文献翻译和原稿

深度卷积神经网络对 ImageNet 数据库实现图像分类

Alex. Krizhevsky	Ilya Sutskever	Geoffrey E. Hinton
University of Toronto	University of Toronto	University of Toronto
kriz@cs.utoronto.ca	ilya@cs.utoronto.ca	hinton@cs.utoronto.ca

摘要

我们训练了一个大型的深度卷积神经网络，对 ImageNet LSVRC-2010 数据库里的 120 万幅高分辨率图像进行分类，将图像分为 1000 个不同的类别。在训练数据的 top-1 和 top-5 分类上，我们的错误率分别是 37.5%和 17.0%，算法显著地超越了以往算法的最好水平。我们训练的神经网络有 6000 万个参数和 65 万神经元，总共有 5 个卷积层，其中一些是最大池化层，还有 3 个全相连的层及最后的 1000 路 Softmax 分类器。为了使训练更快，我们用了非饱和的神经元，同时用一个非常高效的 GPU 来做卷积操作。为了减少全相连层的过适应问题，我们使用最近几年被提出的叫“dropout”的正则化方法，这种方法十分有效。我们还把这个模型稍作修改，参加 ILSVRC-2012 竞赛并在 top-5 上达到 15.3%的错误率，而次好的参赛模型的错误率是 26.2%。

1 介绍

当前对目标识别的研究基本建立在机器学习的方法上。为了改善它们，我们可以收集越来越多的数据集，学习更好的模型，还可以用更好的技术防止过适应。直到最近几年，有标签的数据集依然相对比较小，大约是上万幅图片左右（比如 NORB[16], Caltech-101/256[8, 9], CIFAR-10/100[12]）。这样大小的数据库图片数量可以胜任简单的识别任务，尤其是当它们被标签保留转换增强后。例如，当前对 MNIST 手写数字数据库的最好算法保证错误率小于 0.3%，接近了人类的表现[4]。

不过对真实场景的识别，当前算法表现很不稳定，所以要学习如何识别真实场景需要用更大的训练数据集。而且实际上小的图像数据集的缺点已经被广泛地指出（比如 Pinto[21]等人），但直到最近，收集包含百万幅有标签的图像数据集才成为可能。新的更大的数据集有 LabelMe[23]，它包含数十万幅全分割的图像；还有 ImageNet[6]，它包含超过 1500 万、2200 类有标签的高分辨率图像。

为了从数百万幅图片里学习出上千个目标，我们需要一个模型有很大的学习容量。然而，目标识别任务的巨大复杂度意味着我们不能仅仅依赖于 ImageNet 数据库，模型还需要大量的先验知识来填补所有我们没有的数据。卷积神经网络(CNNs)由一组模型[16, 11, 13, 18, 15, 22, 26]组成，通过调整它们的深度和广度，我们可以控制模型的容量。它们也能对自然图像做出鲁棒的和最正确的假设（也就是说，统计上的平稳和像素依赖于位置）。因此，相比于标准的层间大小相似的前馈神经网络，CNNs 的连接数更少所以它们更易于训练， CNNs 的理论最佳表现也仅仅稍差一些。

CNNs 除了有这些吸引人的特点，除了它们局部特征在相关性上很高效，它们还有很好的延展性来适应高分辨率图像。幸运的是，使用 GPU 进行高度优化过的二维卷积运算，足够训练我们想要的大小的 CNNs。现在例如 ImageNet 的数据集也有足够多的有标签样本来训练这样的模型，而不会有较多的过适应现象。

这篇论文的创新之处在于：我们训练了迄今为止最大的卷积神经网络之一，它在 ImageNet 的 ILSVRC-2010 和 ILSVRC-2012 竞赛的数据集上更新自己，得到的结果远超以往运行在这些数据集上的算法。我们写了一个 GPU 实现的高度优化二维卷积，而其它的操作与卷积神经网络的训练相同，正如我们公开的那样[1]。我们的网络包含许多新的、不常见的特征以改善网络的表现和减少训练时间，这部分内容会在第 3 部分详述。尽管有 120 万有标签训练样本，我们的网络的规模意味着过适应是一个很重要的问题，所以我们采用了一些有效的技术来防止过适应，这部分内容会在第 4 部分详述。最终的网络包含 5 个卷积层和 3 个全相连层，这样的深度很重要：我们发现移除任意的卷积层（每一层包含的模型参数不超过总量的 1%）都会让模型表现更差。

