AlexNet

摘要

我们训练了一个大型深度卷积神经网络来将 ImageNet LSVRC-2010 竞赛的 120 万高分辨率的图像分到 1000 不同的类别中。在测试数据上，我们得到了 top-1 37.5%, top-5 17.0%的错误率，这个结果比目前的最好结果好很多。这个神经网络有 6000 万参数和 650000 个神经元，包含 5 个卷积层（某些卷积层后面带有池化层）和 3 个全连接层，最后是一个 1000 维的 softmax。为了训练的更快，我们使用了非饱和神经元并对卷积操作进行了非常有效的 GPU 实现。为了减少全连接层的过拟合，我们采用了一个最近开发的名为 dropout 的正则化方法，结果证明是非常有效的。我们也使用这个模型的一个变种参加了 ILSVRC-2012 竞赛，赢得了冠军并且与第二名 top-5 26.2%的错误率相比，我们取得了 top-5 15.3%的错误率。

1 引言

当前的目标识别方法基本上都使用了机器学习方法。为了提高目标识别的性能，我们可以收集更大的数据集，学习更强大的模型，使用更好的技术来防止过拟合。直到最近，标注图像的数据集都相对较小--在几万张图像的数量级上（例如，NORB[16]，Caltech-101/256 [8, 9]和 CIFAR-10/100 [12]）。简单的识别任务在这样大小的数据集上可以被解决的相当好，尤其是如果通过标签保留变换进行数据增强的情况下。例如，目前在 MNIST 数字识别任务上（<0.3%）的最好准确率已经接近了人类水平[4]。但真实环境中的对象表现出了相当大的可变性，因此为了学习识别它们，有必要使用更大的训练数据集。实际上，小图像数据集的缺点已经被广泛认识到（例如，Pinto et al. [21]），但收集上百万图像的标注数据仅在最近才变得的可能。新的更大的数据集包括 LabelMe [23]，它包含了数十万张完全分割的图像，ImageNet[6]，它包含了 22000 个类别上的超过 1500 万张标注的高分辨率的图像。

为了从数百万张图像中学习几千个对象，我们需要一个有很强学习能力的模型。然而对象识别任务的巨大复杂性意味着这个问题不能被指定，即使通过像 ImageNet 这样的大数据集，因此我们的模型应该也有许多先验知识来补偿我们所没有的数据。卷积神经网络(CNNs)构成了一个这样的模型[16, 11, 13, 18, 15, 22, 26]。它们的能力可以通过改变它们的广度和深度来控制，它们也可以对图像的本质进行强大且通常正确的假设（也就是说，统计的稳定性和像素依赖的局部性）。因此，与具有层次大小相似的标准前馈神经网络，CNNs 有更少的连接和参数，因此它们更容易训练，而它们理论上的最佳性能可能仅比标准前馈神经网络差一点。

尽管 CNN 具有引人注目的质量，尽管它们的局部架构相当有效，但将它们大规模的应用到到高分辨率图像中仍然是极其昂贵的。幸运的是，目前的 GPU，搭配了

高度优化的 2D 卷积实现，强大到足够促进有趣地大量 CNN 的训练，最近的数据集例如 ImageNet 包含足够的标注样本来训练这样的模型而没有严重的过拟合。

本文具体的贡献如下：我们在 ILSVRC-2010 和 ILSVRC-2012[2]的 ImageNet 子集上训练了到目前为止最大的神经网络之一，并取得了迄今为止在这些数据集上报道过的最好结果。我们编写了高度优化的 2D 卷积 GPU 实现以及训练卷积神经网络内部的所有其它操作，我们把它公开了。我们的网络包含许多新的不寻常的特性，这些特性提高了神经网络的性能并减少了训练时间，详见第三节。即使使用了 120 万标注的训练样本，我们的网络尺寸仍然使过拟合成为一个明显的问题，因此我们使用了一些有效的技术来防止过拟合，详见第四节。我们最终的网络包含 5 个卷积层和 3 个全连接层，深度似乎是非常重要的：我们发现移除任何卷积层（每个卷积层包含的参数不超过模型参数的 1%）都会导致更差的性能。

