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.

基于深层卷积神经网络的图像分类网络

摘要

我们训练了一个庞大的深层卷积神经网络，将 ImageNet LSVRC-2010 比赛中的 120 万张高分辨率图像分为 1000 个不同的类别。在测试数据上，我们取得了 37.5％和 17.0％的 TOP-1 和 TOP-5 错误率，这比以前的先进水平要好得多。具有 6000 万个参数和 650,000 个神经元的神经网络由五个卷积层组成，其中一些连接着最大池化层，接着是三个全连接层以及最后的大小为 1000 的 softmax 输出层。为了加快训练速度，我们使用非饱和神经元和能高效进行卷积运算的 GPU 实现。为了减少全连接层中的过拟合，我们采用了最近开发的称为“dropout”的正则化方法，该方法证明是非常有效的。我们还在 ILSVRC-2012 比赛中使用了这种模式的一个变种，取得了 15.3％的 TIP-5 错误率，而第二名的成绩是 26.2％。

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 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.

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 controlled by varying their depth and

1、介绍

目前，机器学习方法对物体识别非常重要。为了改善他们的表现，我们可以收集更大的数据集，训练更强大的模型，并使用更好的技术来防止过拟合。直到最近，标记好图像的数据集相对还较小——大约上万的数量级（例如，NORB [16]，Caltech-101/256 [8,9] 和 CIFAR-10/100 [12]）。使用这种规模的数据集可以很好地解决简单的识别任务，特别是如果他们增加了保留标签转换（label-preserving transformations）。例如，目前 MNIST 数字识别任务的最低错误率（<0.3%）基本达到了人类的识别水平[4]。但是物体在现实环境中可能表现出相当大的变化性，所以要学会识别它们，就必须使用更大的训练集。事实上，小图像数据集的缺点已是众所周知（例如，Pinto[21]），但直到最近才可以收集到数百万的标记数据集。新的大型数据集包括 LabelMe [23]，其中包含数十万个完全分割的图像，以及 ImageNet [6], 其中包含超过 150,000 万个超过 22,000 个类别的高分辨率图像。

要从数百万图像中学习数千个类别，我们需要一个具有强大学习能力的模型。然而，物体识别任务的巨大复杂性意味着即使是像 ImageNet 这样大的数据集也不能完美地解决这个问题，所以我们的模型也需要使用很多先验知识来弥补我们数据集不足的问题。卷积神经网络（CNN）就构成了一类这样的模型[16,11,13,18,15,22,26]。它们的容量可以通过改变它们的深度和宽度来控制，并且它们也对图像的性质（即统计量的定态假设以及像素局部依赖性假设）做出准确而且全面的假设。因此，与具有相同大小的层的标准前馈神经网络相比，CNN 具有更少的连接和参数，因此它们更容易训练，而其理论最优性能可能稍微弱一些。
尽管 CNN 具有很好的质量，并且尽管其局部结构的效率相对较高，但将它们大规模应用于高分辨率图像时仍然显得非常昂

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.

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.

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 publicly1. 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

贵。幸运的是，当前的 GPU 可以用于高度优化的二维卷积，能够加速许多大型 CNN 的训练，并且最近的数据集（如 ImageNet）包含足够多的标记样本来训练此类模型，而不会出现严重的过度拟合。

本文的具体贡献如下：我们在 ILSVRC-2010 和 ILSVRC-2012 比赛中使用的 ImageNet 子集上训练了迄今为止最大的卷积神经网络之一[2]，并在这些数据集上取得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 实现以及其他训练卷积神经网络的固有操作，并将其公开。我们的网络包含许多新的和不同寻常的功能，这些功能可以提高网络的性能并缩短训练时间，详情请参阅第 3 节。我们的网络规模较大，即使有 120 万个带标签的训练样本，仍然存在过拟合的问题，所以我们采用了几个有效的技巧来阻止过拟合，在第 4 节中有详细的描述。我们最终的网络包含五个卷积层和三个全连接层，并且这个深度似乎很重要：我们发现去除任何卷积层（每个卷积层只包含不超过整个模型参数的 1%的参数）都会使网络的性能变差。

最后，网络的规模主要受限于目前 GPU 上可用的内存量以及我们可接受的训练时间。我们的网络需要在两块 GTX 580 3GB GPU 上花费五到六天的时间来训练。我们所有的实验都表明，通过等待更快的 GPU 和更大的数据集出现，我们的结果可以进一步完善。

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

In the end, the network’s size is limited mainly by 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.

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.

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.

