本文,我们在ImageNet数据集上训练了深度卷积网络.训练得到的网络可以用于对120万张图片进行分类.在测试数据及上,我们的模型的top1和top5错误率分别是37.5%和17.0%,远超了之前的SOTA.我们的神经网络包含六千万个参数,65万个神经元,由5个卷积层组成.其中一些卷积层后续连接最大池化层.最后经过三个全连接层后输出最后的1000个预测分类.为了加速训练,我们使用了非饱和的神经元,并且用高效的GPU实现卷及操作.为了减少全连接层的过拟合.我们使用了最近被提出的并且证明了高效性的称为dropout的正则化方法.

在现在2022年来说,这个已经是上古时期的文章了.那个时候网络如果泛化性能上不去,就会说因为overfit了小的训练集,因为模型学习能力太强,所以学到了小数据集的bias.可是模型的泛化能力差并不一定是overfit了小数据集.完全有可能是因为模型的optimization不好,也就是进行了optimization,但又没有完全进行.意识到泛化性能差可能不是由于overfit而是由于优化存在问题这件事是在2016年的ResNet中发现的,因此在2012-2016年间,大家在做的事情就是寻找能够使得泛化性能更好的方法.不过他们并没有意识到它们的方法其实是提升了优化的效果,还是认为他们减轻了overfit小数据集.

Anyway,写的真的很好

ImageNet Classification with Deep Convolutional Neural Networks

2012年NIPS,Seminal的工作,主要贡献就是使用了relu,大大加速了训练.同时AlexNet唤醒了人们对DL领域研究的热情

Alex Krizhevsky
University of Toronto
kriz@cs.utoronto.ca

Ilya Sutskever
University of Toronto
ilya@cs.utoronto.ca

Geoffrey E. Hinton
University of Toronto
hinton@cs.utoronto.ca

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.

先用的进行物体识别的方法在根本上都使用了机器学习的方法.为了提高模型的性能,我们收集了大量的数据集,学习到了更强大的模型,并且使用了更好的技巧来避免过拟合.但是直到现在,数据集中标记的数据的数量依然只维持在上千上万张图像.

1 Introduction

这个体量的数据集对于一些简单的任务来说已经足够了,例如MNIST手写数字识别问题.然而,显示中的物体却由非常大的多样性,因此学习如何识别现实中的物体就需要更大的训练数据.的确,小数据集带来的问题在21中已经有所讨论,但是直到最近才有可能收集到大型数据集.例如LabelMe中有上百上千张完全分隔的图像,ImageNet中包含了150万张标注的高分辨率图像,一共超过22000分类别

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.

为了能够从上百万张图像中学习到上千个类别,我们需要有大学习能力的模型.然而现实中多种多样复杂的物体远远不是ImageNet中上千个类别所能涵盖的.因此,我们的模型就需要拥有很多的先验知识来弥补我们所欠缺的数据.而卷积神经网络(CNN)就是一类拥有强大学习能力的模型.CNN的哦容量能够通过其深度与广度进行控制.并且他们能够得到强大并且大多数情况下都是正确的假设.

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 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.

相比于标准的前向传播网络(每一层基本都是一样的)全连接网络有更少的连接很参数,因此更加容易进行训练,并且理论上的最佳性能只会有一点点的损失

尽管CNN具有很多非常诱人的性质,并且尽管(由于局部计算)带来的高效性,训练一个神经网络并且把它们放到大规模数据集上进行验证的代价依旧是非常高昂的.幸运的是,最近高算力的GPU以及能够在其上运行的高度优化的2D卷积的实现为训练大规模的CNN提供了动力.并且,最近提出的ImageNet数据集中包含了足够的标注来支持网络的训练而没有很大的过拟合

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.

本文主要的贡献如下:

1.我们在ImageNet的子集上(ILSVRC-2010和ILSVRC-2012中使用的数据)训练了至今为止最大的卷积神经网络,并且实现了至今为止最好的性能

2.我们实现了高效的基于GPU的卷积的实现,同样的,我们还在GPU上实现了网络中其他操作,并且开源了

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[1]. 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.

