论文写作与学术规范

第6讲 三遍读论文法

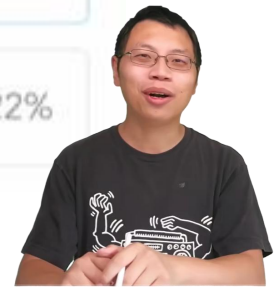
信息学院（智能应用研究院）
欧 新 宇

Thanks to LiMu: Bilibili 跟李沐学AI

Part 01

三遍读论文法

Bilibili 跟李沐学AI



如何读论文

三遍阅读法

	第一遍	第二遍	第三遍
Title	√		
Abstract	√		
Introduction		√	1. 知道每一句话是什么意思，在做什么； 2. 思考如果是你自己写，会如何组织文章； 3. 思考作者描述的实验是否完备，你能不能比作者做得更好； 4. 将自己脑补成作者，能复现全流程的工作。
Related Works		√	
Methods	√ Figures and Tables	√ Figures and Tables	
Experiments	√ Figures and Tables	√ Figures and Tables	
Conclusions	√		
	1. 粗略了解论文在做什么； 2. 判断是否符合你的研究方向。	1. 快速通读全文 2. 搞明白重要图和表都在干什么； 3. 作者方法和对比方法的区别和改进； 4. 确定是否读第三遍； 5. 确定可以研读的相关文献。	

Part
02

第一遍 泛读

如何读论文

三遍阅读法演示

ImageNet Classification with Deep Convolutional Neural Networks

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

第一遍：标题、摘要和结论

第二遍：快速通读全文

第三遍：深入阅读全文

ImageNet Classification with Deep Convolutional Neural Networks.

如何读论文：第一遍

标题

- ImageNet Classification with Deep Convolutional Neural Networks
  - ImageNet Classification (当时最大的数据集)
  - Deep Convolutional Neural Networks
    - ✓ Neural Networks （SVM？决策树？逻辑回归？）
    - ✓ Convolutional
    - ✓ Deep
  - 领域不了解？是否还继续读下去呢？
  - 作者：Alex Krizhevsky？ Ilya Sutskever？ Geoffrey Hinton √

如何读论文：第一遍

第一句，我做了一个啥；第二句，它效果很好。这种写法很少见。

摘要的数值要引起关注，也许是创新的突破，虽然你不一定知道 neuron 是什么，但 parameter 是知道。

摘要

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.

讲了一下它模型的结构是什么

接下来介绍了一些方法，解决了一些问题。创新点?

再次闪现了结果：别人是 26.2%，我是 15.3%。严格说这个摘要写得并不是特别好，更像是技术报告，但是结果确实很赞！所以，只要有一点相关性，可能你就会继续读下去。

如何读论文：第一遍

本文没有结论，只有讨论，这是比较少见的（不推荐）

结论

1. 做了一个很大的网络，证明深度很重要。结论没问题，但论据在今天来看是不足的。移除一层性能下降2%，也可能是调参的问题，并不一定是深度问题。

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.

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.

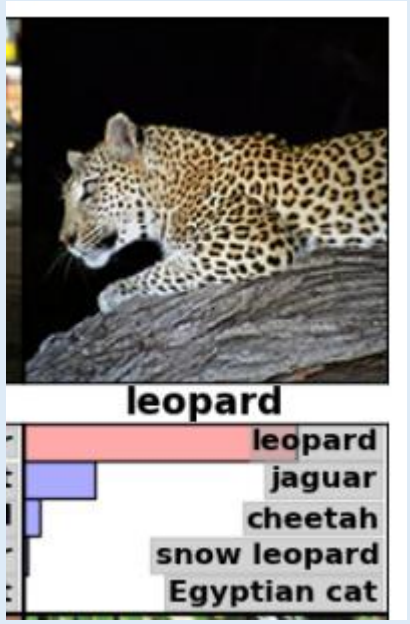
2. 掀起了监督学习范式的风靡，直到2018，19年Bert出现之后，关注点才再次回到非监督学习。

提出了Video的重要性

如何读论文：第一遍

重要的图、表和公式（1/3）

Left: 模型在分类任务（top-5）中效果很好。



Right: 在检索中效果也不错。能从测试集中找出大量与给定样本相似或相同种类的目标。

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.