最后，这个网络的规模主要限制于 GPU 的内存空间，还有我们能够容忍的

[1] http://code.google.com/p/cuda-convnet/

训练时间。我们花费 5 至 6 天在两块 GTX 580 3GB GPU 上训练我们的网络。我们所有的实验都显示出，如果有更快的 GPU 或更大的数据集，我们的结果可以变得更好。

2 数据集

ImageNet数据集拥有超过1500万幅有标签的高分辨率图像，它们分属于大概2200类。这些图像是从网上收集的，由Amazon’s Mechanical Turk crowd-sourcing工具来人工标定。从2010年开始，ImageNet Large-Scale Visual Recognition Challenge (ILSVRC)竞赛每年举行一次，作为Pascal Visual Object Challenge的一部分。ILSVRC竞赛使用ImageNet里共1000类，每类大约1000幅图片作为数据集，总共大约有120万幅训练图像，5万验证图像，和15万测试图像。

ILSVRC-2010是ILSVRC里仅有的有测试集标签的版本，所以我们大多数的实验都在这个版本上进行。不过我们依然用我们的模型参加了ILSVRC-2012竞赛，在第6部分里我们会报告竞赛结果，虽然它没有测试集标签。在ImageNet上，通常需要报告两个错误率：top-1和top-5，top-5错误率指的是图像正确的标签不属于模型预测出的5个最合适的标签。

ImageNet里图像分辨率并不统一，但我们的模型需要输入数据的维度是常量，因此我们将所有图像降采样到256×256分辨率。如果输入的图像是长方形，我们首先把较短的一边重新调节到256长度，然后将正中的256×256图像块拿出。除了对每个像素操作以减少训练集的平均活度，我们不对图像作其它任何预处理，所以我们网络的训练输入是像素的原始RBG值。

3 结构

我们网络的结构在图2里描述。它包括8个学习层：5个卷积层和3个全相连层。下面，我们介绍网络里一些新的和不常见的特征。3.1到3.4部分依据对它们的重要性排序，最重要的在最前面。

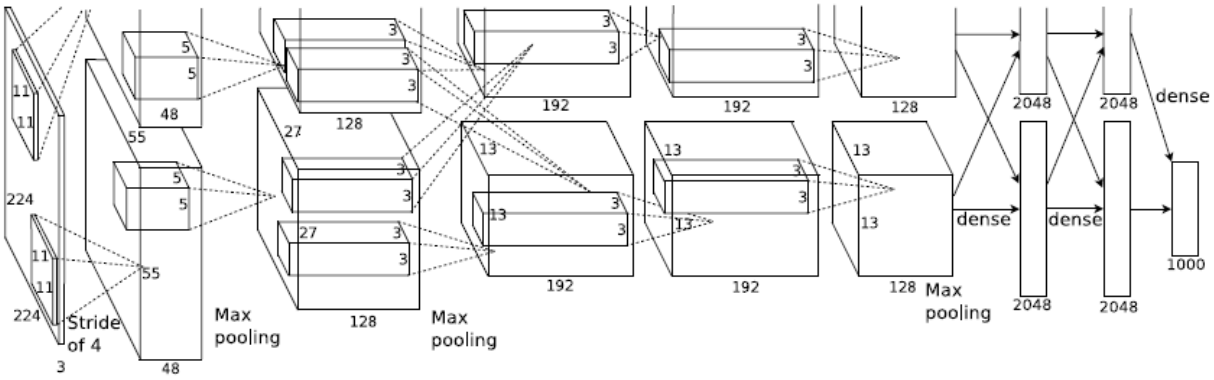

图2：这是我们CNN网络结构的描述。明确展示了两个GPU各自的任务。一个GPU运行在结构高层部分，另一个在底层部分。两个GPU只在某些层有联系。这个网络的输入是150528维，每一层的神经元数分别是253440-186624-64896-64896-43264-4096-4096-1000。

