ImageNet Classification with Deep Convolutional Neural Networks

Alex Krizhevsky, Ilya Sutskever, Geoffrey E. Hinton

翻译：莫天池

版本号：V1.0.0

2016 年 3 月

ImageNet Classification with Deep Convolutional Neural Networks

Abstract

We trained a large, deep convolutional neural network to classify the 1.2 million high-resolution images in the ImageNet LSVRC-2010 contest into the 1000 different classes. On the test data, we achieved top-1 and top-5 error rates of 37.5% and 17.0% which is considerably better than the previous state-of-the-art. The neural network, which has 60 million parameters and 650,000 neurons, consists of five convolutional layers, some of which are followed by max-pooling layers, and three fully-connected layers with a final 1000-way softmax. To make training faster, we used non-saturating neurons and a very efficient GPU implementation of the convolution operation. To reduce overfitting in the fully-connected layers we employed a recently-developed regularization method called "dropout" that proved to be very effective. We also entered a variant of this model in the ILSVRC-2012 competition and achieved a winning top-5 test error rate of 15.3%, compared to 26.2% achieved by the second-best entry.

摘要

我们训练了一个大型深度卷积神经网络来将 ImageNet LSVRC-2010 数据集中的 120 万张高清图片分到 1000 个不同的类别中。在测试数据中，我们将 Top-1 错误（分配的第一个类错误）和 Top-5 错误（分配的前五个类全错）分别降到了 37.5%和 17.0%，这比之前的技术水平要好得多。这个神经网络拥有 6 千万的参数和 65 万个神经元，共有五个卷积层，其中一些卷积层后面跟着最大池化层，还有利用 softmax 函数进行 1000 类分类的最后三个全连接层。为了让训练速度更快，我们使用不饱和【？】神经元，并利用高效的 GPU 实现卷积操作。为了减少全连接层的过拟合，我们采用了一种最近研发出来的正则化方法——"DROPOUT"，它被证明十分有效。我们也在比赛中加入了这一模型的一个变体，第二名的 26.2%相比，我们通过将 TOP-5 错误降到了 15.3%而获胜。

1 Introduction

Current approaches to object recognition make essential use of machine learning methods. To improve their performance, we can collect larger datasets, learn more powerful models, and use better techniques for preventing overfitting. Until recently, datasets of labeled images were relatively small — on the order of tens of thousands of images (e.g., NORB [16], Caltech-101/256 [8, 9], and CIFAR-10/100 [12]). Simple recognition tasks can be solved quite well with datasets of this size, especially if they are

1 引言

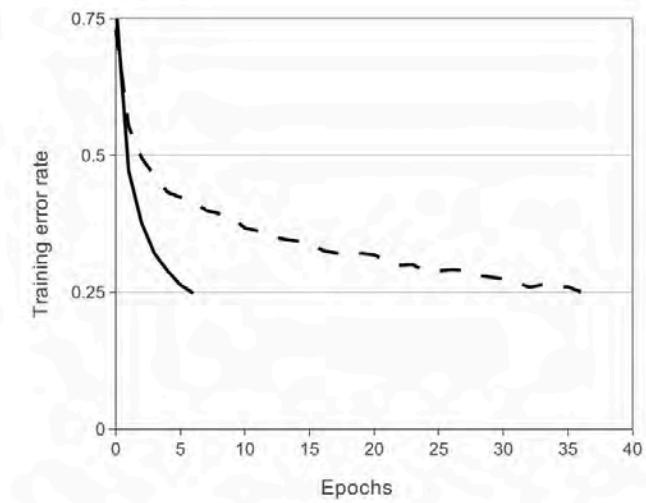
最近的物体识别方法都应用了很重要的机器学习方法，为了提高他们的表现，我们可以收集更大的数据集，学习训练更强大的模型，并用更好的技术来避免过拟合。直到最近，有标签的数据集都是相对较小的，一般只有万张的数量级（比如【16，8，9，12】）。单一的认知任务可以在这个数量级的数据集上得到很好地解决，特别是当其通过标签保存变形技术被放大的时候。比如，现在在 MNIST 数据集上最低的数字识别错误率已经接近了人类的认知水平（<0.3%）【4】。但是模型识别现实背景中物体的能力就表现得不太稳定

augmented with label-preserving transformations. For example, the current best error rate on the MNIST digit-recognition task (<0.3%) approaches human performance [4]. But objects in realistic settings exhibit considerable variability, so to learn to recognize them it is necessary to use much larger training sets. And indeed, the shortcomings of small image datasets have been widely recognized (e.g., Pinto et al. [21]), but it has only recently become possible to collect labeled datasets with millions of images. The new larger datasets include LabelMe [23], which consists of hundreds of thousands of fully-segmented images, and ImageNet [6], which consists of over 15 million labeled high-resolution images in over 22,000 categories.	了，所以为了训练识别这些物体提供大量的数据集是很有必要的。实际上，使用小数据集的缺陷已经被普遍认同了，但直到最近收集百万有标签图片的数据集才成为可能。这些新的大型数据集包括 LabelMe【23】（包含大量被完全分割的图片），还有 ImageNet【6】（由 1500 万张被标记的高清图片组成，覆盖了 2.2 万个类别）。
To learn about thousands of objects from millions of images, we need a model with a large learning capacity. However, the immense complexity of the object recognition task means that this problem cannot be specified even by a dataset as large as ImageNet, so our model should also have lots of prior knowledge to compensate for all the data we don't have. Convolutional neural networks (CNNs) constitute one such class of models [16, 11, 13, 18, 15, 22, 26]. Their capacity can be con- trolled by varying their depth and breadth, and they also make strong and mostly correct assumptions about the nature of images (namely, stationarity of statistics and locality of pixel dependencies). Thus, compared to standard feedforward neural networks with similarly-sized layers, CNNs have much fewer connections and parameters and so they are easier to train, while their theoretically-best performance is likely to be only slightly worse.	为了从百万张图片中学习到数千个物体，我们需要一个有强大学习能力的模型。然而，物体识别任务极高的复杂度意味着即使拥有 ImageNet 这么大的数据集，这个问题也很难被具体化。所以我们的模型也需要大量先验知识去补全所有缺失数据。卷积神经网络（CNNs）就是一种这样的模型[16, 11, 13, 18, 15, 22, 26]。它们的学习能力可以通过控制网络的深度和宽度来调整，它们也可以对图片的本质（高层属性）做出强大而且基本准确的假设（统计上的稳定性，以及像素依赖的局部性特征）。因此，与同样大小的标准的前馈神经网相比，CNNs 有更少的连接、参数，所以更易于训练，而且 CNNs 的理论最佳表现仅比前馈神经网络稍差。

