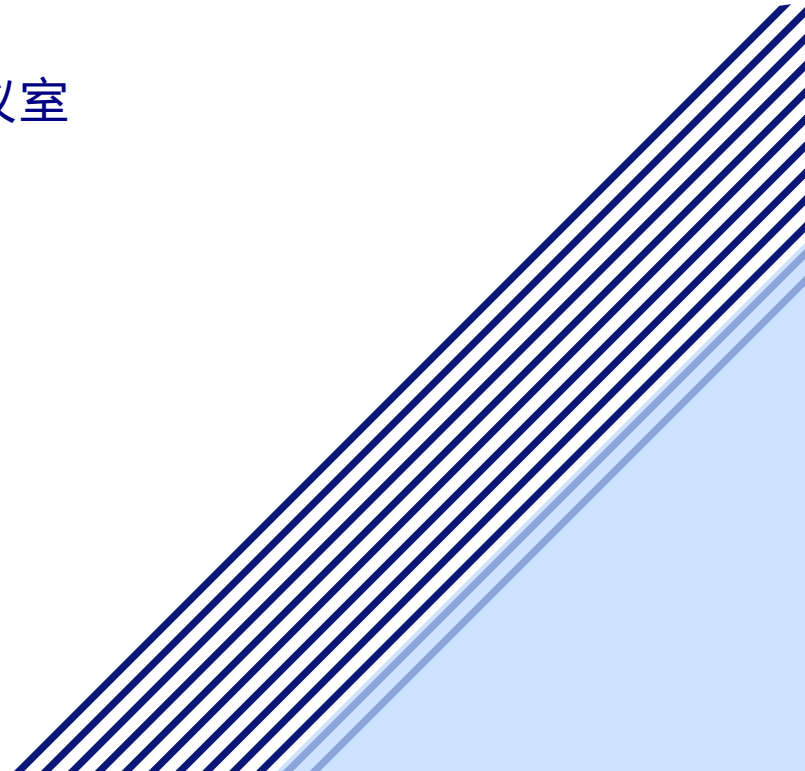
深度神经网络与应用

朱素果 MIL zsg2016@hdu.edu.cn

杭州电子科技大学东区计算机学院 会议室

2019.7.27 下午2:00-4:30

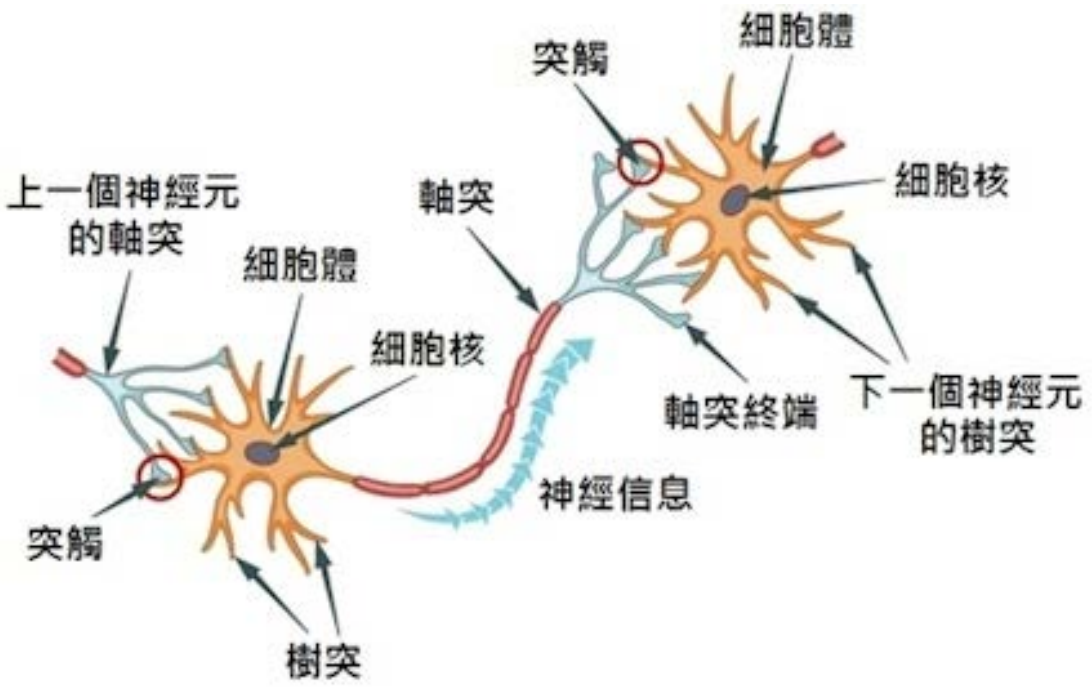


01 网络印象

初衷...