ImageNet consists of variable-resolution images, while our system requires a constant input dimensionality. Therefore, we down-sampled the images to a fixed resolution of 256 * 256. Given a rectangular image, we first rescaled the image such that the shorter side was of length 256, and then cropped out the central 256_256 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

2、数据集

ImageNet 是一个拥有超过 1500 万个已标记高分辨率图像的数据集, 大概有 22,000 个类别 。 图 像 都 是 从 网 上 收 集 ， 并 使 用 Amazon-Mechanical Turk 群智工具人工标记。从 2010 年起，作为 Pascal 视觉对象挑战赛的一部分，即是每年举办一次的名为 ImageNet 大型视觉识别挑战赛（ILSVRC）的比赛。 ILSVRC 使用的是 ImageNet 的一个子集, 每 1000 个类别中大约有 1000 个图像。总共有大约 120 万张训练图像，50,000 张验证图像和 150,000 张测试图像。

ILSVRC-2010 是 ILSVRC 中的唯一可以使用测试集标签的版本，因此这也正是我们进行大 部 分 实 验 的 版 本 。 由 于 我 们 也 在 ILSVRC-2012 比赛中引入了我们的模型，因此在第 6 部分中，我们也会给出此版本数据集的结果，尽管这个版本的测试集标签不可用。在 ImageNet 上，习惯上使用两种错误率：top-1 和 top-5，其中 top-5 错误率是正确标签不在被模型认为最可能的五个标签之中的测试图像的百分率。

ImageNet 由可变分辨率的图像组成，而我们的系统需要固定的输入尺寸。因此，我们将图像下采样到 256×256 的固定分辨率。给定一个矩形图像，我们首先重新缩放图像，使得短边长度为 256，然后从结果中裁剪出中心的 256×256 的图片。除了将每个像素中减去训练集的像素均值之外，我们没有以任何其他方式对图像进行预处理。所以我们在像素的（中心）原始 RGB 值上训练了我们的网络。

our network on the (centered) raw RGB
values of the pixels.

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、结构

图 2 概括了我们所提出网络的结构。它包含八个学习层——五个卷积层和三个全连接层。下面，我们将描述一些所提出网络框架中新颖或不寻常的地方。 3.1-3.4 节按照我们对它们重要性的估计进行排序，其中最重要的是第一个。

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 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.

3.1、ReLU 非线性单元

对一个神经元模型的输出的常规套路是，给他接上一个激活函数：$f(x)=\tanh(x)$或者 $f(x)=(1+e^{-x})^{-1}$。就梯度下降法的训练时间而言，这些饱和非线性函数比非饱和非线性函数如 $f(x)=\max(0,x)$ 慢得多。根据 Nair 和 Hinton 的说法[20]，我们将这种非线性单元称为——修正非线性单元（Rectified Linear Units (ReLUs)）。使用 ReLUs 做为激活函数的卷积神经网络比起使用tanh单元作为激活函数的训练起来快了好几倍。这个结果从图 1 中可以看出来，该图展示了对于一个特定的四层 CNN, CIFAR-10 数据集训练中的误差率达到 25%所需要的迭代次数。从这张图的结果可以看出，如果我们使用传统的饱和神经元模型来训练 CNN，那么我们将无法为这项工作训练如此大型的神经网络。

我们并不是第一个考虑在 CNN 中替换掉传统神经元模型的。例如，Jarrett 等人 [11] 声称，非线性函数 $f(x)=|\tanh(x)|$在他们的对比度归一化问题上，再接上局部均值池化单元，在 Caltech-101 数据集上表现的非常好。然而，在这个数据集中，主要担心的还是防止过拟合，所以他们观察到的效果与我们在使用 ReLU 时观察到的训练集的加速能力还是不一样。加快训练速度对大型数据集上训练的

大型模型的性能有很大的影响。

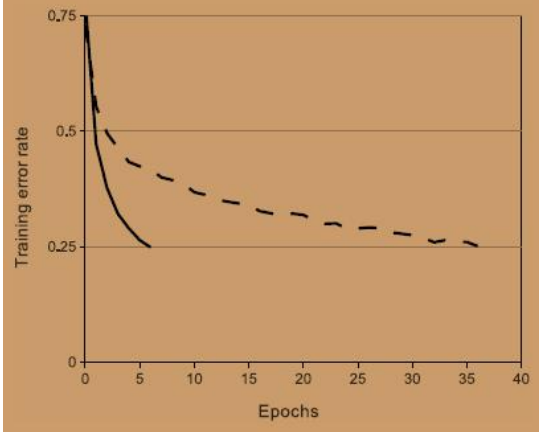


Figure 1: A four-layer convolutional neural network with ReLUs (**solid line**) reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons (**dashed line**). The learning rates for each network were chosen independently to make training as fast as possible. 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 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.