3.1 ReLU非线性

一般方法里，一个神经元的输出 f 对它的输入 x 是 $f(x)=\tanh(x)$，或者 $f(x)=(1+e^{-x})^{-1}$。考虑到梯度下降的训练时间，它们的饱和非线性要远慢于 $f(x)=\max(0,x)$ 这样的不饱和非线性。根据Nair和Hinton[20]的工作，我们把这样的非线性神经元叫做Rectified Linear Units (ReLUs)。含ReLUs的深度卷积神经网络的训练要比 tanh 单元快上几倍。这在图1里有证明，展示了一个四层卷积网络为在CIFAR-10数据集上达到25%的训练误差所需要的迭代次数。这张图表明如果我们用传统的饱和神经元模型，不太可能实现如此巨大的神经网络。

我们并不是第一个考虑替换CNNs里的传统神经元模型的。例如，Jarrett等人[11]发表文章说非线性函数 $f(x)=|\tanh(x)|$ 在用他们自己定义的的对比度归一化方法对Caltech-101数据集做局部平均池化时表现非常好。然而，关于这个数据集以前的研究都集中在防止过适应。所以他们观察到的效应与我们报告的使用ReLUs适应训练数据集获得的加速并不同。更快的学习速度会给在大数据集上训练的大模型产生巨大的影响。

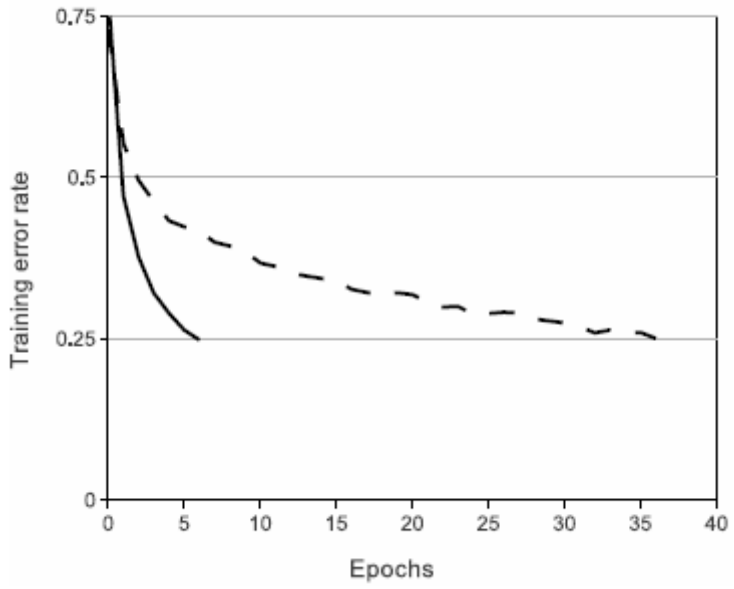

图1：一个使用ReLUs（**实线**）四层卷积网络在CIFAR-10数据集上达到25%的训练误差比使用 tanh 神经元（**虚线**）快6倍。每一个网络的训练比率是独立选择的，以达到最快的训练速度。没有附加任何正则化形式。这里演示的效率幅度会随着网络结构变化，不过使用ReLUs神经元的网络稳定比其余使用饱和神经元的网络快上数倍。

3.2 在多个GPU上训练

单个GTX 580 GPU只有3GB的内存，限制了用其训练的网络的规模。我们发现，用120万训练样本训练网络对于一个GPU来说太多了，因此我们将网络分布在两个GPU上训练。现在的GPU之间可以很好地交叉并行计算，它们可以直接读和写其它GPU的内存，不需要通过主机的内存。我们构建的平行架构分别把一半的核（神经元）放在一个GPU上，和一个附加的改动：GPU之间只在某些层作数据交换。这就意味着，例如第三层的神经元完全接收第2层的神经元的输出，然后第4层神经元只是接收和第3层在同一个GPU上的神经元的输出。选择相连的模式是交叉验证的一个问题，不过这让我们可以精确地调整数据交换的数量，直到这样数量的计算量是可接受为止。

这个最终的结构在一定程度上与Ciresan[5]等人的“柱状”CNN相似，除了我们的柱并不是独立的（见图2）。与使用一个GPU训练一半神经元相比，这个结构分别减少了我们的top-1和top-5错误率1.7%、1.2%，两个GPU网络的训练时间也

一个GPU的训练时间稍少一些[2]。

3.3 局部响应归一化

ReLUs一个吸引人的特点是它不需要对输入归一化以防止输入饱和。如果至少有一些训练样本给一个ReLU提供正输入，这个神经元就会开始学习。然而，我们依然发现下面所说的局部归一化结构对泛化有帮助。$a^i_{x,y}$表示一个神经元在核i和位置(x, y)上的行为，然后对其进行ReLU非线性操作，得到响应归一化后的行为$b^i_{x,y}$：