人脑——神经元

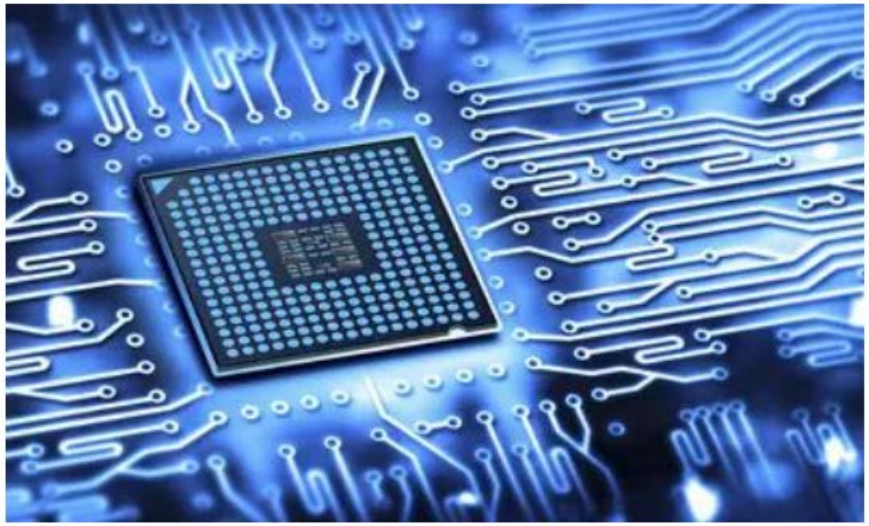
- 1千亿个微小的细胞，称为神经元
- 神经元非常非常小，100个细胞体排列起来才1毫米长



初衷...

电脑——晶体管

- 晶体管
- 20 亿个晶体管
- 2.5 厘米大小的方形集成电路



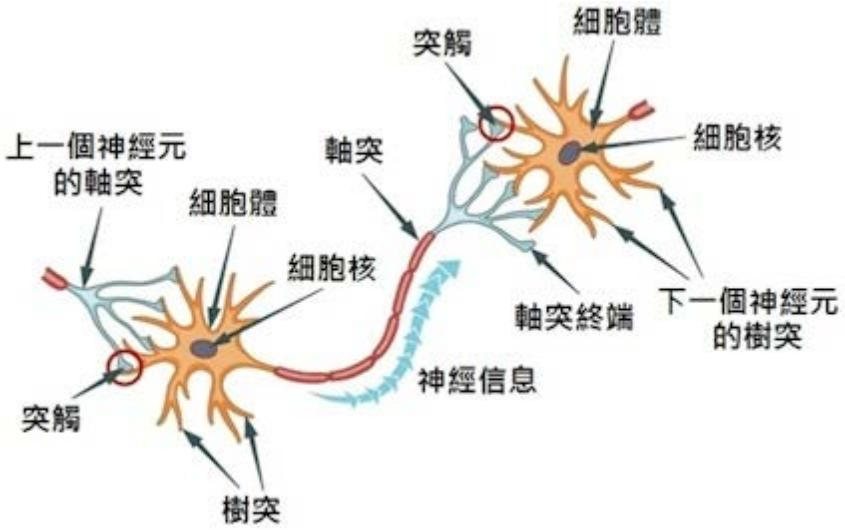
- 电脑中的晶体管只是以一定的顺序相对简单地串联起来（每个晶体管都以基本的逻辑门电路和其它几个晶体管连接），而人脑中的神经元却是以无比复杂的方式并行相连（每一个神经元都可能与其上万个邻近神经元连接）。

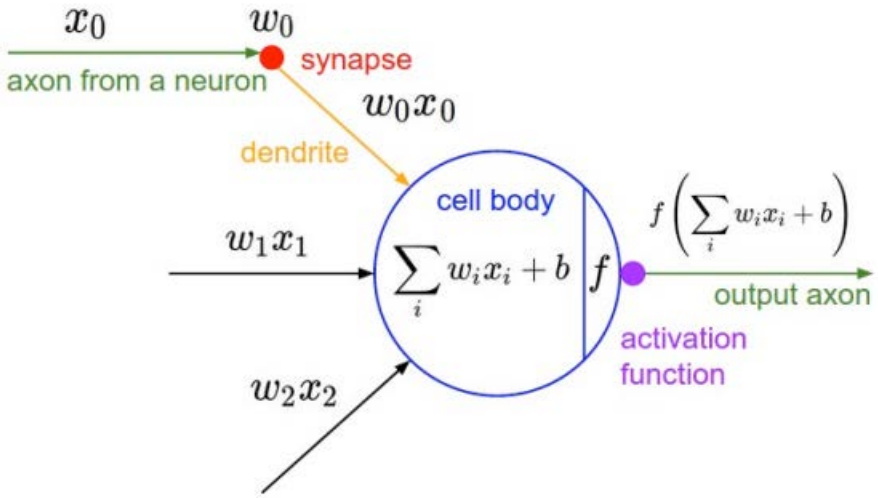
初衷...

