深度之眼
deepshare.net

# 法律声明



**课程详情请咨询**

- 微信公众号：深度之眼
- 客服微信号：deepshare0920

公众号

微 信

深度之眼
deepshare.net

# AlexNet: ImageNet Classification with Deep Convolutional Neural Networks

导师：余老师

深度之眼
deepshare.net

# 第二课：论文精读

The second lesson: the paper details

# 《ImageNet Classification with Deep Convolutional Neural Network》

**基于深度卷积神经网络的图像分类**

作者：Alex Krizhevsky（第一作者）

单位：加拿大多伦多大学

深度之眼
deepshare.net

# 目录

深度之眼
deepshare.net

# 上节回顾

Review in the previous lesson

# 上节回顾

Review in the previous lesson

知识回顾

课程安排：3个课时，导读、精读和代码，6个部分

论文总览：论文包含9个主要部分

研究背景：ILSVRC挑战赛，高性能计算资源GPU

成果意义：ILSVRC冠军，推动CV、ML、DL的发展

深度之眼
deepshare.net

# 论文结构

## Structure of Paper

# 论文结构

Structure of Papers



## 摘要 Abstract

介绍背景及提出AlexNet模型，获得ILSVRC-2012冠军

## 1 Introduction

研究的成功得益于大量数据及高性能GPU；介绍本论文主要贡献

## 2 The Dataset

ILSVRC数据集简介；图片预处理细节

## 3 The Architecture

AlexNet网络结构及其内部细节：ReLU、GPU、LRN、Overlapping Pooling

## 4 Reducing Overfitting

防过拟合技术，数据增强和dropout

## 5 Details of learning

实验参数设置：超参调整，权重初始化

## 6 Results

AlexNet比赛指标、成绩及其详细设置

## 7 Qualitative Evaluations

实验探究，分析卷积核模式，模型输出合理性，高级特征的相似性

## 8 Discussion

强调网络结构之间的强关联提出进一步研究方向

# 论文结构

Structure of Papers

## 摘要

1. 在ILSVRC-2010的120万张图片上训练深度卷积神经网络，获得最优结果，top-1和top-5 error分别为 37.5%, 17%

2. 该网络（AlexNet）由5个卷积层和3个全连接层构成，共计6000万参数，65万个神经元

3. 为加快训练，采用非饱和激活函数——ReLU，采用GPU训练

4. 为减轻过拟合，采用Dropout

5. 基于以上模型及技巧，在ILSVRC-2012以超出第二名10.9个百分点成绩夺冠

原文讲解

# 论文结构

Structure of Papers

深度之眼
deepshare.net

## 论文小标题

1. Introduction
2. The Dataset
3. The Architecture
    - 3.1 ReLU Nonlinearity
    - 3.2 Training on Multiple GPUs
    - 3.3 Local Response Normalization
    - 3.4 Overlapping Pooling
    - 3.5 Overall Architecture

4. Reducing Overfitting
    - 4.1 Data Augmentation
    - 4.2 Dropout
5. Details of learning
6. Results
    - 6.1 Qualitative Evaluations
7. Discussion

深度之眼
deepshare.net

# AlexNet结构

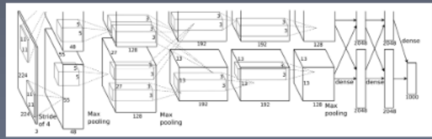
Architecture of AlexNet

# AlexNet结构

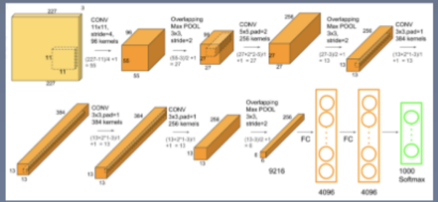
## Architecture of AlexNet

**AlexNet网络结构**

**网络连接方式**



**数据流计算**



**连接参数计算**

| Layer Name | Filter | Padding | Kernel | Stride | Img size | Tensor Size | Weights | Biases | Parameters |
|---|---|---|---|---|---|---|---|---|---|
| Input Image | - | - | - | - | - | 227x227x3 | 0 | 0 | 0 |
| Conv-1 | 96 | 0 | 11 x 11 | 4 | (227-11+2*0)/4+1 = 55 | 55x55x96 | 3*(11*11)*96 = 34848 | 96 | 34,944 |
| MaxPool-1 | 96 | 2 | 3 x 3 | 2 | (55-3)/2+1 = 27 | 27x27x96 | 0(no weight) | 0 | 0 |
| Conv-2 | 256 | 2 | 5 x 5 | 1 | (27-5+2*2)/1+1 = 27 | 27x27x256 | 96*(5*5)*256 = 614400 | 256 | 614,656 |
| MaxPool-2 | 256 | 0 | 3 x 3 | 2 | (27-3)/2+1 = 13 | 13x13x256 | 0 (no weight) | 0 | 0 |
| Conv-3 | 384 | 1 | 3 x 3 | 1 | (13-3+2*1)/1+1 = 13 | 13x13x384 | 256*(3*3)*384 = 884736 | 384 | 885,120 |
| Conv-4 | 384 | 1 | 3 x 3 | 1 | (13-3+2*1)/1+1 = 13 | 13x13x384 | 384*(3*3)*384 = 1327104 | 384 | 1,327,488 |
| Conv-5 | 256 | 1 | 3 x 3 | 1 | (13-3+2*1)/1+1 = 13 | 13x13x256 | 384*(3*3)*256 = 884736 | 256 | 884,992 |
| MaxPool-3 | 256 | 0 | 3 x 3 | 2 | (13-3)/2+1 = 6 | 6x6x256 | 0 (no weight) | 0 | 0 |
| FC-1 | | | | | | 4096×1 | (6*6*256)*4096 = 37748736 | 4,096 | 37,752,832 |
| FC-2 | | | | | | 4096×1 | 4096 * 4096 = 16777216 | 4,096 | 16,781,312 |
| FC-3 | | | | | | 1000×1 | 4096 * 1000 = 4096000 | 1,000 | 4,097,000 |
| Output | | | | | | 1000×1 | 0 | 0 | 0 |
| **Total** | | | | | | | | | **62,378,344** |

