end with a multi-task loss.* | *图1.* *Fast R-CNN架构。输入图像和多个感兴趣区域（RoI）被输入到全卷积网络中。每个RoI被池化到固定大小的特征图中，然后通过全连接层（FC）映射到特征向量。网络对于每个RoI具有两个输出向量：Softmax概率和每类检测框回归偏移量。该架构是使用多任务损失端到端训练的。* |
| 2.1 The RoI Pooling Layer  The RoI pooling layer uses max pooling to convert the features inside any valid region of interest into a small feature map with a fixed spatial extent of H×W(e.g., 7×7), where H and W are layer hyper-parameters that are independent of any particular RoI. In this paper, an RoI is a rectangular window into a conv feature map. Each RoI is defined by a four-tuple (r, c, h, w) that specifies its top-left corner (r, c) and its height and width (h,w).  RoI max pooling works by dividing the h×w RoI window into an H×W grid of sub-windows of approximate size h/H×w/W and then max-pooling the values in each sub-window into the corresponding output grid cell. Pooling is applied independently to each feature map channel, as in standard max pooling. The RoI layer is simply the special-case of the spatial pyramid pooling layer used in SPPnets [11] in which there is only one pyramid level. We use the pooling sub-window calculation given in [11].  2.2 Initializing from Pre-Trained Networks  We experiment with three pre-trained ImageNet [4] networks, each with five max pooling layers and between five and thirteen conv layers (see Section 4.1 for network details). When a pre-trained network initializes a Fast R-CNN network, it undergoes three transformations.  First, the last max pooling layer is replaced by a RoI pooling layer that is configured by setting H and W to be compatible with the net's first fully connected layer (e.g., H=W=7 for VGG16).  Second, the network's last fully connected layer and softmax (which were trained for 1000-way ImageNet classification) are replaced with the two sibling layers described earlier (a fully connected layer and softmax over K+1 categories and category-specific bounding-box regressors).  Third, the network is modified to take two data inputs: a list of images and a list of RoIs in those images.  2.3 Fine-Tuning for Detection  Training all network weights with back-propagation is an important capability of Fast R-CNN. First, let's elucidate why SPPnet is unable to update weights below the spatial pyramid pooling layer.  The root cause is that back-propagation through the SPP layer is highly inefficient when each training sample (i.e. RoI) comes from a different image, which is exactly how R-CNN and SPPnet networks are trained. The inefficiency stems from the fact that each RoI may have a very large receptive field, often spanning the entire input image. Since the forward pass must process the entire receptive field, the training inputs are large (often the entire image).  We propose a more efficient training method that takes advantage of feature sharing during training. In Fast R-CNN training, stochastic gradient descent (SGD) mini-batches are sampled hierarchically, first by sampling N images and then by sampling R/NRoIs from each image. Critically, RoIs from the same image share computation and memory in the forward and backward passes. Making N small decreases mini-batch computation. For example, when using N=2 and R=128, the proposed training scheme is roughly 64× faster than sampling one RoI from 128 different images (i.e., the R-CNN and SPPnet strategy).  