最后，网络尺寸主要受限于目前 GPU 的内存容量和我们能忍受的训练时间。我们的网络在两个 GTX 580 3GB GPU 上训练五六天。我们的所有实验表明我们的结果可以简单地通过等待更快的 GPU 和更大的可用数据集来提高。

2 数据集

ImageNet 数据集有超过 1500 万的标注高分辨率图像，这些图像属于大约 22000 个类别。这些图像是从网上收集的，使用了 Amazon’s Mechanical Turk 的众包工具通过人工标注的。从 2010 年起，作为 Pascal 视觉对象挑战赛的一部分，每年都会举办 ImageNet 大规模视觉识别挑战赛（ILSVRC）。ILSVRC 使用 ImageNet 的一个子集，1000 个类别每个类别大约 1000 张图像。总计，大约 120 万训练图像，50000 张验证图像和 15 万测试图像。

ILSVRC-2010 是 ILSVRC 竞赛中唯一可以获得测试集标签的版本，因此我们大多数实验都是在这个版本上运行的。由于我们也使用我们的模型参加了 ILSVRC-2012 竞赛，因此在第六节我们也报告了模型在这个版本的数据集上的结果，这个版本的测试标签是不可获得的。在 ImageNet 上，按照惯例报告两个错误率：top-1 和 top-5，top-5 错误率是指测试图像的正确标签不在模型认为的五个最可能的便签之中。

ImageNet 包含各种分辨率的图像，而我们的系统要求不变的输入维度。因此，我们将图像进行下采样到固定的 256×256 分辨率。给定一个矩形图像，我们首先缩放图像短边长度为 256，然后从结果图像中裁剪中心的 256×256 大小的图像块。除了在训练集上对像素减去平均活跃度外，我们不对图像做任何其它的预处理。因此我们在原始的 RGB 像素值（中心的）上训练我们的网络

3 架构

我们的网络架构概括为图 2。它包含八个学习层--5 个卷积层和 3 个全连接层。下面，我们将描述我们网络结构中的一些新奇的不寻常的特性。3.1-3.4 小节按照我们对它们评估的重要性进行排序，最重要的最优先。

3.1 ReLU 非线性

将神经元输出 f 建模为输入 x 的函数的标准方式是用 f(x) = tanh(x)或 f(x) = (1 + e-x)-1。考虑到梯度下降的训练时间，这些饱和的非线性比非饱和非线性 f(x) = max(0,x)更慢。根据 Nair 和 Hinton[20]的说法，我们将这种非线性神经元称为修正线性单元(ReLU)。采用 ReLU 的深度卷积神经网络训练时间比等价的 tanh 单元要快几倍。在图 1 中，对于一个特定的四层卷积网络，在 CIFAR-10 数据集上达到 25%的训练误差所需要的迭代次数可以证实这一点。这幅图表明，如果我们采用传统的饱和神经元模型，我们将不能在如此大的神经网络上实验该工作。

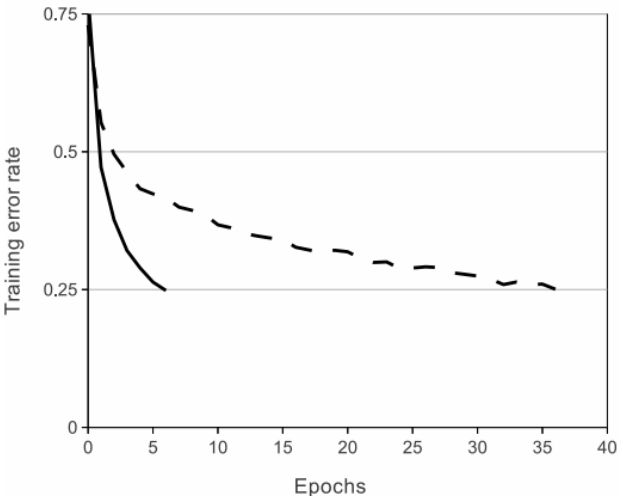


图 1：使用 ReLU 的四层卷积神经网络在 CIFAR-10 数据集上达到 25%的训练误差比使用 tanh 神经元的等价网络（虚线）快六倍。为了使训练尽可能快，每个网络的学习率是单独选择的。没有采用任何类型的正则化。影响的大小随着网络结构的变化而变化，这一点已得到证实，但使用 ReLU 的网络都比等价的饱和神经元快几倍。