并且由于我们的模型具有强大的学习能力,因此过拟合小数据集就成了一个非常严重的问题,即便是ImageNet这种体量的数据集上

最后,我们的模型包含五个卷积层,我们通过实验发现,网络的深度尤其重要,少了任何一层(即便这一层只占1%左右的参数)模型的表现都会导致更差的性能

最后,模型的尺寸主要是收到现在的GPU的内存以及我们所可以接受的最大的训练时间的限制的

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.

我们的模型在两个3GB现存的GPU上花了5~6天训练.我们所有的实验都表明有只要有更快的GPU或者我们愿意等待更长的时间,我们的模型可以变得更加强大

2 The Dataset

ImageNet数据集合一共有超过150万张标注的高分辨率的图像.所有的图像大概分为22000类.所有的图像都是从网络上收集到的,并且由人工手动标注(使用了亚马逊的工具).从2010年开始,ILSVRC挑战赛作为Pascal视觉物体挑战赛中的一部分召开,并且每年举办一次.总的来说,一共有大概120万张训练图像,5万张验证图像和15万张测试数据

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.

由于ILSVRC2010的Label是公开的,因此我们的实验在这个版本的数据集上训练的.虽然ILSVRC2012的测试数据是不公开的,但是因为我们也参加了这个比赛,因此我们也报告了我们在ILSVRC2012上的性能

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

需要注意的一点就是ImageNet中包含不同分辨率的图像,但我们的模型只能够接受固定大小的图像作为输入.因此我们首先把所有输入的图像裁剪成256*256的大小.具体来说就是首先对短边进行了裁剪,然后选取了中间的256个像素作为输入的图像.最后,在输入前我们没有对数据集进行任何的预处理,我们只是单纯的减去了数据集所有图像的均值,然后就直接作为模型的输入了

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.

我们模型的细节在图2中进行了展示,他包含八个可以学习的层,无个卷积层和三个全连接层.3.1-3.4节分别描述了一些我们的网络结构中我们提出的新颖的特征.我们根据估计得到的重要性把重要的内容写到3.1章节

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

通常来说，在标准的网络中建模一个网络输出的函数是tanh函数.但是在训练阶段使用梯度下降的话，由于tanh的饱和性造成了梯度消失，因此使用relu的话就会极大地加快训练的速度.我们跟随20中给的名字，称这种引入非线性的单元的名字为relu.使用relu可以极大地加快训练的速度，如图1所示.在cifar10上训练的一个四层的卷积网络，使用relu收敛速度是使用tanh的六倍

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.

因此这就说明了，如果不使用relu的话，我们是不可能在六天内训练得到我们的模型

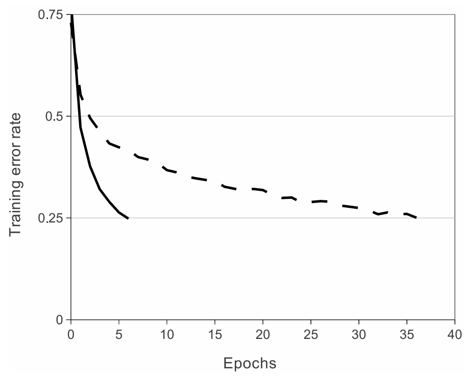


Figure 1: A four-layer convolutional neural network with ReLUs (**solid line**) reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with $\tanh$ neurons (**dashed line**). The learning rates for each network were chosen independently to make training as fast as possible.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

需要注意的是，学习率是针对每个网络分别调试得到的最快收敛的学习率.我们也尝试了不同的网络模型，但是使用relu的是最快收敛的

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.

其实我们并不是第一个考虑到替换非线性神经元的人.Jarrett等人在11中称他们那样的变换对于它们的模型在caltech101上比较好.但是在ImagNet上我们最大的需要是避免过拟合.因此他们的这种变幻的效果与我们使用relu加速训练的效果并不同

3.2 Training on Multiple GPUs

单独的一个GTX580GPU只有大概3GB的缓存，因此限制了可以训练的网络的尺寸.虽然120万张图片足够训练一个模型了，但是对于GPU来说120万张图片实在是太大了.因此我们把网络分散在了两个GPU上.现在GPU很好的支持了跨GPU的并行计算，两个GPU可以互相读写对方的缓存而不需要经过主机的缓存.因此并行计算方案使得我们可以把一半的卷积核或者神经元放到一个GPU上，不过我们使用了一个技巧，即在特定的层GPU间的数据会进行交互

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 Cireşan 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[2].