Despite the attractive qualities of CNNs, and despite the relative efficiency of their local architecture, they have still been prohibitively expensive to apply in large scale to high-resolution images. Luckily, current GPUs, paired with a highly-optimized implementation of 2D convolution, are powerful enough to facilitate the training of interestingly-large CNNs, and recent datasets such as ImageNet contain enough labeled examples to train such models without severe overfitting.	虽然 CNNs 质量很好，而且对于局部结构非常高效，但其应用代价对于大量的高清图片而言还是昂贵到可怕。幸运的是，最近，GPU 可以被应用于高度优化的 2D 卷积的实现，它们足够强大，能够加速大型 CNNs 的训练过程。而且最近的数据集比如 ImageNet 包含了足量的有标签样本，可以用来训练这些模型，而没有太严重的过拟合。
The specific contributions of this paper are as follows: we trained one of the largest convolutional neural networks to date on the subsets of ImageNet used in the ILSVRC-2010 and ILSVRC-2012 competitions [2] and achieved by far the best results ever reported on these datasets. We wrote a highly-optimized GPU implementation of 2D convolution and all the other operations inherent in training convolutional neural networks, which we make available publicly. Our network contains a number of new and unusual features which improve its performance and reduce its training time, which are detailed in Section 3. The size of our network made overfitting a significant problem, even with 1.2 million labeled training examples, so we used several effective techniques for preventing overfitting, which are described in Section 4. Our final network contains five convolutional and three fully-connected layers, and this depth seems to be important: we found that removing any convolutional layer (each of which contains no more than 1% of the model's parameters) resulted in inferior performance.	本文的主要贡献包括：我们在 ImageNet 的 2010 和 2012 数据集集上训练了最大的 CNNs 之一，并且达到了迄今为止最好的结果。我们编写了一个高度优化的 2D 卷积的 GPU 实现，以及其他所有训练 CNNs 的固有操作，并将其公之于众。我们的网络包含一系列新的不同凡响的特征，这提高了它的表现性能，减少了它的训练时间，具体情况在第三章介绍。 即使我们拥有 120 万的标签样例，我们的网络的巨大体积也使得过拟合成了一个严重的问题，所以我们需要一系列技术去克服过拟合，这将在第四章中描述。 我们的网络最终包含 5 个卷积层和 3 个全连接层，这个深度也许是很重要的：我们发现去掉任意一个卷积层都会导致更差的表现，即使每个卷积层仅包含不到 1%的模型参数。
In the end, the network's size is limited mainly by	最后，网络的大小主要被 GPU 中可获得的存储数

the amount of memory available on current GPUs and by the amount of training time that we are willing to tolerate. Our network takes between five and six days to train on two GTX 580 3GB GPUs. All of our experiments suggest that our results can be improved simply by waiting for faster GPUs and bigger datasets to become available.	量，以及可忍受训练时间所限制。 我们的网络需要在两台 GTX 580 3GB GPUs 训练五至六天。我们所有的实验都表明，只要等到更快的 GPU 和更大的数据集出现，其结果能够被进一步提高。
2 The Dataset ImageNet is a dataset of over 15 million labeled high-resolution images belonging to roughly 22,000 categories. The images were collected from the web and labeled by human labelers using Amazon's Mechanical Turk crowd-sourcing tool. Starting in 2010, as part of the Pascal Visual Object Challenge, an annual competition called the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) has been held. ILSVRC uses a subset of ImageNet with roughly 1000 images in each of 1000 categories. In all, there are roughly 1.2 million training images, 50,000 validation images, and 150,000 testing images.	**2 数据集** ImageNet 是一个拥有超过 1500 万张带标签的高清图片的数据集，这些图片大约属于 2.2 万个类别。这些图片收集自网络并由亚马逊的 Turk 群智工具进行人工标记。从 2010 年开始，作为帕斯卡物体可视化挑战的一部分，一项被称为 ILSVRC 的比赛每年都会进行。 ILSVRC 使用 ImageNet 的一个子集，这个子集包含大约 1000 个类别，每个类别大概包涵 1000 张图。总共大概有 120 万张训练图片，5 万张验证图片和 15 万张测试图片。
ILSVRC-2010 is the only version of ILSVRC for which the test set labels are available, so this is the version on which we performed most of our experiments. Since we also entered our model in the ILSVRC-2012 competition, in Section 6 we report our results on this version of the dataset as well, for which test set labels are unavailable. On ImageNet, it is customary to report two error rates: top-1 and top-5, where the top-5 error rate is the fraction of test images for which the correct label is not among the five labels considered most probable by the model.	2010 年的 ILSVRC 数据集是唯一一个测试集标签可得的版本，所以我们用它进行我们的大部分实验。 因为我们也把我们的模型加入了 2012 年的 ILSVRC 比赛，所以我们在第六章也讨论了这个数据集上的实验结果，但其测试集标签不可得，在 ImageNet 上，通常检验两类错误率：TOP-1 和 TOP-5，TOP-5 错误表示测试图片的标签不在模型所认为的 可能性最大的五个标签中。
ImageNet consists of variable-resolution images, while our system requires a constant input	ImageNet 包含各种清晰度的图片，而我们的系统要求输入维度恒定，因此，我们对图片进行采样，