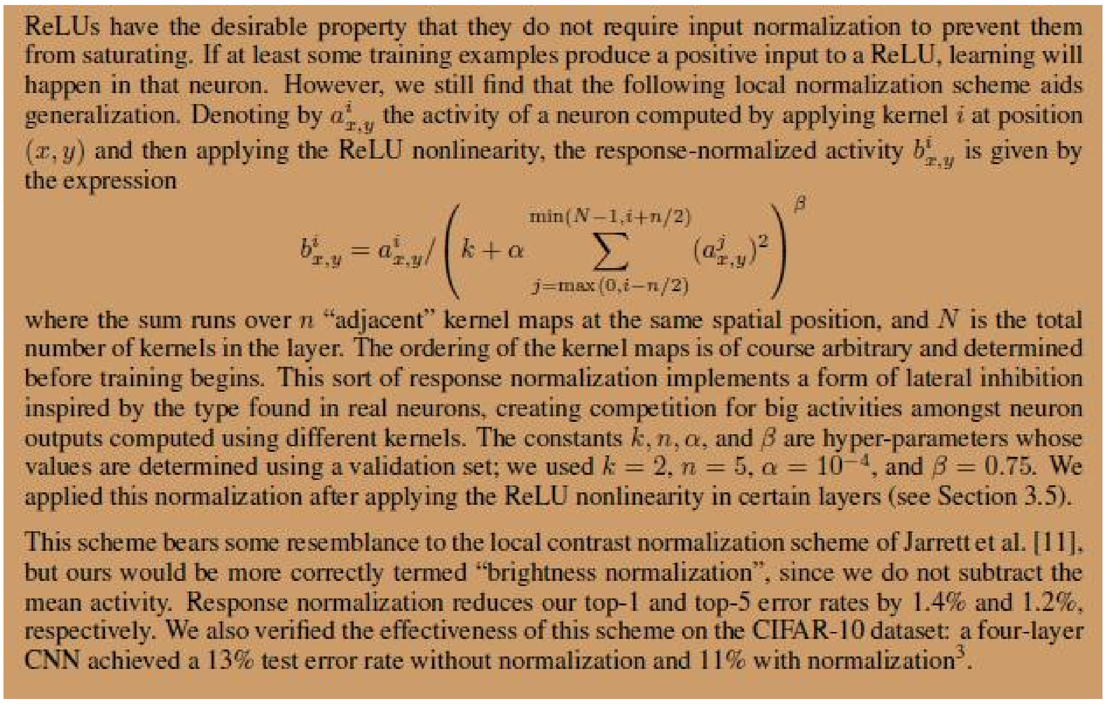
The resultant architecture is somewhat similar to that of the "columnar" CNN employed by Ciresan et al. [5], except that our columns are not independent (see Figure 2). This scheme reduces our top-1 and top-5 error rates by 1.7% and 1.2%, respectively, as compared with a net with half as many kernels in each convolutional layer trained on one GPU. The two-GPU net takes slightly less time to train than the one-GPU net.

3.2、在多个 GPU 上训练

单个 GTX 580 GPU 只有 3GB 内存，这限制了可以在其上训练的网络的最大尺寸。事实证明，120 万个训练样本足以训练那些因规模太大而不适合使用一个 GPU 训练的网络。因此，我们将网络分布在两个 GPU 上。目前的 GPU 很适合于跨 GPU 并行化操作，因为它们能够直接读写对方的内存，而无需通过主机内存。我们采用的并行化方案基本上将半个内核（或神经元）放在各个 GPU 上，另外还有一个技巧：GPU 只在某些层间进行通信。这意味着，例如，第 3 层的内核从第 2 层的所有内核映射（kernel maps）中获取输入。然而，第 4 层中的内核又仅从位于同一 GPU 上的第 3 层中的那些内核映射获取输入。选择连接模式对于交叉验证是一个不小的问题，但这使得我们能够精确调整通信量，直到它的计算量的达到可接受的程度。

由此产生的架构有点类似于 Cire¸san 等人使用的“柱状”CNN[5]，除了我们的每列不是独立的之外（见图 2）。与一个 GPU 上训练的每个卷积层只有一半的内核数量的网络相比，该方案分别将我们的 top-1 和 top-5 错误率分别降低了 1.7%和 1.2%。双 GPU 网络的训练时间比单 GPU 网络更少。