- 仿生学的延续
- 目的：希望机器具有人的智能
- 结果：仿 “神经元” 设计 “人工神经元”

人工神经网络

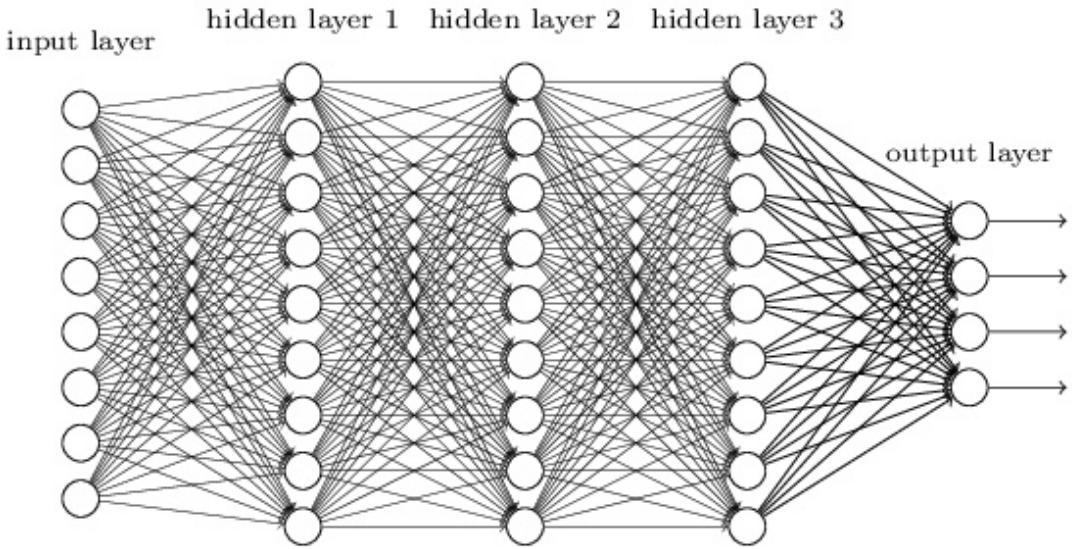
- 人工神经元

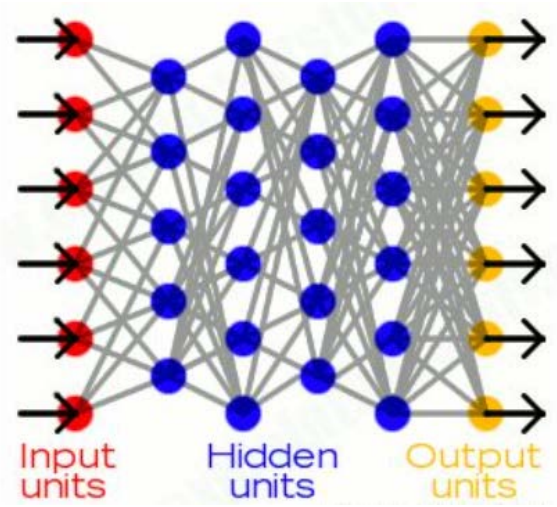




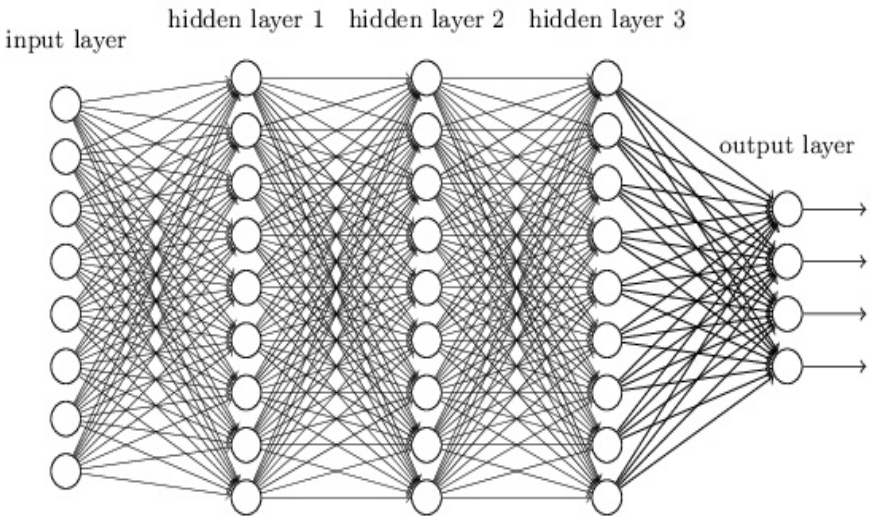
NEURAL NETWORK

- ARTIFICIAL NEURAL NETWORK
- Input layer
- Hidden layer 1~N
- Output layer





BABY!!!