我们不是第一个考虑替代 CNN 中传统神经元模型的人。例如，Jarrett 等人[11]声称非线性函数 f(x) = |tanh(x)|与其对比度归一化一起，然后是局部均值池化，在 Caltech-101 数据集上工作的非常好。然而，在这个数据集上主要的关注点是防止过拟合，因此他们观测到的影响不同于我们使用 ReLU 拟合数据集时的加速能力。更快的学习对大型数据集上大型模型的性能有很大的影响。

3.2 多 GPU 训练

单个 GTX580 GPU 只有 3G 内存，这限制了可以在 GTX580 上进行训练的网络最大尺寸。事实证明 120 万图像用来进行网络训练是足够的，但网络太大因此不能在单个 GPU 上进行训练。因此我们将网络分布在两个 GPU 上。目前的 GPU 非常适合跨 GPU 并行，因为它们可以直接互相读写内存，而不需要通过主机内存。我们采用的并行方案基本上每个 GPU 放置一半的核（或神经元），还有一个额外的技巧：只在某些特定的层上进行 GPU 通信。这意味着，例如，第 3 层的核会将第 2 层的所有核映射作为输入。然而，第 4 层的核只将位于相同 GPU 上的第 3 层的核映射作为输入。连接模式的选择是一个交叉验证问题，但这可以让我们准确地调整通信数量，直到它的计算量在可接受的范围内。

除了我们的列不是独立的之外（看图 2），最终的架构有点类似于 Ciresan 等人[5]采用的“columnar” CNN。与每个卷积层一半的核在单 GPU 上训练的网络相比，这个方案降分别低了我们的 top-1 1.7%， top-5 1.2%的错误率。双 GPU 网络比单 GPU 网络稍微减少了训练时间。

ReLU具有让人满意的特性，它不需要通过输入归一化来防止饱和。如果至少一些训练样本对ReLU产生了正输入，那么那个神经元上将发生学习。然而，我们仍然发现接下来的局部响应归一化有助于泛化。a_x, y^i表示神经元激活，通过在(x, y)位置应用核i，然后应用ReLU非线性来计算，响应归一化激活b^i_x, y通过下式给定：

$$b^i\_x, y = a\_x, y^i / (k + \alpha \sum\_j = max(0, i - n/2)^{min(N-1, i+n/2)} (a\_x, y^i)^2)^\beta$$

求和运算在 n 个“毗邻的”核映射的同一位置上执行，N 是本层的卷积核数目。核映射的顺序当然是任意的，在训练开始前确定。响应归一化的顺序实现了一种侧抑制形式，灵感来自于真实神经元中发现的类型，为使用不同核进行神经元输出计算的较大活动创造了竞争。常量 k， n， α， β 是超参数，它们的值通过验证集确定；我们设 k=2，n=5，α=0.0001，β=0.75。我们在特定的层使用的 ReLU 非线性之后应用了这种归一化（请看 3.5 小节）。

这个方案与 Jarrett 等人[11]的局部对比度归一化方案有一定的相似性，但我们更恰当的称其为“亮度归一化”，因此我们没有减去均值。响应归一化分别减少了 top-1 1.4%， top-5 1.2%的错误率。我们也在 CIFAR-10 数据集上验证了这个方案的有效性：一个乜嘢归一化的四层 CNN 取得了 13%的错误率，而使用归一化取得了 11%的错误率

3.4 重叠池化

CNN 中的池化层归纳了同一核映射上相邻组神经元的输出。习惯上，相邻池化单元归纳的区域是不重叠的（例如[17, 11, 4]）。更确切的说，池化层可看作由池化单元网格组成，网格间距为个像素，每个网格归纳池化单元中心位置大小的邻居。如果设置…，我们会得到通常在 CNN 中采用的传统局部池化。如果设置…，我们会得到重叠池化。这就是我们网络中使用的方法，设置…，…。这个方案分别降低了 top-1 0.4%，top-5 0.3%的错误率，与非重叠方案…相比，输出的维度是相等的。我们在训练过程中通常观察采用重叠池化的模型，发现它更难过拟合。