$F_i\times(K_s\times K_s)\times K_n+ K_n$

# AlexNet结构

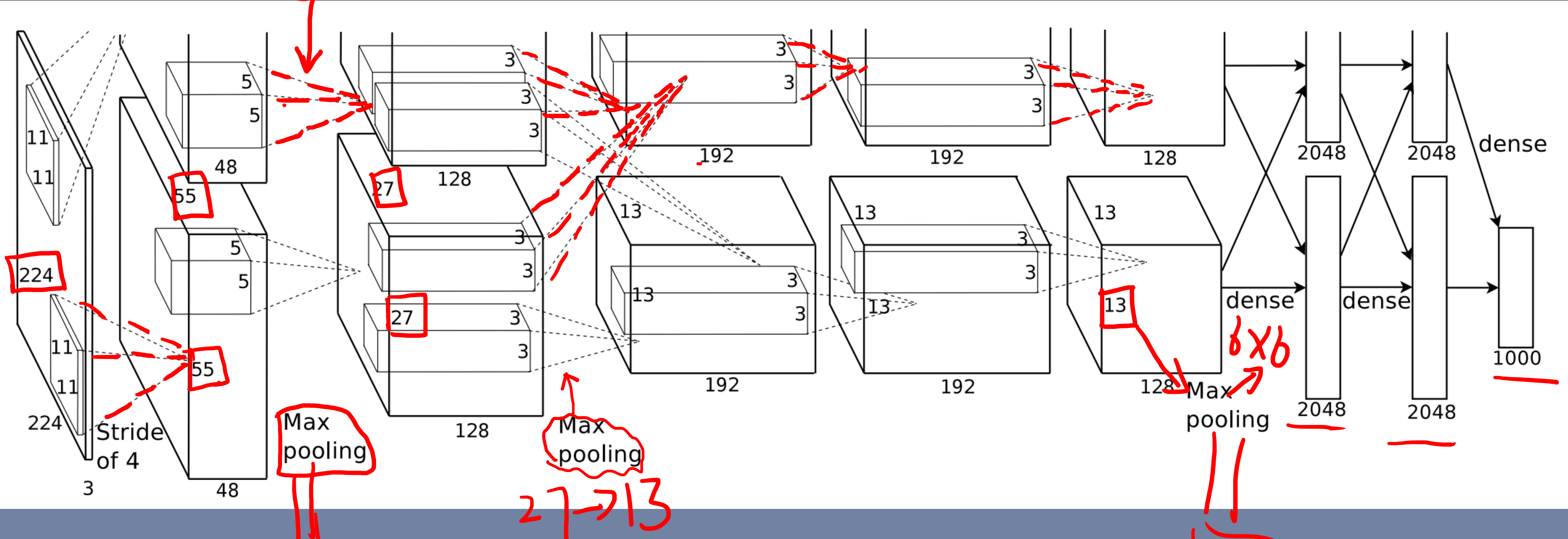
## Architecture of AlexNet



conv2 --> ReLU --> Pool --> LRN

conv4 --> ReLU

重点 重点来了！




conv1 --> ReLU --> Pool --> LRN

conv3 --> ReLU

conv5 --> ReLU --> Pool



原文讲解

# AlexNet结构

## Architecture of AlexNet

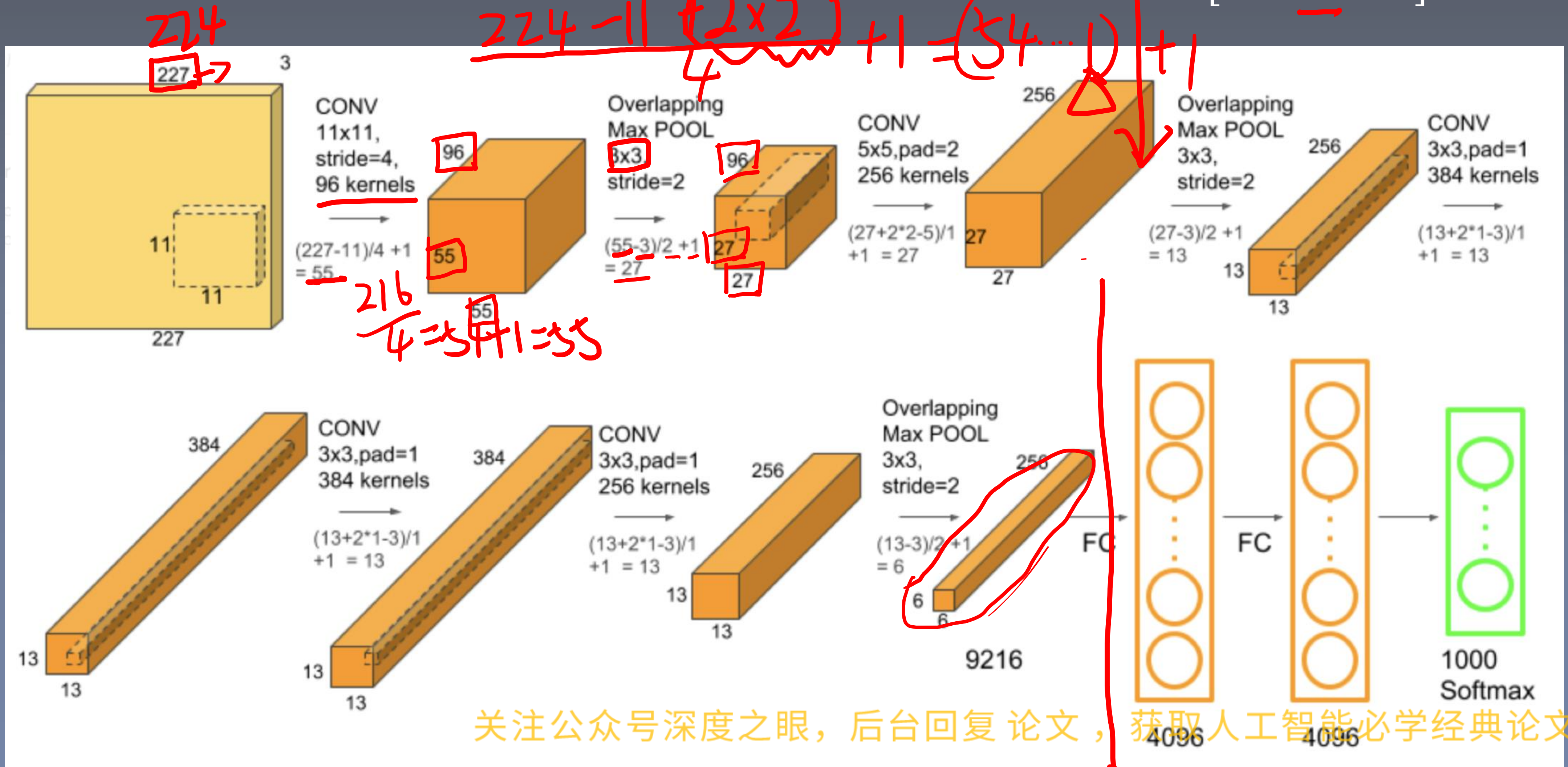
卷积输出特征图:

$$F_o = \left\lfloor \frac{F_{in} - k + 2p}{s} \right\rfloor + 1$$

$\frac{224-11}{4} = \frac{213}{4} = 53 \cdots 1$

$53 + 1 = 54$

$\frac{224-11+2\times 2}{4} + 1 = (54 \cdots 1) + 1$



# AlexNet结构

## Architecture of AlexNet

连接数量计算公式：

$$F_i \times (K_s \times K_s) \times K_n + K_n$$

| Layer Name | Filter | Padding | Kernel | Stride | Img size | Tensor Size | Weights | Biases | Parameters |
|---|---|---|---|---|---|---|---|---|---|
| Input Image | - | - | - | - | - | 227x227x3 | 0 | 0 | 0 |
| Conv-1 | 96 | 0 | 11 x 11 | 4 | (227-11+2*0)/4+1 = 55 | 55x55x96 | 3*(11*11)*96 = 34848 | 96 | 34,944 |
| MaxPool-1 | 96 | 2 | 3 x 3 | 2 | (55-3)/2+1 = 27 | 27x27x96 | 0(no weight) | 0 | 0 |
| Conv-2 | 256 | 2 | 5 x 5 | 1 | (27-5+2*2)/1+1 = 27 | 27x27x256 | 96*(5*5)*256 = 614400 | 256 | 614,656 |
| MaxPool-2 | 256 | 0 | 3 x 3 | 2 | (27-3)/2+1 = 13 | 13x13x256 | 0 (no weight) | 0 | 0 |
| Conv-3 | 384 | 1 | 3 x 3 | 1 | (13-3+2*1)/1+1 = 13 | 13x13x384 | 256*(3*3)*384 = 884736 | 384 | 885,120 |
| Conv-4 | 384 | 1 | 3 x 3 | 1 | (13-3+2*1)/1+1 = 13 | 13x13x384 | 384*(3*3)*384 = 1327104 | 384 | 1,327,488 |
| Conv-5 | 256 | 1 | 3 x 3 | 1 | (13-3+2*1)/1+1 = 13 | 13x13x256 | 384*(3*3)*256 = 884736 | 256 | 884,992 |
| MaxPool-3 | 256 | 0 | 3 x 3 | 2 | (13-3)/2+1 = 6 | 6x6x256 | 0 (no weight) | 0 | 0 |
| FC-1 | | | | | | 4096×1 | (6*6*256)*4096 = 37748736 | 4,096 | 37,752,832 |
| FC-2 | | | | | | 4096×1 | 4096 * 4096 = 16777216 | 4,096 | 16,781,312 |
| FC-3 | | | | | | 1000×1 | 4096 * 1000 = 4096000 | 1,000 | 4,097,000 |
| Output | | | | | | 1000×1 | 0 | 0 | 0 |
| Total | | | | | | | | | 62,378,344 |

# AlexNet结构

Architecture of AlexNet

## AlexNet 结构特点

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.

- The Architecture
  - ReLU Nonlinearity
  - Training on Multiple GPUs
  - Local Response Normalization
  - Overlapping Pooling

原文讲解

# AlexNet结构

Architecture of AlexNet

## ReLU Nonlinearity

**Relu优点：**

1. 使网络训练更快
2. 防止梯度消失（弥散）
3. 使网络具有稀疏性

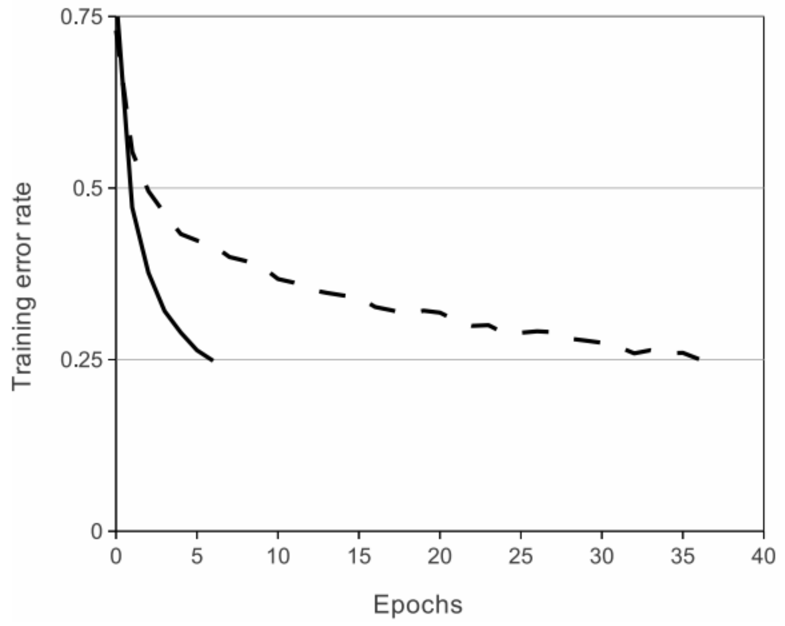



Figure 1: A four-layer convolutional neural network with ReLUs **(solid line)** reaches a 25% training error rate on CIFAR-10 six times faster than an equivalent network with tanh neurons **(dashed line)**. The learning rates for each network were chosen independently to make training as fast as possible.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