需要注意的是，一个GPU上训练的网络的卷积核的数量其实和两个GPU上训练的网络的卷积核数量是一样的.因为网络绝大多数的参数都是在第一个全连接层.因此为了让两个网络有差不多相同的参数量，我们没有减半最后的卷积层的数量，也没有减少全连接的参数量.所以相比于把网络减半，我们这样把网络放到一个GPU上其实是有bias的

[2]The one-GPU net actually has the same number of kernels as the two-GPU net in the final convolutional layer. This is because most of the net's parameters are in the first fully-connected layer, which takes the last convolutional layer as input. So to make the two nets have approximately the same number of parameters, we did not halve the size of the final convolutional layer (nor the fully-conneced layers which follow). Therefore this comparison is biased in favor of the one-GPU net, since it is bigger than "half the size" of the two-GPU net.

relu有用我们预期的所有性质,例如不需要对输入进行预处理以避免饱和性.此外,只要经过矩阵变换之后有大于零的值,那么这个神经元就会被更新.尽管如此,我们发现局部正则化方案能够帮助训练.将(x,y)这个位置卷积核i计算后经过正则化前的值记为a^i_{x,y},那么正则化之后的值b^_{x,y}为相邻的i个卷积层的激活层的值平方和的beta次方.需要注意的是这个指数beta是在训练前指定的超参数.LRN的灵感来源于真实的神经元中的线型,就是大的激活的函数的数值会抑制局部其他的值

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

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

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].

CNN中的池化层可以帮助网络来总结相同卷积核的activation map中相邻的几个激活函数的值.传统上相邻的池化窗口之间是没有重叠的.更加准确的说,假设寄一个池化层是z*z大小的,每次移动s个像素.如果s=z的话那么就是传统的池化,但是如果s<z的话就得到了重叠的池化.我们的网络中用的就是这样的重叠的池化.我们让s=2,z=3

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 $z \times z$ 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.

我们的网络包含八个含有可训练参数的层,前五层是卷积层,后三个是全连接层.最后一个全连接层的输出是100维的向量.这个向量而后被softmax吃进去,计算得到在1000个类别标签上的分布.我们的网络会最大化多个逻辑回归的值.

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.

三四五层的卷积层都只连接到了相同GPU上的前一层的输出,而第三层卷积层则连接到了所有第二层的输出

LRN在第一和第二层后面接着,而最大化池化层在所有层后都接着

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 $224 \times 224 \times 3$ input image with 96 kernels of size $11 \times 11 \times 3$ with a stride of 4 pixels (this is the distance between the receptive field centers of neighboring

[3]We cannot describe this network in detail due to space constraints, but it is specified precisely by the code and parameter files provided here: http://code.google.com/p/cuda-convnet/.

第一个卷积层对224*224*3的输入图像进行卷积,一共有96个11*11*3的卷积核以4为步长进行滑动
PS:他这里是把96个activation map放到两个GPU上去了,一个有48层

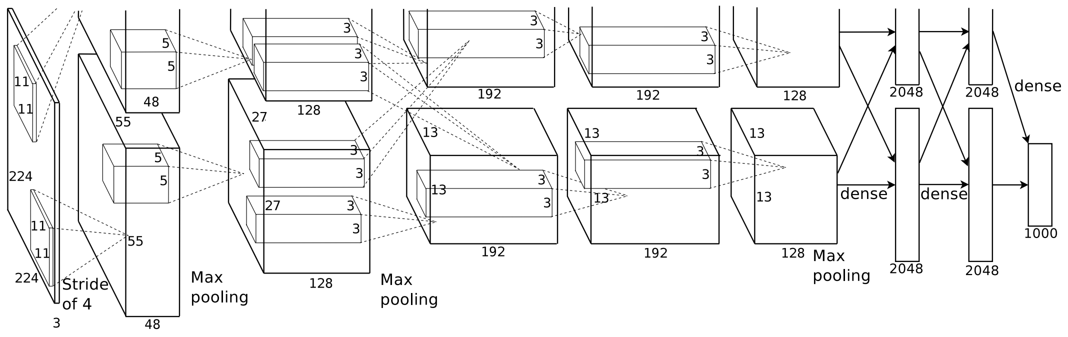


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input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

第二个卷积层输入第一个卷积层池化和LRN之后activation map,一共有256个5*5*48大小的卷积核,注意,第三四五层中间没有池化或者LRN

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 $5 \times 5 \times 48$. 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 $3 \times 3 \times 256$ connected to the (normalized, pooled) outputs of the second convolutional layer. The fourth convolutional layer has 384 kernels of size $3 \times 3 \times 192$, and the fifth convolutional layer has 256 kernels of size $3 \times 3 \times 192$. The fully-connected layers have 4096 neurons each.