3.3 Local Response Normalization



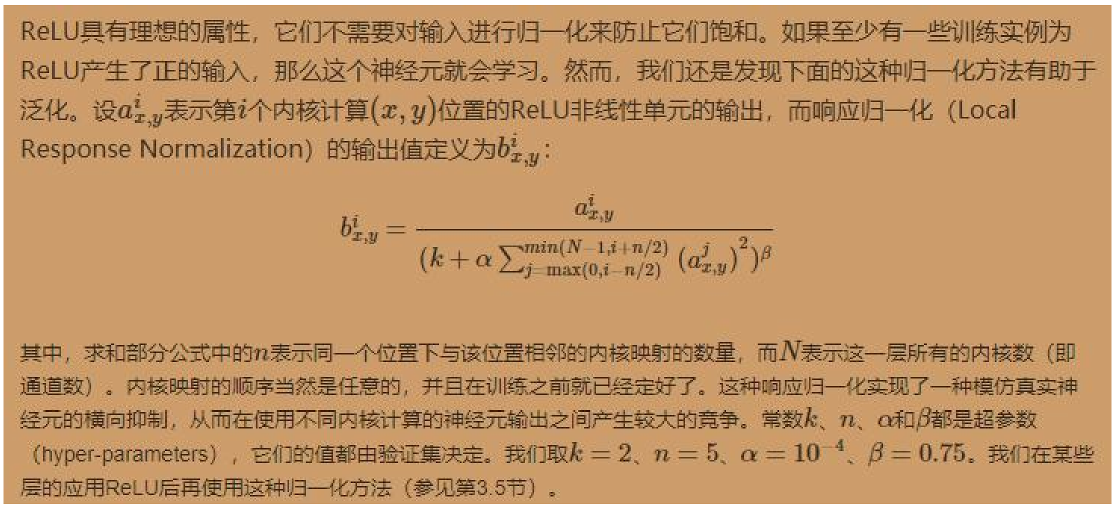
ReLUs have the desirable property that they do not require input normalization to prevent them from saturating. If at least some training examples produce a positive input to a ReLU, learning will happen in that neuron. However, we still find that the following local normalization scheme aids generalization. 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

$$b^i_{x,y} = a^i_{x,y} \Bigg/ \left(k + \alpha \sum_{j=\max(0,i-n/2)}^{\min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^\beta$$

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 $k = 2$, $n = 5$, $\alpha = 10^{-4}$, and $\beta = 0.75$. We applied this normalization after applying the ReLU nonlinearity in certain layers (see Section 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[3].

3.3、局部响应归一化（Local Response Normalization）



ReLU具有理想的属性，它们不需要对输入进行归一化来防止它们饱和。如果至少有一些训练实例为ReLU产生了正的输入，那么这个神经元就会学习。然而，我们还是发现下面的这种归一化方法有助于泛化。设$a^i_{x,y}$表示第i个内核计算(x, y)位置的ReLU非线性单元的输出，而响应归一化（Local Response Normalization）的输出值定义为$b^i_{x,y}$：

$$b^i_{x,y} = \frac{a^i_{x,y}}{(k + \alpha \sum_{j=\max(0,i-n/2)}^{min(N-1,i+n/2)} (a^j_{x,y})^2)^\beta}$$

其中，求和部分公式中的n表示同一个位置下与该位置相邻的内核映射的数量，而N表示这一层所有的内核数（即通道数）。内核映射的顺序当然是任意的，并且在训练之前就已经定好了。这种响应归一化实现了一种模仿真实神经元的横向抑制，从而在使用不同内核计算的神经元输出之间产生较大的竞争。常数k、n、α和β都是超参数（hyper-parameters），它们的值都由验证集决定。我们取$k = 2$、$n = 5$、$\alpha = 10^{-4}$、$\beta = 0.75$。我们在某些层的应用ReLU后再使用这种归一化方法（参见第3.5节）。

3.4 Overlapping Pooling

Pooling layers in CNNs summarize the outputs of 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.

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经元组的输出求和。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.

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 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.