# AlexNet结构

Architecture of AlexNet

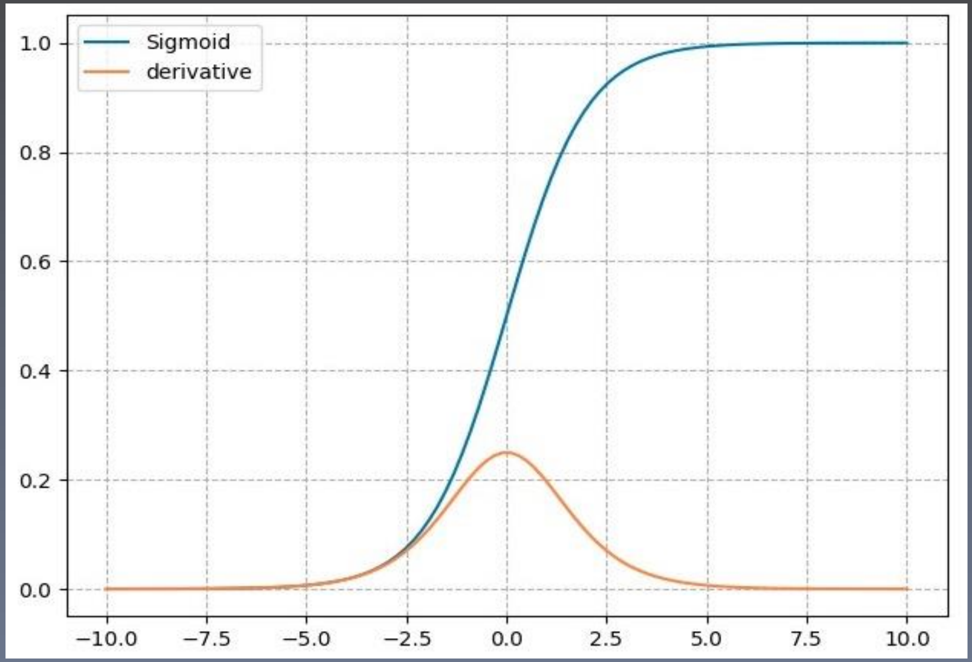
## ReLU Nonlinearity

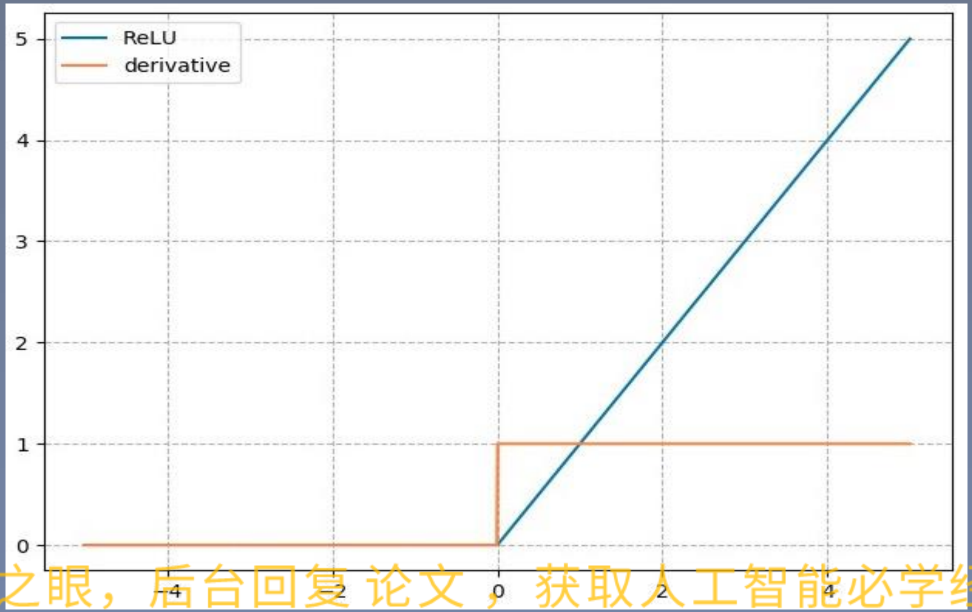
Sigmoid 计算公式：$y = \frac{1}{1+e^{-x}}$

梯度公式：$y' = y * (1 - y)$

ReLu 计算公式：$y = \max(0, x)$

梯度公式：$y' = \begin{cases} 1, & x > 0 \\ undefined, & x = 0 \\ 0, & x < 0 \end{cases}$