input layer
hidden layer 1
hidden layer 2
hidden layer 3
output layer

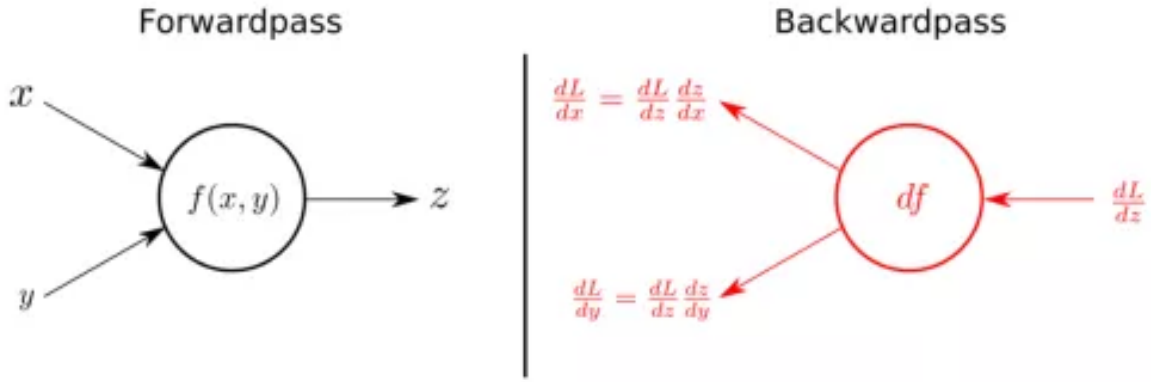

Baby
• COLLECTION •

loss function & cost function

$$\theta^* = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} L(y_i, f(x_i; \theta)) + \lambda \Phi(\theta)$$

- LogLoss对数损失函数（逻辑回归，交叉熵损失）
- center loss
- focal loss
- .....

反向传播



公式 3.44： $dZ^{[2]} = A^{[2]} - Y$ ， $dW^{[2]} = \frac{1}{m} dZ^{[2]} A^{[1]^T}$

公式 3.45： $L = \frac{1}{m} \sum_i^n L(\hat{y}, y)$

公式 3.46： $db^{[2]} = \frac{1}{m} np.sum(dZ^{[2]}, axis = 1, keepdims = True)$

公式 3.47： $dZ^{[1]} = W^{[2]T} dZ^{[2]} * g[1]'(Z^{[1]})$

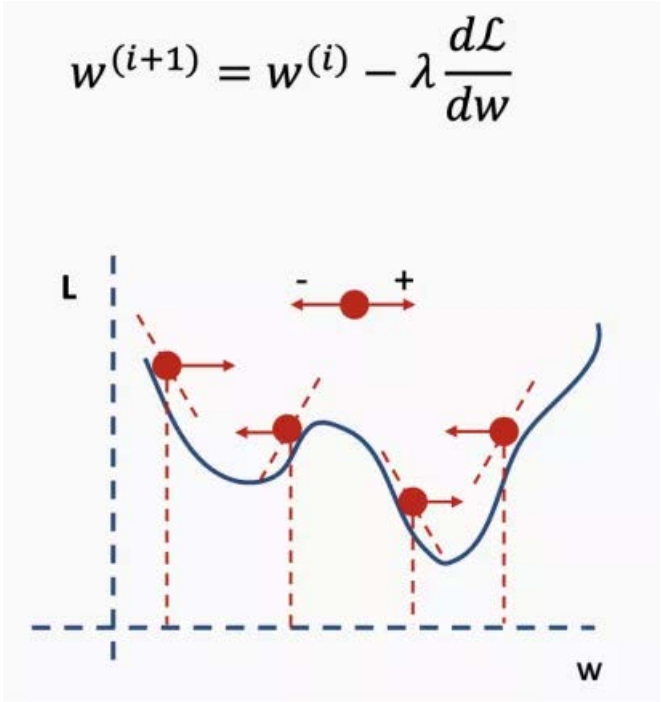
$(n^{[1]}, m)$ $(n^{[1]}, m)$ $(n^{[1]}, m)$

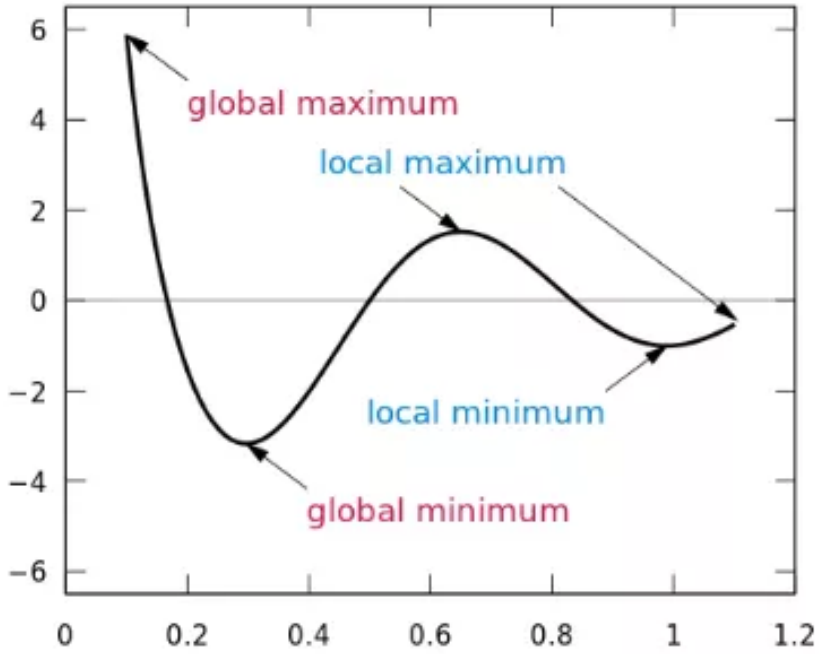
公式 3.48： $dW^{[1]} = \frac{1}{m} dZ^{[1]} x^T$