4 Reducing Overfitting

因为模型中包含的参数比较多,所以即便是ImageNet这样的数据集还是会存在overfit的现象.(其实是optimization失败)

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.

所以下面介绍了我们主要是用两种对抗overfit的方法

4.1 Data Augmentation

最简单并且也是最常用的对抗过拟合的方法就是人工增大数据集.我们使用的两种数据增强的方法消耗的计算量并不大,因此并不需要提前把数据增强之后的图像保存下来

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.

第一种数字增强的方式包括对图像进行平移和水平反转.我们通过从原始的256*256的图像中抽取224*224大小的patch然后对他进行水平翻转.这样做让我们的训练数据增加了2048倍,并且得到的新的训练数据都是高度独立的

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.

注意,在测试阶段,一张图像会选取四个角落和中心的五张patch,然后水平翻转之后得到10张图像.最后再经过softmax分类后,最终的类是经过average之后的argmax

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,

[4]This is the reason why the input images in Figure 2 are $224 \times 224 \times 3$-dimensional.

第二种数据增强的方式就是对训练图像的RGB三个通道调整强度,具体来说就是我们把找到的主成分的值和对应的特征值与标准高斯分布中得到的三个值相乘后载相乘,然后加到原始的图像上去. 不过现在来说,PCA增强已经过时了

具体计算步骤为:1.将图片按照RGB三通道进行normalization处理,均值为0,方差为1.值得一提的是,按照RGB三通道进行处理,因为我们进行的是色彩增强,在RGB三通道的图片中,决定图像色彩的是RGB之间的相对关系,我们不能改变三通道内部的像素值分布.2.将图片按照channel展平成大小为(-1,3)的array 2.求上述array的协方差矩阵 3.对协方差矩阵进行特征分解 4.特征向量组成的矩阵是3x3的,特征值乘以抖动系数组成的array是3x1的,所以点乘得到的array正好是3x1大小的.三个值分别加到原图像的R,G,B三通道上,就是最后得到的增强的图像.

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%.

进行这种增强的意义在于物体的识别对于其光照的强度来说是不变的.因此通过这样的增强模型就可能学到图像的本质,而非学到明暗变化

4.2 Dropout

结合多个不同的模型的预测对于减少测试错误来说是非常常用的方法.但是对于大型神经网络来说,训练它就需要好几天了更别说结合几个不同模型.

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 back-propagation. 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.

不过,最近一种称为dropout的方法能够实现训练多个网络的同时仅仅增加一倍的训练量.dropout的含义就是会随机选择一半的神经元,让他们的值变成0.这样的话这些神经元不会参与到运算,并且也不会进行梯度的反向传播

由于每次被drop out的神经元都是不同的因此每一次输入被送入网络,网络都会从几个不同的网络(因为参数不同)采样得到.此外,这样做也会打断神经元的联合适宜性,使得每一个神经元都会学习到独立的,更加强大的特征

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.

注意,在测试阶段每一层的输出都会乘以0.5,这样做是对所有可能结构的输出的估计

SGD训练的,每一个batch128个样本,动量0.9,衰减0.0005

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

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

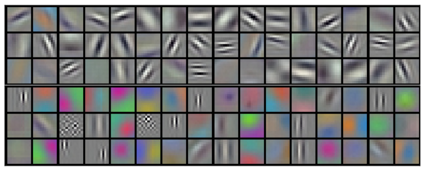


Figure 3: 96 convolutional kernels of size $11\times11\times3$ learned by the first convolutional layer on the $224\times224\times3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

模型在第一层学习到的9个卷积核,上面48个是在第一个gpu上学到的,下面48个是在第二个gpu上学到的

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.

网络参数的初始化是从一个0为均值,0.01为方差的标准正态分布中采样得到的

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.