# AlexNet结构

## Architecture of AlexNet

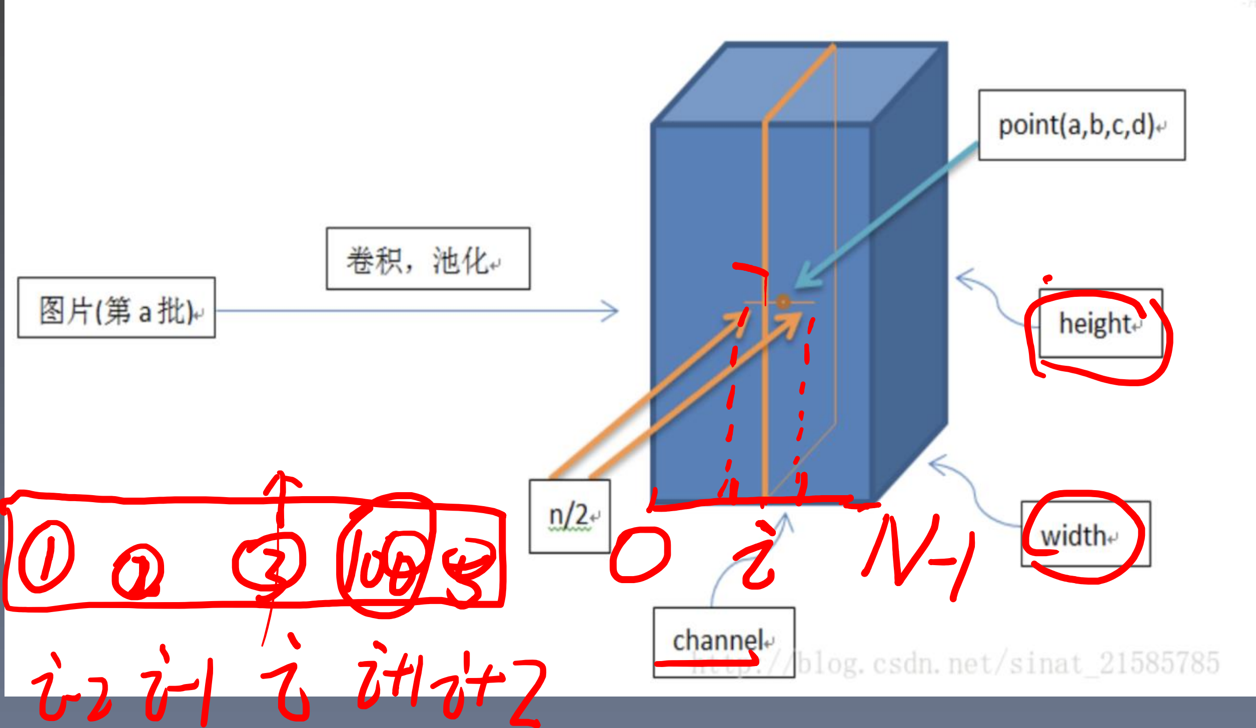
### Local Response Normalization

局部响应标准化：有助于AlexNet**泛化能力的提升**

受真实神经元**侧抑制(lateral inhibition)**启发

**侧抑制**：细胞分化变为不同时，它会对周围细胞产生抑制信号，阻止它们向相同方向分化，最终表现为细胞命运的不同

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




k = 2, n = 5, α = 10^−4, and β = 0.75

i：代表通道 channel

j：平方累加索引，代表从j ~ i的像素值平方求和

x,y：像素的位置，公式中用不到

a：代表feature map里面的 i 对应像素的具体值

N：每个feature map里面最内层向量的列数

k：超参数，由原型中的blas指定

α：超参数，由原型中的alpha指定

n/2：超参数，由原型中的deepth_radius指定

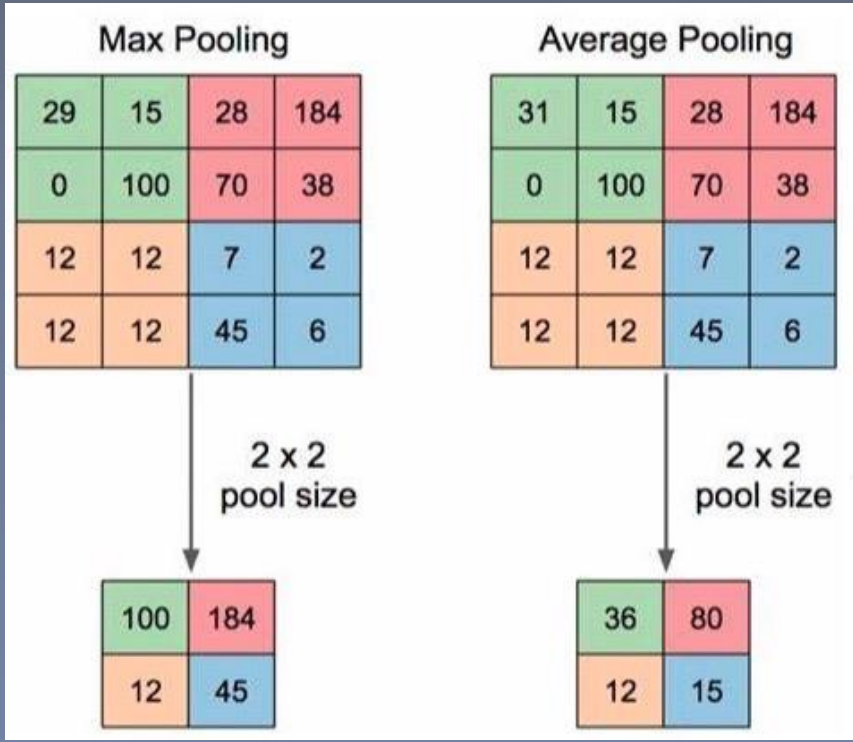
β：超参数，由原型中的belta指定

原文讲解

# AlexNet结构

Architecture of AlexNet

## Overlapping Pooling




常见池化

特征图计算公式：

$$F_o = \left\lfloor \frac{F_{\text{i}} - k + 2p}{s} \right\rfloor + 1$$

原文讲解

深度之眼
deepshare.net

# 训练技巧

Training Tricks

# 训练技巧

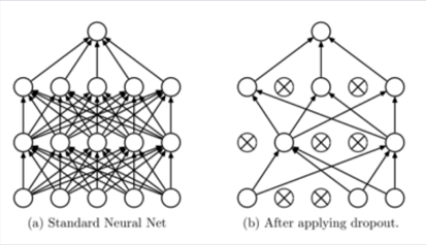
## Training Tricks




Data Augmentation

训练技巧

Dropout



# 训练技巧

Training Tricks

深度之眼
deepshare.net

## Data Augmentation

方法一：针对**位置**

训练阶段：

① 图片统一缩放至256*256

② 随机位置裁剪出224*224区域

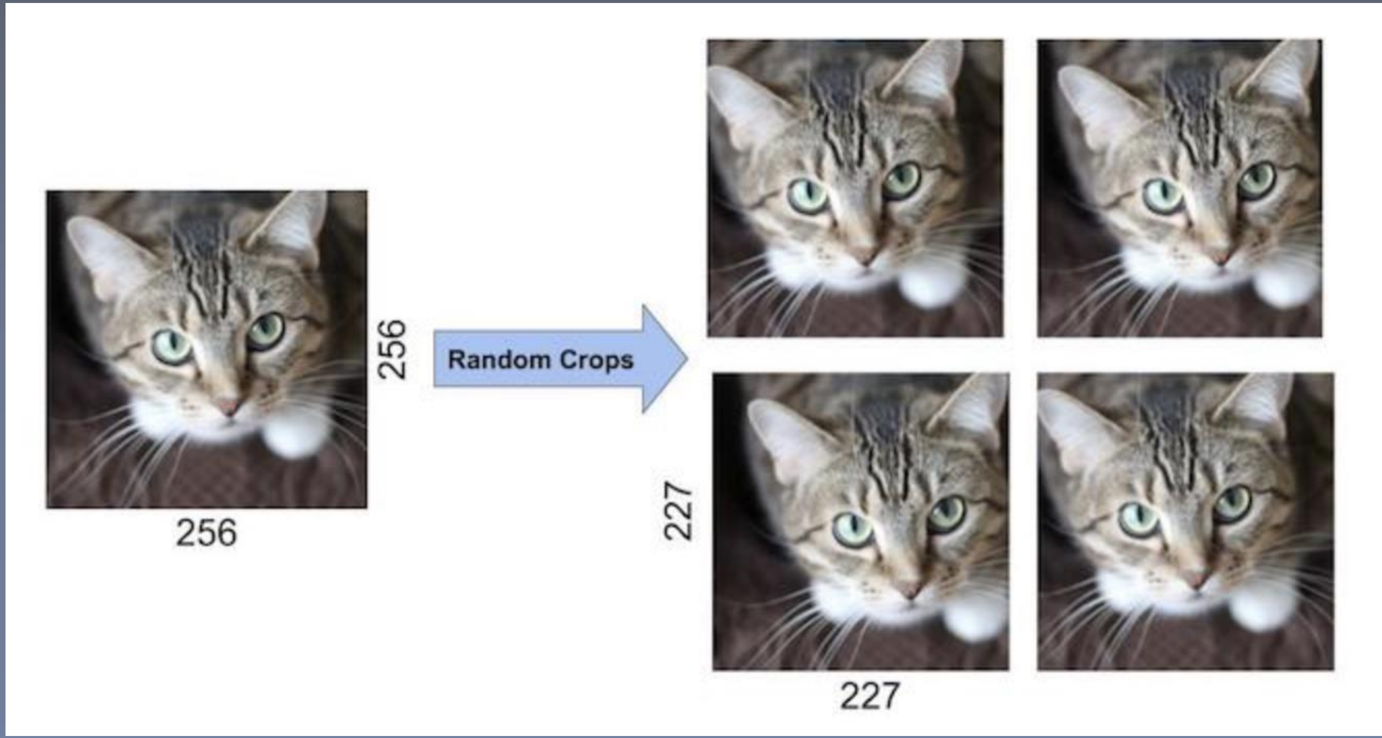
③ 随机进行水平翻转

测试阶段：

① 图片统一缩放至256*256

② 裁剪出5个224*224区域

③ 均进行水平翻转，共得到10张224*224图片




原文讲解

# 训练技巧
Training Tricks

## Data Augmentation

### 方法一：针对**位置**

训练阶段：

① 图片统一缩放至256*256

② 随机位置裁剪出224*224区域

③ 随机进行水平翻转

测试阶段：

① 图片统一缩放至256*256

② 裁剪出5个224*224区域

③ 均进行水平翻转，共得到10张224*224图片

### 方法二：针对**颜色**

通过PCA方法修改RGB通道的像素值，实现颜色扰动，效果有限，仅在top-1提升1个点（top-1 acc约62.5%）