公式 3.49： $db^{[1]} = \frac{1}{m} np.sum(dZ^{[1]}, axis = 1, keepdims = True)$

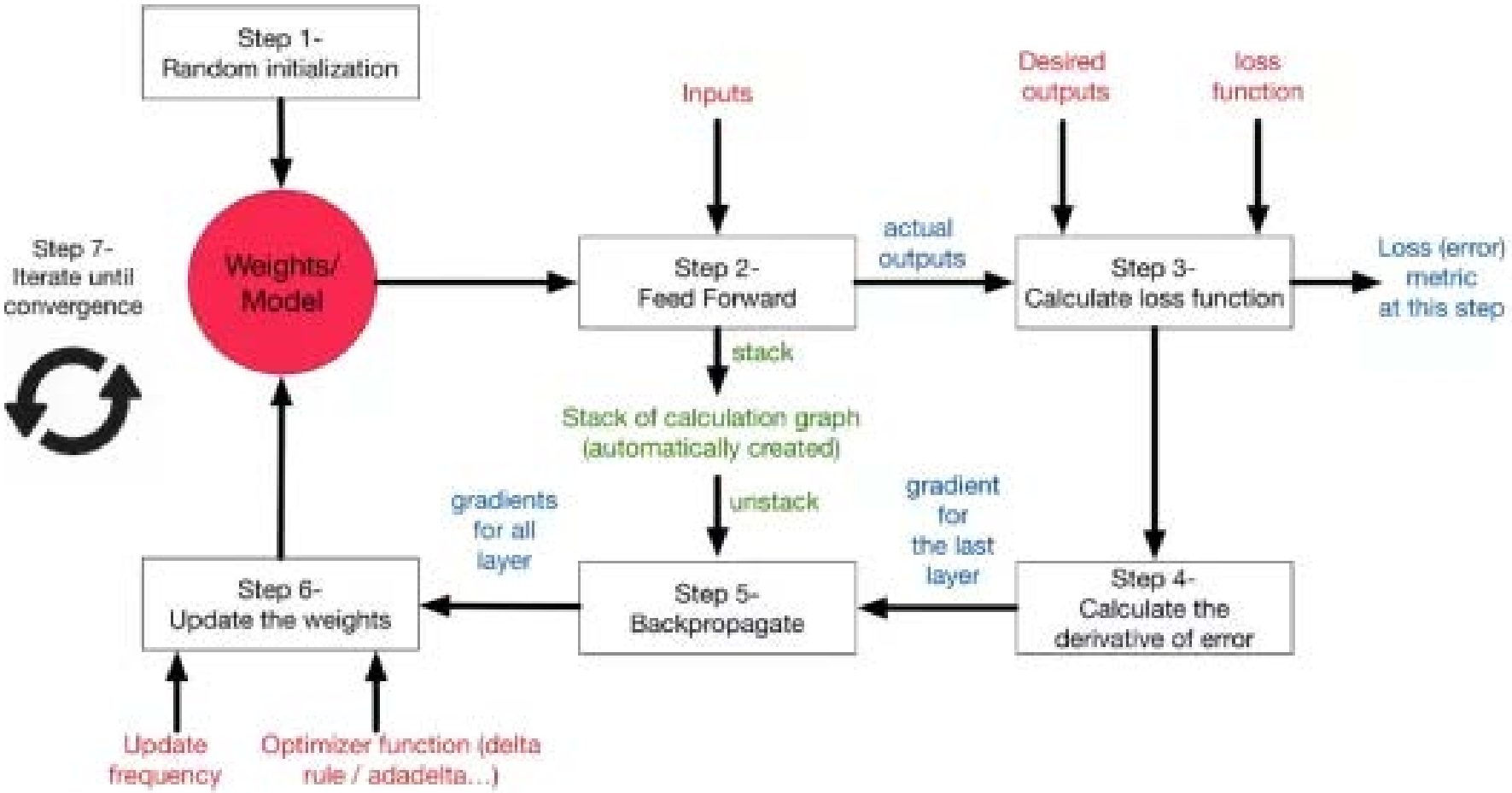
梯度下降

$$w^{(i+1)} = w^{(i)} - \lambda \frac{d\mathcal{L}}{dw}$$





过程

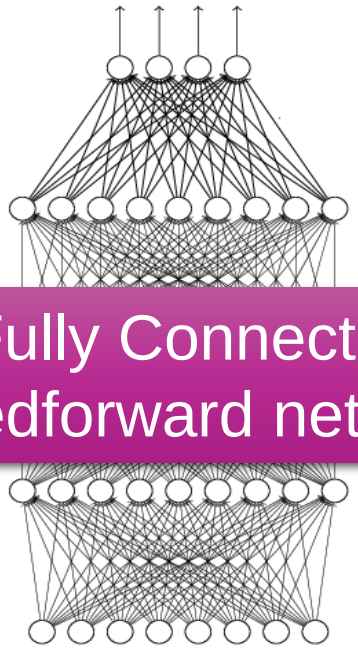

Step 1-
Random initialization
Inputs
Desired outputs
loss function
Step 7-
Iterate until convergence
Weights/ Model
Step 2-
Feed Forward
actual outputs
Step 3-
Calculate loss function
Loss (error) metric at this step
stack
Stack of calculation graph (automatically created)
unstack
gradients for all layer
gradient for the last layer
Step 6-
Update the weights
Step 5-
Backpropagate
Step 4-
Calculate the derivative of error
Update frequency
Optimizer function (delta rule / adadelta...)

02 CNN基本单元

单击此处添加文本具体内容

The whole CNN

cat dog ……



Fully Connected Feedforward network

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Flatten

CNN – Convolution

Those are the network parameters to be learned.

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

Matrix

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

Matrix

⋮ ⋮

Property 1

Each filter detects a small pattern (3 x 3).

CNN – Convolution

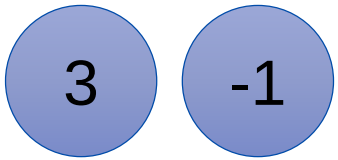
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image



CNN – Convolution

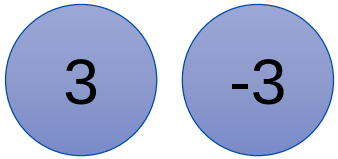
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

If stride=2

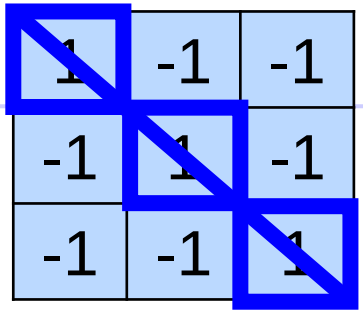
1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image



