Fully Convolutional Networks for Semantic Segmentation

Jonathan Long* Evan Shelhamer* Trevor Darrell
UC Berkeley
{jonlong,shelhamer,trevor}@cs.berkeley.edu

Abstract

Convolutional networks are powerful visual models that yield hierarchies of features. We show that convolutional networks by themselves, trained end-to-end, pixels-to-pixels, exceed the state-of-the-art in semantic segmentation. Our key insight is to build "fully convolutional" networks that take input of arbitrary size and produce correspondingly-sized output with efficient inference and learning. We define and detail the space of fully convolutional networks, explain their application to spatially dense prediction tasks, and draw connections to prior models. We adapt contemporary classification networks (AlexNet [19], the VGG net [31], and GoogLeNet [32]) into fully convolutional networks and transfer their learned representations by fine-tuning [4] to the segmentation task. We then define a novel architecture that combines semantic information from a deep, coarse layer with appearance information from a shallow, fine layer to produce accurate and detailed segmentations. Our fully convolutional network achieves state-of-the-art segmentation of PASCAL VOC (20% relative improvement to 62.2% mean IU on 2012), NYUDv2, and SIFT Flow, while inference takes less than one fifth of a second for a typical image.

1. Introduction

Convolutional networks are driving advances in recognition. Convnets are not only improving for whole-image classification [19, 31, 32], but also making progress on local tasks with structured output. These include advances in bounding box object detection [29, 12, 17], part and keypoint prediction [39, 24], and local correspondence [24, 9].

The natural next step in the progression from coarse to fine inference is to make a prediction at every pixel. Prior approaches have used convnets for semantic segmentation [27, 2, 8, 28, 16, 14, 11], in which each pixel is labeled with the class of its enclosing object or region, but with shortcomings that this work addresses.

*Authors contributed equally

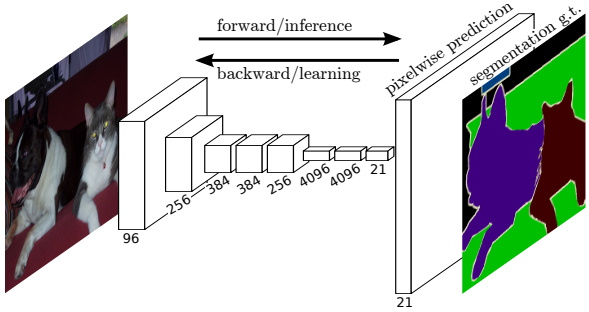

Figure 1. Fully convolutional networks can efficiently learn to make dense predictions for per-pixel tasks like semantic segmentation.

We show that a fully convolutional network (FCN), trained end-to-end, pixels-to-pixels on semantic segmentation exceeds the state-of-the-art without further machinery. To our knowledge, this is the first work to train FCNs end-to-end (1) for pixelwise prediction and (2) from supervised pre-training. Fully convolutional versions of existing networks predict dense outputs from arbitrary-sized inputs. Both learning and inference are performed whole-image-at-a-time by dense feedforward computation and backpropagation. In-network upsampling layers enable pixelwise prediction and learning in nets with subsampled pooling.

This method is efficient, both asymptotically and absolutely, and precludes the need for the complications in other works. Patchwise training is common [27, 2, 8, 28, 11], but lacks the efficiency of fully convolutional training. Our approach does not make use of pre- and post-processing complications, including superpixels [8, 16], proposals [16, 14], or post-hoc refinement by random fields or local classifiers [8, 16]. Our model transfers recent success in classification [19, 31, 32] to dense prediction by reinterpreting classification nets as fully convolutional and fine-tuning from their learned representations. In contrast, previous works have applied small convnets without supervised pre-training [8, 28, 27].

Semantic segmentation faces an inherent tension between semantics and location: global information resolves what while local information resolves where. Deep feature

用于语义分割的全卷积网络

乔纳森·朗* 埃文·谢尔哈默* 特雷弗·达雷尔 加州大学伯克利分校

{jonlong,shelhamer,trevor}@cs.berkeley.edu

摘要

Convolutional networks are powerful visual models that yield hierarchies of features. We show that convolutional networks by themselves, trained end-to-end, pixels-to-pixels, exceed the state-of-the-art in semantic segmentation. Our key insight is to build "fully convolutional" networks that take input of arbitrary size and produce correspondingly-sized output with efficient inference and learning. We define and detail the space of fully convolutional networks, explain their application to spatially dense prediction tasks, and draw connections to prior models. We adapt contemporary classification networks (AlexNet [19], the VGG net [31], and GoogLeNet [32]) into fully convolutional networks and transfer their learned representations by fine-tuning [4] to the segmentation task. We then define a novel architecture that combines semantic information from a deep, coarse layer with appearance information from a shallow, fine layer to produce accurate and detailed segmentations. Our fully convolutional network achieves state-of-the-art segmentation of PASCAL VOC (20% relative improvement to 62.2% mean IU on 2012), NYUDv2, and SIFT Flow, while inference takes less than one fifth of a second for a typical image.

1. 引言

卷积网络正在推动识别领域的进步。卷积网络不仅在整图分类方面持续改进[19, 31, 32]，还在具有结构化输出的局部任务中取得进展。这些进展包括边界框目标检测[29, 12, 17]、部件与关键点预测[39, 24]以及局部对应关系建模[24, 9]等领域。

从粗到细推理的自然下一步是在每个像素上进行预测。先前的方法已使用卷积网络进行语义分割[27, 2, 8, 28, 16, 14, 11]，即用其所属物体或区域的类别标记每个像素，但存在本工作所解决的缺陷。

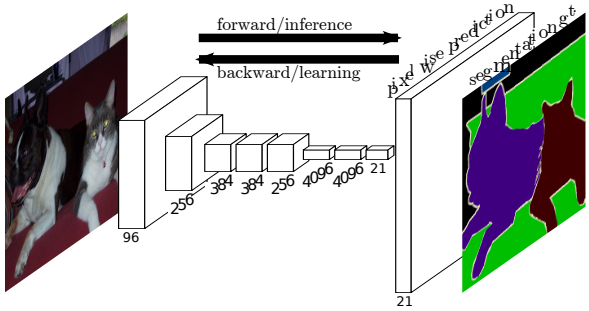

图1. 全卷积网络能够高效学习，为像素级任务（如语义分割）进行密集预测。

我们证明，一个全卷积网络（FCN）在语义分割任务上以端到端、像素到像素的方式进行训练，无需额外机制即可超越现有技术水平。据我们所知，这是首次（1）为像素级预测而（2）从有监督预训练出发，端到端地训练FCN的工作。现有网络的全卷积版本能够从任意尺寸的输入中预测密集输出。学习和推断均通过密集的前向计算与反向传播以整图一次处理的方式完成。网络内的上采样层使得在具有下采样池化的网络中能够进行像素级预测和学习。

该方法在渐近和绝对意义上均高效，且避免了其他工作中复杂的处理步骤。分块训练是常见做法[27, 2, 8, 28, 11]，但效率不及全卷积训练。我们的方法无需前后处理环节的复杂操作，例如超像素分割[8, 16]、候选区域生成[16, 14]，或通过随机场与局部分类器进行后处理优化[8, 16]。通过将分类网络重新解读为全卷积结构并基于其已学习的表征进行微调，我们的模型将近期在分类任务中的成功[19, 31, 32]迁移至密集预测任务。相比之下，先前的研究多采用未经监督预训练的小型卷积网络[8, 28, 27]。

语义分割面临语义与位置之间的固有张力：全局信息解决"是什么"，而局部信息解决"在哪里"。深度特征

*Authors contributed equally

hierarchies jointly encode location and semantics in a local-to-global pyramid. We define a novel "skip" architecture to combine deep, coarse, semantic information and shallow, fine, appearance information in Section 4.2 (see Figure 3).

In the next section, we review related work on deep classification nets, FCNs, and recent approaches to semantic segmentation using convnets. The following sections explain FCN design and dense prediction tradeoffs, introduce our architecture with in-network upsampling and multi-layer combinations, and describe our experimental framework. Finally, we demonstrate state-of-the-art results on PASCAL VOC 2011-2, NYUDv2, and SIFT Flow.

2. Related work

Our approach draws on recent successes of deep nets for image classification [19, 31, 32] and transfer learning [4, 38]. Transfer was first demonstrated on various visual recognition tasks [4, 38], then on detection, and on both instance and semantic segmentation in hybrid proposal-classifier models [12, 16, 14]. We now re-architect and fine-tune classification nets to direct, dense prediction of semantic segmentation. We chart the space of FCNs and situate prior models, both historical and recent, in this framework.

Fully convolutional networks To our knowledge, the idea of extending a convnet to arbitrary-sized inputs first appeared in Matan *et al*. [25], which extended the classic LeNet [21] to recognize strings of digits. Because their net was limited to one-dimensional input strings, Matan *et al*. used Viterbi decoding to obtain their outputs. Wolf and Platt [37] expand convnet outputs to 2-dimensional maps of detection scores for the four corners of postal address blocks. Both of these historical works do inference and learning fully convolutionally for detection. Ning *et al*. [27] define a convnet for coarse multiclass segmentation of *C. elegans* tissues with fully convolutional inference.

Fully convolutional computation has also been exploited in the present era of many-layered nets. Sliding window detection by Sermanet *et al*. [29], semantic segmentation by Pinheiro and Collobert [28], and image restoration by Eigen *et al*. [5] do fully convolutional inference. Fully convolutional training is rare, but used effectively by Tompson *et al*. [35] to learn an end-to-end part detector and spatial model for pose estimation, although they do not exposit on or analyze this method.

Alternatively, He *et al*. [17] discard the non-convolutional portion of classification nets to make a feature extractor. They combine proposals and spatial pyramid pooling to yield a localized, fixed-length feature for classification. While fast and effective, this hybrid model cannot be learned end-to-end.

Dense prediction with convnets Several recent works have applied convnets to dense prediction problems, including semantic segmentation by Ning *et al*. [27], Farabet *et al*. [8], and Pinheiro and Collobert [28]; boundary prediction for electron microscopy by Ciresan *et al*. [2] and for natural images by a hybrid neural net/nearest neighbor model by Ganin and Lempitsky [11]; and image restoration and depth estimation by Eigen *et al*. [5, 6]. Common elements of these approaches include

- small models restricting capacity and receptive fields;
- patchwise training [27, 2, 8, 28, 11];
- post-processing by superpixel projection, random field regularization, filtering, or local classification [8, 2, 11];
- input shifting and output interlacing for dense output [28, 11] as introduced by OverFeat [29];
- multi-scale pyramid processing [8, 28, 11];
- saturating $\tanh$ nonlinearities [8, 5, 28]; and
- ensembles [2, 11],

whereas our method does without this machinery. However, we do study patchwise training 3.4 and "shift-and-stitch" dense output 3.2 from the perspective of FCNs. We also discuss in-network upsampling 3.3, of which the fully connected prediction by Eigen *et al*. [6] is a special case.

Unlike these existing methods, we adapt and extend deep classification architectures, using image classification as supervised pre-training, and fine-tune fully convolutionally to learn simply and efficiently from whole image inputs and whole image ground thruths.

Hariharan *et al*. [16] and Gupta *et al*. [14] likewise adapt deep classification nets to semantic segmentation, but do so in hybrid proposal-classifier models. These approaches fine-tune an R-CNN system [12] by sampling bounding boxes and/or region proposals for detection, semantic segmentation, and instance segmentation. Neither method is learned end-to-end.

They achieve state-of-the-art results on PASCAL VOC segmentation and NYUDv2 segmentation respectively, so we directly compare our standalone, end-to-end FCN to their semantic segmentation results in Section 5.

3. Fully convolutional networks

Each layer of data in a convnet is a three-dimensional array of size $h \times w \times d$, where h and w are spatial dimensions, and d is the feature or channel dimension. The first layer is the image, with pixel size $h \times w$, and d color channels. Locations in higher layers correspond to the locations in the image they are path-connected to, which are called their *receptive fields*.