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

公式中的求和经过n个在同一空间位置的“相邻“内核映射，而N是该层核的总数。内核映射的次序是任意的，在训练开始前决定。这种响应归一化在某种形式上实现了真正的神经元带给我们的灵感：在不同的核中计算神经元的输出以创造竞争。常量k，n，α和β被一个有效的验证集决定：我们使$k = 2$，$n = 5$，$\alpha = 10^{-4}$，和$\beta = 0.75$。在进行ReLU非线性操作后我们在某些层进行这样的归一化（见3.5部分）。

这个结构与Jarrett[11]等人的局部对比归一化有一点相似，不过我们的更可以被称为“亮度归一化”，因为我们并不做去平均的操作。响应归一化分别减少了我们网络的top-1和top-5错误率1.4%、1.2%。我们还在CIFAR-10数据集上验证了这个结构的效率：一个4层的CNN，当没有归一化时的测试错误率是13%，有归一化的错误率11%[3]。

3.4 重叠池化

CNNs里的池化层对同一个内核映射里相邻的神经元组的输出求和。传统上，

---

[2] 一个 GPU 的网络实际上在最后卷积层与两个 GPU 的网络有一样多的神经元。这是因为网络的大多数参数都在第一个全相连层，它将最后一个卷积层的的输出作为输入。所以我们让这两个网络有相同数目的参数，并不减半最后卷积层的大小（亦不减半后面跟着的全相连层）。因此这个比较有一些偏向一个 GPU 的网络，因为它比两个 GPU 网络的“一半“要大。

[3] 限于文章限制，我们不能具体描述这个网络，不过你可以通过它的代码和参数文件来准确了解：http://code.google.com/p/cuda-convnet/

相连池化单元的求和不重叠（例如文献[17, 11, 4]）。为了更加精确，一个池化层可以认为是一个网格被分离出 s 个像素的池化单元，每个都对以池化单元为中心、大小为 $z \times z$ 的相邻区域求和。如果设 $s = z$，就得到经常用在CNNs里的传统局部池化。如果设 $s < z$，就是重叠池化。我们将重叠池化用在我们的网络里，同时设定 $s = 2$ 和 $z = 3$。与 $s = 2$ 和 $z = 2$ 相同维度的输出相比，这个设定分别减少了top-1和top-5错误率0.4%和0.3%。我们还观察到用重叠池化训练的模型更难以过适应。

3.5 总体结构

现在我们可以介绍我们的CNN的整体结构了。就像图2所描述的那样，这个网络包含8个权重网络：前5个是卷积层，剩下的3个是全相连层，最后全相连层的输出通过一个1000路softmax分类器，分配超过1000个类标签。我们的网络最大化了逻辑回归目标，相当于最大化了预测分类时正确标签的训练样本的平均对数概率。

第2、4、5层卷积层的核仅被连接到前一层处于相同GPU的核上（见图2），第3层卷积层的核全部连接到第二层映射的核上，全相连层的神经元全部连接到前一层上。响应归一化层跟在第1层和第2层卷积层后面。3.4部分描述的最大池化层放在响应归一化层和第4层卷积层后。ReLU非线性操作在每一个卷积层和全相连层的输出后进行。

第一个卷积层对224×224×3的输入图片用96个大小为11×11×3的核做步长为4个像素（这是一个内核映射里相邻神经元感受野中心的距离）的滤波操作。第二层卷积层将第一层的输出（经过了响应归一化和池化）作为输入，并用256个大小为5×5×48的核做滤波操作。第3、4、5层卷积层之间都没有归一化或池化操作。第3层卷积层有384个大小为3×3×256的核连接到第2层卷积层的输出（经过了归一化和池化）。第4层卷积层有384个大小为3×3×192的核，第5层卷积层有256个大小为3×3×192的核。每个全相连层有4096个神经元。

4. 减少过适应

我们的神经网络结构有6000万个参数。ILSVRC里需要将图像分为1000类，所

以每个训练样从图像到标签需要10个比特来表示。除了相当大的过适应，看起来并不需要如此多的参数。接下来，我们将介绍两个我们用来防止过适应的基本方法。

4.1 数据增广

最简单和最常见的用来减少图像数据过适应的方法是使用标签保护变换（例如[25, 4, 5]）来人工增大数据集。因为只需对原始图像做很少的计算就可以得到变换后的图像，所以变换后的图像并不需要存储在硬盘上。在我们的实验里，当GPU在训练前一批图像时，CPU上的Python代码生成新的变换后图像，所以实际上这个数据增广结构从计算代价来说是免费的。