# 训练技巧

Training Tricks

## Dropout

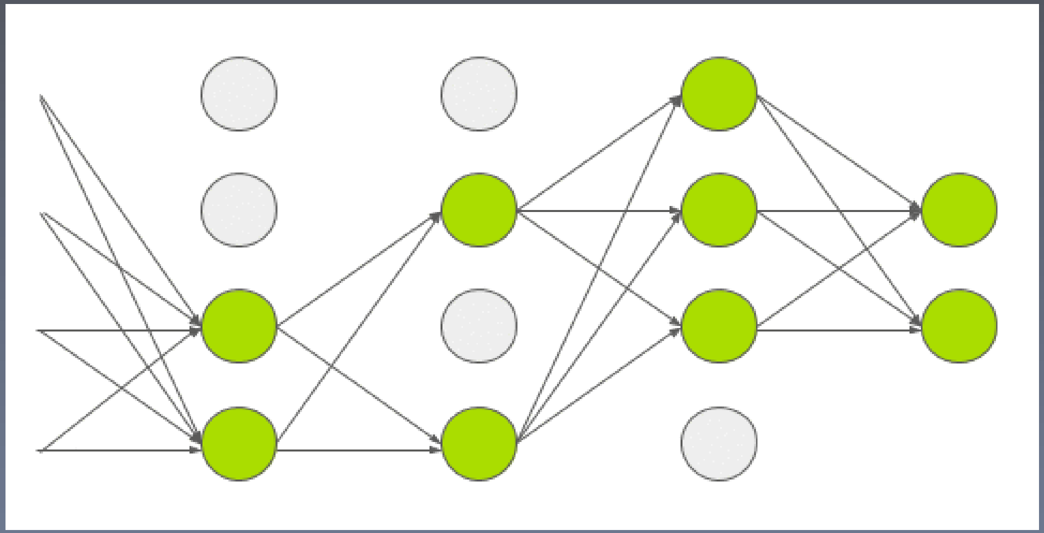
Dropout：**随机失活**

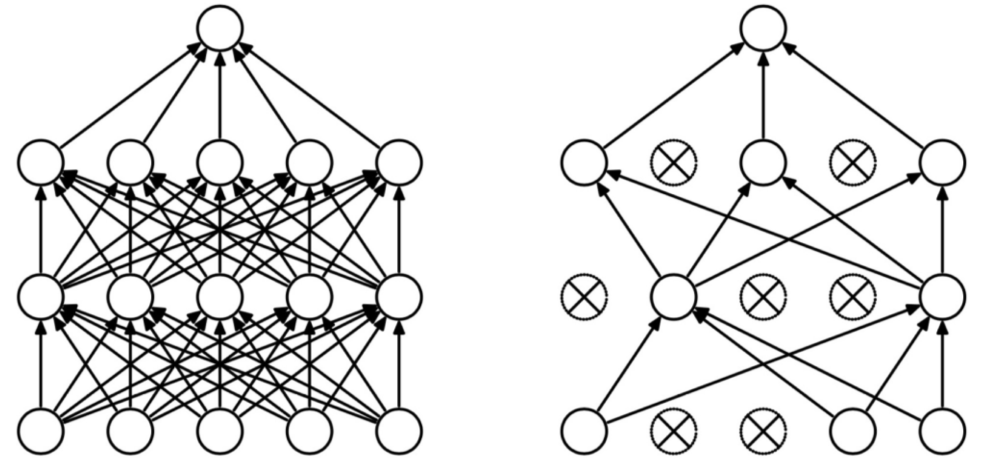
**随机**：dropout probability （eg：p=0.5）

**失活**：weight = 0

**注意事项：训练和测试两个阶段的数据尺度变化**

**测试时，神经元输出值需要乘以 p**





(a) Standard Neural Net (b) After applying dropout.

深度之眼
deepshare.net

# 实验结果及分析

## Results and Discussion

# 实验结果及分析

Results and Discussion



**实现结果及分析**

**ILSVERC-2012比赛**

| 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%** |

**卷积核可视化**

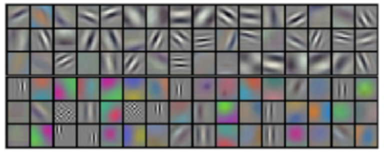


Figure 3: 96 convolutional kernels of size $11\times11\times3$ learned by the first convolutional layer on the $224\times224\times3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

**特征相似性**

# 实验结果及分析

Results and Discussion

## ILSVRC-2012 分类指标

SIFT+FVS：ILSVRC-2012 分类任务**第二名**

1CNN： 训练一个AlexNet

5CNNs： 训练五个AlexNet取平均值

1CNN*在最后一个池化层之后，额外添加第六个卷积层，并使用ImageNet 2011（秋）数据集上**预训练**

7CNNs* 两个预训练微调，与5CNNs取平均值

| 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%** |

# 实验结果及分析

Results and Discussion



## Qualitative Evaluations

### 卷积核可视化

- 卷积核呈现出不同的频率、方向和颜色
- 两个GPU还呈现分工学习

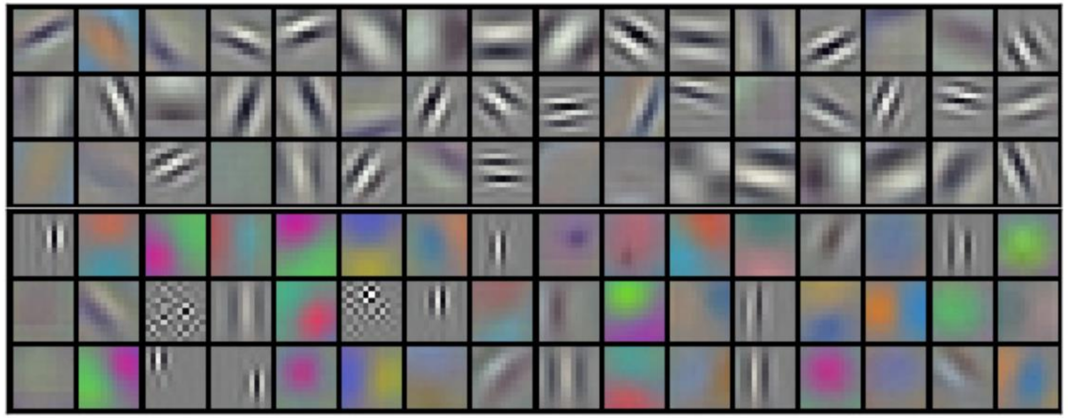


Figure 3: 96 convolutional kernels of size $11 \times 11 \times 3$ learned by the first convolutional layer on the $224 \times 224 \times 3$ input images. The top 48 kernels were learned on GPU 1 while the bottom 48 kernels were learned on GPU 2. See Section 6.1 for details.