Convnets are built on translation invariance. Their basic components (convolution, pooling, and activation functions) operate on local input regions, and depend only on *relative* spatial coordinates. Writing $\mathbf{x}_{ij}$ for the data vector at location (i, j) in a particular layer, and $\mathbf{y}_{ij}$ for the follow-

层次结构在局部到全局的金字塔中共同编码位置和语义。我们在第4.2节定义了一种新颖的“跳跃”架构，将深层的、粗糙的语义信息与浅层的、精细的表观信息相结合（见图3）。

在下一节中，我们将回顾关于深度分类网络、全卷积网络（FCN）以及近期利用卷积网络进行语义分割的相关工作。随后的章节将阐述FCN的设计与密集预测的权衡，介绍我们采用网络内上采样和多层组合的架构，并描述我们的实验框架。最后，我们在PASCAL VOC 2011-2、NYUDv2和SIFT Flow数据集上展示了最先进的结果。

2. 相关工作

我们的方法借鉴了深度网络在图像分类[19, 31, 32]和迁移学习[4, 38]领域的最新成果。迁移学习最初在各种视觉识别任务[4, 38]中得到验证，随后扩展至检测任务，并在混合建议-分类器模型[12, 16, 14]中同时应用于实例分割与语义分割。我们现通过重构并微调分类网络，将其直接用于密集的语义分割预测。我们系统梳理了全卷积网络的设计空间，并将历史与近期的相关模型置于这一框架中进行定位。

全卷积网络 据我们所知，将卷积网络扩展至任意尺寸输入的想法最早出现在Matan *et al*。[25] 将经典的LeNet [21] 扩展用于识别数字串。由于他们的网络仅限于一维输入字符串，Matan *et al* 采用维特比解码来获取输出。Wolf和Platt [37] 将卷积网络输出扩展为邮政地址块四个角点检测分数的二维映射图。这两项早期研究均以全卷积方式进行检测的推断与学习。Ning *et al* [27] 提出一种卷积网络，通过全卷积推断实现*C. elegans*组织的粗粒度多类分割。

全卷积计算在当今多层网络时代也得到了应用。Sermanet *et al* [29] 的滑动窗口检测、Pinheiro 与 Collobert [28] 的语义分割，以及 Eigen *et al* [5] 的图像修复均采用了全卷积推理。全卷积训练虽较为少见，但 Tompson *et al* [35] 已将其有效用于端到端部位检测器与姿态估计空间模型的学习，尽管他们未对该方法进行详细阐述或分析。

另一种方法是，He *et al*。[17] 舍弃了分类网络中的非卷积部分，构建了一个特征提取器。他们通过结合候选区域与空间金字塔池化，生成了用于分类的局部化定长特征。尽管这种混合模型快速有效，却无法实现端到端学习。

使用卷积网络进行密集预测 最近的一些研究已将卷积网络应用于密集预测问题，包括Ning *et al*的语义分割。[27], Farabet *et al*。[8]以及Pinheiro和Collobert[28]；Ciresan *et al*用于电子显微镜的边界预测[2]，以及Ganin和Lempitsky[11]通过混合神经网络/最近邻模型用于自然图像的边界预测；还有Eigen *et al*用于图像恢复和深度估计[5, 6]。这些方法的共同要素包括

- 小型模型限制了容量和感受野；
- 分块训练 [27, 2, 8, 28, 11]；
- 通过超像素投影、随机场正则化、滤波或局部分类进行后处理[8, 2, 11]；

- 输入平移和输出交错以实现密集输出[28, 11]，如OverFeat[29]所引入；
- 多尺度金字塔处理 [8, 28, 11]；
- 饱和tanh非线性[8, 5, 28]；以及
- 集成方法 [2, 11]，

而我们的方法无需这种机制。不过，我们确实从全卷积网络的角度研究了分块训练（3.4节）和“平移缝合”密集输出（3.2节）。我们还讨论了网络内上采样（3.3节），其中Eigen *et al* [6]的全连接预测是该方法的一个特例。

与现有方法不同，我们调整并扩展了深度分类架构，以图像分类作为监督式预训练，并通过全卷积方式进行微调，从而简单高效地从完整图像输入和完整图像真实标注中学习。

Hariharan *et al*。 [16] 和 Gupta *et al*。 [14] 同样将深度分类网络适配于语义分割，但采用的是混合提议-分类器模型。这些方法通过对检测、语义分割和实例分割任务采样边界框和/或区域提议，对 R-CNN 系统 [12] 进行微调。两种方法均未实现端到端学习。

它们在PASCAL VOC分割和NYUDv2分割上分别取得了最先进的结果，因此我们在第5节中直接将我们独立的端到端FCN与它们的语义分割结果进行比较。

3. 全卷积网络

卷积网络的每一层数据都是一个大小为 $h \times w \times d$ 的三维数组，其中 h 和 w 是空间维度，d 是特征或通道维度。第一层是图像，其像素尺寸为 $h \times w$，并包含 d 个颜色通道。更高层中的位置对应于它们在图像中通过路径连接的位置，这些位置被称为它们的 *receptive fields*。

卷积网络建立在平移不变性的基础上。它们的基本组件（卷积、池化和激活函数）在局部输入区域上操作，并且仅依赖于*relative*空间坐标。将特定层中位置(i, j)的数据向量记为$\mathbf{x}_{ij}$，并将后续层记为$\mathbf{y}_{ij}$——

ing layer, these functions compute outputs $\mathbf{y}_{ij}$ by

$$\mathbf{y}_{ij} = f_{ks}\left(\{\mathbf{x}_{si+\delta i, sj+\delta j}\}_{0 \le \delta i, \delta j \le k}\right)$$

where k is called the kernel size, s is the stride or subsampling factor, and f_{ks} determines the layer type: a matrix multiplication for convolution or average pooling, a spatial max for max pooling, or an elementwise nonlinearity for an activation function, and so on for other types of layers.

This functional form is maintained under composition, with kernel size and stride obeying the transformation rule

$$f_{ks} \circ g_{k's'} = (f \circ g)_{k'+(k-1)s', ss'}.$$

While a general deep net computes a general nonlinear function, a net with only layers of this form computes a nonlinear *filter*, which we call a *deep filter* or *fully convolutional network*. An FCN naturally operates on an input of any size, and produces an output of corresponding (possibly resampled) spatial dimensions.

A real-valued loss function composed with an FCN defines a task. If the loss function is a sum over the spatial dimensions of the final layer, $\ell(\mathbf{x}; \theta) = \sum_{ij} \ell'(\mathbf{x}_{ij}; \theta)$, its gradient will be a sum over the gradients of each of its spatial components. Thus stochastic gradient descent on ℓ computed on whole images will be the same as stochastic gradient descent on ℓ', taking all of the final layer receptive fields as a minibatch.

When these receptive fields overlap significantly, both feedforward computation *and* backpropagation are much more efficient when computed layer-by-layer over an entire image instead of independently patch-by-patch.

We next explain how to convert classification nets into fully convolutional nets that produce coarse output maps. For pixelwise prediction, we need to connect these coarse outputs back to the pixels. Section 3.2 describes a trick that OverFeat [29] introduced for this purpose. We gain insight into this trick by reinterpreting it as an equivalent network modification. As an efficient, effective alternative, we introduce deconvolution layers for upsampling in Section 3.3. In Section 3.4 we consider training by patchwise sampling, and give evidence in Section 4.3 that our whole image training is faster and equally effective.

3.1. Adapting classifiers for dense prediction

Typical recognition nets, including LeNet [21], AlexNet [19], and its deeper successors [31, 32], ostensibly take fixed-sized inputs and produce nonspatial outputs. The fully connected layers of these nets have fixed dimensions and throw away spatial coordinates. However, these fully connected layers can also be viewed as convolutions with kernels that cover their entire input regions. Doing so casts them into fully convolutional networks that take input of any size and output classification maps. This transformation

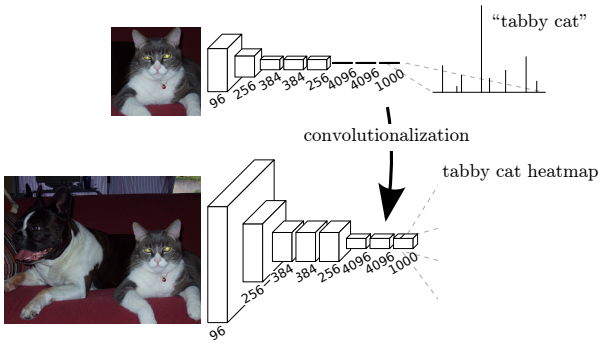

Figure 2. Transforming fully connected layers into convolution layers enables a classification net to output a heatmap. Adding layers and a spatial loss (as in Figure 1) produces an efficient machine for end-to-end dense learning.

is illustrated in Figure 2. (By contrast, nonconvolutional nets, such as the one by Le *et al.* [20], lack this capability.)

Furthermore, while the resulting maps are equivalent to the evaluation of the original net on particular input patches, the computation is highly amortized over the overlapping regions of those patches. For example, while AlexNet takes 1.2 ms (on a typical GPU) to produce the classification scores of a 227×227 image, the fully convolutional version takes 22 ms to produce a 10×10 grid of outputs from a 500×500 image, which is more than 5 times faster than the naïve approach[1].

The spatial output maps of these convolutionalized models make them a natural choice for dense problems like semantic segmentation. With ground truth available at every output cell, both the forward and backward passes are straightforward, and both take advantage of the inherent computational efficiency (and aggressive optimization) of convolution.

The corresponding backward times for the AlexNet example are 2.4 ms for a single image and 37 ms for a fully convolutional 10×10 output map, resulting in a speedup similar to that of the forward pass. This dense backpropagation is illustrated in Figure 1.

While our reinterpretation of classification nets as fully convolutional yields output maps for inputs of any size, the output dimensions are typically reduced by subsampling. The classification nets subsample to keep filters small and computational requirements reasonable. This coarsens the output of a fully convolutional version of these nets, reducing it from the size of the input by a factor equal to the pixel stride of the receptive fields of the output units.

[1] Assuming efficient batching of single image inputs. The classification scores for a single image by itself take 5.4 ms to produce, which is nearly 25 times slower than the fully convolutional version.

在输入层，这些函数通过计算输出 $\mathbf{y}_{ij}$

$$\mathbf{y}_{ij} = f_{ks}\left(\{\mathbf{x}_{si+\delta i, sj+\delta j}\}_{0\le\delta i,\delta j\le k}\right)$$

其中k被称为核大小，s是步长或下采样因子，而f_{ks}决定了层的类型：对于卷积或平均池化是矩阵乘法，对于最大池化是空间最大值操作，对于激活函数是逐元素非线性变换，其他类型的层也依此类推。

这种函数形式在组合下保持不变，其核大小和步长遵循变换规则

$$f_{ks} \circ g_{k's'} = (f \circ g)_{k'+(k-1)s', ss'}.$$

虽然一般的深度网络计算的是普通的非线性函数，但仅由这种形式的层构成的网络计算的是非线性*filter*，我们称之为*deep filter*或*fully convolu-tional network*。全卷积网络（FCN）天然适用于任意尺寸的输入，并生成具有相应（可能经过重采样的）空间维度的输出。

一个由全卷积网络（FCN）与实值损失函数组合而成的任务被定义。若损失函数是最终层空间维度上的求和，即$\ell(\mathbf{x}、\theta) = \sum_{ij}\ell'(\mathbf{x}_{ij}、\theta)$，其梯度将是各空间分量梯度的总和。因此，在整个图像上计算ℓ的随机梯度下降，将与将所有最终层感受野作为小批量处理ℓ'的随机梯度下降相同。

当这些感受野显著重叠时，无论是前馈计算*and*还是反向传播，在整个图像上逐层计算都比独立地逐块计算要高效得多。

接下来，我们将解释如何将分类网络转换为能够生成粗略输出图的全卷积网络。为了实现像素级预测，我们需要将这些粗略输出重新连接到像素上。第3.2节描述了OverFeat[29]为此目的引入的一种技巧。通过将其重新解释为等效的网络修改，我们深入理解了这一技巧。作为一种高效且有效的替代方案，我们在第3.3节中引入了用于上采样的反卷积层。在第3.4节中，我们考虑了通过分块采样进行训练，并在第4.3节中提供证据，表明我们的整图训练方法更快且同样有效。