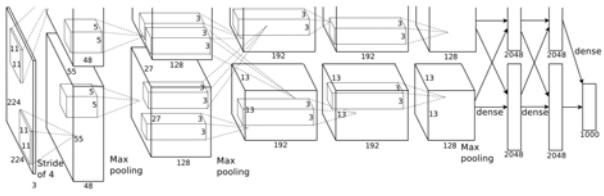
dimensionality. Therefore, we down-sampled the images to a fixed resolution of 256X256. Given a rectangular image, we first rescaled the image such that the shorter side was of length 256, and then cropped out the central 256X256 patch from the resulting image. We did not pre-process the images in any other way, except for subtracting the mean activity over the training set from each pixel. So we trained our network on the (centered) raw RGB values of the pixels.	获得固定大小的 256X256 的分辨率，对于每张长方形的图，我们将短边按比例调整为 256，然后取中心区域的 256X256 像素。我们并未使用其他方法对图片进行预处理，除了把每个像素减去整个训练集的平均值【？？？ except for subtracting the mean activity over the training set from each pixel.】所以我们的模型是在原始的 RGB 像素值上训练出来的。
3 The Architecture The architecture of our network is summarized in Figure 2. It contains eight learned layers — five convolutional and three fully-connected. Below, we describe some of the novel or unusual features of our network's architecture. Sections 3.1-3.4 are sorted according to our estimation of their importance, with the most important first.	**3 模型体系结构** 网络的体系结构如图 2.它包含 8 个学习层——五个卷积层 3 个全连接层。接下来，我们讨论一些我们的网络中创新的，或者不常见的结构。3.1~3.4 节按照我们心目中对它们重要性的评估进行排序，越重要越靠前。
3.1 ReLU Nonlinearity The standard way to model a neuron's output f as a function of its input x is with $f(x) = \tanh(x)$ or $f(x) = (1 + e^{-x})^{-1}$. In terms of training time with gradient descent, these saturating nonlinearities are much slower than the non-saturating nonlinearity $f(x) = \max(0,x)$. Following Nair and Hinton [20], we refer to neurons with this nonlinearity as Rectified Linear Units (ReLUs). Deep convolutional neural networks with ReLUs train several times faster than their equivalents with tanh units. This is demonstrated in Figure 1, which shows the number of iterations required to reach 25% training error on the CIFAR-10 dataset for a particular four-layer convolutional network. This plot shows that we would not have been able to experiment with such large neural networks for this work if we had used traditional saturating	3.1 ReLU 非线性 对神经元输出 f 的标准建模方法是将输入 x 函数变换为 $f(x) = \tanh(x)$或 $f(x) = (1 + e^{-x})^{-1}$。 从梯度下降的训练时间而言，这种饱和【？saturating】的非线性比使用非饱和的非线性 $f(x) = \max(0,x)$要慢得多。根据 Nair 和 Hinton 说法【20】，我们让神经元使用这种非线性——修正线性单元（ReLUs）。使用 ReLU 的深度卷积神经网络比使用 tanh 的网络训练速度快几倍。 图一展示了一个特定的四层 CNN 在 CIFAR-10 数据集上达到 25%训练错误所需要的迭代次数。 这张图显示，如果我们采用传统的饱和神经元，我们将不可能为这项工作训练如此庞大的神经网络。

neuron models.	
We are not the first to consider alternatives to traditional neuron models in CNNs. For example, Jarrett et al. [11] claim that the nonlinearity f (x) = |tanh(x)| works particularly well with their type of contrast normalization followed by local average pooling on the Caltech-101 dataset. However, on this dataset the primary concern is preventing overfitting, so the effect they are observing is different from the accelerated ability to fit the training set which we report when using ReLUs. Faster learning has a great influence on the performance of large models trained on large datasets.	我们并不是最早考虑替换传统 CNN 神经元模型的人。比如，J【11】等人宣称利用 f (x) = |tanh(x)| 非线性在 Caltech-101 数据集上做对比度归一化（Contrast Normalization，CN）和局部平均值池化表现得很好。 然而，关于这个数据集的主要问题是要防止过拟合，所以他们观察到的效果，与我们报告的使用 ReLUs 时对训练集的适应（fit）累积能力不同。更快的学习对于在大型数据集上训练大型模型的表现有重大影响。
 Figure 1: A four-layer convolutional neural network with ReLUs (**solid line**) reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons (**dashed line**). The learning rates for each network were chosen independently to make training as fast as possible. No regularization of any kind was employed. The magnitude of the effect demonstrated here varies with network architecture, but networks with ReLUs consistently learn several times faster than equivalents with saturating neurons.	
3.2 Training on Multiple GPUs A single GTX 580 GPU has only 3GB of memory, which limits the maximum size of the networks	**3.2 多 GPU 并行训练** 单个的 GTX580GPU 只有 3G 的存储空间，这会限制能够在其上训练的网络大小。充分训练网络需

that can be trained on it. It turns out that 1.2 million training examples are enough to train networks which are too big to fit on one GPU. Therefore we spread the net across two GPUs. Current GPUs are particularly well-suited to cross-GPU parallelization, as they are able to read from and write to one another's memory directly, without going through host machine memory. The parallelization scheme that we employ essentially puts half of the kernels (or neurons) on each GPU, with one additional trick: the GPUs communicate only in certain layers. This means that, for example, the kernels of layer 3 take input from all kernel maps in layer 2. However, kernels in layer 4 take input only from those kernel maps in layer 3 which reside on the same GPU. Choosing the pattern of connectivity is a problem for cross-validation, but this allows us to precisely tune the amount of communication until it is an acceptable fraction of the amount of computation.	要 120 万张训练样本图，这对于一个 GPU 而言量太大了，所以我们将网络分布在两个 GPU 上。 现在的 GPU 非常适合做跨 GPU 并行运算，因为它们可以直接向彼此的存储中做读写操作，而无需通过宿主机存储。 我们采用的这种并行模式主要是将各一半的网络内核（或神经元）放在每个 GPU 上，然后再采用一个小技巧：将 GPU 通信限制在某些特定的层上。这意味着，比如，第三层的内核从所有的第二层内核映射（kernel map）中获得输入，但是，第四层的内核只从和自己在同一个 GPU 上的第三层内核中获得输入。 选择一种连接模式对于交互验证是个问题，但这允许我们精确调整连接的数量，直到计算量落入一个可接受的范围内。
The resultant architecture is somewhat similar to that of the "columnar" CNN employed by Ciresan et al. [5], except that our columns are not independent (see Figure 2). This scheme reduces our top-1 and top-5 error rates by 1.7% and 1.2%, respectively, as compared with a net with half as many kernels in each convolutional layer trained on one GPU. The two-GPU net takes slightly less time to train than the one-GPU net.	由此产生的结构会和所谓的"柱状（columnar）"CNN 有些类似（由 Ciresan 等人【5】开发），只是我们的"柱子"不是独立的（见图 2）。与用一个 GPU 训练每个卷积层只有一半的内核的网络相比，这种结构将我们的 TOP-1 错误和 TOP-5 错误分别降低了 1.7%和 1.2%。双 GPU 结构网络比单 GPU 网络所需的训练时间要稍微少一些。
3.3 Local Response Normalization ReLUs have the desirable property that they do not require input normalization to prevent them from saturating. If at least some training examples produce a positive input to a ReLU, learning will happen in that neuron. However, we	**3.3 局部反应归一化（Local Response Normalization）** ReLUs 有一个很赞的属性，它们无需对输入数据进行归一化来避免其饱和【？】。如果至少有一些训练样例为 ReLU 产生了正输入，那么这个神经元就会进行学习。