最后一个隐层的特征向量对于表征样本具有非常好的性能。然而，当时并没有很详细地讨论这个问题，在几个月后的其他研究中，我们发现这个结论确实很有道理，大家都去follow了这个规律。

如何读论文：第一遍

重要的图、表和公式（2/3）

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.

和别人结果的对比，主打就是效果好，也是本文的卖点。在摘要中也提到了。

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.

提出了两种改进方案的结果，一个是pre-trained，另一个是多模型融合。

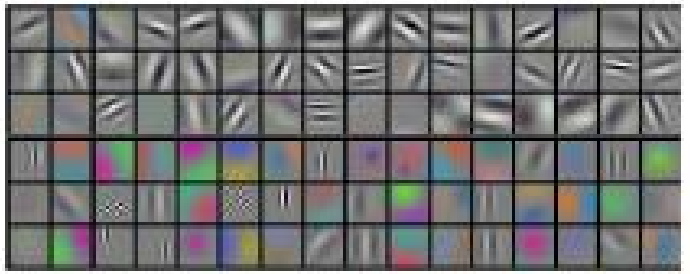
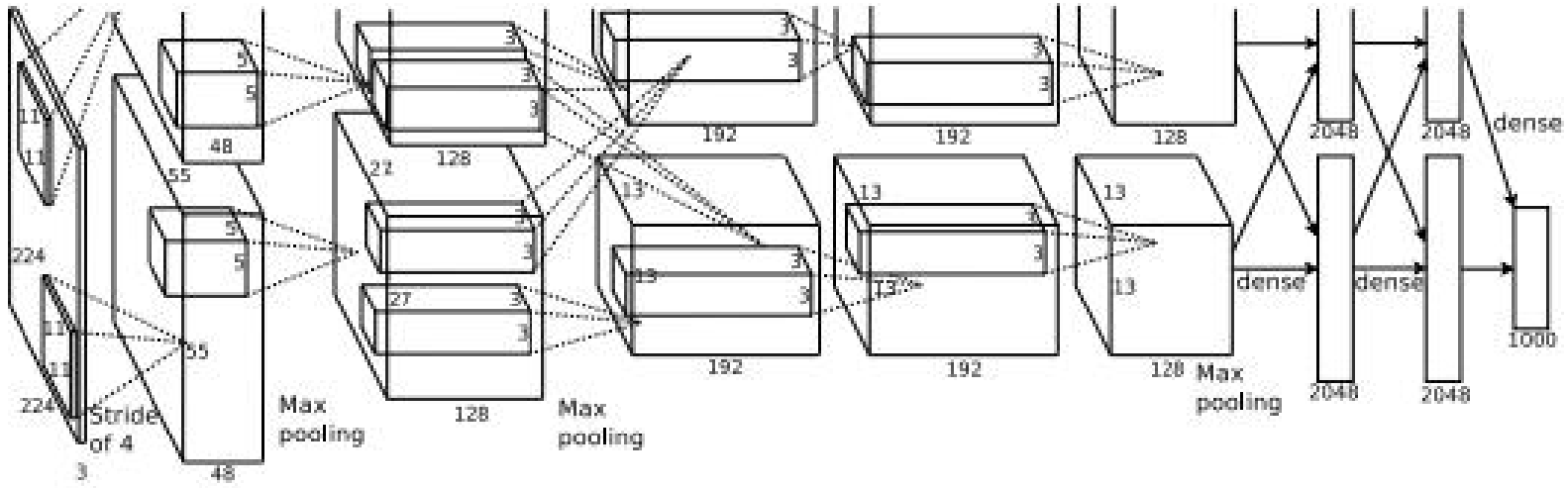