3.1. 为密集预测任务调整分类器

典型的识别网络，包括LeNet [21]、AlexNet [19]及其更深的后续模型[31, 32]，表面上接受固定尺寸的输入并产生非空间输出。这些网络的全连接层具有固定维度，且会丢弃空间坐标信息。然而，这些全连接层也可被视为覆盖整个输入区域的卷积核所执行的卷积操作。通过这种转换，它们可转化为全卷积网络，能够接受任意尺寸的输入并输出分类图。这一转换

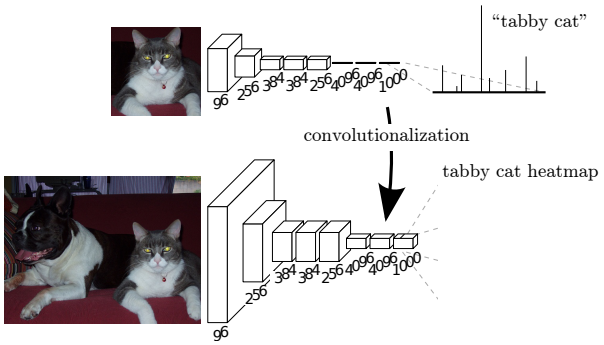

图2. 将全连接层转换为卷积层使分类网络能够输出热力图。添加层和空间损失（如图1所示）构建了一个高效的端到端密集学习机制。

如图2所示。（相比之下，非卷积网络（如Le *et al*提出的网络[20]）则不具备这种能力。）

此外，虽然生成的特征图等价于原始网络在特定输入图像块上的评估结果，但计算过程在这些图像块的重叠区域上得到了高度分摊。例如，AlexNet在典型GPU上需要1.2毫秒来生成一张227×227图像的分类得分，而全卷积版本仅需22毫秒即可从一张500×500图像中生成10×10的输出网格，这比原始方法快了5倍以上。

这些卷积化模型的空间输出映射使它们成为密集任务（如语义分割）的自然选择。由于每个输出单元都有可用的真实标签，前向传播和反向传播都变得直接明了，并且两者都能利用卷积固有的计算效率（以及激进的优化）。

AlexNet示例对应的反向传播时间分别为单张图像2.4毫秒和全卷积10×10输出图的37毫秒，其加速效果与前向传播相似。这种密集反向传播过程如图1所示。

尽管我们将分类网络重新解释为全卷积网络，使其能对任意尺寸的输入生成输出映射图，但输出维度通常会因下采样而降低。分类网络通过下采样来保持滤波器的小尺寸和计算需求的合理性。这导致这些网络的全卷积版本输出变得粗糙，使其尺寸相较于输入按输出单元感受野的像素步长比例缩小。

[1] Assuming efficient batching of single image inputs. The classification scores for a single image by itself take 5.4 ms to produce, which is nearly 25 times slower than the fully convolutional version.

3.2. Shift-and-stitch is filter rarefaction

Input shifting and output interlacing is a trick that yields dense predictions from coarse outputs without interpolation, introduced by OverFeat [29]. If the outputs are downsampled by a factor of f, the input is shifted (by left and top padding) x pixels to the right and y pixels down, once for every value of $(x, y) \in \{0, \ldots, f-1\} \times \{0, \ldots, f-1\}$. These f^2 inputs are each run through the convnet, and the outputs are interlaced so that the predictions correspond to the pixels at the *centers* of their receptive fields.

Changing only the filters and layer strides of a convnet can produce the same output as this shift-and-stitch trick. Consider a layer (convolution or pooling) with input stride s, and a following convolution layer with filter weights f_{ij} (eliding the feature dimensions, irrelevant here). Setting the lower layer's input stride to 1 upsamples its output by a factor of s, just like shift-and-stitch. However, convolving the original filter with the upsampled output does not produce the same result as the trick, because the original filter only sees a reduced portion of its (now upsampled) input. To reproduce the trick, rarefy the filter by enlarging it as

$$f'_{ij} = \begin{cases} f_{i/s,j/s} & \text{if } s \text{ divides both } i \text{ and } j; \\ 0 & \text{otherwise,} \end{cases}$$

(with i and j zero-based). Reproducing the full net output of the trick involves repeating this filter enlargement layer-by-layer until all subsampling is removed.

Simply decreasing subsampling within a net is a tradeoff: the filters see finer information, but have smaller receptive fields and take longer to compute. We have seen that the shift-and-stitch trick is another kind of tradeoff: the output is made denser without decreasing the receptive field sizes of the filters, but the filters are prohibited from accessing information at a finer scale than their original design.

Although we have done preliminary experiments with shift-and-stitch, we do not use it in our model. We find learning through upsampling, as described in the next section, to be more effective and efficient, especially when combined with the skip layer fusion described later on.

3.3. Upsampling is backwards strided convolution

Another way to connect coarse outputs to dense pixels is interpolation. For instance, simple bilinear interpolation computes each output y_{ij} from the nearest four inputs by a linear map that depends only on the relative positions of the input and output cells.

In a sense, upsampling with factor f is convolution with a *fractional* input stride of $1/f$. So long as f is integral, a natural way to upsample is therefore *backwards convolution* (sometimes called *deconvolution*) with an *output* stride of f. Such an operation is trivial to implement, since it simply reverses the forward and backward passes of convolution. Thus upsampling is performed in-network for end-to-end learning by backpropagation from the pixelwise loss.

Note that the deconvolution filter in such a layer need not be fixed (e.g., to bilinear upsampling), but can be learned. A stack of deconvolution layers and activation functions can even learn a nonlinear upsampling.

In our experiments, we find that in-network upsampling is fast and effective for learning dense prediction. Our best segmentation architecture uses these layers to learn to upsample for refined prediction in Section 4.2.

3.4. Patchwise training is loss sampling

In stochastic optimization, gradient computation is driven by the training distribution. Both patchwise training and fully-convolutional training can be made to produce any distribution, although their relative computational efficiency depends on overlap and minibatch size. Whole image fully convolutional training is identical to patchwise training where each batch consists of all the receptive fields of the units below the loss for an image (or collection of images). While this is more efficient than uniform sampling of patches, it reduces the number of possible batches. However, random selection of patches within an image may be recovered simply. Restricting the loss to a randomly sampled subset of its spatial terms (or, equivalently applying a DropConnect mask [36] between the output and the loss) excludes patches from the gradient computation.

If the kept patches still have significant overlap, fully convolutional computation will still speed up training. If gradients are accumulated over multiple backward passes, batches can include patches from several images.[2]

Sampling in patchwise training can correct class imbalance [27, 8, 2] and mitigate the spatial correlation of dense patches [28, 16]. In fully convolutional training, class balance can also be achieved by weighting the loss, and loss sampling can be used to address spatial correlation.

We explore training with sampling in Section 4.3, and do not find that it yields faster or better convergence for dense prediction. Whole image training is effective and efficient.

4. Segmentation Architecture

We cast ILSVRC classifiers into FCNs and augment them for dense prediction with in-network upsampling and a pixelwise loss. We train for segmentation by fine-tuning. Next, we build a novel skip architecture that combines coarse, semantic and local, appearance information to refine prediction.

For this investigation, we train and validate on the PASCAL VOC 2011 segmentation challenge [7]. We train with

[2] Note that not every possible patch is included this way, since the receptive fields of the final layer units lie on a fixed, strided grid. However, by shifting the image left and down by a random value up to the stride, random selection from all possible patches may be recovered.

3.2. 移位缝合是滤波器稀疏化

输入平移与输出交错是一种技巧，无需插值即可从粗糙输出中获得密集预测，由OverFeat[29]提出。若输出下采样因子为f，则输入（通过左方和上方填充）向右平移x像素、向下平移y像素，对每个$(x, y) \in \{0, \ldots, f-1\} \times \{0, \ldots, f-1\}$的值执行一次。这$f^2$个输入分别通过卷积网络处理，并将输出交错排列，使得预测结果对应其感受野*centers*处的像素。

仅改变卷积网络的滤波器和层步长，就能产生与这种移位拼接技巧相同的输出。考虑一个输入步长为s的层（卷积或池化），以及一个后续的卷积层，其滤波器权重为f_{ij}（此处省略了特征维度，因其无关紧要）。将较低层的输入步长设为1，会将其输出上采样s倍，正如移位拼接所做的那样。然而，用原始滤波器与上采样后的输出进行卷积，并不会产生与该技巧相同的结果，因为原始滤波器只能看到其（现已上采样的）输入的一小部分。为了复现该技巧，需通过扩大滤波器来稀释它，具体如下：

$$f'_{ij} = \begin{cases} f_{i/s,j/s} & \text{if } s \text{ divides both } i \text{ and } j; \\ 0 & \text{otherwise,} \end{cases}$$

(将i和j设为零基)。重现该技巧的完整网络输出需要逐层重复此滤波器放大操作，直至所有子采样被移除。

单纯减少网络中的下采样是一种权衡：滤波器能看到更精细的信息，但感受野会变小且计算耗时更长。我们已经看到，平移-拼接技巧是另一种权衡：输出变得更密集且不减小滤波器的感受野大小，但滤波器无法访问比原始设计更精细尺度的信息。

尽管我们已经对移位拼接法进行了初步实验，但并未在我们的模型中使用它。我们发现，通过下一节将描述的上采样方法进行学习更为有效和高效，特别是与后文将介绍的跳跃层融合相结合时。

3.3. 上采样是反向步进卷积

另一种将粗糙输出连接到密集像素的方法是插值。例如，简单的双线性插值通过仅依赖于输入和输出单元相对位置的线性映射，从最近的四个输入计算每个输出y_{ij}。

从某种意义上说，以因子f进行上采样就是与输入步幅为$1/f$的*fractional*进行卷积。只要f是整数，一种自然的上采样方式就是*backwards convolution*（有时称为*deconvolution*），其步幅为f的*output*。这种操作实现起来很简单，因为它只是简单地反转了卷积的前向和反向传播过程。

因此，上采样在网络上执行，以便通过从像素级损失的反向传播进行端到端学习。

需要注意的是，此类层中的反卷积滤波器无需固定（例如固定为双线性上采样），而是可以学习得到。通过堆叠反卷积层和激活函数，甚至能够学习非线性上采样。

在我们的实验中，我们发现网络内上采样对于学习密集预测是快速且有效的。我们最佳的分割架构在4.2节中使用这些层来学习上采样，以实现精细预测。

3.4. 逐块训练是一种损失采样方法

在随机优化中，梯度计算由训练分布驱动。尽管分块训练和全卷积训练的相对计算效率取决于重叠度和小批量大小，但两者均可配置为生成任意分布。整图全卷积训练等同于分块训练，其中每个批次包含图像（或图像集合）损失函数下方单元的所有感受野。虽然这比均匀采样图像块更高效，但减少了可能的批次数量。然而，图像内随机选择图像块的方法可以轻松恢复：将损失函数限制在其空间项的随机采样子集上（或等效地在输出与损失函数间应用DropConnect掩码[36]），即可将特定图像块排除在梯度计算之外。

如果保留的补丁仍有显著重叠，全卷积计算仍将加速训练。如果在多个反向传播过程中累积梯度，批次可以包含来自多个图像的补丁。[2]

在分块训练中，采样可以纠正类别不平衡问题[27, 8, 2]，并缓解密集块的空间相关性[28, 16]。在全卷积训练中，类别平衡也可以通过加权损失来实现，并且可以使用损失采样来解决空间相关性问题。

我们在第4.3节探讨了采样训练方法，但并未发现其在密集预测任务中能带来更快或更好的收敛效果。全图像训练既高效又有效。

4. 分割架构

我们将ILSVRC分类器转化为全卷积网络，并通过网络内上采样和逐像素损失增强其密集预测能力。通过微调进行分割训练。接着，我们构建了一种新颖的跳跃式架构，结合粗粒度的语义信息与局部的表观信息，以优化预测结果。

对于本次研究，我们在PASCAL VOC 2011分割挑战赛[7]上进行训练与验证。我们使用

[2]Note that not every possible patch is included this way, since the receptive fields of the final layer units lie on a fixed, strided grid. However, by shifting the image left and down by a random value up to the stride, random selection from all possible patches may be recovered.

a per-pixel multinomial logistic loss and validate with the standard metric of mean pixel intersection over union, with the mean taken over all classes, including background. The training ignores pixels that are masked out (as ambiguous or difficult) in the ground truth.