第一种数据增广包括图像的平移和水平反射。我们从256×256的图像里随机地提取出224×224图像块（和它们的水平反射），然后使用这些图像块训练我们的网络[4]。这将我们的训练数据集提升了2048倍大小。如果没有这个结构，我们的网络会产生大量的过适应，迫使我们只能选择小得多的网络。在测试的时候，网络对提取出的5个224×224图像块（4个角落的图像块和1个中心图像块）及它们的水平反射作预测（因此总共是10个图像块），然后将网络softmax层对这10个图像块的预测结果作平均。

第二种数据增广是改变训练图像的RGB值。特别的，我们对ImageNet训练集里的RGB像素值作PCA变换。对每一幅训练图像，我们成倍地加上基本主成分，倍数是与特征值大小成正比的数乘以一个均值为0、标准差为0.1的高斯变量。因此，对每个RBG像素值 $I_{xy}=[I_{xy}^R, I_{xy}^G, I_{xy}^B]^T$ 我们乘以下式：

$$[p_1, p_2, p_3][\alpha_1\lambda_1, \alpha_2\lambda_2, \alpha_3\lambda_3]^T$$

其中 p_i 是第 i 个特征向量，λ_i 是RGB像素值的3×3协方差矩阵的特征值，α_i 是前面所述的随机变量。一幅训练图像的的所有像素只使用一个 α_i 值，直到这幅图像被再次用来训练，α_i 值会被重新选取。这个结构大致契合自然图像的一个重要性质：物体的属性不随照度的大小和颜色改变。这个结构减少了超过1%的top-1错误率。

---

[4] 这就是为什么图 2 里的输入图像的维度是 224×224×3。

4.2 舍弃

将许多不同模型的预测结合起来可以很成功地减少测试错误率[1,3]，不过对于一个已经训练了数天的巨大神经网络来说这看起来十分昂贵。然而，现在有一个十分有效的模型合并技术叫作“舍弃”[10]，它只花费二分之一的时间来训练，以0.5的概率随机将任意隐藏神经元的输出置零。被“舍弃”的神经元不再对前向传播产生影响，也不参与到反向传播里。所以每当有一次输入，神经网络都是不同的结构，不过所有的这些结构共享权重。这种技术减少了神经元间复杂的互适应，因为一个神经元并不依赖于其它神经元而存在。这就是说，网络会学习更鲁棒的特征以用在与其它神经元的许多不同随机子集的连接里。在测试的时候，我们使用了所有的神经元，不过对它们的输出乘上0.5，这是对指数方式产生的预测分布几何平均的有理由近似——许多网络被舍弃掉了。

我们在图2中前两层全相连层使用舍弃。如果不加舍弃，我们的网络表现出严重的过适应。舍弃大约翻倍了网络要收敛所需的迭代次数。

5 学习的细节

我们用随机梯度下降法训练我们的模型，块大小是128个样本，动量是0.9，权重衰减参数是0.0005。我们发现这个很小的权重衰减对于学习模型很重要，换句话说，这里的权重衰减不仅仅是一个正则项，它还减少了模型的训练误差。权重 w 的更新规则是：

$$v_{i+1} := 0.9 \cdot vi - 0.0005 \cdot \varepsilon \cdot w_i - \varepsilon \cdot \left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}$$

$$w_{i+1} := w_i + v_{i+1}$$

这里 i 是迭代序号，v 是动量变量，ε 是学习速率，$\left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}$ 是在 w_i 上计算的与 w 相关的 D_i 上第 i 个目标导数的平均。

我们在每一层上用零均值、标准差为0.01的高斯分布来初始化权重。我们在第2、4、5层卷积层及全相连隐层上用常数1来初始化神经元的偏置。这个初始化给ReLUs提供了正输入，因此加速了神经网络早期的学习。我们将余下层的神经

元偏置设为常数0。

我们给所有层相等的学习速率，在训练时加以手动调整。我们遵循一个启发式的方法，即在当前的学习速率下验证错误率停止改善时，我们将学习速率除以10。学习速率被初始化为0.01，从一开始到最后会减少3次。我们用120万幅图片的训练集循环训练网络大约90次，在两块GTX 580 3GB GPU上花费了5至6天时间。