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

3.5、整体结构

现在我们已经准备好描述 CNN 的整体架构了。如图 2 所示，这个网络包含了八层权重;前五个是卷积层，其余三个为全连接层。最后的全连接层的输出被送到 1000 维的 softmax 函数，其产生 1000 个类的预测。我们的网络最大化多项逻辑回归目标，这相当于在预测的分布下最大化训练样本中正确标签对数概率的平均值。

第二，第四和第五个卷积层的内核仅与上一层存放在同一 GPU 上的内核映射相连（见图 2）。第三个卷积层的内核连接到第二层中的所有内核映射。全连接层中的神经元连接到前一层中的所有神经元。响应归一化层紧接着第一个和第二个卷积层。 在 3.4 节中介绍的最大池化层，后面连接响应归一化层以及第五个卷积层。将 ReLU 应用于每个卷积层和全连接层的输出。

第一个卷积层的输入为 224×224×3 的图像，对其使用 96 个大小为 11×11×3、步长为 4（步长表示内核映射中相邻神经元感受野中心之间的距离）的内核来处理输入图像。第二个卷积层将第一个卷积层的输出（响应归一化以及池化）作为输入，并使用 256 个内核处理图像，每个内核大小为 5×5×48。第三个、第四个和第五个卷积层彼此连接而中间没有任何池化或归一化层。第三个卷积层有 384 个内核，每个的大小为 3×3×256，其输入为第二个卷积层的输出。第四个卷积层有 384 个内核，每个内核大小为 3×3×192。第五个卷积层有 256 个内核，每个内核大小为 3×3×192。全连接层各有 4096 个神经元。

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.

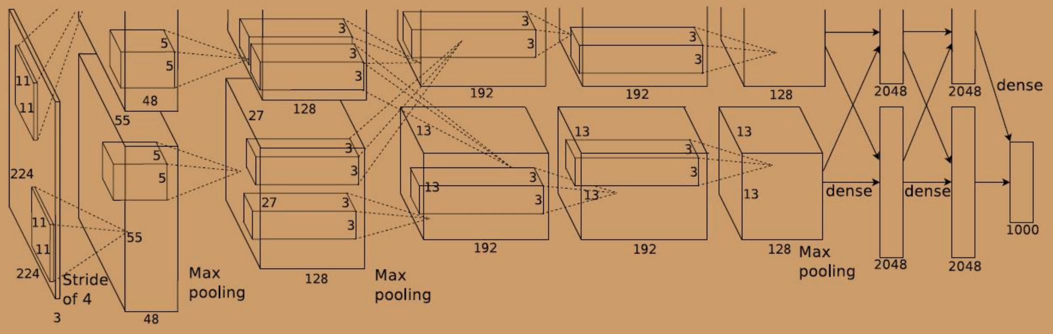


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input 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.

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.

4.1 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.

The first form of data augmentation consists of generating image translations and horizontal reflections. We do this by extracting random 224*224 patches (and their horizontal reflections) from the 256*256 images and training our network on these extracted patches[4]. 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 * 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.

4、减少过拟合

我们的神经网络架构拥有 6000 万个参数。尽管 ILSVRC 的 1000 个类别使得每个训练样本从图像到标签的映射被限制在了 10 bit 之内，但这不足以保证训练这么多参数而不出现过拟合。下面，我们将介绍对付过度拟合的两个方法。

4.1、数据增强（Data Augmentation）

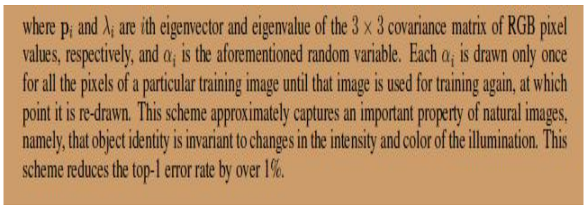
减小过拟合的最简单且最常用的方法就是，使用标签保留转换（label-preserving transformations，例如[25,4,5]），人为地放大数据集。我们采用两种不同形式的数据增强方法，它们都允许通过很少的计算就能从原始图像中生成转换图像，所以转换后的图像不需要存储在硬盘上。在我们实现过程中，转换后的图像是使用 CPU 上的 Python 代码生成的，在生成这些转换图像的同时，GPU 还在训练上一批图像数据。所以这些数据增强方案实际上是很高效的。

数据增强的第一种形式包括平移图像和水平映射。我们通过从 256×256 图像中随机提取 224×224 的图像块（及其水平映射）并在这些提取的图像块上训练我们的网络来做到这一点。这使我们的训练集的规模增加了 2048 倍，尽管由此产生的训练样本当然还是高度相互依赖的。如果没有这个方案，我们的网络就可能会遭受大量的的过拟合，可能会迫使我们不得不使用更小的网络。在测试时，网络通过提取 5 个 224×224 的图像块（四个角块和中心块）以及它们的水平映射（因此总共包括 10 个块）来进行预测，并求网络的 softmax 层的上的十个预测结果的均值。

第二种形式的数据增强包括改变训练图像中 RGB 通道的灰度。具体而言，我们在整个 ImageNet 训练集的图像的 RGB 像素值上使用 PCA。对于每个训练图像，我们添加多个通过 PCA 找到的主成分，

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 $I_{xy} = [I^R_{xy}; I^G_{xy}; I^B_{xy}]^T$ we add the following quantity:

$$[\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$$


where $\mathbf{p}_i$ and λ_i are ith eigenvector and eigenvalue of the 3×3 covariance matrix of RGB pixel values, respectively, and α_i is the aforementioned random variable. Each α_i 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 re-drawn. This 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%.