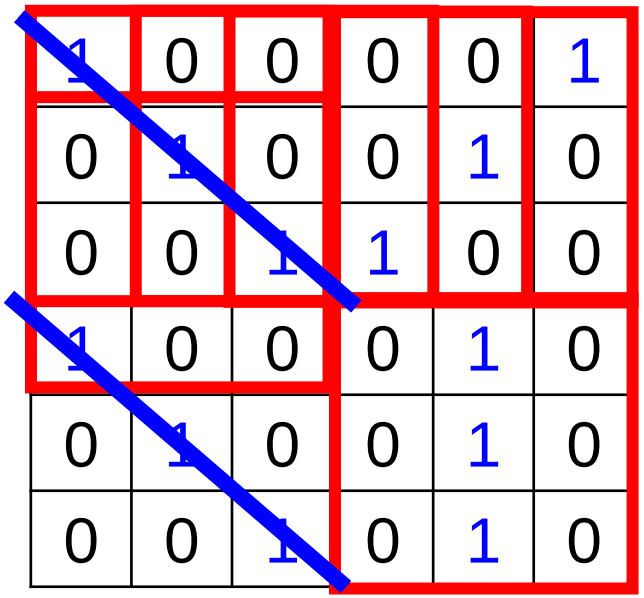
We set stride=1 below

CNN – Convolution

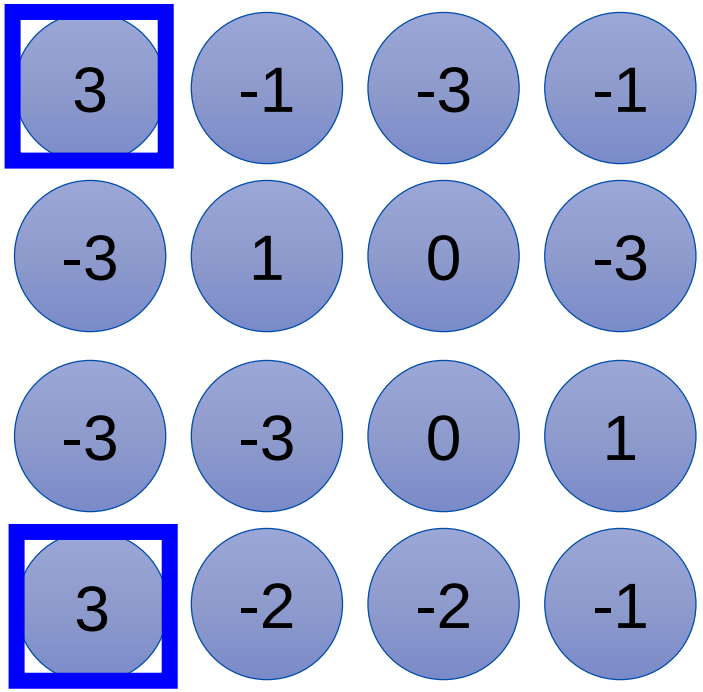


Filter 1

stride=1



6 x 6 image



Property 2

CNN – Convolution

-1	1	-1
-1	1	-1
-1	1	-1