还是比较早期的工作,没有用kaiming初始化

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

所有层的学习率都是相同的,并且在训练的过程中会手动调整.如果模型在当前学习阶段停止了学习,那么就会除以10,初始学习率设置为0.01.在停止训练前会减少三次.我们用120万张图像训练了大概90个epoch 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.

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].

因为ILSVRC2012的测试集不公开,因此我们没有办法报告不同层数的模型的表现.下面的表2是我们在validation集合上测试的结果.前面说的结构的网络实现了top5错误率18.2%,

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 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].

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.

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.

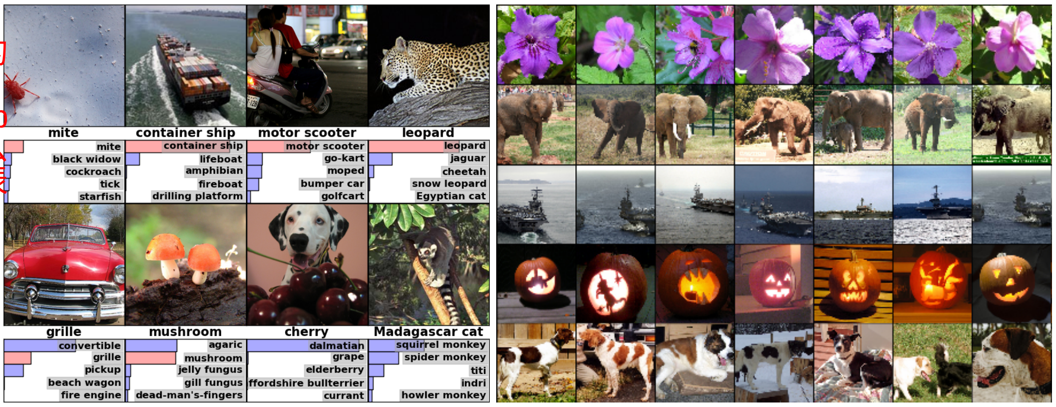
6.1 Qualitative Evaluations

图三展示了模型第一层学习到的所有卷积核.模型学习到了很多种不同频率,方向选择性的卷积核.需要注意的是,第一个GPU上的卷积核学习到了颜色无关的方向信息,而第二个GPU上的卷积核学习到了和颜色相关的特征.这种区别是通用的,即多次训练每次都能够观察到这个现象

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).

[5]The error rates without averaging predictions over ten patches as described in Section 4.1 are 39.0% and 18.3%.

左边的图像是2010的测试图像,gt被写在了下面.如果模型top5预测中存在gt,那么就会用红色的柱子表示



右侧第一列图像是测试中的ground truth.而剩下六列都是训练数据中和测试数据在最后一层得到的特征向量欧式距离最接近的点

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.

通过图四左侧的图像,能够观察到:1.即便是偏离中心的图像依然能够被我们的网络识别出来 2.绝大多数top5的预测label都是合理的.例如猎豹的前五个预测都是猫科动物.而一些推测失败的图像则是由于图像本身具有很大的模糊性

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.

另外一种从视觉上理解网络的方法就是观察线性层的向量.如果两张图像输出的activation向量比较接近,我们就可以认为模型认为这两张图像是相似的

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 dogs and elephants appear in a variety of poses. We present the results for many more test images in the supplementary material.

需要注意的是,模型从训练数据中检索到的图像和测试图像之间的L2距离还是很大的.例如大象和狗都有不同的姿势

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.

需要注意的是,对4096维度的向量计算欧几里得距离其实并不是非常好的,因为高维度的向量都是比较接近的,因此如果使用Auto-encoder来计算特征向量的话获取的更好的结果

我们的结果表明,使用一个大型的深度卷积网络是可以在极具挑战性的数据及上获得破纪录的性能的,并且仅仅使用监督训练即可.此外,还需要注意的就是我们的模型的性能随着网络层数的变浅,性能是会有极度的下降.例如随便移除一个卷积层最后的top1准确率都会有2%左右的损失.因此深度是我们实现如此好的性能的重要因素之一.

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.

为了简化训练和讨论,我们并没有使用任何无监督的预训练,尽管我们认为这样做绝对会提升我们的性能.尤其是我们如果拥有更大的算力和数据集.甚至只要我们愿意等的更久一些,模型的性能都会继续有所提升

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.