4.1. From classifier to dense FCN

We begin by convolutionalizing proven classification architectures as in Section 3. We consider the AlexNet[3] architecture [19] that won ILSVRC12, as well as the VGG nets [31] and the GoogLeNet[4] [32] which did exceptionally well in ILSVRC14. We pick the VGG 16-layer net[5], which we found to be equivalent to the 19-layer net on this task. For GoogLeNet, we use only the final loss layer, and improve performance by discarding the final average pooling layer. We decapitate each net by discarding the final classifier layer, and convert all fully connected layers to convolutions. We append a 1×1 convolution with channel dimension 21 to predict scores for each of the PASCAL classes (including background) at each of the coarse output locations, followed by a deconvolution layer to bilinearly upsample the coarse outputs to pixel-dense outputs as described in Section 3.3. Table 1 compares the preliminary validation results along with the basic characteristics of each net. We report the best results achieved after convergence at a fixed learning rate (at least 175 epochs).

Fine-tuning from classification to segmentation gave reasonable predictions for each net. Even the worst model achieved $\sim 75\%$ of state-of-the-art performance. The segmentation-equippped VGG net (FCN-VGG16) already appears to be state-of-the-art at 56.0 mean IU on val, compared to 52.6 on test [16]. Training on extra data raises performance to 59.4 mean IU on a subset of val[7]. Training details are given in Section 4.3.

Despite similar classification accuracy, our implementation of GoogLeNet did not match this segmentation result.

4.2. Combining what and where

We define a new fully convolutional net (FCN) for segmentation that combines layers of the feature hierarchy and refines the spatial precision of the output. See Figure 3.

While fully convolutionalized classifiers can be fine-tuned to segmentation as shown in 4.1, and even score highly on the standard metric, their output is dissatisfyingly coarse (see Figure 4). The 32 pixel stride at the final prediction layer limits the scale of detail in the upsampled output.

We address this by adding links that combine the final prediction layer with lower layers with finer strides. This

[3]Using the publicly available `CaffeNet` reference model.

[4]Since there is no publicly available version of GoogLeNet, we use our own reimplementation. Our version is trained with less extensive data augmentation, and gets 68.5% top-1 and 88.4% top-5 ILSVRC accuracy.

[5]Using the publicly available version from the Caffe model zoo.

Table 1. We adapt and extend three classification convnets to segmentation. We compare performance by mean intersection over union on the validation set of PASCAL VOC 2011 and by inference time (averaged over 20 trials for a 500×500 input on an NVIDIA Tesla K40c). We detail the architecture of the adapted nets as regards dense prediction: number of parameter layers, receptive field size of output units, and the coarsest stride within the net. (These numbers give the best performance obtained at a fixed learning rate, not best performance possible.)

	FCN-AlexNet	FCN-VGG16	FCN-GoogLeNet[4]
mean IU	39.8	**56.0**	42.5
forward time	50 ms	210 ms	59 ms
conv. layers	8	16	22
parameters	57M	134M	6M
rf size	355	404	907
max stride	32	32	32

turns a line topology into a DAG, with edges that skip ahead from lower layers to higher ones (Figure 3). As they see fewer pixels, the finer scale predictions should need fewer layers, so it makes sense to make them from shallower net outputs. Combining fine layers and coarse layers lets the model make local predictions that respect global structure. By analogy to the multiscale local jet of Florack *et al*. [10], we call our nonlinear local feature hierarchy the *deep jet*.

We first divide the output stride in half by predicting from a 16 pixel stride layer. We add a 1×1 convolution layer on top of `pool4` to produce additional class predictions. We fuse this output with the predictions computed on top of `conv7` (convolutionalized `fc7`) at stride 32 by adding a $2\times$ upsampling layer and summing[6] both predictions. (See Figure 3). We initialize the $2\times$ upsampling to bilinear interpolation, but allow the parameters to be learned as described in Section 3.3. Finally, the stride 16 predictions are upsampled back to the image. We call this net FCN-16s. FCN-16s is learned end-to-end, initialized with the parameters of the last, coarser net, which we now call FCN-32s. The new parameters acting on `pool4` are zero-initialized so that the net starts with unmodified predictions. The learning rate is decreased by a factor of 100.

Learning this skip net improves performance on the validation set by 3.0 mean IU to 62.4. Figure 4 shows improvement in the fine structure of the output. We compared this fusion with learning only from the `pool4` layer (which resulted in poor performance), and simply decreasing the learning rate without adding the extra link (which results in an insignificant performance improvement, without improving the quality of the output).

We continue in this fashion by fusing predictions from `pool3` with a $2\times$ upsampling of predictions fused from `pool4` and `conv7`, building the net FCN-8s. We obtain

[6]Max fusion made learning difficult due to gradient switching.

采用逐像素的多项逻辑损失，并使用平均像素交并比这一标准指标进行验证，该平均值涵盖所有类别，包括背景。训练过程中会忽略真实标注中被遮蔽（因模糊或难以判断）的像素。

4.1. 从分类器到密集全卷积网络

我们首先按照第3节的方法，将成熟的分类架构进行卷积化改造。我们采用了在ILSVRC12中获胜的AlexNet[3]架构[19]，以及在ILSVRC14中表现极为出色的VGG网络[31]和GoogLeNet[4][32]。我们选择了VGG的16层网络[5]，因为在此任务中我们发现其效果与19层网络相当。对于GoogLeNet，我们仅使用最终的损失层，并通过移除最后的平均池化层来提升性能。我们通过丢弃最终的分类器层对每个网络进行“斩首”处理，并将所有全连接层转换为卷积层。随后，我们添加一个通道维度为21的1×1卷积层，用于在每个粗略输出位置预测PASCAL类别（包括背景）的得分，再按第3.3节所述，通过一个反卷积层对粗略输出进行双线性上采样，得到像素级密集输出。表1对比了初步验证结果以及各网络的基本特性。我们报告了在固定学习率下收敛后（至少175个训练周期）获得的最佳结果。

从分类到分割的微调为每个网络提供了合理的预测。即使是最差的模型也达到了当前最佳性能的~ 75%。配备分割功能的VGG网络（FCN-VGG16）在验证集上已达到当前最佳水平，平均IU为56.0，而测试集上为52.6[16]。使用额外数据训练后，在验证集子集[7]上的性能提升至59.4平均IU。训练细节详见第4.3节。

尽管分类准确率相似，我们的GoogLeNet实现未能达到这一分割结果。

4.2. 结合内容与位置

我们定义了一种新的全卷积网络（FCN）用于分割，它结合了特征层次的多层信息并提升了输出的空间精度。参见图3。

虽然全卷积化的分类器可以如4.1节所示微调用于分割任务，甚至在标准指标上获得高分，但其输出结果在细节上显得过于粗糙（见图4）。最终预测层32像素的步长限制了上采样输出中细节的尺度。

我们通过添加链接来解决这个问题，这些链接将最终预测层与具有更精细步幅的较低层结合起来。这

表1. 我们将三种分类卷积网络进行适应和扩展以用于分割任务。通过在PASCAL VOC 2011验证集上的平均交并比性能以及推理时间（在NVIDIA Tesla K40c上对500 {v*} 500输入进行20次试验的平均值）进行比较。我们详细说明了适应网络在密集预测方面的架构：参数层数量、输出单元的感受野大小以及网络内部最粗略的步幅。（这些数值是在固定学习率下获得的最佳性能，并非可能达到的最高性能。）

	FCN-AlexNet	FCN-VGG16	FCN-GoogLeNet[4]
mean IU	39.8	**56.0**	42.5
forward time	50 ms	210 ms	59 ms
conv. layers	8	16	22
parameters	57M	134M	6M
rf size	355	404	907
max stride	32	32	32

将线型拓扑转变为有向无环图，其边可从较低层跳跃至较高层（图3）。由于精细尺度预测所需处理的像素更少，其所需网络层数也应更少，因此从较浅的网络输出层进行此类预测是合理的。通过融合精细层与粗糙层，模型得以在遵循全局结构的前提下进行局部预测。类比Florack *et al*提出的多尺度局部喷流模型[10]，我们将这种非线性局部特征层次结构称为*deep jet*。

我们首先通过从步长为16像素的层进行预测，将输出步长减半。我们在pool4之上添加一个1×1卷积层，以产生额外的类别预测。我们将此输出与在步长为32的conv7（卷积化的fc7）之上计算的预测进行融合，通过添加一个2倍上采样层并将两个预测相加（见图3）。我们将2倍上采样初始化为双线性插值，但允许参数按照第3.3节所述进行学习。最后，将步长为16的预测上采样回原始图像尺寸。我们将此网络称为FCN-16s。FCN-16s通过端到端学习，以最后一个较粗糙网络（现称为FCN-32s）的参数进行初始化。作用于pool4的新参数以零初始化，使网络从未经修改的预测开始训练。学习率降低了100倍。

学习这个跳跃网络将验证集的性能提高了3.0平均IU，达到62.4。图4显示了输出精细结构的改进。我们将这种融合与仅从pool4层学习（这导致性能较差）进行了比较，并与仅降低学习率而不添加额外链接（这带来的性能提升不显著，且未改善输出质量）进行了对比。

我们继续以这种方式，将来自pool3的预测与来自pool4和conv7融合预测的2×上采样进行融合，构建网络FCN-8s。我们得到

[3]Using the publicly available `CaffeNet` reference model.

[4]Since there is no publicly available version of GoogLeNet, we use our own reimplementation. Our version is trained with less extensive data augmentation, and gets 68.5% top-1 and 88.4% top-5 ILSVRC accuracy.

[5]Using the publicly available version from the Caffe model zoo.

[6]Max fusion made learning difficult due to gradient switching.

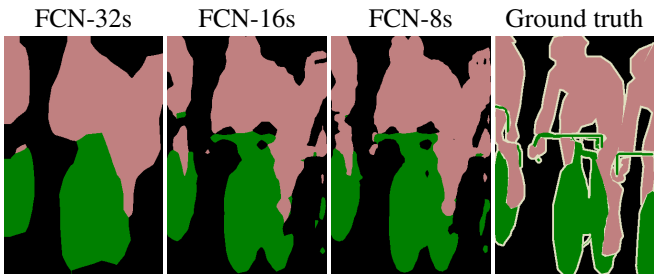

Figure 4. Refining fully convolutional nets by fusing information from layers with different strides improves segmentation detail. The first three images show the output from our 32, 16, and 8 pixel stride nets (see Figure 3).

Table 2. Comparison of skip FCNs on a subset of PASCAL VOC2011 validation[7]. Learning is end-to-end, except for FCN-32s-fixed, where only the last layer is fine-tuned. Note that FCN-32s is FCN-VGG16, renamed to highlight stride.

	pixel acc.	mean acc.	mean IU	f.w. IU
FCN-32s-fixed	83.0	59.7	45.4	72.0
FCN-32s	89.1	73.3	59.4	81.4
FCN-16s	90.0	75.7	62.4	83.0
FCN-8s	**90.3**	**75.9**	**62.7**	**83.2**

a minor additional improvement to 62.7 mean IU, and find a slight improvement in the smoothness and detail of our output. At this point our fusion improvements have met diminishing returns, both with respect to the IU metric which emphasizes large-scale correctness, and also in terms of the improvement visible e.g. in Figure 4, so we do not continue fusing even lower layers.

Refinement by other means Decreasing the stride of pooling layers is the most straightforward way to obtain finer predictions. However, doing so is problematic for our VGG16-based net. Setting the `pool5` layer to have stride 1 requires our convolutionalized `fc6` to have a kernel size of 14×14 in order to maintain its receptive field size. In addition to their computational cost, we had difficulty learning such large filters. We made an attempt to re-architect the layers above `pool5` with smaller filters, but were not successful in achieving comparable performance; one possible explanation is that the initialization from ImageNet-trained weights in the upper layers is important.

Another way to obtain finer predictions is to use the shift-and-stitch trick described in Section 3.2. In limited experiments, we found the cost to improvement ratio from this method to be worse than layer fusion.

4.3. Experimental framework

Optimization We train by SGD with momentum. We use a minibatch size of 20 images and fixed learning rates of 10^{-3}, 10^{-4}, and 5^{-5} for FCN-AlexNet, FCN-VGG16, and FCN-GoogLeNet, respectively, chosen by line search. We use momentum 0.9, weight decay of 5^{-4} or 2^{-4}, and doubled the learning rate for biases, although we found training to be insensitive to these parameters (but sensitive to the learning rate). We zero-initialize the class scoring convolution layer, finding random initialization to yield neither better performance nor faster convergence. Dropout was included where used in the original classifier nets.