3.5 整体架构

现在我们准备描述我们的 CNN 的整体架构。如图 2 所示，我们的网络包含 8 个带权重的层；前 5 层是卷积层，剩下的 3 层是全连接层。最后一层全连接层的输出是 1000 维 softmax 的输入，softmax 会产生 1000 类标签的分布。我们的网络最大化多项逻辑回归的目标，这等价于最大化预测分布下训练样本正确标签的对数概率的均值。

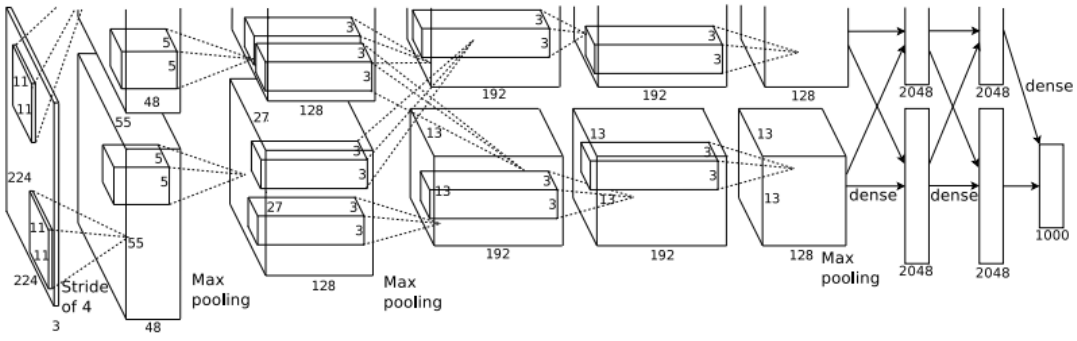


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input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

第 2，4，5 卷积层的核只与位于同一 GPU 上的前一层的核映射相连接（看图 2）。第 3 卷积层的核与第 2 层的所有核映射相连。全连接层的神经元与前一层的所有神经元相连。第 1，2 卷积层之后是响应归一化层。3.4 节描述的这种最大池化层在响应归一化层和第 5 卷积层之后。ReLU 非线性应用在每个卷积层和全连接层的输出上。

第 1 卷积层使用 96 个核对 224 × 224 × 3 的输入图像进行滤波，核大小为 11 × 11 × 3，步长是 4 个像素（核映射中相邻神经元感受野中心之间的距离）。第 2 卷积层使用用第 1 卷积层的输出（响应归一化和池化）作为输入，并使用 256 个核进行滤波，核大小为 5 × 5 × 48。第 3，4，5 卷积层互相连接，中间没有接入池化层或归一化层。第 3 卷积层有 384 个核，核大小为 3 × 3 × 256，与第 2 卷积层的输出（归一化的，池化的）相连。第 4 卷积层有 384 个核，核大小为 3 × 3 × 192，第 5 卷积层有 256 个核，核大小为 3 × 3 × 192。每个全连接层有 4096 个神经元。

4 减少过拟合

我们的神经网络架构有 6000 万参数。尽管 ILSVRC 的 1000 类使每个训练样本从图像到标签的映射上强加了 10 比特的约束，但这不足以学习这么多的参数而没有相当大的过拟合。下面，我们会描述我们用来克服过拟合的两种主要方式。

4.1 数据增强

图像数据上最简单常用的用来减少过拟合的方法是使用标签保留变换（例如[25, 4, 5]）来人工增大数据集。我们使用了两种独特的数据增强方式，这两种方式都可以从原始图像通过非常少的计算量产生变换的图像，因此变换图像不需要存储在硬盘上。在我们的实现中，变换图像通过 CPU 的 Python 代码生成，而此时 GPU 正在训练前一批图像。因此，实际上这些数据增强方案是计算免费的。