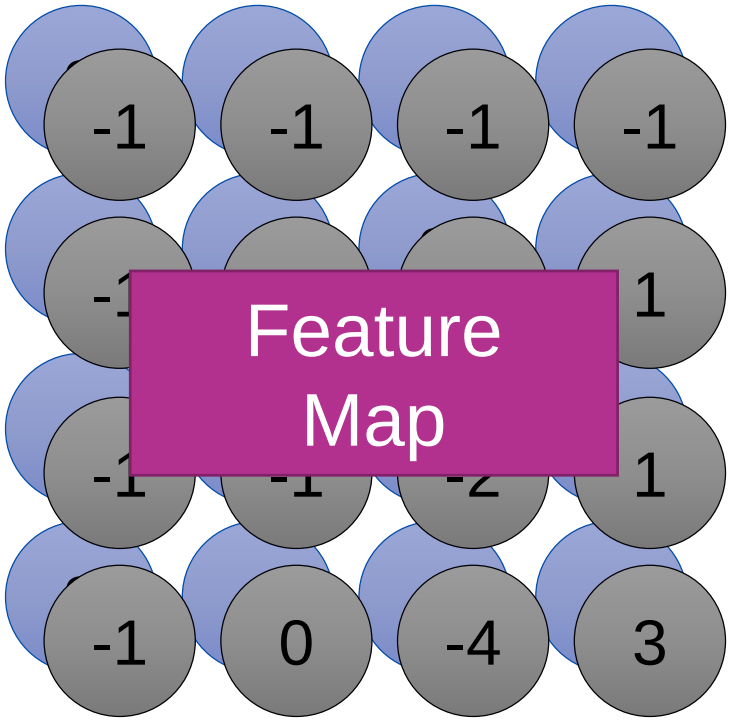
Filter 2

stride=1

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

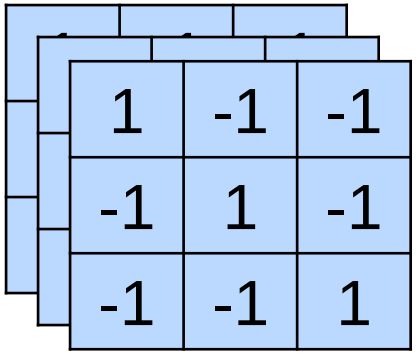
6 x 6 image

Do the same process for every filter