Fine-tuning We fine-tune all layers by back-propagation through the whole net. Fine-tuning the output classifier alone yields only 70% of the full fine-tuning performance as compared in Table 2. Training from scratch is not feasible considering the time required to learn the base classification nets. (Note that the VGG net is trained in stages, while we initialize from the full 16-layer version.) Fine-tuning takes three days on a single GPU for the coarse FCN-32s version, and about one day each to upgrade to the FCN-16s and FCN-8s versions.

Patch Sampling As explained in Section 3.4, our full image training effectively batches each image into a regu-

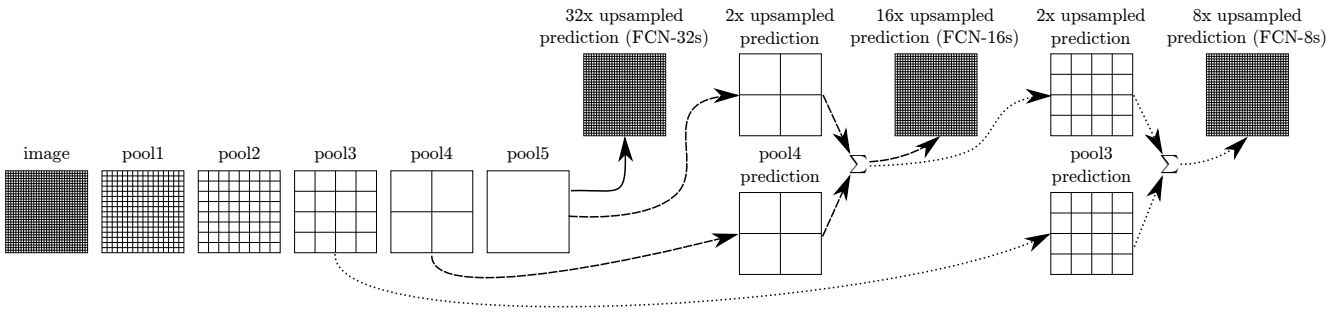

Figure 3. Our DAG nets learn to combine coarse, high layer information with fine, low layer information. Layers are shown as grids that reveal relative spatial coarseness. Only pooling and prediction layers are shown; intermediate convolution layers (including our converted fully connected layers) are omitted. Solid line (FCN-32s): Our single-stream net, described in Section 4.1, upsamples stride 32 predictions back to pixels in a single step. Dashed line (FCN-16s): Combining predictions from both the final layer and the `pool4` layer, at stride 16, lets our net predict finer details, while retaining high-level semantic information. Dotted line (FCN-8s): Additional predictions from `pool3`, at stride 8, provide further precision.

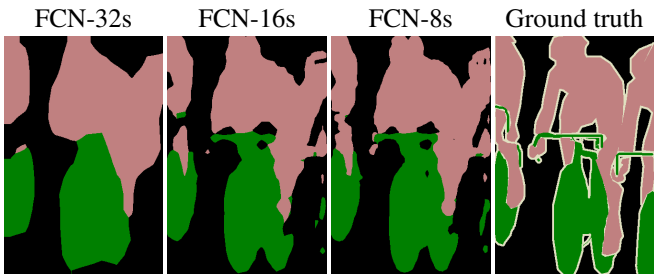

图4. 通过融合不同步幅层的信息来优化全卷积网络，提升了分割细节。前三张图像展示了我们32、16和8像素步幅网络的输出结果（参见图3）。

表2. 在PASCAL VOC2011验证集子集[7]上跳跃FCN的对比。除FCN-32s-fixed仅微调最后一层外，其余均为端到端学习。注意FCN-32s即FCN-VGG16，重命名以突出其步长。

	pixel acc.	mean acc.	mean IU	f.w. IU
FCN-32s-fixed	83.0	59.7	45.4	72.0
FCN-32s	89.1	73.3	59.4	81.4
FCN-16s	90.0	75.7	62.4	83.0
FCN-8s	**90.3**	**75.9**	**62.7**	**83.2**

在62.7的平均IU上仅有微小的额外提升，同时发现输出结果的平滑度和细节略有改善。至此，我们的融合改进已呈现收益递减趋势——无论是从强调大尺度准确性的IU指标来看，还是从如图4所示的视觉改进效果而言皆是如此，因此我们不再继续融合更底层网络。

通过其他方式细化　降低池化层的步长是获得更精细预测的最直接方法。然而，这样做对于我们基于VGG16的网络来说是有问题的。将pool5层的步长设为1，要求我们卷积化的fc6具有 14×14 以保持其感受野大小。除了计算成本高之外，我们也难以训练如此大的滤波器。我们尝试用较小的滤波器重新设计 pool5 之上的层结构，但未能达到相当的性能；一种可能的解释是，上层中由 ImageNet 预训练权重进行的初始化至关重要。

另一种获得更精细预测的方法是使用第3.2节中描述的移位拼接技巧。在有限的实验中，我们发现这种方法带来的改进与成本之比不如层融合方法。

4.3. 实验框架

优化 我们使用带动量的随机梯度下降（SGD）进行训练。我们采用20张图像的小批量大小，并通过线性搜索为FCN-AlexNet、FCN-VGG16和FCN-GoogLeNet分别设定了固定的学习率：10^{-3}、10^{-4}和5^{-5}。我们使用动量0.9，权重衰减为5^{-4}或2^{-4}，并将偏置的学习率加倍，尽管我们发现训练对这些参数不敏感（但对学习率敏感）。我们将类别评分卷积层初始化为零，发现随机初始化既不能带来更好的性能，也不能加快收敛速度。在原始分类网络中使用的dropout被保留。

微调 我们通过整个网络的反向传播对所有层进行微调。仅微调输出分类器只能达到全网络微调性能的70%，如表2所示。考虑到学习基础分类网络所需的时间，从头开始训练是不可行的（注意VGG网络是分阶段训练的，而我们是从完整的16层版本初始化的）。在单个GPU上，粗略的FCN-32s版本微调需要三天时间，升级到FCN-16s和FCN-8s版本各需约一天。

补丁采样 如第3.4节所述，我们的完整图像训练将每张图像有效批处理为规整的

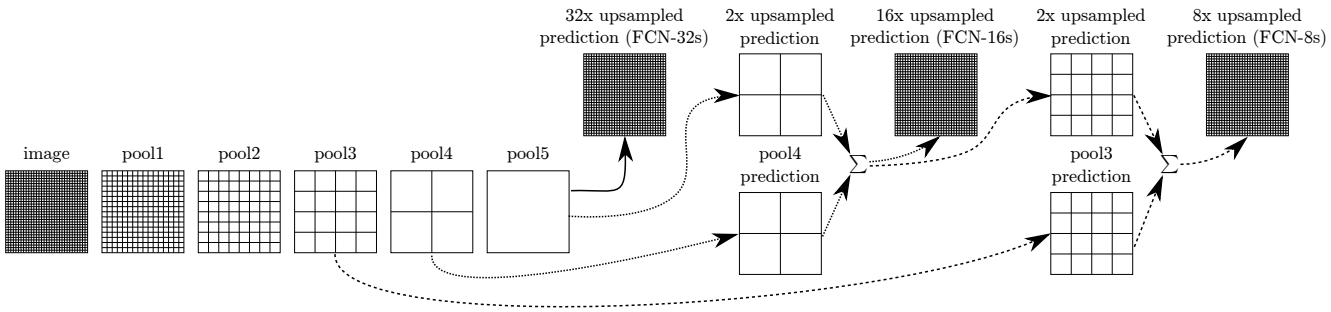

图3. 我们的DAG网络学会将粗糙的高层信息与精细的低层信息相结合。各层以网格形式展示，揭示相对空间粗糙度。图中仅显示池化层和预测层；中间卷积层（包括我们转换的全连接层）已省略。实线（FCN-32s）：如4.1节所述，我们的单流网络通过单步将步幅32的预测上采样至像素级。虚线（FCN-16s）：结合最终层和pool4层的预测（步幅16），使网络在保留高层语义信息的同时预测更精细的细节。点线（FCN-8s）：从pool3层（步幅8）添加的预测进一步提升了精度。

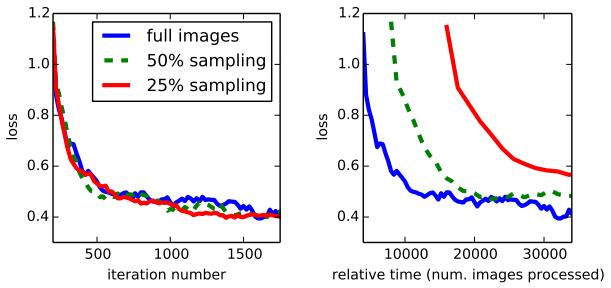

Figure 5. Training on whole images is just as effective as sampling patches, but results in faster (wall time) convergence by making more efficient use of data. Left shows the effect of sampling on convergence rate for a fixed expected batch size, while right plots the same by relative wall time.

lar grid of large, overlapping patches. By contrast, prior work randomly samples patches over a full dataset [27, 2, 8, 28, 11], potentially resulting in higher variance batches that may accelerate convergence [22]. We study this trade-off by spatially sampling the loss in the manner described earlier, making an independent choice to ignore each final layer cell with some probability $1-p$. To avoid changing the effective batch size, we simultaneously increase the number of images per batch by a factor $1/p$. Note that due to the efficiency of convolution, this form of rejection sampling is still faster than patchwise training for large enough values of p (e.g., at least for $p > 0.2$ according to the numbers in Section 3.1). Figure 5 shows the effect of this form of sampling on convergence. We find that sampling does not have a significant effect on convergence rate compared to whole image training, but takes significantly more time due to the larger number of images that need to be considered per batch. We therefore choose unsampled, whole image training in our other experiments.

Class Balancing Fully convolutional training can balance classes by weighting or sampling the loss. Although our labels are mildly unbalanced (about $3/4$ are background), we find class balancing unnecessary.

Dense Prediction The scores are upsampled to the input dimensions by deconvolution layers within the net. Final layer deconvolutional filters are fixed to bilinear interpolation, while intermediate upsampling layers are initialized to bilinear upsampling, and then learned. Shift-and-stitch (Section 3.2), or the filter rarefaction equivalent, are not used.

Augmentation We tried augmenting the training data by randomly mirroring and "jittering" the images by translating them up to 32 pixels (the coarsest scale of prediction) in each direction. This yielded no noticeable improvement.

More Training Data The PASCAL VOC 2011 segmentation challenge training set, which we used for Table 1, labels 1112 images. Hariharan *et al.* [15] have collected labels for a much larger set of 8498 PASCAL training images, which was used to train the previous state-of-the-art system, SDS [16]. This training data improves the FCN-VGG16 validation score[7] by 3.4 points to 59.4 mean IU.

Implementation All models are trained and tested with Caffe [18] on a single NVIDIA Tesla K40c. The models and code will be released open-source on publication.

5. Results

We test our FCN on semantic segmentation and scene parsing, exploring PASCAL VOC, NYUDv2, and SIFT Flow. Although these tasks have historically distinguished between objects and regions, we treat both uniformly as pixel prediction. We evaluate our FCN skip architecture[8] on each of these datasets, and then extend it to multi-modal input for NYUDv2 and multi-task prediction for the semantic and geometric labels of SIFT Flow.

Metrics We report four metrics from common semantic segmentation and scene parsing evaluations that are variations on pixel accuracy and region intersection over union (IU). Let n_{ij} be the number of pixels of class i predicted to belong to class j, where there are n_{cl} different classes, and let $t_i = \sum_j n_{ij}$ be the total number of pixels of class i. We compute:

- pixel accuracy: $\sum_i n_{ii} / \sum_i t_i$
- mean accuracy: $(1/n_{\text{cl}}) \sum_i n_{ii}/t_i$
- mean IU: $(1/n_{\text{cl}}) \sum_i n_{ii} / \left(t_i + \sum_j n_{ji} - n_{ii}\right)$
- frequency weighted IU: $\left(\sum_k t_k\right)^{-1} \sum_i t_i n_{ii} / \left(t_i + \sum_j n_{ji} - n_{ii}\right)$

PASCAL VOC Table 3 gives the performance of our FCN-8s on the test sets of PASCAL VOC 2011 and 2012, and compares it to the previous state-of-the-art, SDS [16], and the well-known R-CNN [12]. We achieve the best results on mean IU[9] by a relative margin of 20%. Inference time is reduced $114\times$ (convnet only, ignoring proposals and refinement) or $286\times$ (overall).