6. 结果

我们在ILSVRC-2010上的训练结果展示在表1里，测试集的top-1和top-5错误率分别是37.5%和17.0%[5]。ILSVRC-2010竞赛中的最佳成绩分别是47.1%和28.2%，使用的是在不同特征上训练的6个稀疏编码模型的预测取平均[2]。至今为止公开发表的最好结果是45.7%和25.7%，他们从两种密度采样得到的特征在Fisher Vectors(FVs)上训练二分类器并对预测取平均[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%**

表1：比较ILSVRC-2010测试集结果。*斜体字*是其他人取得的最好结果。

我们还用模型参加了ILSVRC-2012竞赛，在表2中报告我们的结果。鉴于ILSVRC-2012测试集标签没有公开，我们不能报告我们试过的所有模型的错误率。在这段里，验证错误率和测试错误率是几乎等同的，因为在我们的实验里它们没有相差超过0.1%（见表2）。这篇文章所描述的CNN达到18.2%的top-5错误率，对5个相似的CNN的预测取平均可以达到16.4%的错误率。我们还训练了一个CNN，在最后池化层后附加6个额外的卷积层，来对整个ImageNet Fall 2011数据集（1500万幅图像，22000类）做分类，微调后把它用在ILSVRC-2012里，达到16.6%的错误率。我们还对两个CNN的预测做平均，这两个CNN用了前面所述的5个在整个Fall 2011数据集上训练过的CNN作网络预训练，结果达到15.3%。这次竞赛第二好的成绩是26.2%错误率，他们从不同密度采样得到的特征在FVs上训

[5] 4.1 部分描述的没有对 10 块做平均的预测错误率分别是 39.0%和 18.3%。

练多分类器并对预测取平均[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%**

表2：比较ILSVRC-2012验证集和测试集的错误率。*斜体字*是其他人取得的最好结果。带星号*的模型是经过了整个ImageNet Fall 2011数据集作预训练的，请参见第6部分的详细内容。

最后，我们还要报告我们在ImageNet Fall 2009版本上的错误率，这个数据集包含10184类、890万幅图片。在这个数据集上，我们依照约定使用一半图片作训练，一半图片作测试。由于没有现成的测试集，我们对数据集的分解方式必定与以前的研究不同，不过这并不会对结果有很大影响。我们的top-1和top-5错误率在这个数据集上分别是**67.4%**和**40.9%**，使用的是在最后池化层后附加6个额外卷积层的网络。这个数据集上公开发表的最好结果是78.1%和60.9%[19]。

6.1 定性评估

图3展示了这个网络的两个数据连接层学习到的卷积核。这个网络学习了一系列频率和方向选择的核，如图中的各种斑点。注意两个GPU表现出的特性，即3.5部分所述的限制连接的结果。GPU 1上的核对颜色很不敏感，而GPU 2上的核对颜色很敏感。在每一次运行里都会产生这种特性，与任何特定的随机权重都无关（为GPU按模重新编号）。

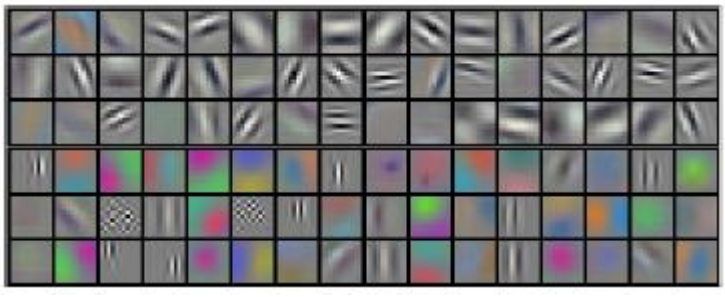

图3：224×224×3输入图像通过第一层卷积层学习到的96个大小为11×11×3的卷积核。上面的48个核是GPU 1学习到的，下面的48个核是GPU 2学习到的。参见6.1部分的详细内容。