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

计算机视觉的可视化图。当然，如果对计算机视觉领域不了解，对卷积神经网络不了解，你也许可能看不懂是什么。那么，暂时跳过就可以了。

如何读论文：第一遍

重要的图、表和公式（3/3）



AlexNet 的 architecture。今天我们知道，凡是涉及architecture的图一般都很重要，但对于十多年前的你来说，或许你真的是完全看不懂。

- 首先，作为初学者，在第一遍中可能真的看不太懂，第一遍能看懂的通常是实验结果，或者是一些你所熟悉的领域。
- 其次，这篇文章是开创性的工作；
- 最后，它真的有些复杂，很多隐含的概念，在图中都没有写明。

如何读论文：第一遍

现在你知道了什么？

- 这篇文章结果特别的好，它是用神经网络来做的
- 但是具体为什么好？具体怎么做的？由于你的背景知识不够，你可能并不了解。

□ So what?

1. 暂时放弃这些为什么
2. 决定是否要继续往下读?
   - ✓ 很多人可能会选择放弃，因为神经网络在当时确实很少人做，可能和你的研究无关你就放弃了。
   - ✓ 但是如果你是做图片分类的，即便后面的技术不是很了解，但是因为它确实性能不错，赢得了比赛，所以估计明年会有很多人去follow，所以你还是应该去读一下。
   - ✓ 所以，此时你考虑的应该是这个领域是你关心的，而不是这个技术是你关心的。

- 一般来说，第一遍需要花费你20～30分钟的时间

Part
03

第二遍 精读

如何读论文：第二遍

基本原则

- 全文通读一遍
- 对于特别不懂的，可以暂时留下来，留给第三遍
- 明白大多数细节是干什么
- 更重要的是明白作者是怎么想的，作者是怎么描述问题的。通过读论文去了解作者对整个问题的看法，以及他的认知角度和思考方式。
- PS：学会形成自己的观点，特别是看问题和理解问题的关键，以及分析问题和处理问题的方法

- 事后读论文，有时候我们可能会发现论文的很多观点和方法是错误的，也可能有一些做法是多余的，非常的engineering。因为对于一篇经典的论文，大家都会去研究它的每一个细节。因此，对于一篇有历史的论文，我们需要带着一些批判的思维去理解和认识它。
- 然而在当时，可能这些观点都是很具有启发性的，也会是开创一个研究领域的先驱，这是我们读一篇新的论文需要去挖掘的。

如何读论文：第二遍

Introduction（1/5）

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.

第一段：

1. 讲一个故事：我们要做什么研究，我们做了什么，它是哪个方向的，这个方向有什么，为什么很重要。

2. 接下来就要说一下ImageNet，这里不是介绍数据集的细节。而是说了一下这个数据集是在什么样的背景下出现的，目的是让别人觉得你在这个数据集上做的实验结果是有价值，可参考的。故事的内容没有错，但是在写作逻辑上看，还是有点容易让读者对动机产生困惑。

如何读论文：第二遍

Introduction（2/5）

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.

第二段：

1. 强化动机。为了……，所以我们需要……

2. 直奔主题，讲了应该用神经网络来做，特别是CNN。

- **注意：这种写法并不是很好的写法。你不应该基于自己对神经网络非常熟悉的前提，就直奔主题。因为当时的主流并不是神经网络，很多人并不是很了解这个东西。**
- **比较好的做法是简单提一下别人是怎么做，主流是哪些方法（与Related Work呼应），然后再转到自己的方法。**

如何读论文：第二遍

Introduction（3/5）

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 overfitti

第三段：提炼一下本文使用GPU来实现，以及为什么使用GPU来实现。

如何读论文：第二遍

Introduction（4/5）

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.

总结贡献：

1. 我们做了一个超大型CNN在ILSVRC上获得了很好的性能； ○
2. 我们写了一个优化很好的GPU实现的CNN框架；
3. 我们使用了一些新的、与众不同的方法来改进性能并降低训练时间。 √
4. 我们使用一些有效的技术来降低过拟合。 √
5. 我们设计了一个新的网络，而且发现深度挺重要的。 √

哪一个更重要?