Table 3. Our fully convolutional net gives a 20% relative improvement over the state-of-the-art on the PASCAL VOC 2011 and 2012 test sets, and reduces inference time.

	mean IU VOC2011 test	mean IU VOC2012 test	inference time
R-CNN [12]	47.9	-	-
SDS [16]	52.6	51.6	$\sim$ 50 s
FCN-8s	**62.7**	**62.2**	$\sim$ **175 ms**

NYUDv2 [30] is an RGB-D dataset collected using the

[7] There are training images from [15] included in the PASCAL VOC 2011 val set, so we validate on the non-intersecting set of 736 images. An earlier version of this paper mistakenly evaluated on the entire val set.

[8] Our models and code are publicly available at `https://github.com/BVLC/caffe/wiki/Model-Zoo#fcn`.

[9] This is the only metric provided by the test server.

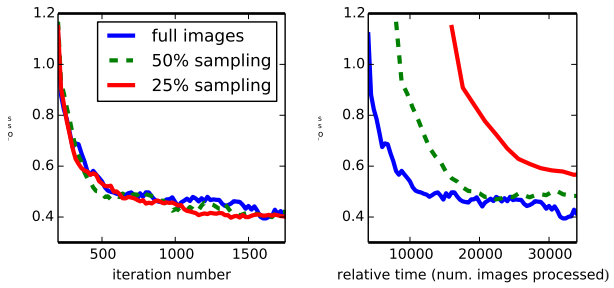

图5. 在全图像上进行训练与采样图像块同样有效，但通过更高效地利用数据，能实现更快的（实际时间）收敛。左图展示了固定预期批次大小时采样对收敛速度的影响，右图则按相对实际时间绘制了相同结果。

采用大面积重叠斑块的规则网格。相比之下，先前的研究在整个数据集中随机采样斑块[27, 2, 8, 28, 11]，这可能导致方差更高的批次，从而可能加速收敛[22]。我们通过前文描述的方式对损失进行空间采样来研究这种权衡，即以概率$1-p$独立选择忽略每个最终层单元。为避免改变有效批次大小，我们同时将每批图像数量增加$1/p$倍。需要注意的是，由于卷积的高效性，对于足够大的p (值（例如根据第3.1)节的数据，至少p >0. 2时），这种形式的拒绝采样仍比逐块训练更快。图5展示了这种采样方式对收敛的影响。我们发现，与整图训练相比，采样对收敛速度没有显著影响，但由于每批需要考虑的图像数量更多，耗时显著增加。因此我们在其他实验中选择了非采样的整图训练方式。

类别平衡 全卷积训练可以通过加权或采样损失来平衡类别。尽管我们的标签存在轻微的不平衡（大约3/4是背景），但我们发现类别平衡并非必要。

密集预测 分数通过网络内的反卷积层上采样至输入维度。最终层的反卷积滤波器固定为双线性插值，而中间上采样层则初始化为双线性上采样后通过学习优化。未使用移位拼接（第3.2节）或其等效的滤波器稀疏化方法。

增强 我们尝试通过随机镜像和“抖动”图像来增强训练数据，即在每个方向上平移最多32个像素（预测的最粗尺度）。这并未带来明显的改进。

更多训练数据 我们用于表1的PASCAL VOC 2011分割挑战训练集标注了1112张图像。Hariharan {v*}等人[15]已收集用于训练先前最先进系统SDS [16]的更大规模PASCAL训练集（共8498张图像）的标签。该训练数据将FCN-VGG16的验证分数[7]提升了3.4个点，达到59.4的平均IU。

实现 所有模型均在单个NVIDIA Tesla K40c上使用Caffe[18]进行训练和测试。模型与代码将在发表时开源发布。

5. 结果

我们在语义分割和场景解析任务上测试了全卷积网络，探索了PASCAL VOC、NYUDv2和SIFT Flow数据集。尽管这些任务历来区分对象与区域，我们将两者统一视为像素级预测处理。我们评估了全卷积网络的跳跃式架构[8]在各项数据集上的表现，随后将其扩展至NYUDv2的多模态输入处理，并针对SIFT Flow数据集的语义标注与几何标注实现了多任务预测。

指标 我们报告了来自常见语义分割和场景解析评估的四项指标，这些指标是像素精度和区域交并比（IU）的变体。设 n_{ij} 为预测属于类别 j 的类别 i 的像素数，其中共有 n_{cl} 个不同类别，并设 $t_i = \sum_j n_{ij}$ 为类别 i 的总像素数。我们计算：

- 像素精度：$\sum_i n_{ii} / \sum_i t_i$
- 平均准确率: $(1/n_{\text{cl}}) \sum_i n_{ii}/t_i$
- 平均IU：$(1/n_{\text{cl}}) \sum_i n_{ii} / \left(t_i + \sum_j n_{ji} - n_{ii}\right)$
- 频率加权IU：$\left(\sum_k t_k\right)^{-1} \sum_i t_i n_{ii} / \left(t_i + \sum_j n_{ji} - n_{ii}\right)$

PASCAL VOC 表3展示了我们的FCN-8s在PASCAL VOC 2011和2012测试集上的性能，并与之前的最先进方法SDS[16]以及著名的R-CNN[12]进行了比较。我们在平均IU[9]上以20%的相对优势取得了最佳结果。推理时间减少了114×（（仅考虑卷积网络，忽略建议框和优化））或286×（（整体计算））。

表3. 我们的全卷积网络在PASCAL VOC 2011和2012测试集上相比现有最优技术实现了20%的相对性能提升，并减少了推理时间。

	mean IU VOC2011 test	mean IU VOC2012 test	inference time
R-CNN [12]	47.9	-	-
SDS [16]	52.6	51.6	~ 50 s
FCN-8s	**62.7**	**62.2**	~ **175 ms**

NYUDv2 [30] is an RGB-D dataset collected using the

[7]There are training images from [15] included in the PASCAL VOC 2011 val set, so we validate on the non-intersecting set of 736 images. An earlier version of this paper mistakenly evaluated on the entire val set.

[8]Our models and code are publicly available at `https://github.com/BVLC/caffe/wiki/Model-Zoo#fcn`.

[9]This is the only metric provided by the test server.

Table 4. Results on NYUDv2. *RGBD* is early-fusion of the RGB and depth channels at the input. *HHA* is the depth embedding of [14] as horizontal disparity, height above ground, and the angle of the local surface normal with the inferred gravity direction. *RGB-HHA* is the jointly trained late fusion model that sums RGB and HHA predictions.

	pixel acc.	mean acc.	mean IU	f.w. IU
Gupta *et al.* [14]	60.3	-	28.6	47.0
FCN-32s RGB	60.0	42.2	29.2	43.9
FCN-32s RGBD	61.5	42.4	30.5	45.5
FCN-32s HHA	57.1	35.2	24.2	40.4
FCN-32s RGB-HHA	64.3	44.9	32.8	48.0
FCN-16s RGB-HHA	**65.4**	**46.1**	**34.0**	**49.5**

Microsoft Kinect. It has 1449 RGB-D images, with pixel-wise labels that have been coalesced into a 40 class semantic segmentation task by Gupta *et al.* [13]. We report results on the standard split of 795 training images and 654 testing images. (Note: all model selection is performed on PASCAL 2011 val.) Table 4 gives the performance of our model in several variations. First we train our unmodified coarse model (FCN-32s) on RGB images. To add depth information, we train on a model upgraded to take four-channel RGB-D input (early fusion). This provides little benefit, perhaps due to the difficultly of propagating meaningful gradients all the way through the model. Following the success of Gupta *et al.* [14], we try the three-dimensional HHA encoding of depth, training nets on just this information, as well as a "late fusion" of RGB and HHA where the predictions from both nets are summed at the final layer, and the resulting two-stream net is learned end-to-end. Finally we upgrade this late fusion net to a 16-stride version.

SIFT Flow is a dataset of 2,688 images with pixel labels for 33 semantic categories ("bridge", "mountain", "sun"), as well as three geometric categories ("horizontal", "vertical", and "sky"). An FCN can naturally learn a joint representation that simultaneously predicts both types of labels. We learn a two-headed version of FCN-16s with semantic and geometric prediction layers and losses. The learned model performs as well on both tasks as two independently trained models, while learning and inference are essentially as fast as each independent model by itself. The results in Table 5, computed on the standard split into 2,488 training and 200 test images,[10] show state-of-the-art performance on both tasks.

[10]Three of the SIFT Flow categories are not present in the test set. We made predictions across all 33 categories, but only included categories actually present in the test set in our evaluation. (An earlier version of this paper reported a lower mean IU, which included all categories either present or predicted in the evaluation.)

Table 5. Results on SIFT Flow[10] with class segmentation (center) and geometric segmentation (right). Tighe [33] is a non-parametric transfer method. Tighe 1 is an exemplar SVM while 2 is SVM + MRF. Farabet is a multi-scale convnet trained on class-balanced samples (1) or natural frequency samples (2). Pinheiro is a multi-scale, recurrent convnet, denoted RCNN$_3$ ($\circ^3$). The metric for geometry is pixel accuracy.

	pixel acc.	mean acc.	mean IU	f.w. IU	geom. acc.
Liu *et al.* [23]	76.7	-	-	-	-
Tighe *et al.* [33]	-	-	-	-	90.8
Tighe *et al.* [34] 1	75.6	41.1	-	-	-
Tighe *et al.* [34] 2	78.6	39.2	-	-	-
Farabet *et al.* [8] 1	72.3	50.8	-	-	-
Farabet *et al.* [8] 2	78.5	29.6	-	-	-
Pinheiro *et al.* [28]	77.7	29.8	-	-	-
FCN-16s	**85.2**	**51.7**	39.5	76.1	**94.3**

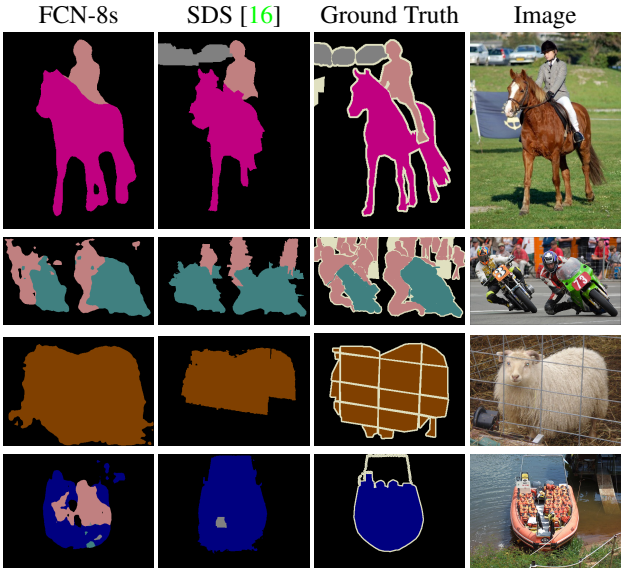

Figure 6. Fully convolutional segmentation nets produce state-of-the-art performance on PASCAL. The left column shows the output of our highest performing net, FCN-8s. The second shows the segmentations produced by the previous state-of-the-art system by Hariharan *et al.* [16]. Notice the fine structures recovered (first row), ability to separate closely interacting objects (second row), and robustness to occluders (third row). The fourth row shows a failure case: the net sees lifejackets in a boat as people.

6. Conclusion

Fully convolutional networks are a rich class of models, of which modern classification convnets are a special case. Recognizing this, extending these classification nets to segmentation, and improving the architecture with multi-resolution layer combinations dramatically improves the state-of-the-art, while simultaneously simplifying and speeding up learning and inference.