still find that the following local normalization scheme aids generalization. Denoting by $a^i_{x,y}$ the activity of a neuron computed by applying kernel i at position (x, y) and then applying the ReLU nonlinearity, the response-normalized activity $b^i_{x,y}$ is given by the expression	然而，我们还是发现下面这种归一化的模式能够更好地泛化。 设由第 i 个内核计算(x, y)位置的 ReLU 非线性的活动为 $a^i_{x,y}$，反应归一化（response-normalized）活动 $b^i_{x,y}$ 如下公式所示： $$b^i_{x,y} = a^i_{x,y} / \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^{\beta}$$
where the sum runs over n "adjacent" kernel maps at the same spatial position, and N is the total number of kernels in the layer. The ordering of the kernel maps is of course arbitrary and determined before training begins. This sort of response normalization implements a form of lateral inhibition inspired by the type found in real neurons, creating competition for big activities amongst neuron outputs computed using different kernels. The constants k, n, α, and β are hyper-parameters whose values are determined using a validation set; we used k = 2, n = 5, $\alpha = 10^{-94}$, and 9 β = 0.75. We applied this normalization after applying the ReLU nonlinearity in certain layers (see Section 3.5).	其中，累加公式中的 n 表示同一空间【GPU？】上邻接于该位置的所有内核映射的数量，N 表示这一层的所有内核数。内核映射的顺序当然是任意的，并且是在训练之前就定好了的。 这种反应归一化（response normalization）实现了一种模仿生物神经元的横向抑制【层间不通讯？？】，让神经元在利用不同内核进行计算的大规模活动中产生竞争。常数 k，n，α 和 β 是超系数，它们的值由验证集决定。我们取 k = 2, n = 5, $\alpha = 10^{-4}$, and β = 0.75.我们在特定层使用 ReLU 非线性之后应用这种归一化（见 3.5）
This scheme bears some resemblance to the local contrast normalization scheme of Jarrett et al. [11], but ours would be more correctly termed "brightness normalization", since we do not subtract the mean activity. Response normalization reduces our top-1 and top-5 error rates by 1.4% and 1.2%, respectively. We also verified the effectiveness of this scheme on the CIFAR-10 dataset: a four-layer CNN achieved a 13% test error rate without normalization and 11% with normalization.	这种模式与 J【11】提出的局部对比度归一化有点类似，但我们的方法更准确的描述应该是亮度归一化，因为我们并不减去均值。 反应归一化将我们的 TOP-1 和 TOP-5 错误分别降低了 1.4%和 1.2%。我们还在 CIFAR-10 数据集上验证了该模式的效果：四层 CNN 不用归一化错误率为 13%，用了之后降到了 11%。
3.4 Overlapping Pooling Pooling layers in CNNs summarize the outputs of	**3.4 重叠池化** CNN 中的池化层负责对同一内核映射中相邻的神

neighboring groups of neurons in the same kernel map. Traditionally, the neighborhoods summarized by adjacent pooling units do not overlap (e.g., [17, 11, 4]). To be more precise, a pooling layer can be thought of as consisting of a grid of pooling units spaced s pixels apart, each summarizing a neighborhood of size z * z centered at the location of the pooling unit. If we set s = z, we obtain traditional local pooling as commonly employed in CNNs. If we set s < z, we obtain overlapping pooling. This is what we use throughout our network, with s = 2 and z = 3. This scheme reduces the top-1 and top-5 error rates by 0.4% and 0.3%, respectively, as compared with the non-overlapping scheme s = 2, z = 2, which produces output of equivalent dimensions. We generally observe during training that models with overlapping pooling find it slightly more difficult to overfit.	经元组的输出求和【？summarize the outputs of neighboring groups of neurons in the same kernel map.】。一般地，被邻接的池化单元求和的邻居节点是没有重复的【17，11，4】。 为了更加精确，一个池化层可以看做由相隔 s 个像素占据的池化单元组成的网格所构成，每个单元负责对相邻的 z*z 范围的中心区域求和。若设 s=z，我们就能够获得用于大多数 CNN 的传统的局部池化方法。若设 s<z，我们就得到了有重叠的池化。 这就是我们在自己的网络中使用的方法，s=2，z=3. 与无重叠的 s=z=2 相比，这一模式在产生相同维度的输出时分别将 TOP1 和 TOP5 降低了 0.4%和 0.3%。 我们还观察到，采用有重叠的池化能稍稍让模型更难过拟合。
3.5 Overall Architecture Now we are ready to describe the overall architecture of our CNN. As depicted in Figure 2, the net contains eight layers with weights; the first five are convolutional and the remaining three are fully- connected. The output of the last fully-connected layer is fed to a 1000-way softmax which produces a distribution over the 1000 class labels. Our network maximizes the multinomial logistic regression objective, which is equivalent to maximizing the average across training cases of the log-probability of the correct label under the prediction distribution.	**3.5 整体结构** 现在我们可以描述我们的 CNN 的整体结构了。如图 2，这个网络包含 8 个加权的层：前五个是卷积层，后三个是全连接层。 最后一个全连接层输出一个 1000 维的 softmax 来表达对于 1000 个类别的预测。 我们的网络采取取最大值的多标量罗吉斯回归，它与 maximizing the average across training cases of the log-probability of the correct label under the prediction distribution【？】等价
The kernels of the second, fourth, and fifth convolutional layers are connected only to those kernel maps in the previous layer which reside on the same GPU (see Figure 2). The kernels of the	第 2、4、5 个卷积层的内核只与前一层与自己同在一个 GPU 上的内核映射相连接。 第三层的内核与全部的第二层内核映射相连接。 全连接层的神经元与上层神经元全都有连接。