如何读论文：第二遍

Introduction（5/5）

In the end, the network's size is limited mainly by the amount of memory available o and by the amount of training time that we are willing to tolerate. Our network tak and six days to train on two GTX 580 3GB GPUs. All of our experiments suggest can be improved simply by waiting for faster GPUs and bigger datasets to become a

多余的

Us ve lts

接下来，作者谈了一下具体的实现和问题：

1. 训练这么一个大型的网络时间很长，而且很耗费GPU的资源

2. So，使用了5~6天来训练，并且用了2个GTX580的GPU（cuda-convnet, 多卡并行）

3. 用更好的GPU和更大的数据集可以改进性能。

然而，对于一个研究论文来说，以上内容意义不大。别人不会因为你的工作量大，工程复杂而认为你的作品具有价值，特别是对于技术上的工作来说，工程上的细节很难源远流长。事实上，后面基本没有任何工作是基于cuda-convnet来完成，包括 Alex 本人都没有继续使用它。

如何读论文：第二遍

The Dataset

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

第2章，就直接介绍数据集，这是非常少见的，但是作为大牛的作品，因为他卖的就是ImageNet的性能，所以也可以接受。（但不推荐这么去写）

此处，隐含了两个非常重要的工作，但作者并没有将其作为创新点来描述，仅仅只是说了他是怎么做（这也是现在说这篇论文写得不好的点之一）

1. 数据预处理
2. 从 raw RGB 开始训练（End-to-end）—— 深度学习方法论的核心创新

如何读论文：第二遍

The Architecture（1/6）（主要贡献之一：CNN网络架构）

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.

先阐述了一下最终版本的完整架构是 Figure2（就是第一遍中很复杂的那个）

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.

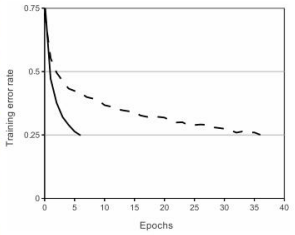


Figure 1: A four-layer convolutional neural network with ReLUs **(solid line)** reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons **(dashed line)**. The learning rates for each network were chosen independently to make training as fast as possible.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

3.1 ReLU

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 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 [illegible] and [illegible]%, [illegible] we a net with half as many kernels in each convoluti[illegible] trained [illegible] GPU and the two-GPU net takes slightly less time to train than the one-GPU net[illegible].

3.2 多GPU

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

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

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 LRN

3.4 Overlapping Pooling

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.

3.5 Overall Architecture

3.5 整体架构

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.

如何读论文：第二遍

The Architecture（2/6）（主要贡献之一：CNN网络架构）

粗略的认识是：Standard way是….，我们的方法是…..。这里提到了有关**饱和非线性**（saturating）和**非饱和非线性**（non-saturating）的概念。一般来说，你是不懂的，有兴趣可以去看看Hinton的[20]，不过第二遍不看也没事（打个问号）。**但你可以看到的是饱和的慢，而ReLU是非饱和的。**

这一段，整体就是说，我们提出了一种激活函数叫 ReLU，它就是max(0, x)。名字取得很好！

为了证明ReLU好，作者做了一个比较简单的对比实验。就是在CIFAR10上和Tanh比了一下。

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.