Acknowledgements This work was supported in part

表4. NYUDv2数据集上的结果。*RGBD*是在输入端对RGB和深度通道进行的早期融合。*HHA*是[14]中提出的深度嵌入方法，使用水平视差、地面以上高度以及局部表面法线与推断重力方向之间的夹角作为特征。*RGB-HHA*是联合训练的后期融合模型，对RGB和HHA的预测结果进行求和。

	pixel acc.	mean acc.	mean IU	f.w. IU
Gupta *et al*. [14]	60.3	-	28.6	47.0
FCN-32s RGB	60.0	42.2	29.2	43.9
FCN-32s RGBD	61.5	42.4	30.5	45.5
FCN-32s HHA	57.1	35.2	24.2	40.4
FCN-32s RGB-HHA	64.3	44.9	32.8	48.0
FCN-16s RGB-HHA	**65.4**	**46.1**	**34.0**	**49.5**

微软Kinect。它包含1449张RGB-D图像，带有像素级标签，这些标签已被Gupta *et al* [13]整合为40类语义分割任务。我们在标准划分（795张训练图像和654张测试图像）上报告结果。（注：所有模型选择均在PASCAL 2011验证集上进行。）表4展示了我们模型几种变体的性能。首先，我们在RGB图像上训练未经修改的粗粒度模型（FCN-32s）。为加入深度信息，我们升级模型以接受四通道RGB-D输入（早期融合）并进行训练。这带来的改进甚微，可能是由于难以在整个模型中传播有效的梯度。借鉴Gupta *et al* [14]的成功经验，我们尝试使用深度的三维HHA编码，仅基于此信息训练网络，同时尝试RGB与HHA的“晚期融合”——将两个网络的预测结果在最终层求和，并以端到端方式学习这个双流网络。最后，我们将该晚期融合网络升级为16步长版本。

SIFT Flow 是一个包含 2,688 张图像的数据集，每张图像带有 33 个语义类别（如“桥梁”、“山脉”、“太阳”）以及三个几何类别（“水平”、“垂直”和“天空”）的像素级标签。全卷积网络（FCN）能够自然地学习一种联合表示，同时预测这两种类型的标签。我们训练了一个双头版本的 FCN-16s 模型，该模型包含语义和几何预测层以及相应的损失函数。学习得到的模型在两项任务上的表现均与两个独立训练的模型相当，同时其训练和推理速度基本与每个独立模型自身相当。表 5 中的结果基于标准划分（2,488 张训练图像和 200 张测试图像）计算得出，{v*} 显示该模型在两项任务上均达到了最先进的性能水平。

[10]Three of the SIFT Flow categories are not present in the test set. We made predictions across all 33 categories, but only included categories actually present in the test set in our evaluation. (An earlier version of this paper reported a lower mean IU, which included all categories either present or predicted in the evaluation.)

表5. 在SIFT Flow[10]数据集上使用类别分割（中）与几何分割（右）的结果。Tighe [33]是一种非参数迁移方法。Tighe 1为示例SVM，而2为SVM + MRF。Farabet是在类别平衡样本(1)或自然频率样本(2)上训练的多尺度卷积网络。Pinheiro为多尺度循环卷积网络，记作RCNN。

$_3$ (∘3)。几何度量的指标是像素精度。

	pixel acc.	mean acc.	mean IU	f.w. IU	geom. acc.
Liu *et al*. [23]	76.7	-	-	-	-
Tighe *et al*. [33]	-	-	-	-	90.8
Tighe *et al*. [34] 1	75.6	41.1	-	-	-
Tighe *et al*. [34] 2	78.6	39.2	-	-	-
Farabet *et al*. [8] 1	72.3	50.8	-	-	-
Farabet *et al*. [8] 2	78.5	29.6	-	-	-
Pinheiro *et al*. [28]	77.7	29.8	-	-	-
FCN-16s	**85.2**	**51.7**	39.5	76.1	**94.3**

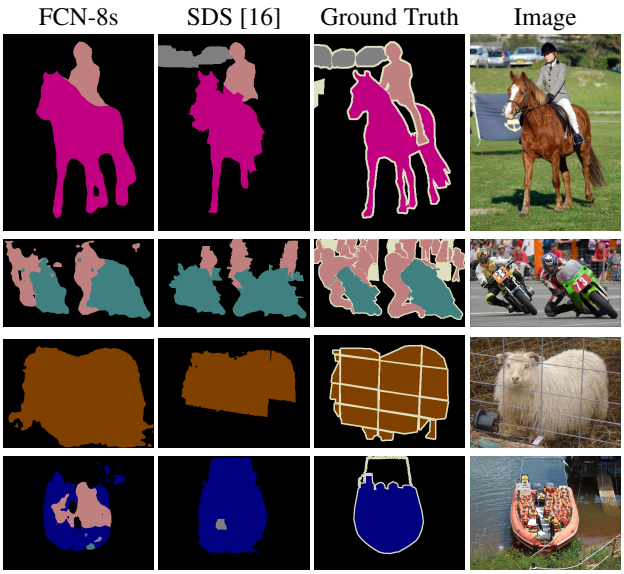

图6. 全卷积分割网络在PASCAL数据集上实现了最先进的性能。左列展示了我们性能最高的网络FCN-8s的输出结果。第二列显示了由Hariharan等人*et al*提出的先前最先进系统产生的分割结果[16]。请注意恢复的精细结构（第一行）、分离紧密交互物体的能力（第二行）以及对遮挡物的鲁棒性（第三行）。第四行展示了一个失败案例：网络将船上的救生衣误识别为人。

6. 结论

全卷积网络是一类丰富的模型，现代分类卷积网络是其中的一个特例。认识到这一点后，将这些分类网络扩展到分割任务，并通过多分辨率层组合改进架构，极大地提升了现有技术水平，同时简化和加速了学习与推理过程。

致谢 本工作部分得到了

by DARPA's MSEE and SMISC programs, NSF awards IIS-1427425, IIS-1212798, IIS-1116411, and the NSF GRFP, Toyota, and the Berkeley Vision and Learning Center. We gratefully acknowledge NVIDIA for GPU donation. We thank Bharath Hariharan and Saurabh Gupta for their advice and dataset tools. We thank Sergio Guadarrama for reproducing GoogLeNet in Caffe. We thank Jitendra Malik for his helpful comments. Thanks to Wei Liu for pointing out an issue wth our SIFT Flow mean IU computation and an error in our frequency weighted mean IU formula.

A. Upper Bounds on IU

In this paper, we have achieved good performance on the mean IU segmentation metric even with coarse semantic prediction. To better understand this metric and the limits of this approach with respect to it, we compute approximate upper bounds on performance with prediction at various scales. We do this by downsampling ground truth images and then upsampling them again to simulate the best results obtainable with a particular downsampling factor. The following table gives the mean IU on a subset of PASCAL 2011 val for various downsampling factors.

factor	mean IU
128	50.9
64	73.3
32	86.1
16	92.8
8	96.4
4	98.5

Pixel-perfect prediction is clearly not necessary to achieve mean IU well above state-of-the-art, and, conversely, mean IU is a not a good measure of fine-scale accuracy.

B. More Results

We further evaluate our FCN for semantic segmentation.

PASCAL-Context [26] provides whole scene annotations of PASCAL VOC 2010. While there are over 400 distinct classes, we follow the 59 class task defined by [26] that picks the most frequent classes. We train and evaluate on the training and val sets respectively. In Table 6, we compare to the joint object + stuff variation of Convolutional Feature Masking [3] which is the previous state-of-the-art on this task. FCN-8s scores 35.1 mean IU for an 11% relative improvement.

Table 6. Results on PASCAL-Context. *CFM* is the best result of [3] by convolutional feature masking and segment pursuit with the VGG net. *O_2P* is the second order pooling method [1] as reported in the *errata* of [26]. The 59 class task includes the 59 most frequent classes while the 33 class task consists of an easier subset identified by [26].

59 class	pixel acc.	mean acc.	mean IU	f.w. IU
O_2P	-	-	18.1	-
CFM	-	-	31.5	-
FCN-32s	63.8	42.7	31.8	48.3
FCN-16s	65.7	46.2	34.8	50.7
FCN-8s	**65.9**	**46.5**	**35.1**	**51.0**
33 class				
O_2P	-	-	29.2	-
CFM	-	-	46.1	-
FCN-32s	69.8	65.1	50.4	54.9
FCN-16s	**71.8**	**68.0**	53.4	57.5
FCN-8s	**71.8**	67.6	**53.5**	**57.7**

Changelog

The arXiv version of this paper is kept up-to-date with corrections and additional relevant material. The following gives a brief history of changes.

v2 Add Appendix A giving upper bounds on mean IU and Appendix B with PASCAL-Context results. Correct PASCAL validation numbers (previously, some val images were included in train), SIFT Flow mean IU (which used an inappropriately strict metric), and an error in the frequency weighted mean IU formula. Add link to models and update timing numbers to reflect improved implementation (which is publicly available).

References

[1] J. Carreira, R. Caseiro, J. Batista, and C. Sminchisescu. Semantic segmentation with second-order pooling. In *ECCV*, 2012. 9

[2] D. C. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. Deep neural networks segment neuronal membranes in electron microscopy images. In *NIPS*, pages 2852–2860, 2012. 1, 2, 4, 7

[3] J. Dai, K. He, and J. Sun. Convolutional feature masking for joint object and stuff segmentation. *arXiv preprint arXiv:1412.1283*, 2014. 9

[4] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell. DeCAF: A deep convolutional activation feature for generic visual recognition. In *ICML*, 2014. 1, 2

[5] D. Eigen, D. Krishnan, and R. Fergus. Restoring an image taken through a window covered with dirt or rain. In *Computer Vision (ICCV), 2013 IEEE International Conference on*, pages 633–640. IEEE, 2013. 2

[6] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. *arXiv preprint arXiv:1406.2283*, 2014. 2

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes

由DARPA的MSEE和SMISC项目、NSF奖项IIS-1427425、IIS-1212798、IIS-1116411，以及NSF GRFP、丰田公司和伯克利视觉与学习中心资助。我们衷心感谢NVIDIA捐赠的GPU。感谢Bharath Hariharan和Saurabh Gupta的建议与数据集工具。感谢Sergio Guadarrama在Caffe中复现GoogLeNet。感谢Jitendra Malik提出的宝贵意见。感谢Wei Liu指出我们SIFT Flow平均IU计算中的一个问题以及频率加权平均IU公式中的一处错误。

A. IU的上界

在本文中，即使使用粗糙的语义预测，我们也在平均IU分割指标上取得了良好性能。为了更好地理解该指标以及此方法在指标上的局限性，我们计算了不同尺度下预测性能的近似上限。具体做法是通过下采样真实标注图像，再重新上采样以模拟特定下采样因子下可获得的最佳结果。下表展示了PASCAL 2011验证集子集在不同下采样因子下的平均IU值。

factor	mean IU
128	50.9
64	73.3
32	86.1
16	92.8
8	96.4
4	98.5

像素级完美预测显然不是实现远高于当前最佳水平的平均IU所必需的，并且反过来，平均IU也不是衡量精细尺度准确性的良好指标。

B. 更多结果

我们进一步评估了用于语义分割的全卷积网络。

PASCAL-Context [26] 提供了对 PASCAL VOC 2010 的完整场景标注。尽管数据集中包含超过 400 个不同的类别，我们遵循 [26] 定义的 59 类任务，选取了最常见的类别。我们分别在训练集和验证集上进行训练与评估。在表 6 中，我们与卷积特征掩码 [3] 的联合物体 {v*} 材质变体进行了比较，后者是此前在此任务上的最先进方法。FCN-8s 取得了 35.1 的平均 IU 分数，实现了 11% 的相对提升。

表6. PASCAL-Context数据集上的结果。*CFM*是文献[3]采用VGG网络通过卷积特征掩码与区域追踪得到的最佳结果。*O_2P*为文献[1]提出的二阶池化方法，其数值引用自文献[26]的*errata*。59类任务包含出现频率最高的59个类别，而33类任务则由文献[26]确定的更简单子集构成。

59 class	pixel acc.	mean acc.	mean IU	f.w. IU
O_2P	-	-	18.1	-
CFM	-	-	31.5	-
FCN-32s	63.8	42.7	31.8	48.3
FCN-16s	65.7	46.2	34.8	50.7
FCN-8s	**65.9**	**46.5**	**35.1**	**51.0**
33 class				
O_2P	-	-	29.2	-
CFM	-	-	46.1	-
FCN-32s	69.8	65.1	50.4	54.9
FCN-16s	**71.8**	**68.0**	53.4	57.5
FCN-8s	**71.8**	67.6	**53.5**	**57.7**