大小与相应的特征值成比例，乘以一个随机值，该随机值属于均值为 0、标准差为 0.1 的高斯分布。因此，对于每个图像的 RGB 像素有：Ixy=[IRxyIGxyIBxy]TIxy=[IxyRIxyGIxyB]T，我们加入如下的值：

[p1p2p3][α1λ1α2λ2α3λ3]T

其中，pi 和λi 分别是 3x3 的 RGB 协方差矩阵的第 i 个特征向量和第 i 个的特征值，而αi 是前面所说的随机值。对于一张特定图像中的所有像素，每个αi 只会被抽取一次，知道这张图片再次用于训练时，才会重新提取随机变量。这个方案近似地捕捉原始图像的一些重要属性，对象的身份不受光照的强度和颜色变化影响。这个方案将 top-1 错误率降低了 1%以上。

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.

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 converge.

4.2、Dropout

结合许多不同模型的预测结果是减少测试错误率的一种非常成功的方法[1,3]，但对于已经花费数天时间训练的大型神经网络来说，它似乎成本太高了。然而，有一种非常有效的模型组合方法，在训练期间，只需要消耗 1/2 的参数。这个新发现的技术叫做“Dropout”[10]，它会以 50%的概率将隐含层的神经元输出置为 0。以这种方法被置 0 的神经元不参与网络的前馈和反向传播。因此，每次给网络提供了输入后，神经网络都会采用一个不同的结构，但是这些结构都共享权重。这种技术减少了神经元的复杂适应性，因为神经元无法依赖于其他特定的神经元而存在。因此，它被迫学习更强大更鲁棒的功能，使得这些神经元可以与其他神经元的许多不同的随机子集结合使用。在测试时，我们试着使用了所有的神经元，并将它们的输出乘以 0.5。这与采用大量 dropout 的网络产生的预测结果分布的几何均值近似。

我们在图 2 中的前两个全连接层上使用了 dropout。没有 dropout，我们的网络会出现严重的过拟合。Dropout 大概会使达到收敛的迭代次数翻倍。

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

$$\begin{aligned} v_{i+1} &:= 0.9 \cdot v_i - 0.0005 \cdot \epsilon \cdot w_i - \epsilon \cdot \left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i} \\ w_{i+1} &:= w_i + v_{i+1} \end{aligned}$$

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.

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.

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 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.

5、训练细节

我们使用随机梯度下降法来训练我们的模型，每个 batch 有 128 个样本，动量（momentum）为 0.9，权重衰减（weight decay）为 0.0005。我们发现这种较小的权重衰减对于模型的训练很重要。换句话说，权重衰减在这里不仅仅是一个正则化方法：它减少了模型的训练误差。权重ω的更新法则是：

$$\begin{aligned} v_{i+1} &:= 0.9 \cdot v_i - 0.0005 \cdot \epsilon \cdot \omega_i - \epsilon \cdot \langle \frac{\partial L}{\partial \omega}|_{\omega_i} \rangle_{D_i} \\ \omega_{i+1} &:= \omega_i + v_{i+1} \end{aligned}$$

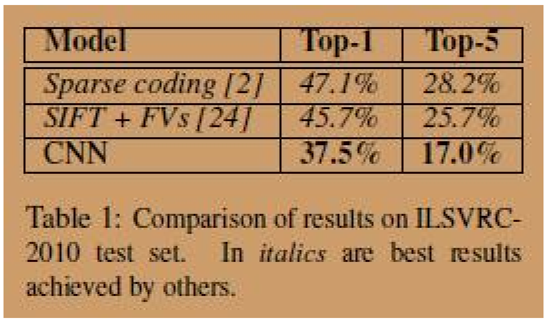
其中，i 表示当前的迭代次数，v 表示动量（momentum），ε表示学习率，$\langle \frac{\partial L}{\partial \omega}|_{\omega_i} \rangle_{D_i}$ 是第 i 批次的目标函数关于 w 的导数（wi 的偏导数）Di 的平均值。