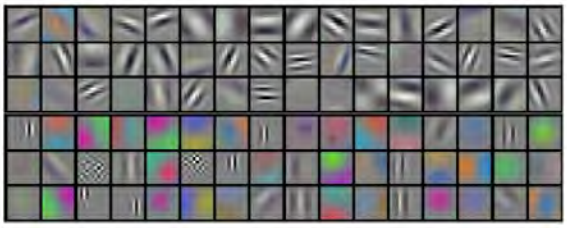
third convolutional layer are connected to all kernel maps in the second layer. The neurons in the fully-connected layers are connected to all neurons in the previous layer. Response-normalization layers follow the first and second convolutional layers. Max-pooling layers, of the kind described in Section 3.4, follow both response-normalization layers as well as the fifth convolutional layer. The ReLU non-linearity is applied to the output of every convolutional and fully-connected layer.	反应归一化层跟在第二个卷积层后面。最大值池化层（如 3.4 所讨论的）跟在反应归一化层后面和第五个卷积层后面。ReLU 非线性被应用在每个卷积层和全连接层
The first convolutional layer filters the 224 × 224 × 3 input image with 96 kernels of size 11 × 11 × 3 with a stride of 4 pixels (this is the distance between the receptive field centers of neighboring neurons in a kernel map). The second convolutional layer takes as input the (response-normalized and pooled) output of the first convolutional layer and filters it with 256 kernels of size 5 × 5 × 48. The third, fourth, and fifth convolutional layers are connected to one another without any intervening pooling or normalization layers. The third convolutional layer has 384 kernels of size 3 × 3 × 256 connected to the (normalized, pooled) outputs of the second convolutional layer. The fourth convolutional layer has 384 kernels of size 3 × 3 × 192 , and the fifth convolutional layer has 256 kernels of size 3 × 3 × 192. The fully-connected layers have 4096 neurons each.	第一个卷积层的输入是 224 × 224 × 3【注：3 是 RGB】的图像，然后用 96 个 11 × 11 × 3 的步长为 4 像素的内核去过滤（步长是相邻神经元感知区域中心之间的距离）。 第二个卷积层将第一个卷积层的输出作为输入（反应归一化并池化），然后用 256 个 5 × 5 × 48 的内核进行过滤。 第三、四、五层卷积层前后相连，之间没有池化层和归一化层。第三个卷积层有 384 个 3 × 3 × 256 的内核，连接着第二个卷积层的输出（归一化+池化）。第四个卷积层有 384 个 3 × 3 × 192 的内核，第五个卷积层有 256 个 3 × 3 × 192 的内核。每个全连接层各有 4096 个神经元。
 Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of	图 2：一个关于我们的 CNN 的结构的描述，明确地勾勒出两个 GPU 之间的对应关系。一个 GPU 运行某一个层 画在这幅图上部的那部分 的同时，另一个 GPU 会运行同一层 画在这幅图下部的那部分。两个 GPU 只在特定的层通讯。

responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input is 150,528-dimensional, and the number of neurons in the network's remaining layers is given by 253,440–186,624–64,896–64,896–43,264– 4096–4096–1000.	网络的输入是 150,528 维的，而除输入层外，余下五个卷积层和三个全连接层分别有 253440，186624，64896，64896，43264，4096，4096，1000 个神经元。
4 Reducing Overfitting Our neural network architecture has 60 million parameters. Although the 1000 classes of ILSVRC make each training example impose 10 bits of constraint on the mapping from image to label, this turns out to be insufficient to learn so many parameters without considerable overfitting. Below, we describe the two primary ways in which we combat overfitting.	**4. 减少过拟合** 我们的神经网络拥有 6000 万的参数，虽然 ILSVRC 的 1000 个类别将从图片到标签的映射限制在 10 个 bits，这依然不足以训练这么多的参数而不造成过拟合。下面，我们将介绍两种对付过拟合的基本方法。
4.1 Data Augmentation The easiest and most common method to reduce overfitting on image data is to artificially enlarge the dataset using label-preserving transformations (e.g., [25, 4, 5]). We employ two distinct forms of data augmentation, both of which allow transformed images to be produced from the original images with very little computation, so the transformed images do not need to be stored on disk. In our implementation, the transformed images are generated in Python code on the CPU while the GPU is training on the previous batch of images. So these data augmentation schemes are, in effect, computationally free.	**4.1 数据集放大** 最简单最常用的减少过拟合的方法就是利用标签保存变形技术人工放大数据集【如文献 25，4，5】。我们采取了两种不同形式的数据放大，它们都允许在仅对原图做少量计算的情况下产生变形的新图，所以变形后的新图无需存储在硬盘中。 在我们的实现中，变形的新图由 Python 在 CPU 上计算产生，与此同时，GPU 仍在计算其他的之前批次的图片。所以这种放大数据集的方式是很高效很节省计算资源的。
The first form of data augmentation consists of generating image translations and horizontal reflections. We do this by extracting random 224 X 224 patches (and their horizontal reflections)	第一种放大数据集（产生新图）的方式由图片翻译【？translations】和水平镜像组成，我们通过从 256X256 的图片中随机抽取 224X224 的区块（及其水平镜像）来实现这种方法，并在这些抽