One concern over this strategy is it may cause slow training convergence because RoIs from the same image are correlated. This concern does not appear to be a practical issue and we achieve good results with N=2 and R=128 using fewer SGD iterations than R-CNN.  In addition to hierarchical sampling, Fast R-CNN uses a streamlined training process with one fine-tuning stage that jointly optimizes a softmax classifier and bounding-box regressors, rather than training a softmax classifier, SVMs, and regressors in three separate stages [9], [11]. The components of this procedure (the loss, mini-batch sampling strategy, back-propagation through RoI pooling layers, and SGD hyper-parameters) are described below.  **Multi-Task Loss**  A Fast R-CNN network has two sibling output layers. The first outputs a discrete probability distribution (per RoI), p=(p0, …,pK), over K+1 categories. As usual, p is computed by a softmax over the K+1 outputs of a fully connected layer. The second sibling layer outputs bounding-box regression offsets, tk=(tkx, tky, tkw, tkh), for each of the K object classes, indexed by k. We use the parameterization for tk given in [9], in which tk specifies a scale-invariant translation and log-space height/width shift relative to an object proposal.  Each training RoI is labeled with a ground-truth class u and a ground-truth bounding-box regression target v. We use a multi-task loss L on each labeled RoI to jointly train for classification and bounding-box regression: | 2.1 RoI池化层  RoI池化层使用最大池化将任何有效的RoI内的特征转换成具有H×W（例如，7×7）的固定空间范围的小特征图，其中H和W是层的超参数，独立于任何特定的RoI。在本文中，RoI是卷积特征图中的一个矩形窗口。 每个RoI由指定其左上角(r,c)及其高度和宽度(h,w)的四元组(r,c,h,w)定义。  RoI最大池化通过将大小为h×w的RoI窗口分割成H×W个网格，子窗口大小约为h/H×w/W，然后对每个子窗口执行最大池化，并将输出合并到相应的输出网格单元中。同标准的最大池化一样，池化操作独立应用于每个特征图通道。RoI层只是SPPnets[11]中使用的空间金字塔池层的特殊情况，其只有一个金字塔层。我们使用[11]中给出的池化子窗口计算方法。  2.2 从预训练网络初始化  我们实验了三个预训练的ImageNet [4]网络，每个网络有五个最大池化层和五到十三个卷积层（网络详细信息，请参见实验配置）。当预训练网络初始化fast R-CNN网络时，其经历三个变换。  首先，最后的最大池化层由RoI池层代替，其将H和W设置为与网络的第一个全连接层兼容的配置（例如，对于VGG16，H=W=7）。  然后，网络的最后一个全连接层和Softmax（其被训练用于1000类ImageNet分类）被替换为前面描述的两个同级层（全连接层和K+1个类别的Softmax以及类别特定的检测框回归）。  最后，网络被修改为采用两个数据输入：图像的列表和这些图像中的RoI的列表。  2.3 微调  用反向传播训练所有网络权重是Fast R-CNN的重要能力。首先，让我们阐明为什么SPPnet无法更新低于空间金字塔池化层的权重。  根本原因是当每个训练样本（即RoI）来自不同的图像时，通过SPP层的反向传播是非常低效的，这正是训练R-CNN和SPPnet网络的方法。低效的部分是因为每个RoI可能具有非常大的感受野，通常跨越整个输入图像。由于正向传播必须处理整个感受野，训练输入很大（通常是整个图像）。  我们提出了一种更有效的训练方法，利用训练期间的特征共享。在Fast RCNN网络训练中，随机梯度下降（SGD）的小批量是被分层采样的，首先采样N个图像，然后从每个图像采样R/N个 RoI。关键的是，来自同一图像的RoI在向前和向后传播中共享计算和内存。减小N，就减少了小批量的计算。例如，当N=2和R=128时，得到的训练方案比从128幅不同的图采样一个RoI（即R-CNN和SPPnet的策略）快64倍。  这个策略的一个令人担心的问题是它可能导致训练收敛变慢，因为来自相同图像的RoI是相关的。这个问题似乎在实际情况下并不存在，当N=2和R=128时，我们使用比R-CNN更少的SGD迭代就获得了良好的结果。  除了分层采样，Fast R-CNN使用了一个精细的训练过程，在微调阶段联合优化Softmax分类器和检测框回归，而不是分别在三个独立的阶段训练softmax分类器，SVM和回归器[9], [11]。 下面将详细描述该过程（损失，小批量采样策略，通过RoI池化层的反向传播和SGD超参数）。  **多任务损失**  Fast R-CNN网络具有两个同级输出层。第一个输出在K+1个类别上的离散概率分布(每个RoI)，p=(p0,…,pK)。 通常，通过全连接层的K+1个输出上的Softmax来计算p。第二个输出层输出检测框回归偏移，tk=(tkx, tky, tkw, tkh)，对于由k索引的K个类别中的每一个。 我们使用[9]中给出的tk的参数化，其中tk指定相对于候选框的尺度不变转换和对数空间高度/宽度移位。  每个训练的RoI用类真值u和检测框回归目标真值v标记。我们对每个标记的RoI使用多任务损失LL以联合训练分类和检测框回归： |