最后,我们的模型通过使用更大,更深的网络实现了很强的性能,但是我们如果继续使用人类视觉系统的时间信息流,那么性能可能会有更强的提升.因此,我们接下来希望 把更大,更深的卷积网络运用到视频领域,因为对于视频来说建模时间就够将回提供更多的静态图像中无法提供的时间信息

References

[1] R.M. Bell and Y. Koren. Lessons from the netflix prize challenge. *ACM SIGKDD Explorations Newsletter*, 9(2):75–79, 2007.

[2] A. Berg, J. Deng, and L. Fei-Fei. Large scale visual recognition challenge 2010. www.image-net.org/challenges. 2010.

[3] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[4] D. Cireşan, U. Meier, and J. Schmidhuber. Multi-column deep neural networks for image classification. *Arxiv preprint arXiv:1202.2745*, 2012.

[5] D.C. Cireşan, U. Meier, J. Masci, L.M. Gambardella, and J. Schmidhuber. High-performance neural networks for visual object classification. *Arxiv preprint arXiv:1102.0183*, 2011.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[7] J. Deng, A. Berg, S. Satheesh, H. Su, A. Khosla, and L. Fei-Fei. *ILSVRC-2012*, 2012. URL `http://www.image-net.org/challenges/LSVRC/2012/`.

[8] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.

[9] G. Griffin, A. Holub, and P. Perona. Caltech-256 object category dataset. Technical Report 7694, California Institute of Technology, 2007. URL `http://authors.library.caltech.edu/7694`.

[10] G.E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R.R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[11] K. Jarrett, K. Kavukcuoglu, M. A. Ranzato, and Y. LeCun. What is the best multi-stage architecture for object recognition? In *International Conference on Computer Vision*, pages 2146–2153. IEEE, 2009.

[12] A. Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, Department of Computer Science, University of Toronto, 2009.

[13] A. Krizhevsky. Convolutional deep belief networks on cifar-10. *Unpublished manuscript*, 2010.

[14] A. Krizhevsky and G.E. Hinton. Using very deep autoencoders for content-based image retrieval. In *ESANN*, 2011.

[15] Y. Le Cun, B. Boser, J.S. Denker, D. Henderson, R.E. Howard, W. Hubbard, L.D. Jackel, et al. Handwritten digit recognition with a back-propagation network. In *Advances in neural information processing systems*, 1990.

[16] Y. LeCun, F.J. Huang, and L. Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–97. IEEE, 2004.

[17] Y. LeCun, K. Kavukcuoglu, and C. Farabet. Convolutional networks and applications in vision. In *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pages 253–256. IEEE, 2010.

[18] H. Lee, R. Grosse, R. Ranganath, and A.Y. Ng. Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 609–616. ACM, 2009.

[19] T. Mensink, J. Verbeek, F. Perronnin, and G. Csurka. Metric Learning for Large Scale Image Classification: Generalizing to New Classes at Near-Zero Cost. In *ECCV - European Conference on Computer Vision*, Florence, Italy, October 2012.

[20] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proc. 27th International Conference on Machine Learning*, 2010.

[21] N. Pinto, D.D. Cox, and J.J. DiCarlo. Why is real-world visual object recognition hard? *PLoS computational biology*, 4(1):e27, 2008.

[22] N. Pinto, D. Doukhan, J.J. DiCarlo, and D.D. Cox. A high-throughput screening approach to discovering good forms of biologically inspired visual representation. *PLoS computational biology*, 5(11):e1000579, 2009.

[23] B.C. Russell, A. Torralba, K.P. Murphy, and W.T. Freeman. Labelme: a database and web-based tool for image annotation. *International journal of computer vision*, 77(1):157–173, 2008.

[24] J. Sánchez and F. Perronnin. High-dimensional signature compression for large-scale image classification. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 1665–1672. IEEE, 2011.

[25] P.Y. Simard, D. Steinkraus, and J.C. Platt. Best practices for convolutional neural networks applied to visual document analysis. In *Proceedings of the Seventh International Conference on Document Analysis and Recognition*, volume 2, pages 958–962, 2003.

[26] S.C. Turaga, J.F. Murray, V. Jain, F. Roth, M. Helmstaedter, K. Briggman, W. Denk, and H.S. Seung. Convolutional networks can learn to generate affinity graphs for image segmentation. *Neural Computation*, 22(2):511–538, 2010.