from the 256X256 images and training our network on these extracted patches. This increases the size of our training set by a factor of 2048, though the resulting training examples are, of course, highly interdependent. Without this scheme, our network suffers from substantial overfitting, which would have forced us to use much smaller networks. At test time, the network makes a prediction by extracting five 224 x 224 patches (the four corner patches and the center patch) as well as their horizontal reflections (hence ten patches in all), and averaging the predictions made by the network's softmax layer on the ten patches.	取后得到的区块上训练我们的神经网络。 这种方法为我们的训练集增加了 2048 个因子【？This increases the size of our training set by a factor of 2048】，虽然这些生成的训练图片明显是相互关联的。如果不采用这种方法，我们的网络会出现严重的过拟合，进而迫使我们采用更小的网络。在测试过程中，网络会抽取五个（四角和中间）224 x 224 的区块及其水平镜像进行预测，然后将 softmax 层对这十个区块做出的预测取平均。
The second form of data augmentation consists of altering the intensities of the RGB channels in training images. Specifically, we perform PCA on the set of RGB pixel values throughout the ImageNet training set. To each training image, we add multiples of the found principal components, with magnitudes proportional to the corresponding eigenvalues times a random variable drawn from a Gaussian with mean zero and standard deviation 0.1. Therefore to each RGB image pixel $I_{xy}=[I_{xy}^R, I_{xy}^G, I_{xy}^B]^T$ we add the following quantity:	第二种放大数据集的方法是对训练图片的 RGB 频谱密度进行改变。特别地，我们在整个 ImageNet 训练集上对 RGB 像素进行主成分分析（PCA），对于每张训练图像，我们通过均值为 0，方差为 0.1 的高斯分布产生一个随机值（译者：设为 a），然后通过向图像中加入更大比例的相应的本征值的 a 倍，把其主成分翻倍【we add multiples of the found principal components, with magnitudes proportional to the corresponding eigenvalues times a random variable drawn from a Gaussian with mean zero and standard deviation 0.1】。因此，对于每个 RGB 像素 $I_{xy}=[I_{xy}^R, I_{xy}^G, I_{xy}^B]^T$ 我们加入的值如下：
$[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$ where p_i and λ_i are ith eigenvector and eigenvalue of the 3 x 3 covariance matrix of RGB pixel values, respectively, and αi is the aforementioned random variable. Each αi is drawn only once for all the pixels of a particular training image until that image is used for training again, at which point it is redrawn. This	$[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$ 其中，p_i 和 λ_i 分别是第 i 个特征向量和第 i 个 3x3RGB 协方差矩阵的本征值。而 αi 是前面所述的随机变量。对于一张特定的训练图片的所有像素，每个 αi 仅被抽取一次，直到这张图再次被用于训练才会再次提取随机变量。

scheme approximately captures an important property of natural images, namely, that object identity is invariant to changes in the intensity and color of the illumination. This scheme reduces the top-1 error rate by over 1%.	这一方案能够近似地捕捉原始图像的一些重要特征，即那些不随光线强度与颜色变化的物体特质。这一方法把 top-1 错误降低了 1%。
4.2 Dropout Combining the predictions of many different models is a very successful way to reduce test errors [1, 3], but it appears to be too expensive for big neural networks that already take several days to train. There is, however, a very efficient version of model combination that only costs about a factor of two during training. The recently-introduced technique, called "dropout" [10], consists of setting to zero the output of each hidden neuron with probability 0.5. The neurons which are "dropped out" in this way do not contribute to the forward pass and do not participate in backpropagation. So every time an input is presented, the neural network samples a different architecture, but all these architectures share weights. This technique reduces complex co-adaptations of neurons, since a neuron cannot rely on the presence of particular other neurons. It is, therefore, forced to learn more robust features that are useful in conjunction with many different random subsets of the other neurons. At test time, we use all the neurons but multiply their outputs by 0.5, which is a reasonable approximation to taking the geometric mean of the predictive distributions produced by the exponentially-many dropout networks.	**4.2 DROPOUT** 降低测试错误的一种有效方法是联立多种不同模型的预测结果【1，3】，但这种方法对于大型神经网络来说似乎太昂贵了，需要好几天去训练。然而，有一种非常高效的模型联立方法，只需要在训练过程中消耗一到两个因子。这种新近研究出来的技术叫做“DROPOUT”【10】，它会以 50%的概率将每个隐藏层神经元置零。 以这种方法被置零的神经元不再参与前馈和 BP 过程。 所以每次一个输入进来之后，这个神经网都会被置于不同的结构，但所有这些结构共享同一套参数。这种技术降低了神经元间相互适应的复杂性，因为每个神经元都不可能依赖其他特定某个神经元的表现。 因此，模型被迫学习更加健壮的特征，使之能够被许多不同的随机神经元子集使用。在测试中，我们使用所有的神经元，但是把它们的输出乘以 0.5，这是一种对大量 dropout 网络产生的预测分布的几何均值的合理近似。
We use dropout in the first two fully-connected layers of Figure 2. Without dropout, our network exhibits substantial overfitting. Dropout roughly doubles the number of iterations required to	我们在图 2 中的前两个全连接层使用 dropout。否则，我们的网络会表现出严重的过拟合。dropout 大概会让达到收敛所需要的迭代次数翻倍。

converge.	
5 Details of learning We trained our models using stochastic gradient descent with a batch size of 128 examples, momentum of 0.9, and weight decay of 0.0005. We found that this small amount of weight decay was important for the model to learn. In other words, weight decay here is not merely a regularizer: it reduces the model's training error. The update rule for weight w was $$v_{i+1} := 0.9 \cdot v_i - 0.0005 \cdot \epsilon \cdot w_i - \epsilon \cdot \left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}$$ $$w_{i+1} := w_i + v_{i+1}$$ where i is the iteration index, v is the momentum variable, ε is the learning rate, and $\left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}$ is the average over the ith batch D_i of the derivative of the objective with respect to w, evaluated at w_i.	**5. 学习过程的细节** 我们每个训练批次有 128 个样本，在其上采用随机梯度下降进行训练。设置增量【？momentum】为 0.9，权值衰退因子为 0.0005.我们发现小的权重衰退因子对于模型学习很重要，换句话说，权重衰退因子在这里不光是个正则化因子，它还可以减少模型错误。权值 w 的更新规则是： $$v_{i+1} := 0.9 \cdot v_i - 0.0005 \cdot \epsilon \cdot w_i - \epsilon \cdot \left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}$$ $$w_{i+1} := w_i + v_{i+1}$$ 其中，i 是迭代次数，v 是增量【？momentum】，ε 是学习速率，$\left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}$是第 i 批次的目标函数关于 w 的导数（wi 的偏导数）Di 的平均值【？is the average over the ith batch D_i of the derivative of the objective with respect to w, evaluated at w_i.】
We initialized the weights in each layer from a zero-mean Gaussian distribution with standard deviation 0.01. We initialized the neuron biases in the second, fourth, and fifth convolutional layers, as well as in the fully-connected hidden layers, with the constant 1. This initialization accelerates the early stages of learning by providing the ReLUs with positive inputs. We initialized the neuron biases in the remaining layers with the constant 0.	我们将每一层的权值利用均值为 0 方差为 0.01 的高斯分布随机初始化，我们用常数 1 初始化第 2、4、5 卷积层和全连接隐藏层的偏置神经元（常数单元）。这种初始化通过向 ReLUs 提供正输入，加速了学习的早期过程。我们将其它层的偏置神经元初始化为 0.
We used an equal learning rate for all layers, which we adjusted manually throughout training. The heuristic which we followed was to divide the learning rate by 10 when the validation error rate stopped improving with the current learning rate. The learning rate was initialized at 0.01 and reduced three times prior to termination. We	在整个学习过程中，我们在所有层都使用人工调整的相等的学习速率。我们采用的启发式方法是当验证误差不在降低时，就把当前的学习速率除以 10。学习速率初始化为 0.01，并在结束前减小 3 次。（做三次除以 10） 我们大概用 120 万张图片把我们的网络训练了约 90 轮，在两个 NVIDIA GTX 580 3GB GPU 上这大