第一种数据增强方式包括产生图像变换和水平翻转。我们从 256×256 图像上通过随机提取 224 × 224 的图像块实现了这种方式，然后在这些提取的图像块上进行训练。这通过一个 2048 因子增大了我们的训练集，尽管最终的训练样本是高度相关的。没有这个方案，我们的网络会有大量的过拟合，这会迫使我们使用更小的网络。在测试时，网络会提取 5 个 224 × 224 的图像块（四个角上的图像块和

对

第二种数据增强方式包括改变训练图像的RGB通道的强度。具体地，我们在整个ImageNet训练集上对RGB像素值集合执行PCA。对于每幅训练图像，我们加上多倍找到的主成分，大小成正比的对应特征值乘以一个随机变量，随机变量通过均值为0，标准差为0.1的高斯分布得到。因此对于每幅RGB图像像素$I_xy = [I^R_xy, I^G_xy, I^B_xy]^T$，我们加上下面的数量：

$$[p\_1, p\_2, p\_3][\alpha\_1\lambda\_1, \alpha\_2\lambda\_2, \alpha\_3\lambda\_3]^T$$

p_i，λ_i分别是RGB像素值3 × 3协方差矩阵的第i个特征向量和特征值，α_i是前面提到的随机变量。对于某个训练图像的所有像素，每个α_i只获取一次，直到图像进行下一次训练时才重新获取。这个方案近似抓住了自然图像的一个重要特性，即光照的颜色和强度发生变化时，目标身份是不变的。这个方案减少了`top 1`错误率1%以上。

4.2 失活(Dropout)

将许多不同模型的预测结合起来是降低测试误差[1, 3]的一个非常成功的方法，但对于需要花费几天来训练的大型神经网络来说，这似乎太昂贵了。然而，有一个

非常有效的模型结合版本，它只花费两倍的训练成本。这种最近引入的技术，叫做“dropout”[10]，它会以 0.5 的概率对每个隐层神经元的输出设为 0。那些“失活的”的神经元不再进行前向传播并且不参与反向传播。因此每次输入时，神经网络会采样一个不同的架构，但所有架构共享权重。这个技术减少了复杂的神经元互适应，因为一个神经元不能依赖特定的其它神经元的存在。因此，神经元被强迫学习更鲁棒的特征，它在与许多不同的其它神经元的随机子集结合时是有用的。在测试时，我们使用所有的神经元但它们的输出乘以 0.5，对指数级的许多失活网络的预测分布进行几何平均，这是一种合理的近似。

我们在图 2 中的前两个全连接层使用失活。如果没有失活，我们的网络表现出大量的过拟合。失活大致上使要求收敛的迭代次数翻了一倍。

5 学习细节

我们使用随机梯度下降来训练我们的模型，样本的batch size为128，动量为0.9，权重衰减为0.0005。我们发现少量的权重衰减对于模型的学习是重要的。换句话说，权重衰减不仅仅是一个正则项：它减少了模型的训练误差。权重w的更新规则是

$$v\_i + 1 := 0.9 \bullet v\_i - 0.0005 \bullet \varepsilon \bullet w\_i - \varepsilon \bullet \langle \frac{\partial L}{\partial w} |\_w\_i \rangle \_D\_i$$

i是迭代索引，v是动量变量，ε是学习率，$\langle \frac{\partial L}{\partial w} |_w_i \rangle _D_i$是目标函数对$w$，在$w_i$上的第$i$批微分$D_i$的平均。

我们使用均值为 0，标准差为 0.01 的高斯分布对每一层的权重进行初始化。我们在第 2，4，5 卷积层和全连接隐层将神经元偏置初始化为常量 1。这个初始化通过为 ReLU 提供正输入加速了学习的早期阶段。我们在剩下的层将神经元偏置初始化为 0。

我们对所有的层使用相等的学习率，这个是在整个训练过程中我们手动调整得到的。当验证误差在当前的学习率下停止提供时，我们遵循启发式的方法将学习率除以 10。学习率初始化为 0.01，在训练停止之前降低三次。我们在 120 万图像的训练数据集上训练神经网络大约 90 个循环，在两个 NVIDIA GTX 580 3GB GPU 上花费了五到六天。

6 结果

我们在 ILSVRC-2010 上的结果概括为表 1。我们的神经网络取得了 `top-1 37.5%`，`top-5 17.0%`的错误率。在 ILSVRC-2010 竞赛中最佳结果是 `top-1 47.1%`，`top-5 28.2%`，使用的方法是对 6 个在不同特征上训练的稀疏编码模型生成的预测进行平均，从那时起已公布的最好结果是 `top-1 45.7%`，`top-5 25.7%`，使用的方法是平均在 Fisher 向量（FV）上训练的两个分类器的预测结果，Fisher 向量是通过两种密集采样特征计算得到的[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.