原文讲解

# 实验结果及分析

Results and Discussion

深度之眼
deepshare.net

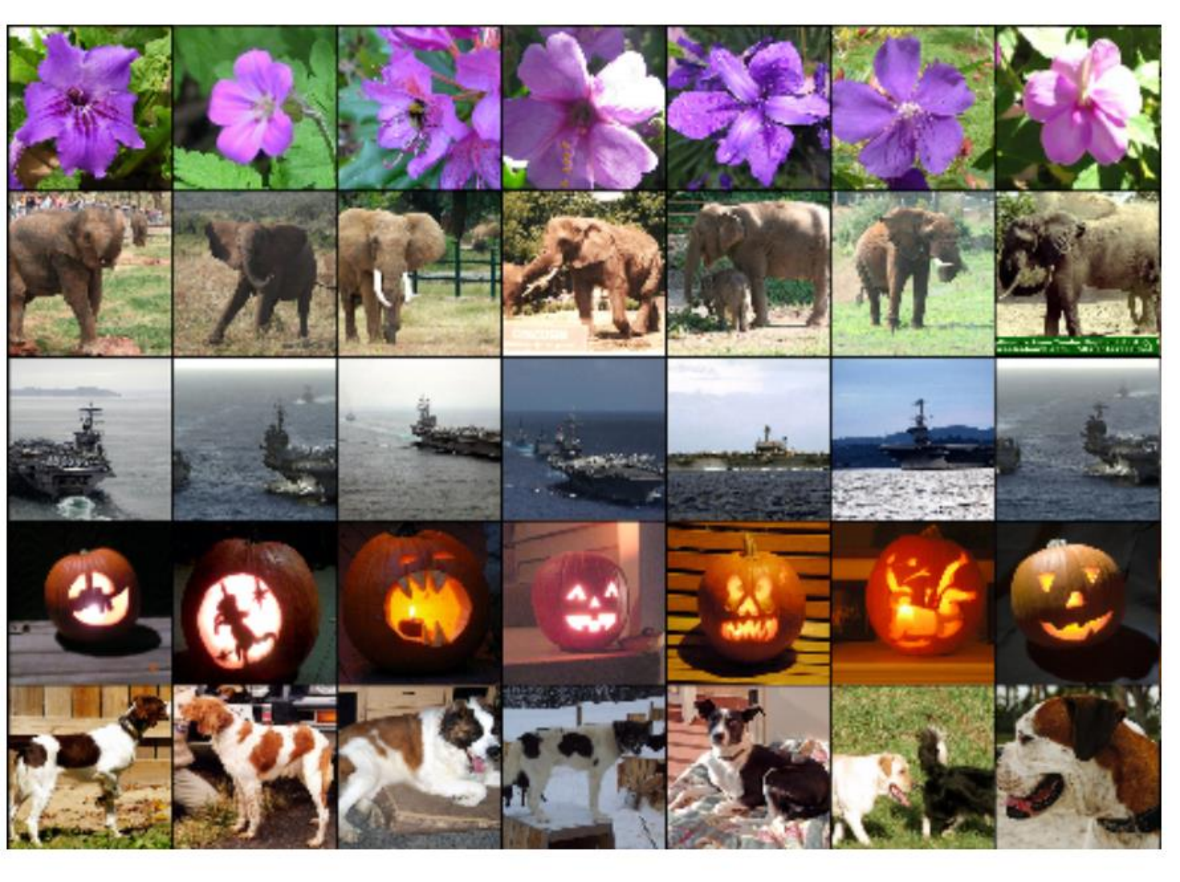
## Qualitative Evaluations

**特征的相似性**

相似图片的第二个全连接层输出**特征向量**的欧式距离**相近**

启发：

可用AlexNet提取高级特征进行图像检索、图像聚类、图像编码



原文讲解

深度之眼
deepshare.net

# 论文总结

## Summary of the paper

# 论文总结

Summary of the paper

## 关键点

- 大量带标签数据——ImageNet
- 高性能计算资源——GPU
- 合理算法模型——深度卷积神经网络

## 创新点

- 采用ReLu加快大型神经网络训练
- 采用LRN提升大型网络泛化能力
- 采用Overlapping Pooling提升指标
- 采用随机裁剪翻转及色彩扰动增加数据多样性
- 采用Drpout减轻过拟合

# 论文总结

Summary of the paper

深度之眼 deepshare.net

## 启发点

- 深度与宽度可决定网络能力

Their capacity can be controlled by varying their depth and breadth.（1 Introduction p2）

- 更强大GPU及更多数据可进一步提高模型性能

All of our experiments suggest that our results can be improved simply by waiting for faster GPUs and bigger datasets to become available. （1 Introduction p5）

- 图片缩放细节，对短边先缩放

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.(2 Dataset p3)

- ReLU不需要对输入进行标准化来防止饱和现象，即说明sigmoid/tanh激活函数有必要对输入进行标准化

ReLUs have the desirable property that they do not require input normalization to prevent them from saturating(3.3 LRN p1)

# 论文总结

## Summary of the paper

### 启发点

- 卷积核学习到频率、方向和颜色特征

The network has learned a variety of frequency- and orientation-selective kernels, as well as various colored blobs.(6.1 p1)

- 相似图片具有“相近”的高级特征

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.(6.1 p3)

- 图像检索可基于高级特征，效果应该优于基于原始图像

This should produce a much better image retrieval method than applying autoencoders to the raw pixels.(6.1 p4)

- 网络结构具有相关性，不可轻易移除某一层

It is notable that our network’ s performance degrades if a single convolutional layer is removed.(7 Discussion p1)

- 采用视频数据，可能有新突破

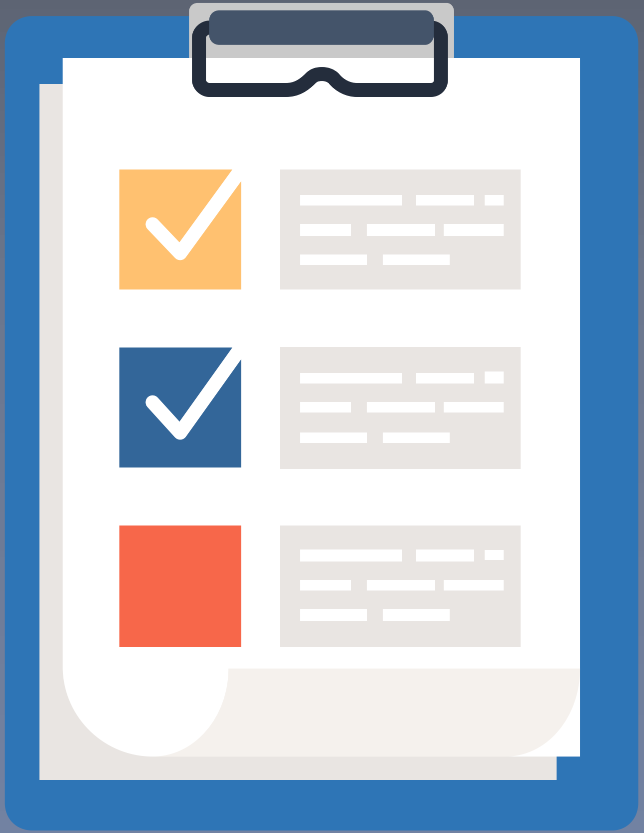
Ultimately we would like to use very large and deep convolutional nets on video sequences.(7 Discussion p2)