trained the network for roughly 90 cycles through the training set of 1.2 million images, which took five to six days on two NVIDIA GTX 580 3GB GPUs.	概要 5 到 6 天。
 Figure 3: 96 convolutional kernels of size $11\times11\times3$ learned by the first convolutional layer on the $224\times224\times3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.	图 3：96 个通过第一个卷积层学习 224x224x3 的图片得到的 11x11x3 的卷积内核。上面 48 个和下面 48 个分别由两个 GPU 学习得到，详见 6.1.
6 Results Our results on ILSVRC-2010 are summarized in Table 1. Our network achieves top-1 and top-5 test set error rates of 37.5% and 17.0%. The best performance achieved during the ILSVRC- 2010 competition was 47.1% and 28.2% with an approach that averages the predictions produced from six sparse-coding models trained on different features [2], and since then the best published results are 45.7% and 25.7% with an approach that averages the predictions of two classifiers trained on Fisher Vectors (FVs) computed from two types of densely-sampled features [24].	**6 实验结果** 我们在 ILSVRC-2010 数据集上的实验结果归纳在表 1 里。我们的网络 top-1 和 top-5 测试误差分别是 37.5%和 17.0%。在此之前 ILSVRC- 2010 数据集上的最好的比赛纪录是对在不同特征上训练的留个稀疏自编码器取平均，top-1 和 top-5 测试误差分别是 47.1%和 28.2%【2】。 之后，已出版的最佳结果是一种 对两个在不同取样密度的 费舍向量上训练的 分类器取平均的方法，结果是 45.7%和 25.7%【24】。 **Model** / **Top-1** / **Top-5** *Sparse coding [2]* / *47.1%* / *28.2%* *SIFT + FVs [24]* / *45.7%* / *25.7%* CNN / **37.5%** / **17.0%** Table 1: Comparison of results on ILSVRC-2010 test set. In *italics* are best results achieved by others.
We also entered our model in the ILSVRC-2012 competition and report our results in Table 2. Since the ILSVRC-2012 test set labels are not publicly available, we cannot report test error	我们也让我们的模型参加了 ILSVRC-2012 的比赛，并在表 2 中展示了我们的结果。因为 ILSVRC-2012 测试集的标签并未公开，所以我们不能报告我们所有试过的模型的测试错误率。在

rates for all the models that we tried. In the remainder of this paragraph, we use validation and test error rates interchangeably because in our experience they do not differ by more than 0.1% (see Table 2). The CNN described in this paper achieves a top-5 error rate of 18.2%. Averaging the predictions of five similar CNNs gives an error rate of 16.4%. Training one CNN, with an extra sixth convolutional layer over the last pooling layer, to classify the entire ImageNet Fall 2011 release (15M images, 22K categories), and then "fine-tuning" it on ILSVRC-2012 gives an error rate of 16.6%. Averaging the predictions of two CNNs that were pre-trained on the entire Fall 2011 release with the aforementioned five CNNs gives an error rate of 15.3%. The second-best contest entry achieved an error rate of 26.2% with an approach that averages the predictions of several classifiers trained on FVs computed from different types of densely-sampled features [7].

这一段的余下部分，我们使用验证误差代替测试误差，因为根据我们的经验，它们的差距不会大于 0.1%（见表 2）。本文介绍的卷积神经网络达到了 Top-5 错误 18.2%的水平。5 个相同 CNN 平均 TOP-5 错误为 16.4%。

训练一个比之前说的五个卷积层还多一个卷积层的 CNN 去分类整个 ImageNet Fall 2011 数据集（1500 万张图，22000 个类别），然后对其进行调整，在 ILSVRC-2012 上可以达到 16.6%的 TOP-5 错误。

两个在 ImageNet Fall 2011 数据集上预训练的 CNN，加上前面提到的五个 CNN，平均 TOP-5 为 15.3%。比赛的第二名达到了 26.2%的 TOP-5，他们用的是对 几个在特征取样密度不同的费舍向量上训练的分类器的预测结果取平均的方法【7】。

Model	Top-1 (val)	Top-5 (val)	Top-5 (test)
SIFT + FVs [7]	—	—	*26.2%*
1 CNN	40.7%	18.2%	—
5 CNNs	38.1%	16.4%	**16.4%**
1 CNN*	39.0%	16.6%	—
7 CNNs*	36.7%	15.4%	**15.3%**

Table 2: Comparison of error rates on ILSVRC-2012 validation and test sets. In *italics* are best results achieved by others. Models with an asterisk* were "pre-trained" to classify the entire ImageNet 2011 Fall release. See Section 6 for details.

6.1 Qualitative Evaluations

Figure 3 shows the convolutional kernels learned by the network's two data-connected layers. The network has learned a variety of frequency-and orientation-selective kernels, as well as various colored blobs. Notice the specialization exhibited by the two GPUs, a result of the restricted connectivity described in Section 3.5. The kernels on GPU 1 are largely color-agnostic, while the kernels on GPU 2 are largely color-specific. This kind of specialization occurs during every run and is independent of any particular random weight initialization (modulo a renumbering of the GPUs).