表 1：ILSVRC-2010 测试集上的结果对比。斜体是其它人取得的最好结果。

我们也用我们的模型参加了 ILSVRC-2012 竞赛并在表 2 中报告了我们的结果。由于 ILSVRC-2012 的测试集标签不可以公开得到，我们不能报告我们尝试的所有模型的测试错误率。在这段的其余部分，我们会使用验证误差率和测试误差率互换，因为在我们的实验中它们的差别不会超过 0.1%（看图 2）。本文中描述的 CNN 取得了 top-5 18.2%的错误率。五个类似的 CNN 预测的平均误差率为 16.4%。为了对 ImageNet 2011 秋季发布的整个数据集（1500 万图像，22000 个类别）进行分类，我们在最后的池化层之后有一个额外的第 6 卷积层，训练了一个 CNN，然后在它上面进行“fine-tuning”，在 ILSVRC-2012 取得了 16.6%的错误率。对在 ImageNet 2011 秋季发布的整个数据集上预训练的两个 CNN 和前面提到的五个 CNN 的预测进行平均得到了 15.3%的错误率。第二名的最好竞赛输入取得了 26.2%的错误率，他的方法是对 FV 上训练的一些分类器的预测结果进行平均，FV 在不同类型密集采样特征计算得到的。

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.

表 2：ILSVRC-2012 验证集和测试集的误差对比。斜线部分是其它人取得的最好的结果。带星号的是“预训练的”对 ImageNet 2011 秋季数据集进行分类的模型。更多细节请看第六节。

最后，我们也报告了我们在 ImageNet 2009 秋季数据集上的误差率，ImageNet 2009 秋季数据集有 10,184 个类，890 万图像。在这个数据集上我们按照惯例用一半的图像来训练，一半的图像来测试。由于没有建立测试集，我们的数据集分割有必要不同于以前作者的数据集分割，但这对结果没有明显的影响。我们在这个数据集上的的 top-1 和 top-5 错误率是 67.4%和 40.9%，使用的是上面描述的在最后的池化层之后有一个额外的第 6 卷积层网络。这个数据集上公开可获得的最好结果是 78.1%和 60.9%[19]。

6.1 定性评估

图 3 显示了网络的两个数据连接层学习到的卷积核。网络学习到了大量的频率核、方向选择核，也学到了各种颜色点。注意两个 GPU 表现出的专业化，3.5 小节中描述的受限连接的结果。GPU 1 上的核主要是没有颜色的，而 GPU 2 上的核主要是针对颜色的。这种专业化在每次运行时都会发生，并且是与任何特别的随机权重初始化（以 GPU 的重新编号为模）无关的。

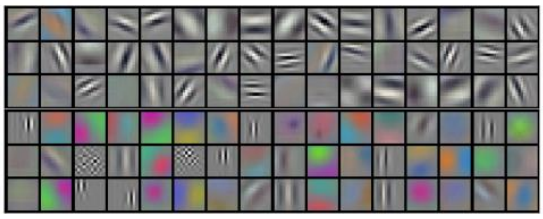
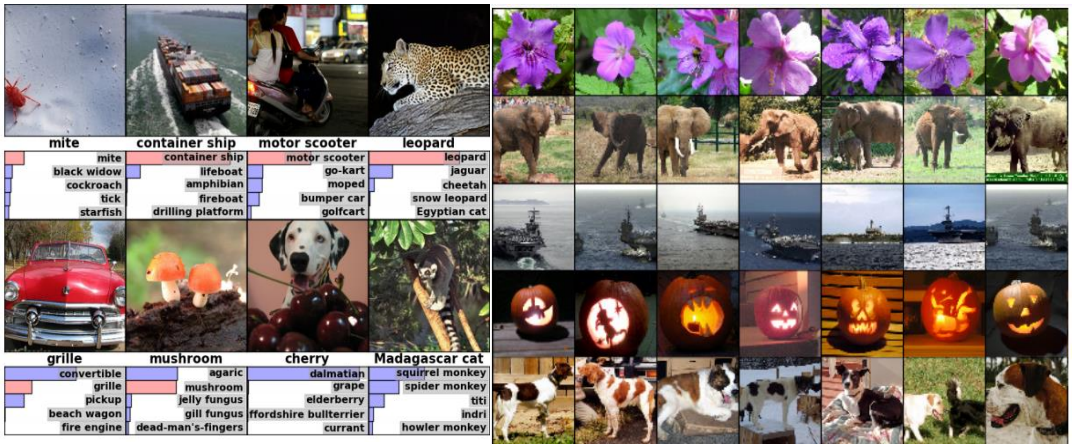