更新日志

本文的arXiv版本会随着修正和补充相关材料而保持更新。以下简要列出变更历史。

v2 增加附录A提供平均IU的上界，以及附录B包含PASCAL-Context结果。修正PASCAL验证数据（此前部分验证图像被包含在训练集中）、SIFT Flow平均IU（原使用了不恰当的严格度量标准）以及频率加权平均IU公式中的一处错误。添加模型链接并更新时序数据以反映已公开的改进实现。

参考文献

[1] J. Carreira, R. Caseiro, J. Batista, 和 C. Sminchisescu。基于二阶池化的语义分割。发表于 *ECCV*, 2012年。9[2] D. C. Ciresan, A. Giusti, L. M. Gambardella, 和 J. Schmidhuber。深度神经网络在电子显微镜图像中分割神经元膜。发表于 *NIPS*, 第2852–2860页, 2012年。1, 2, 4, 7[3] J. Dai, K. He, 和 J. Sun。用于联合物体与背景分割的卷积特征掩码。*arXiv preprint arXiv:1412.1283*, 2014年。9[4] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, 和 T. Darrell。DeCAF：一种用于通用视觉识别的深度卷积激活特征。发表于 *ICML*, 2014年。1, 2[5] D. Eigen, D. Krishnan, 和 R. Fergus。恢复透过覆盖灰尘或雨水的窗户拍摄的图像。发表于 *Computer Vision (ICCV), 2013 IEEE International Conference on*, 第633–640页。IEEE, 2013年。2[6] D. Eigen, C. Puhrsch, 和 R. Fergus。使用多尺度深度网络从单张图像预测深度图。*arXiv preprint arXiv:1406.2283*, 2014年。2[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, 和 A. Zisserman。PASCAL视觉物体类别

Challenge 2011 (VOC2011) Results. http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html. 4

[8] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2013. 1, 2, 4, 7, 8

[9] P. Fischer, A. Dosovitskiy, and T. Brox. Descriptor matching with convolutional neural networks: a comparison to SIFT. *CoRR*, abs/1405.5769, 2014. 1

[10] L. Florack, B. T. H. Romeny, M. Viergever, and J. Koenderink. The gaussian scale-space paradigm and the multiscale local jet. *International Journal of Computer Vision*, 18(1):61–75, 1996. 5

[11] Y. Ganin and V. Lempitsky. N^4-fields: Neural network nearest neighbor fields for image transforms. In *ACCV*, 2014. 1, 2, 7

[12] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Computer Vision and Pattern Recognition*, 2014. 1, 2, 7

[13] S. Gupta, P. Arbelaez, and J. Malik. Perceptual organization and recognition of indoor scenes from RGB-D images. In *CVPR*, 2013. 8

[14] S. Gupta, R. Girshick, P. Arbelaez, and J. Malik. Learning rich features from RGB-D images for object detection and segmentation. In *ECCV*. Springer, 2014. 1, 2, 8

[15] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, and J. Malik. Semantic contours from inverse detectors. In *International Conference on Computer Vision (ICCV)*, 2011. 7

[16] B. Hariharan, P. Arbeláez, R. Girshick, and J. Malik. Simultaneous detection and segmentation. In *European Conference on Computer Vision (ECCV)*, 2014. 1, 2, 4, 5, 7, 8

[17] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014. 1, 2

[18] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014. 7

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 2, 3, 5

[20] Q. V. Le, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, and A. Y. Ng. Building high-level features using large scale unsupervised learning. In *ICML*, 2012. 3

[21] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. Backpropagation applied to hand-written zip code recognition. In *Neural Computation*, 1989. 2, 3

[22] Y. A. LeCun, L. Bottou, G. B. Orr, and K.-R. Müller. Efficient backprop. In *Neural networks: Tricks of the trade*, pages 9–48. Springer, 1998. 7

[23] C. Liu, J. Yuen, and A. Torralba. Sift flow: Dense correspondence across scenes and its applications. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):978–994, 2011. 8

[24] J. Long, N. Zhang, and T. Darrell. Do convnets learn correspondence? In *NIPS*, 2014. 1

[25] O. Matan, C. J. Burges, Y. LeCun, and J. S. Denker. Multi-digit recognition using a space displacement neural network. In *NIPS*, pages 488–495. Citeseer, 1991. 2

[26] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, and A. Yuille. The role of context for object detection and semantic segmentation in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 891–898. IEEE, 2014. 9

[27] F. Ning, D. Delhomme, Y. LeCun, F. Piano, L. Bottou, and P. E. Barbano. Toward automatic phenotyping of developing embryos from videos. *Image Processing, IEEE Transactions on*, 14(9):1360–1371, 2005. 1, 2, 4, 7

[28] P. H. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In *ICML*, 2014. 1, 2, 4, 7, 8

[29] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. Overfeat: Integrated recognition, localization and detection using convolutional networks. In *ICLR*, 2014. 1, 2, 3, 4

[30] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 7

[31] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014. 1, 2, 3, 5

[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. 1, 2, 3, 5

[33] J. Tighe and S. Lazebnik. Superparsing: scalable nonparametric image parsing with superpixels. In *ECCV*, pages 352–365. Springer, 2010. 8

[34] J. Tighe and S. Lazebnik. Finding things: Image parsing with regions and per-exemplar detectors. In *CVPR*, 2013. 8

[35] J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. *CoRR*, abs/1406.2984, 2014. 2

[36] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1058–1066, 2013. 4

[37] R. Wolf and J. C. Platt. Postal address block location using a convolutional locator network. *Advances in Neural Information Processing Systems*, pages 745–745, 1994. 2

[38] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *Computer Vision–ECCV 2014*, pages 818–833. Springer, 2014. 2

[39] N. Zhang, J. Donahue, R. Girshick, and T. Darrell. Part-based r-cnns for fine-grained category detection. In *Computer Vision–ECCV 2014*, pages 834–849. Springer, 2014. 1

挑战赛2011（VOC2011）结果。http://www.pascal-network.org/challenges/VOC/voc2011/workshop/index.html。4 [8] C. Farabet, C. Couprie, L. Najman, 和 Y. LeCun。用于场景标注的层次特征学习。*Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 2013。1, 2, 4, 7, 8 [9] P. Fischer, A. Dosovitskiy, 和 T. Brox。使用卷积神经网络进行描述符匹配：与SIFT的比较。*CoRR*, abs/1405.5769, 2014。1 [10] L. Florack, B. T. H. Romeny, M. Viergever, 和 J. Koenderink。高斯尺度空间范式与多尺度局部jet。*International Journal of Computer Vision*, 18(1):61–75, 1996。5 [11] Y. Ganin 和 V. Lempitsky。N^4-场：用于图像变换的神经网络最近邻场。收录于 *ACCV*, 2014。1, 2, 7 [12] R. Girshick, J. Donahue, T. Darrell, 和 J. Malik。用于精确目标检测和语义分割的丰富特征层次结构。收录于 *Computer Vision and Pattern Recognition*, 2014。1, 2, 7 [13] S. Gupta, P. Arbelaez, 和 J. Malik。从RGB-D图像中进行室内场景的感知组织与识别。收录于 *CVPR*, 2013。8 [14] S. Gupta, R. Girshick, P. Arbelaez, 和 J. Malik。从RGB-D图像中学习丰富特征以进行目标检测与分割。收录于 *ECCV*。Springer, 2014。1, 2, 8 [15] B. Hariharan, P. Arbelaez, L. Bourdev, S. Maji, 和 J. Malik。来自逆向检测器的语义轮廓。收录于 *International Conference on Computer Vision (ICCV)*, 2011。7 [16] B. Hariharan, P. Arbeláez, R. Girshick, 和 J. Malik。同步检测与分割。收录于 *European Confer- ence on Computer Vision (ECCV)*, 2014。1, 2, 4, 5, 7, 8 [17] K. He, X. Zhang, S. Ren, 和 J. Sun。用于视觉识别的深度卷积网络中的空间金字塔池化。收录于 *ECCV*, 2014。1, 2 [18] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, 和 T. Darrell。Caffe：用于快速特征嵌入的卷积架构。*arXiv preprint arXiv:1408.5093*, 2014。7 [19] A. Krizhevsky, I. Sutskever, 和 G. E. Hinton。使用深度卷积神经网络进行ImageNet分类。收录于 *NIPS*, 2012。1, 2, 3, 5 [20] Q. V. Le, R. Monga, M. Devin, K. Chen, G. S. Corrado, J. Dean, 和 A. Y. Ng。使用大规模无监督学习构建高层特征。收录于 *ICML*, 2012。3 [21] Y. LeCun, B. Boser, J. Denker, D. Henderson, R. E. Howard, W. Hubbard, 和 L. D. Jackel。反向传播应用于手写邮政编码识别。收录于 *Neural Computation*, 1989。2, 3 [22] Y. A. LeCun, L. Bottou, G. B. Orr, 和 K.-R. Müller。高效反向传播。收录于 *Neural networks: Tricks of the trade*, 第9–48页。Springer, 1998。7 [23] C. Liu, J. Yuen, 和 A. Torralba。SIFT流：跨场景的密集对应及其应用。*Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 33(5):978–994, 2011。8 [24] J. Long, N. Zhang, 和 T. Darrell。卷积网络学习对应关系吗？发表于 *NIPS*，2014年。1[25] O. Matan, C. J. Burges, Y. LeCun, 和 J. S. Denker。使用空间位移神经网络进行多位数识别。发表于 *NIPS*，第488–495页。Citeseer，1991年。2[26] R. Mottaghi, X. Chen, X. Liu, N.-G. Cho, S.-W. Lee, S. Fidler, R. Urtasun, 和 A. Yuille。野外环境中上下文对于目标检测和语义分割的作用。发表于 *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*，第891–898页。IEEE，2014年。9[27] F. Ning, D. Delhomme, Y. LeCun, F. Piano, L. Bottou, 和 P. E. Barbano。基于视频对发育胚胎进行自动表型分析。*Image Processing, IEEE Transactions on*，14(9):1360–1371，2005年。1, 2, 4, 7[28] P. H. Pinheiro 和 R. Collobert。用于场景标注的循环卷积神经网络。发表于 *ICML*，2014年。1, 2, 4, 7, 8[29] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, 和 Y. LeCun。Overfeat：使用卷积网络进行集成识别、定位和检测。发表于 *ICLR*，2014年。1, 2, 3, 4[30] N. Silberman, D. Hoiem, P. Kohli, 和 R. Fergus。从RGBD图像进行室内分割和支持推理。发表于 *ECCV*，2012年。7[31] K. Simonyan 和 A. Zisserman。用于大规模图像识别的极深度卷积网络。*CoRR*，abs/1409.1556，2014年。1, 2, 3, 5[32] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, 和 A. Rabinovich。更深入的卷积网络。*CoRR*，abs/1409.4842，2014年。1, 2, 3, 5[33] J. Tighe 和 S. Lazebnik。超解析：使用超像素的可扩展非参数图像解析。发表于 *ECCV*，第352–365页。Springer，2010年。8[34] J. Tighe 和 S. Lazebnik。寻找物体：使用区域和逐样本检测器的图像解析。发表于 *CVPR*，2013年。8[35] J. Tompson, A. Jain, Y. LeCun, 和 C. Bregler。用于人体姿态估计的卷积网络与图模型的联合训练。*CoRR*，abs/1406.2984，2014年。2[36] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, 和 R. Fergus。使用DropConnect正则化神经网络。发表于 *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*，第1058–1066页，2013年。4[37] R. Wolf 和 J. C. Platt。使用卷积定位网络定位邮政地址块。*Advances in Neural Infor- mation Processing Systems*，第745–745页，1994年。2[38] M. D. Zeiler 和 R. Fergus。可视化和理解卷积网络。发表于 *Computer Vision–ECCV 2014*，第818–833页。Springer，2014年。2[39] N. Zhang, J. Donahue, R. Girshick, 和 T. Darrell。用于细粒度类别检测的基于部分的R-CNN。发表于 *Com- puter Vision–ECCV 2014*，第834–849页。Springer，2014年。1