我们使用标准差为 0.01、均值为 0 的高斯分布来初始化各层的权重。我们使用常数 1 来初始化了网络中的第二个、第四个和第五个卷积层以及全连接层中的隐含层中的所有偏置参数。这种初始化权重的方法通过向 ReLU 提供了正的输入，来加速前期的训练。我们使用常数 0 来初始化剩余层中的偏置参数。

我们对所有层都使用相同的学习率，在训练过程中又手动进行了调整。我们遵循的启发式方法是：以当前的学习速率训练，验证集上的错误率停止降低时，将学习速率除以 10.学习率初始时设为 0.01，并且在终止前减少 3 次。我们使用 120 万张图像的训练集对网络进行了大约 90 次迭代的训练，这在两块 NVIDIA GTX 580 3GB GPU 上花费了大约 5 到 6 天的时间。

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%[5]. 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].



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 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

6、结果

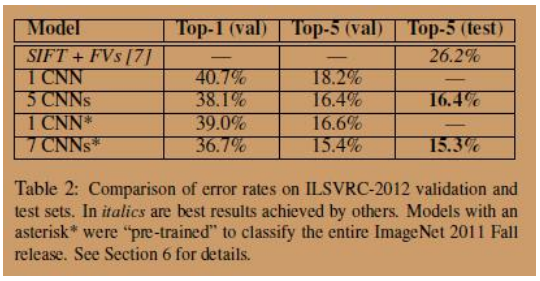
我们在ILSVRC-2010上取得的结果如表1所示。我们的网络的 top-1 和 top-5 测试集错误率分别为 37.5％和 17.0％。在 ILSVRC-2010 比赛期间取得的最佳成绩是 47.1％和 28.2％，其方法是对六种不同的稀疏编码模型所产生的预测结果求平均[2]。此后公布的最佳结果为 45.7%、25.7%，其方法是对两种经过密集采样的特征[24]计算出来的 Fisher 向量（FV）训练的两个分类器取平均值。

我们还在 ILSVRC-2012 竞赛中使用了我们的模型，并在表 2 中给出了我们的结果。由于 ILSVRC-2012 测试集标签未公开，因此我们无法给出我们测试过的所有模型在测试集上的错误率。在本节的其余部分中，我们将验证集和测试集的错误率互换，因为根据我们的经验，它们之间的差值不超过 0.1%（见表 2）。本文描述的 CNN 的 top-5 错误率达到了 18.2%。对五个相似 CNN 的预测结果计算均值，得到的错误率为 16.4%。单独一个 CNN，在最后一个池化层之后，额外添加第六个卷积层，对整个 ImageNet Fall 2011 release(15M images, 22K categories) 进行分类，然后在 ILSVRC-2012 上“微调”（fine-tuning）网络，得到的错误率为 16.6%。对整个 ImageNet Fall 2011 版本的数据集下预训练的两个 CNN，求他们输出的预测值与前面提到的 5 个不同的 CNN 输出的预测值的均值，得到的错误率为 15.3%。比赛的第二名达到了 26.2% 的 top-5 错误率，他们的方法是：对几个在特征取样密度不同的 Fisher 向量上训练的分类器的预测结果取平均的方法[7]。

最后，我们还在 ImageNet Fall 2009 版本的数据集上提交了错误率，总共有 10,184 个类别和 890 万张图像。在这个数据集

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].

Finally, we also report our error rates on the Fall 2009 version of ImageNet with 10,184 categories and 8.9 million images. On this dataset we follow the convention in the literature of using half of the images for training and half for testing. Since there is no established test set, our split necessarily differs from the splits used by previous authors, but this does not affect the results appreciably. Our top-1 and top-5 error rates on this dataset are 67.4% and 40.9%, attained by the net described above but with an additional, sixth convolutional layer over the last pooling layer. The best published results on this dataset are 78.1% and 60.9% [19].



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.

中，我们遵循文献中的使用一半图像用于训练，一半图像用于测试的惯例。由于没有建立测试集，所以我们的拆分方法有必要与先前作者使用的拆分方法不同，但这并不会对结果产生显著的影响。我们在这个数据集上的top-1和top-5错误率分别是 67.4％和 40.9％，是通过前面描述的网络获得的，但是在最后的池化层上还有额外的第 6 个卷积层。该数据集此前公布的最佳结果是 78.1％和 60.9％[19]。

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 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).

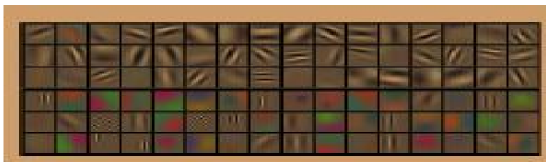


Figure 3: 96 convolutional kernels of size $11\times11\times3$ learned by the first convolutional layer on the $224\times224\times3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

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.