Figure 3: 96 convolutional kernels of size $11\times11\times3$ learned by the first convolutional layer on the $224\times224\times3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

图 3：第一卷积层在 224×224×3 的输入图像上学习到的大小为 11×11×3 的 96 个卷积核。上面的 48 个核是在 GPU 1 上学习到的而下面的 48 个卷积核是在 GPU 2 上学习到的。更多细节请看 6.1 小节。



在图 4 的左边部分，我们通过在 8 张测试图像上计算它的 top-5 预测定性地评估了网络学习到的东西。注意即使是不在图像中心的目标也能被网络识别，例如左上角的小虫。大多数的 top-5 标签似乎是合理的。例如，对于美洲豹来说，只有其它类型的猫被认为是看似合理的标签。在某些案例（格栅，樱桃）中，网络在意的图片焦点真的很含糊。

图 4：（左）8 张 ILSVRC-2010 测试图像和我们的模型认为最可能的 5 个标签。每张图像的下面是它的正确标签，正确标签的概率用红条表示（如果正确标签在 top 5 中）。（右）第一列是 5 张 ILSVRC-2010 测试图像。剩下的列展示了 6 张训练图像，这些图像在最后的隐藏层的特征向量与测试图像的特征向量有最小的欧氏距离。

探索网络可视化知识的另一种方式是思考最后的 4096 维隐藏层在图像上得到的特征激活。如果两幅图像生成的特征激活向量之间有较小的欧式距离，我们可以认为神经网络的更高层特征认为它们是相似的。图 4 表明根据这个度量标准，测试集的 5 张图像和训练集的 6 张图像中的每一张都是最相似的。注意在像素级别，检索到的训练图像与第一列的查询图像在 L2 上通常是不接近的。例如，检索的狗和大象似乎有很多姿态。我们在补充材料中对更多的测试图像呈现了这种结果。

通过两个 4096 维实值向量间的欧氏距离来计算相似性是效率低下的，但通过训练一个自动编码器将这些向量压缩为短二值编码可以使其变得高效。这应该会产生一种比将自动编码器应用到原始像素上[14]更好的图像检索方法，自动编码器应用到原始像素上的方法没有使用图像标签，因此会趋向于检索与要检索的图像具有相似边缘模式的图像，无论它们是否是语义上相似。

7 探讨

我们的结果表明一个大型深度卷积神经网络在一个具有高度挑战性的数据集上使用纯有监督学习可以取得破纪录的结果。值得注意的是，如果移除一个卷积层，我们的网络性能会降低。例如，移除任何中间层都会引起网络损失大约 2%的 top-1 性能。因此深度对于实现我们的结果非常重要。

为了简化我们的实验，我们没有使用任何无监督的预训练，尽管我们希望它会有所帮助，特别是在如果我们能获得足够的计算能力来显著增加网络的大小而标注的数据量没有对应增加的情况下。到目前为止，我们的结果已经提高了，因为我们的网络更大、训练时间更长，但为了匹配人类视觉系统的下颞线（视觉专业术语）我们仍然有许多数量级要达到。最后我们想在视频序列上使用非常大的深度卷积网络，视频序列的时序结构会提供非常有帮助的信息，这些信息在静态图像上是缺失的或远不那么明显。