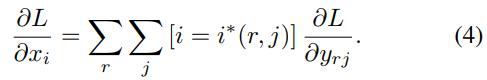
|  |  |
| --- | --- |
| in which Lcls(p, u)=−logpU is log loss for true class u.  The second task loss, Lloc, is defined over a tuple of true bounding-box regression targets for class u, v=(vx, vy, vw, vh), and a predicted tuple tu=(tux, tuy, tuw, tuh), again for class u. The Iverson bracket indicator function [u≥1] evaluates to 1 when u≥1 and 0 otherwise. By convention the catch-all background class is labeled u=0. For background RoIs there is no notion of a ground-truth bounding box and hence Lloc is ignored. For bounding-box regression, we use the loss | 其中Lcls(p, u)=−logpU，是类真值u的log损失。  对于类真值u，第二个损失Lloc是定义在检测框回归目标真值元组u, v=(vx, vy, vw, vh)和预测元组tu=(tux, tuy, tuw, tuh)上的损失。Iverson括号指示函数 [u≥1]当u≥1的时候为值1，否则为0。按照惯例，背景类标记为u=0。对于背景RoI，没有检测框真值的概念，因此Lloc被忽略。对于检测框回归，我们使用损失 |



|  |  |
| --- | --- |
| in which | 其中 |



|  |  |
| --- | --- |
| is a robust L1 loss that is less sensitive to outliers than the L2 loss used in R-CNN and SPPnet. When the regression targets are unbounded, training with L2 loss can require careful tuning of learning rates in order to prevent exploding gradients. Eq. 3 eliminates this sensitivity.  The hyper-parameter λ in Eq. 1 controls the balance between the two task losses. We normalize the ground-truth regression targets vi to have zero mean and unit variance. All experiments use λ=1.  We note that [6] uses a related loss to train a class-agnostic object proposal network. Different from our approach, [6] advocates for a two-network system that separates localization and classification. OverFeat [19], R-CNN [9], and SPPnet [11] also train classifiers and bounding-box localizers, however these methods use stage-wise training, which we show is suboptimal for Fast R-CNN (Section 5.1).  **Mini-Batch Sampling**  During fine-tuning, each SGD mini-batch is constructed from N=2 images, chosen uniformly at random (as is common practice, we actually iterate over permutations of the dataset). We use mini-batches of size R=128, sampling 64 RoIs from each image. As in [9], we take 25% of the RoIs from object proposals that have intersection over union (IoU) overlap with a ground-truth bounding box of at least 