在图4的左半部分，我们通过计算这个网络在8幅测试图片上的top-5预测，定性地估计它学习到了什么。注意到尽管物体偏离图片中央，就像左上角图片里的

小虫子，网络仍然可以识别它。大多数的top-5标签看起来很合理。例如，网络认为只有其它种类的猫是符合美洲豹的标签。在一些例子里（护栅，樱桃），图片的标签有歧义。

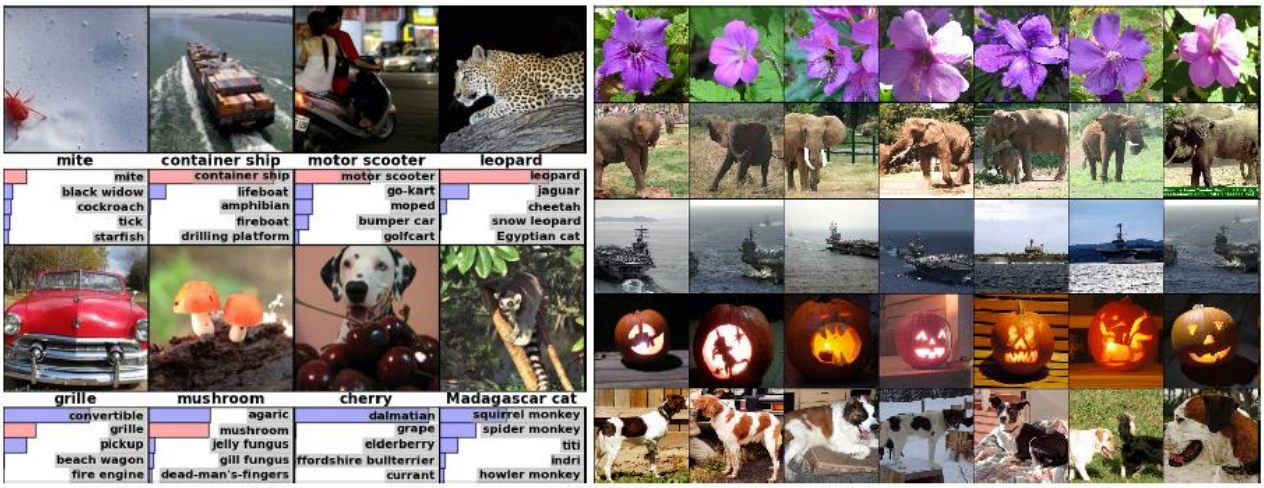

图4：(左)8幅ILSVRC-2010测试图片和5个分别由我们的模型给出的最合适标签。正确的标签写在每幅图片下方，每个正确标签对应的概率也用红色条标注出来（如果它在top 5里）。（**右**）第一列是5幅ILSVRC-2010测试图片，剩下的6列分别是6幅图片，它们在最后的隐层里的特征的向量与测试图片的特征的向量有最小欧拉距离。

另一种调查网络的视觉知识的方法是考虑一幅图片在最后一层4096维的隐层上的变化。如果两幅图片变化后的特征的向量之间欧拉距离较小，我们可以说神经网络的高层部分认为它们相似。图4展示了5幅测试集里的图片，和分别6幅来自训练集的图片，它们在这种方法下被认为最相似。注意到在像素层面，检索到的来自训练集的图片与第一列图片相差很大。例如，检索到的大象和狗有很多种姿势。在补充材料里我们会展示更多图片的测试结果。

在两个4096维的实值向量间计算欧拉距离效率非常低，如果训练一个自动编码器将这些向量压缩成二值编码，会很大程度上提升效率。这是一个比在原始像素上施加自动编码器更好的图片检索方法[14]，并不需要用到图片标签，从而用相似的边缘模式检索图片成为了趋势，无论他们在语义上是否相似。

7 讨论

我们的结果告诉我们一个巨大的深度卷积神经网络进行完全有监督的学习，能在一个有很高挑战的数据集上取得突破性的结果。值得注意的是，如果移除一

个卷积层，我们的网络表现就会变差。例如，移除中间的任意一层都会引起top-1错误率提高2%。所以实际上深度对我们得到的结果十分重要。

为简化我们的实验，我们并没有用任何无监督预训练尽管我们认为这会有帮助，特别当我们的计算能力足够扩大网络的规模但我们没有足够多的有标签数据。到目前为止，因为网络的增大和训练时间的增加，我们的结果已经变得更好，但相比人类视觉系统的下颞皮层通路，我们仍相差许多数量级。我们希望以后能对视频序列使用巨大和深层的卷积网络，以获得静态图像里漏掉或很不明显的有用信息。