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 in L2 to the query images in the first column. For example, the retrieved

6.1、定性评估

图 3 显示了由网络的两个数据连接层学习得到的卷积内核。该网络已经学习到许多频率和方向提取的内核，以及各种色块。请注意两个 GPU 所展现的不同特性，这也是 3.5 节中介绍的限制互连的结果。GPU1 上的内核在很大程度上与颜色无关， 然而 GPU2 上的内核在很大程度上都于颜色有关。这种特异性在每次迭代期间都会发生，并且独立于任何特定的随机权重初始化过程（以 GPU 的重新编号为模）。

在图 4 的左边， 我们通过计算 8 张测试图像的 top-5 预测来定性评估网络的训练结果。请注意，即使是偏离中心的物体，如左上角的螨虫，也可以被网络识别出来。大多数 top-5 的标签都显得比较合理。例如，只有其他类型的猫才被认为是豹子的可能标签。在某些情况下（栅栏、樱桃），照片的关注点存在模糊性，不知道到底该关注哪个。

另一个研究可视化的网络的方法是，考虑由最后一个 4096 维隐含层中的图像的特征的激活函数输出值。如果两幅图像产生有的欧氏距离，我们可以认为高层次的神经网络认为它们是相似的。图 4 显示了测试集中的 5 个图像和来自训练集的 6 个图像，这些图像根据这种度量方法来比较它们中的哪一个与其最相似。请注意，在像素层次上，待检测的训练图像通常不会与第一列中的查询图像有较小的 L2 距离。例如，检索到的狗和大象有各种不同的姿势。我们在补充材料中提供了更多测试图像的结果。

通过使用欧式距离来计算两个 4096 维实值向量的相似性，效率不高，但是通过训练自编码器可以将这些向

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 autoencoders 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.

量压缩为较短的二进制码，能够使其更高效。与应用自编码器到原始像素[14]相比，这应该是更好的图像检索方法。它不使用图像标签，因此更倾向于检索具有相似图案边缘的图像，不管它们的图像语义是否相似。

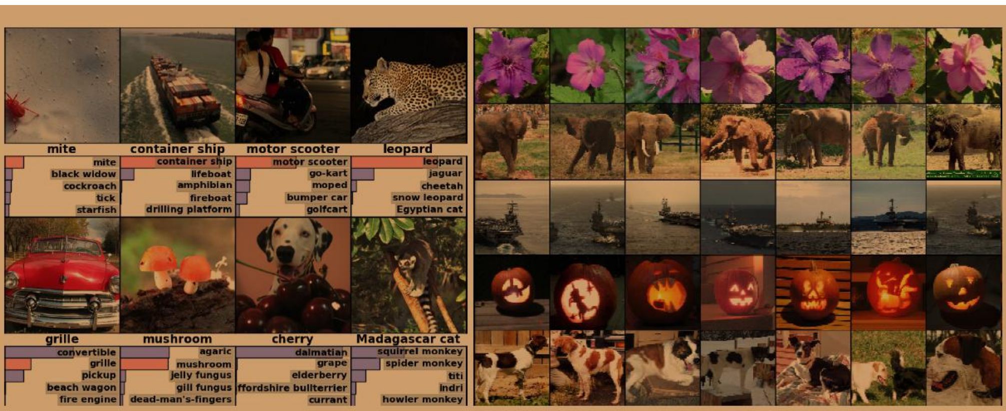


Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010 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.

7 Discussion

Our results show that a large, deep convolutional neural network is capable of achieving recordbreaking 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.

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 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.

7、讨论

我们的研究结果表明，一个大的深层卷积神经网络能够在纯粹使用监督学习的情况下，在极具挑战性的数据集上实现破纪录的结果。值得注意的是，如果移除任何一个卷积层，网络的性能就会下降。例如，删除任何中间层的结果会导致网络性能的 top-1 准确率下降 2%。因此网络的深度对于实现我们的结果真的很重要。

为了简化我们的实验，我们没有使用任何无监督的预训练方法，尽管这样可能会有所帮助，特别是如果我们获得了足够的计算能力来显著地增加网络的大小而不会相应地增加已标记数据的数量。到目前为止，我们的结果已经获得了足够的进步，因为我们已经使网络更大，并且训练了更长时间。但我们仍然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。最后，我们希望对视频序列使用非常大的深度卷积神经网路，其中时间结构提供了非常有用的信息，这些信息往往在静态图像中丢失了，或者说不太明显。