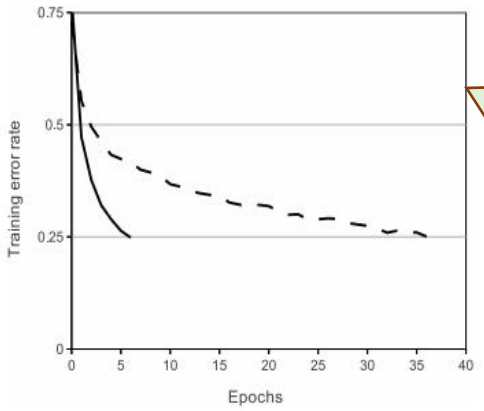


Figure 1: A four-layer convolutional neural network with ReLUs **(solid line)** reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons **(dashed line)**. The learning rates for each network were chosen independently to make training as fast as possible.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

如果文字看不懂，图应该是可以看得懂的。
1. 在CIFAR-10上对ReLU和Tanh进行了对比。
2. 两者可以达到基本相当精度（error rate = 0.25）
3. ReLU比Tanh快6倍。

本小节的结论是：
作者引入了一种它们自己提出来的激活函数ReLU，以前没怎么用过，但在本文的CNN里，效果不错，至少速度很快。对比的是当时常用的Sigmoid和Tanh。今天我们依然用ReLU，但主要原因是它简单。

如何读论文：第二遍

The Architecture（3/6）（主要贡献之一：CNN网络架构）

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 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].

➡ 基本动机，显卡显存太小，不足以放下整个模型。动机很重要！

- 实际上，这部分内容工程性非常强，作为机器学习的论文，写不写影响都不是特别大。但为了应对不必要的质疑（比如别人怀疑你的模型太大/你的算法太复杂，而你的硬件、你的资源不足以实现你的想法），你可以适当提一下。
- 但目前也提倡开源，开源和项目网站，通常也可以解决这些质疑。
- 所以，其实关于工程实现的部分，你可以不去过多阅读，除非你想学一下怎么实现。

如何读论文：第二遍

The Architecture（4/6）（主要贡献之一：CNN网络架构）

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

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

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].

在第二遍中，理论上我们应该搞清楚为什么要做这个工作。

对于 Local Respopnse Normalization 来说，作者介绍的是：ReLU不需要输入正则来避免它饱和；但做点正则还是挺好的，至少可以提高泛化能力。

但实际上来说，以上的解释，并没有讲清楚为什么要使用LRN，只是说我们要怎么做。其实，这也是不太好的写作方法。动机不明！

该公式挺复杂，但如果当年按照作者的思路，你基本也知道它的目的是提高泛化能力。只是作者没有提为什么可以提高泛化。对于第二遍阅读，知道这些也就够了。

如何读论文：第二遍

The Architecture（5/6）（主要贡献之一：CNN网络架构）

Pooling 我们现在很熟悉了，它很简单，也很有用。但当时CNN刚刚出来，大家其实并不了解什么是Pooling，更别说Overlapping Pooling。此时，你可以先粗略读一下前几句，了解这个技术到底是干啥。如果还不懂，可以暂时打个问号。

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.

实现细节，在第二遍中可以暂时忽略。

根据写作的基本结构“总分”或“总分总”，你可以重点观察一下开头几句和最后几句话，找找有没有重点。这一段简单说了一下，overlapping pooling可以让overfit更困难。如果你是做机器学习的，那么你应该了解的是，作者的意思其实是说这个技术可以改善过拟合问题。其实，也就是动机所在。

如何读论文：第二遍

The Architecture（6/6）（主要贡献之一：CNN网络架构）

在第二遍中，如果你英语还行，你可以顺着读一下整体框架的介绍。一般来说，Architecture都是机器学习论文的核心，或者说是起点，所以通常阅读难度不会很高。通读一下，有助于理解作者的核心设计思想。**注意：阅读的重点是对照Figure2.**

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 $224 \times 224 \times 3$ 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/.