0.5. These RoIs comprise the examples labeled with a foreground object class, i.e. u≥1. The remaining RoIs are sampled from object proposals that have a maximum IoU with ground truth in the interval [0.1, 0.5), following [11]. These are the background examples and are labeled with u=0. The lower threshold of 0.1 appears to act as a heuristic for hard example mining [8]. During training, images are horizontally flipped with probability 0.5. No other data augmentation is used.  **Back-Propagation Through RoI Pooling Layers**  Back-propagation routes derivatives through the RoI pooling layer. For clarity, we assume only one image per mini-batch (N=1), though the extension to N>1 is straightforward because the forward pass treats all images independently.  Let xi∈R be the i-th activation input into the RoI pooling layer and let yrj be the layer's j−th output from the r-th RoI. The RoI pooling layer computes yrj=xi∗(r,j), in which i∗(r, j)=argmaxi′∈R(r,j. R(r, j) is the index set of inputs in the sub-window over which the output unit yrj max pools. A single xi may be assigned to several different outputs yrj.  The RoI pooling layer's backwards function computes partial derivative of the loss function with respect to each input variable xi by following the argmax switches: | 是鲁棒的L1损失，对于异常值比在R-CNN和SPPnet中使用的L2损失更不敏感。当回归目标无界时，具有L2损失的训练可能需要仔细调整学习速率，以防止爆炸梯度。公式(3)消除了这种敏感度。  公式(1)中的超参数λ控制两个任务损失之间的平衡。我们将回归目标真值vi归一化为具有零均值和单位方差。所有实验都使用λ=1。  我们注意到[6]使用相关损失来训练一个类别无关的目标候选网络。与我们的方法不同的是[6]倡导一个分离定位和分类的双网络系统。OverFeat[19]，R-CNN[9]和SPPnet[11]也训练分类器和检测框定位器，但是这些方法使用逐级训练，这对于Fast RCNN来说不是最好的选择(5.1节)。  **小批量采样**  在微调期间，每个SGD的小批量由N=2个图像构成，均匀地随机选择（如通常的做法，我们实际上迭代数据集的排列）。我们使用大小为R=128的小批量，从每个图像采样64个RoI。 如在[9]中，我们从候选框中获取25％的RoI，这些候选框与检测框真值的IoU至少为0.5。 这些RoI只包括用前景对象类标记的样本，即u≥1。 剩余的RoI从候选框中采样，该候选框与检测框真值的最大IoU在区间[0.1,0.5)上[11]。 这些是背景样本，并用u=0标记。0.1的阈值下限似乎充当难负样本重训练的启发式算法[8]。在训练期间，图像以概率0.5水平翻转。不使用其他数据增强。  **通过RoI池化层的反向传播**  反向传播通过RoI池化层。为了清楚起见，我们假设每个小批量(N=1)只有一个图像，扩展到N>1是显而易见的，因为前向传播独立地处理所有图像。  令xi∈R是到RoI池化层的第i个激活输入，并且令yrj是来自第r个RoI层的第j个输出。RoI池化层计算yrj=xi∗(r,j)，其中i∗(r, j)=argmaxi′∈R(r,j。R(r,j)是输出单元yrj最大池化的子窗口中的输入的索引集合。单个xi可以被分配给几个不同的输出yrj。  RoI池化层反向传播函数通过遵循argmax switches来计算关于每个输入变量xi的损失函数的偏导数： |