6.1 定量分析

图 3 展示了网络的两个数据连接层学到的卷积内核。网络学到了一系列“频率+方向选择”【frequency-and orientation-selective】的内核，还有一系列色块。请注意两个 GPU 表现除出了不同的特性，这是 3.5 节介绍的限制互联方式的结果。GPU 1 上的内核基本上不在意颜色，而 GPU 2 上的内核就是色彩专家。这种专一性每次都会出现，与权值的随机初始化无关（GPU 重新编号）。

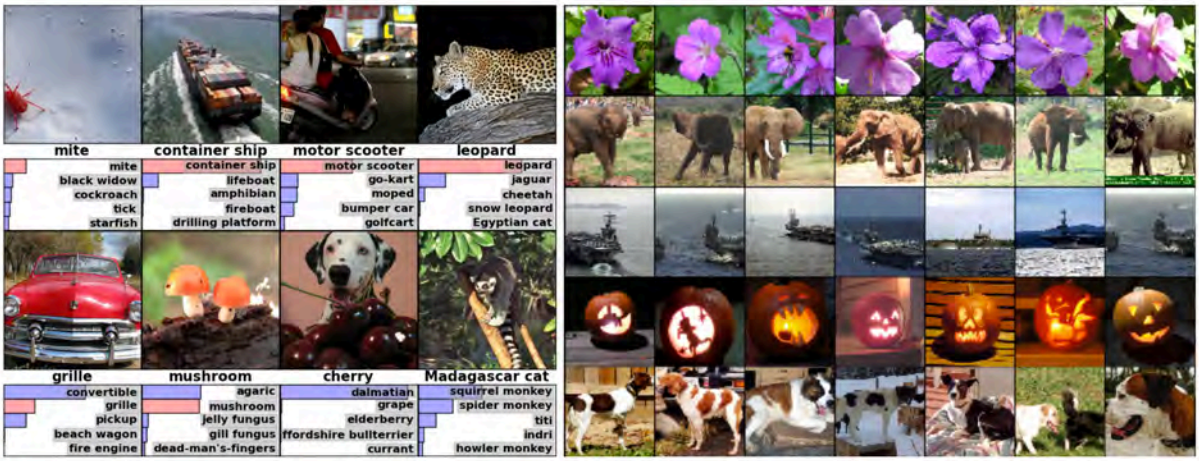


Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010 test images in the first column. The remaining columns show the six training images that produce feature vectors in the last hidden layer with the smallest Euclidean distance from the feature vector for the test image.

In the left panel of Figure 4 we qualitatively assess what the network has learned by computing its top-5 predictions on eight test images. Notice that even off-center objects, such as the mite in the top-left, can be recognized by the net. Most of the top-5 labels appear reasonable. For example, only other types of cat are considered plausible labels for the leopard. In some cases (grille, cherry) there is genuine ambiguity about the intended focus of the photograph.	在图四的左侧，我们定量地展示了对于 8 张图片网络所学习到的前五个预测。注意对于偏离中心的物体，比如左上角的那只螨虫，网络依然可以识别出来。大多数前五个标签看起来都比较合理，比如，只有其他类别的猫科动物才被判别是豹子的可能标签。在一些例子中，比如栅栏，樱桃，确实对于究竟该关注哪个物体存在歧义。
Another way to probe the network's visual knowledge is to consider the feature activations induced by an image at the last, 4096-dimensional hidden layer. If two images produce feature activation vectors with a small Euclidean separation, we can say that the higher levels of the neural network consider them to be similar. Figure 4 shows five images from the test set and the six images from the training set that are most similar to each of them according to this measure. Notice that at the pixel level, the retrieved training images are generally not close	另一个研究可视化网络所学知识的方法是考虑最后一个 4096 维隐层所激活的特征向量。 如果两张图的向量欧氏距离很小，我们可以说很大程度上神经网络认为它们是相似的。图 4 展示了五张测试集中的图片，以及按照上述方法找出的分别与这五张图最相似的 6 张训练集图片。 注意在像素尺度上，找出来的训练集图片不一定在 L2【？】上和第一列的测试集图片很相似。比如，找出来的狗狗和大象摆出了不同的造型。我们用更多的测试集图片支持证明了这一观点。

in L2 to the query images in the first column. For example, the retrieved dogs and elephants appear in a variety of poses. We present the results for many more test images in the supplementary material.	
Computing similarity by using Euclidean distance between two 4096-dimensional, real-valued vectors is inefficient, but it could be made efficient by training an auto-encoder to compress these vectors to short binary codes. This should produce a much better image retrieval method than applying auto-encoders to the raw pixels [14], which does not make use of image labels and hence has a tendency to retrieve images with similar patterns of edges, whether or not they are semantically similar.	通过两个 4096 维的实数向量之间的欧氏距离来计算相似度显然效率很低，但可以通过训练一个自编码器去把这些向量压缩为二进制编码来提高效率。这应该能够产生一种比 对原始像素进行自编码【14】 更好的图像检索方法，因为（对原始像素进行自编码）用不到标签，因此它倾向于找出具有同样边缘模式的图片，而不是语义上相似的图。
7 Discussion Our results show that a large, deep convolutional neural network is capable of achieving record-breaking results on a highly challenging dataset using purely supervised learning. It is notable that our network's performance degrades if a single convolutional layer is removed. For example, removing any of the middle layers results in a loss of about 2% for the top-1 performance of the network. So the depth really is important for achieving our results.	**7 讨论** 我们的结果显示一个大型深度卷积神经网络能够在一个极具挑战的数据集上进行破纪录的纯粹的监督学习。值得注意的是，如果把我们的网络去掉一层卷积层，表现就会变差。比如，去掉任意隐藏层会让 top-1 错误增加 2%，所以深度对于我们的成功真的很重要。
To simplify our experiments, we did not use any unsupervised pre-training even though we expect that it will help, especially if we obtain enough computational power to significantly increase the size of the network without obtaining a corresponding increase in the amount of labeled data. Thus far, our results have improved as we have made our network larger and trained it longer but we still have many orders of	为了简化我们的实验，我们并未使用非监督的预训练，即使我们知道这样会有帮助，特别是如果我们能够获得足够的计算力去大幅提升网络规模却不相应地增加标签数据的数量。至此，我们的结果已经通过增大我们的网络规模、进行更长时间的训练而得到优化。 但我们还有很大的空间去优化网络使之能够像人类的视觉系统一样感知时序。 最终我们希望在视频序列上使用极大极深的卷积

magnitude to go in order to match the infero-temporal pathway of the human visual system. Ultimately we would like to use very large and deep convolutional nets on video sequences where the temporal structure provides very helpful information that is missing or far less obvious in static images.	神经网，因为视频序列的时序结构能够提供丰富的信息，这些信息在静态图片上丢失了，或远远没有那么明显。