但反过来说，本文其实写的不是很好，并没有介绍为什么这么设计，而是直接说了怎么做。（更像一个技术报告）所以，对于不熟悉CNN的人来说，阅读起来其实并不容易。特别是图Figure2也很工程。

如何读论文：第二遍

The Architecture（6/6）（主要贡献之一：CNN网络架构）

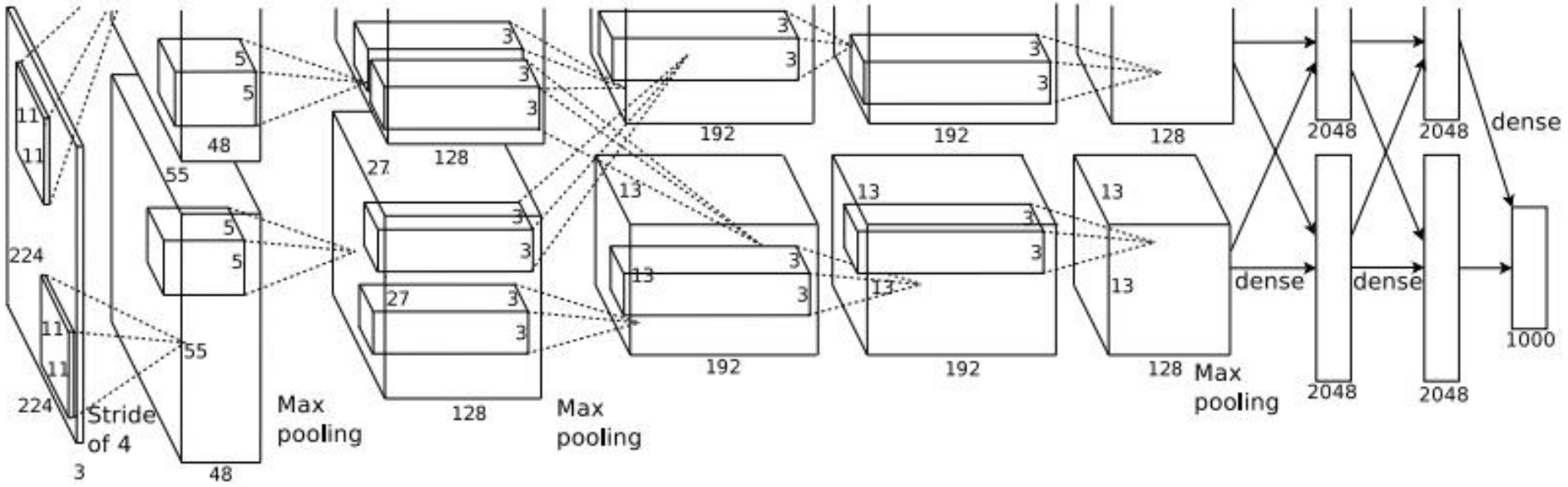


Figure 2: An illustration of the architecture of our CNN, explicitly showing the delineation of responsibilities between the two GPUs. One GPU runs the layer-parts at the top of the figure while the other runs the layer-parts at the bottom. The GPUs communicate only at certain layers. The network's input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

对于不熟悉的图、代码和方法，建议按照IPO的原则进行解读。Input, Processing, Output

如何读论文：第二遍

Reducing Overfitting（主要贡献之二：缓解过拟合）

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

4.2 Dropout

本章是论文的一个比较重要的创新点，当然，它所解决的问题也是卷积神经网络在刚出现的时候很容易出现的问题。

- 一个是数据量不足导致的过拟合，所以使用数据增广；
- 另一个是模型设计中全连接层带来的问题，所以提出了Dropout。

- 严格说，这两个技术都算不上是原始创新，但是作为开创性工作中新出现的问题，引入其他方法来解决问题，还是可以接受的。
- 这两个小节写作总体不错，包括了问题出现的原因，解决问题的基本方法，具体的实现和细节都介绍得很清楚。

如何读论文：第二遍

Details of learning

5 Details of learning

We trained our models using stochastic gradient descent with a batch size of 128 examples, momentum of 0.9, and weight decay of 0.0005. We found that this small amount of weight decay was important for the model to learn. In other words, weight decay here is not merely a regularizer: it reduces the model's training error. The update rule for weight w was

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

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

where i is the iteration index, v is the momentum variable, ϵ is the learning rate, and $\left\langle \frac{\partial L}{\partial w}\Big|_{w_i} \right\rangle_{D_i}$ is the average over the ith batch D_i of the derivative of the objective with respect to w, evaluated at w_i.