|  |  |
| --- | --- |
| In words, for each mini-batch RoI r and for each pooling output unit yrj, the partial derivative ∂L/∂yrj is accumulated if i is the argmax selected for yrj by max pooling. In back-propagation, the partial derivatives ∂L/∂yrj are already computed by the backwards function of the layer on top of the RoI pooling layer.  **SGD Hyper-Parameters**  The fully connected layers used for softmax classification and bounding-box regression are initialized from zero-mean Gaussian distributions with standard deviations 0.01 and 0.001, respectively. Biases are initialized to 0. All layers use a per-layer learning rate of 1 for weights and 2 for biases and a global learning rate of 0.001. When training on VOC07 or VOC12 trainval we run SGD for 30k mini-batch iterations, and then lower the learning rate to 0.0001 and train for another 10k iterations. When we train on larger datasets, we run SGD for more iterations, as described later. A momentum of 0.9 and parameter decay of 0.0005 (on weights and biases) are used.  2.4 Scale Invariance  We explore two ways of achieving scale invariant object detection: (1) via “brute force” learning and (2) by using image pyramids. These strategies follow the two approaches in [11]. In the brute-force approach, each image is processed at a pre-defined pixel size during both training and testing. The network must directly learn scale-invariant object detection from the training data.  The multi-scale approach, in contrast, provides approximate scale-invariance to the network through an image pyramid. At test-time, the image pyramid is used to approximately scale-normalize each object proposal. During multi-scale training, we randomly sample a pyramid scale each time an image is sampled, following [11], as a form of data augmentation. We experiment with multi-scale training for smaller networks only, due to GPU memory limits. | 换句话说，对于每个小批量RoI r和对于每个池化输出单元yrj，如果i是yrj通过最大池化选择的argmax，则将这个偏导数∂L/∂yrj积累下来。在反向传播中，偏导数∂L/∂yrj已经由RoI池化层顶部的层的反向传播函数计算。  **SGD超参数**  用于Softmax分类和检测框回归的全连接层的权重分别使用具有方差0.01和0.001的零均值高斯分布初始化。偏置初始化为0。所有层的权重学习率为1倍的全局学习率，偏置为2倍的全局学习率，全局学习率为0.001。 当对VOC07或VOC12 trainval训练时，我们运行SGD进行30k次小批量迭代，然后将学习率降低到0.0001，再训练10k次迭代。当我们训练更大的数据集，我们运行SGD更多的迭代，如下文所述。使用0.9的动量和0.0005的参数衰减（权重和偏置）。  2.4 尺度不变性  我们探索两种实现尺度不变对象检测的方法：（1）通过“brute force”学习和（2）通过使用图像金字塔。这些策略遵循[11]中的两种方法。 在“brute force”方法中，在训练和测试期间以预定义的像素大小处理每个图像。网络必须直接从训练数据学习尺度不变性目标检测。  相反，多尺度方法通过图像金字塔向网络提供近似尺度不变性。在测试时，图像金字塔用于大致缩放-规范化每个候选框。在多尺度训练期间，我们在每次图像采样时随机采样金字塔尺度，遵循[11]，作为数据增强的形式。由于GPU内存限制，我们只对较小的网络进行多尺度训练。 |
| 1. Fast R-CNN Detection   Once a Fast R-CNN network is fine-tuned, detection amounts to little more than running a forward pass (assuming object proposals are pre-computed). The network takes as input an image (or an image pyramid, encoded as a list of images) and a list of R object proposals to score. At test-time, R is typically around 2000, although we will consider cases in which it is larger (≈45k). When using an image pyramid, each RoI is assigned to the scale such that the scaled RoI is closest to 2242 pixels in area [11].  For each test RoI r, the forward pass outputs a class posterior probability distribution p and a set of predicted bounding-box offsets relative to r (each of the K classes gets its own refined bounding-box prediction). We assign a detection confidence to r for each object class k using the estimated probability Pr(class =k|r)≜pk. We then perform non-maximum suppression independently for each class using the algorithm and settings from R-CNN [9]. | 1. **Fast R-CNN检测**   一旦Fast R-CNN网络被微调完毕，检测相当于运行前向传播（假设候选框是预先计算的）。网络将图像（或图像金字塔，编码为图像列表）和待计算概率的R个候选框的列表作为输入。在测试的时候，R通常在2000左右，虽然我们将考虑将它变大（约45k）的情况。当使用图像金字塔时，每个RoI被缩放，使其最接近[11]中的2242个像素。  对于每个测试的RoI r，正向传播输出类别后验概率分布p和相对于r的预测的检测框框偏移集合（K个类别中的每一个获得其自己的精细检测框预测）。我们使用估计的概率Pr(class =k|r)≜pk为每个对象类别k分配r的检测置信度。然后，我们使用R-CNN算法的设置和对每个类别独立执行非最大抑制[9]。 |

1. References

[1] Beltrán J, Guindel C, Moreno F M, et al. Birdnet: a 3D object detection framework from LiDAR information[C]//2018 21st International Conference on Intelligent Transportation Systems (ITSC). IEEE, 2018: 3517-3523.

[2] Chen X, Kundu K, Zhang Z, et al. Monocular 3d object detection for autonomous driving[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2016: 2147-2156.

[3] Chen X, Kundu K, Zhu Y, et al. 3d object proposals for accurate object class detection[C]//Advances in Neural Information Processing Systems (NIPS). 2015: 424-432.

[4] Chen X, Kundu K, Zhu Y, et al. 3d object proposals using stereo imagery for accurate object class detection[J]. IEEE transactions on pattern analysis and machine intelligence (PAMI), 2017, 40(5): 1259-1272.

[5] Chen X, Ma H, Wan J, et al. Multi-view 3d object detection network for autonomous driving[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2017: 1907-1915.