# 本课回顾

Review in the lesson

# 本课回顾

Review in the lesson



## 01 AlexNet网络结构及参数计算

讲解AlexNet网络的构成，为什么是8层，哪里用了LRN，哪里用了Pooling Layer，6000万网络参数如何计算

## 02 AlexNet网络特色及训练技巧

AlexNet网络结构特色及关键操作介绍，同时学习训练技巧，包含数据增强操作

## 03 实验结果及结果分析

实验结果分析对比；卷积核可视化分析，高级特征相似性分析

## 04 论文总结

总结论文中创新点、关键点及启发点

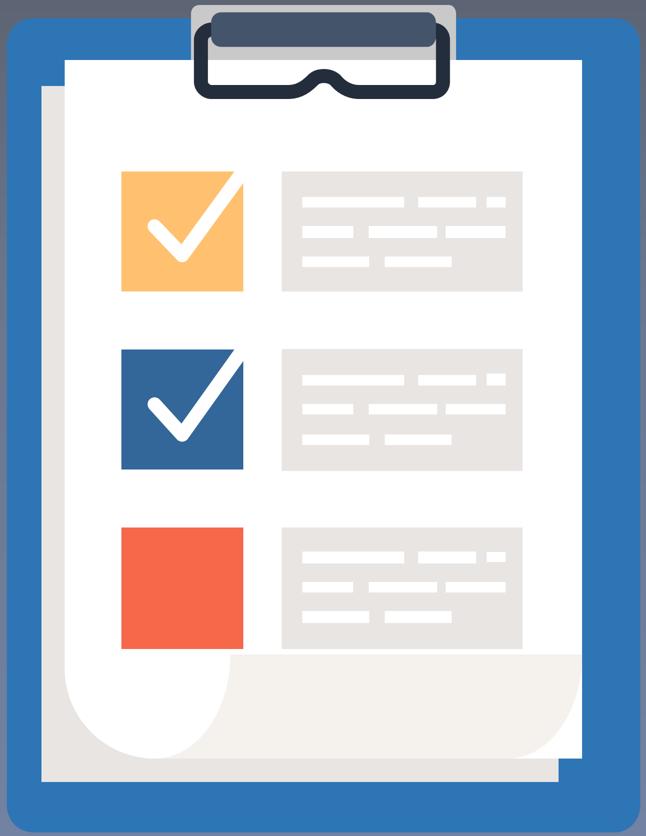
深度之眼
deepshare.net

# 下节预告

Preview of next lesson

# 下节预告

Preview of next lesson



01 搭建AlexNet网络代码介绍

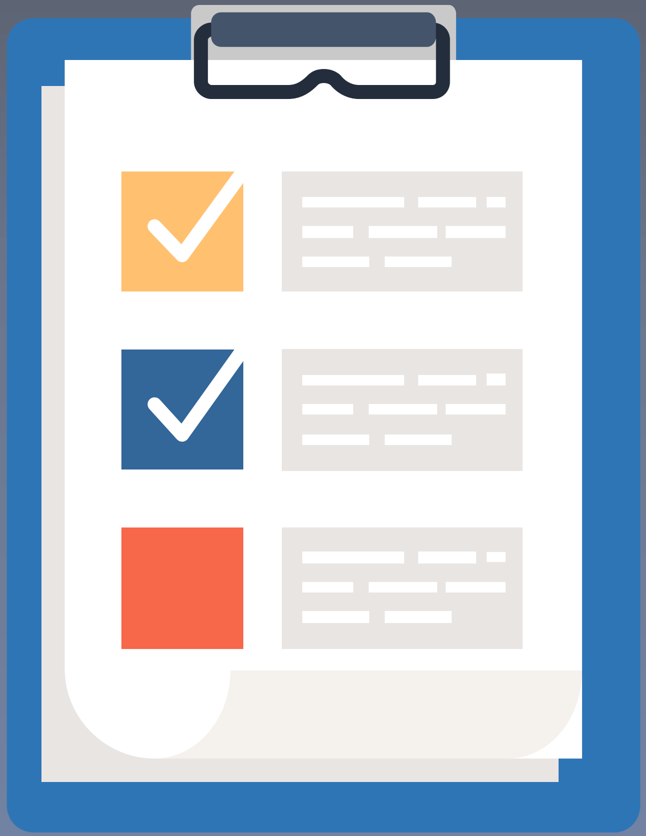
02 AlexNet可视化及结果分析

03 猫狗大战数据集简介及下载

04 基于AlexNet训练猫狗大战

# 下节预告

Preview of next lesson



## 下节课前准备

- 再次阅读AlexNet论文
- 熟悉AlexNet模型结构及数据预处理方式
- 配置PyTorch开发环境
- 下载两个文件

alexnet 预训练模型
https://download.pytorch.org/models/alexnet-owt-4df8aa71.pth

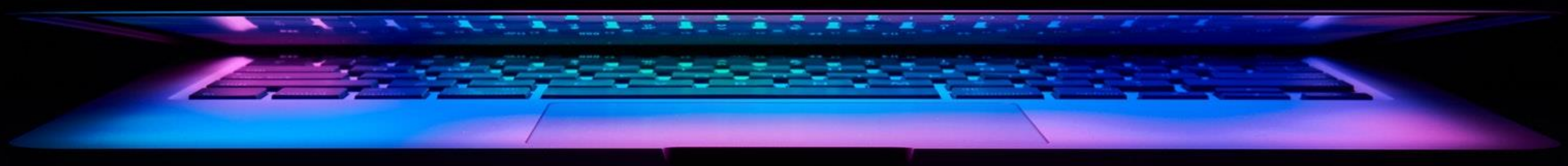
猫狗数据集：
https://www.kaggle.com/c/dogs-vs-cats-redux-kernels-edition/data

# 结 语

在这次课程中，介绍了AlexNet论文中详细知识点

在下次课程中，我们将会学习

AlexNet代码实现（基于PyTorch）



深度之眼
deepshare.net

联系我们:

电话: 18001992849

邮箱: service@deepshare.net

Q Q: 2677693114

公众号

客服微信