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

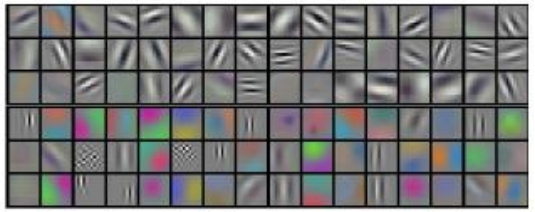


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

第5章，学习的细节实际对应的是Experiments的实验细节，所以可见的是章节的设置并不是特别合理。但这部分写作内容是比较充实，你可以关注的几点是：

1. 作为实验细节，实际上你已经不需要写太多原理了，关键是挖掘出实验的步骤是什么，实验的关键参数是什么，实验的实现方法是什么等等。根据这部分的描述，你应该能重现整个实验的所有步骤。

2. 关注一些重点、通用、可浮现的设置。

- ✓ **SDG训练方法**
- ✓ **权重衰减 weight decay = L2 Normalization**
- ✓ **更新方法**
- ✓ **动量法**
- ✓ **权重：zero-mean，方差为1的（Gaussian）随机初始化；偏置：0 或 1**
- ✓ **步进学习率，初始学习率为0.01**
- ✓ **当验证损失不变时，学习率降低10倍，共三次**
- ✓ **训练90个周期，花了5-6天**

如何读论文：第二遍

Results

6 Results

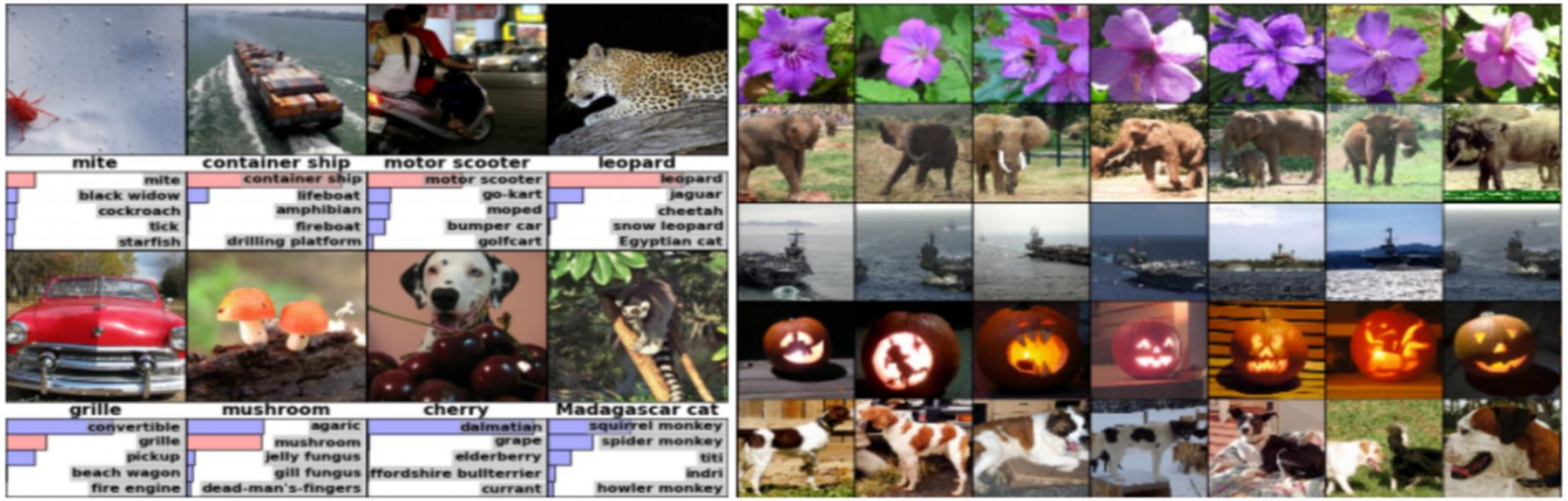


Figure 4: **(Left)** Eight ILSVRC-2010 test images and the five labels considered most probable by our model. The correct label is written under each image, and the probability assigned to the correct label is also shown with a red bar (if it happens to be in the top 5). **(Right)** Five ILSVRC-2010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

In the left panel of Figure 4 we qualitatively assess what the network has learned by computing its top-5 predictions on eight test images. Notice that even off-center objects, such as the mite in the top-left, can be recognized by the net. Most of the top-5 labels appear reasonable. For example, only other types of cat are considered plausible labels for the leopard. In some cases (grille, cherry) there is genuine ambiguity about the intended focus of the photograph.

实验结果对于一篇论文来说是比较重要的，但是你也可以说它并不是那么重要。

重要之处在于它是不是印证了一开始的设计动机；它是不是实现了你的设计思想；它是不是有效；或者它是不是有启发性。

所以，实验部分，你阅读的时候应该关注的是什么？

1.评价指标是什么？为什么用这些指标去比，目的是什么

2. 消融实验，在哪几个方面进行展开的，结果如何

3. SOTA对比试验和哪些论文做了对比，结果如何

4. 更重要的是：针对以上的对比，作者是如何解释的。关键不是数值有差异，而是数值为什么有差异。

如何读论文：第二遍

现在你知道了什么?

- **一般来说，第二遍阅读需要花费你1-2个小时。**
- 这篇文章主要工作的大部分原理和动机
- 文章涉及哪些创新技术和创新的思路，它们解决了什么问题?
- 这篇文章做了哪些实验，为什么要做这些实验，分别是怎么做的，结果怎么样。
- 这一遍读完之后，你应该能够获得比大多数博客更重要的东西：思想、动机和Why？这比技术本身更为重要
- 这篇文章是否值得细扣下去，是否值得你follow，是否值得你搞懂每一个细节。
- 和这篇文章相关的文献还有哪些？它们是不是值得继续阅读。此时，你应该整理出一个随后继续阅读的文章列表。
- 对于大多数的论文来说，读两遍也就差不多了；除非你需要重点follow，或者需要借鉴。

Part
04

第三遍 理解与升华

如何读论文：第三遍

第三遍怎么读?

- 深究每一个重要的技术细节，特别是你可能会用到的技术，包括：
  - 卷积神经网络是什么?
  - 过拟合是怎么回事?
  - ReLU是什么
  - 随机梯度下降是怎么回事
  - 不懂的内容，如果作者写了你可以认真看一下，如果没写，那它引用的文章是最好的选择
  - 细节搞懂之后，建议再从头到尾、快速、重新复盘阅读一下全文。
- 训练是如何完成？如果要进行实验的复现，此时你可能需要去找一下源代码，记录一下论文中作者写好的关键超参数。
- 思考一下作者的实验是否完备，你能不能做得比作者更好。（本文其实是不够充分的）
- 除了论文提出的问题，该技术（CNN）、该领域（Image Classification）是否还有其他值得解决的问题（挖掘更多的研究动机）？
- 思考如果是你自己写，会如何组织文章（本文的结构也不是很好，写作也不有缺陷）

如何读论文

Suggestion

- 读论文绝对比读博客收获更多
- 英文对于很多人都是个坎，但读论文可以快速提高你的英文阅读和写作能力
- 你可以借助ChatGPT和文心一言之类的大模型来辅助你阅读，特别是初步了解论文在讲什么，以及英文翻译
- 依然建议，你能够按照三遍法去阅读原版论文，并整理自己的思路，读论文是提高研究水平和论文写作水平的捷径
- 尽量不要把读论文完全交给大模型和博客

欧老师的联系方式

读万卷书 行万里路 只为最好的修炼

论文是为研究工作服务的

研究是为产业落地服务的

所有的研究动机都来源于个人的内生动力

QQ：14777591 (宇宙骑士)
Email：ouxinyu@alumni.hust.edu.cn
Website: http://ouxinyu.cn
Tel：18687840023