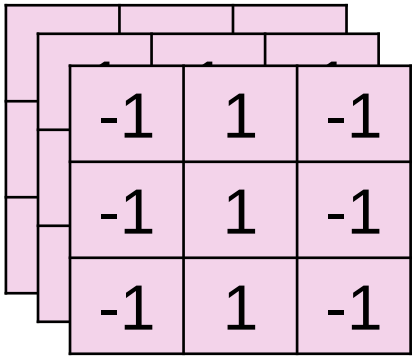


4 x 4 image

CNN – Colorful image

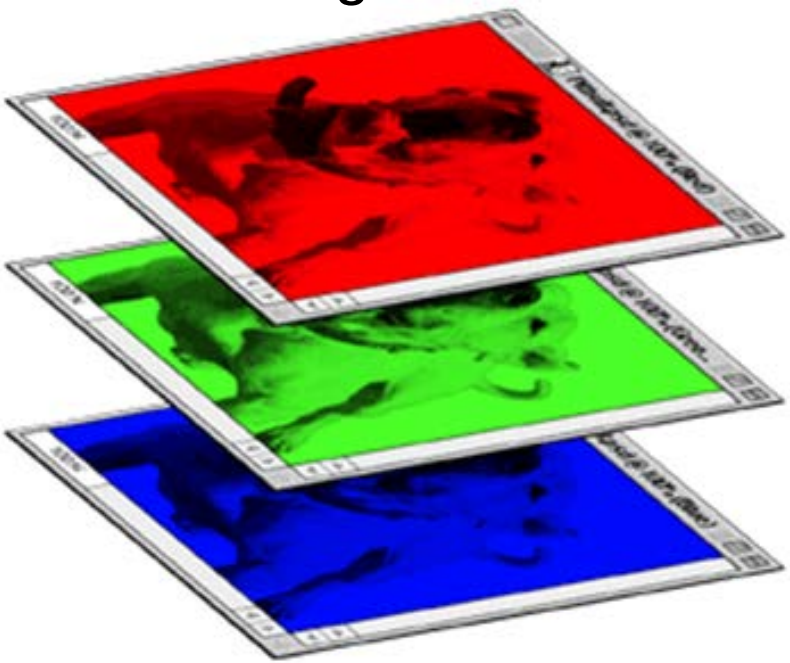


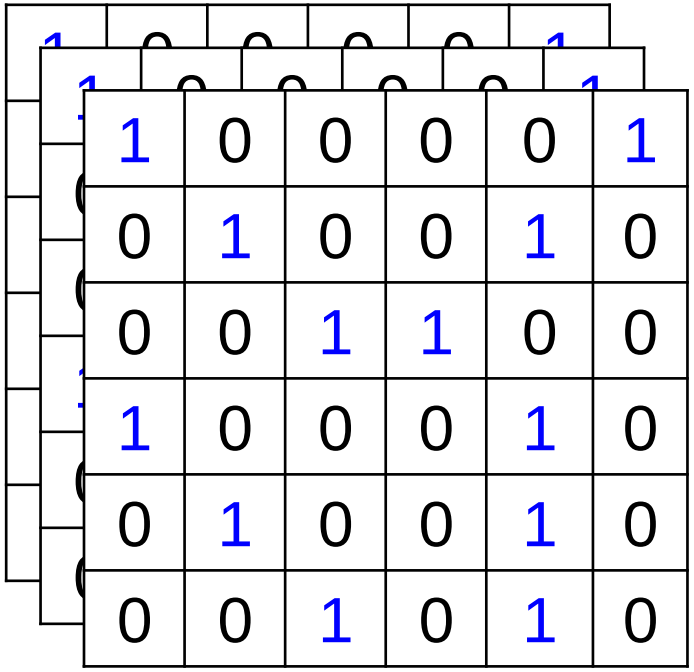
Filter 1



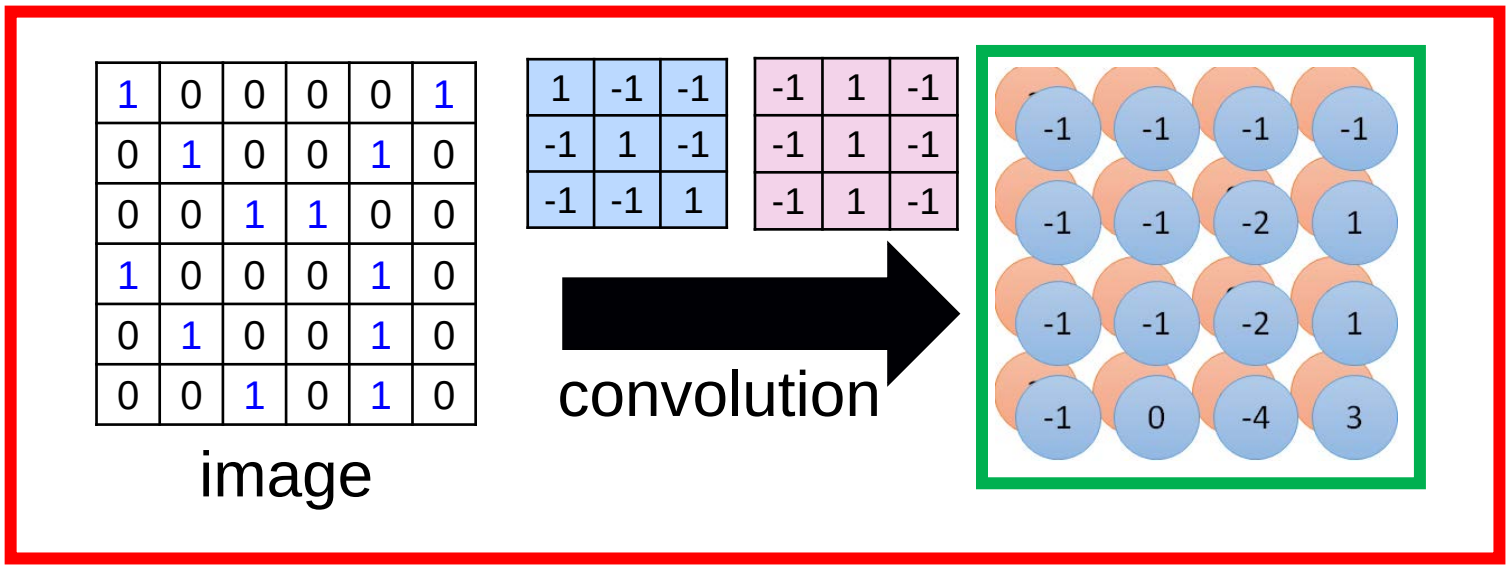
Filter 2

Colorful image

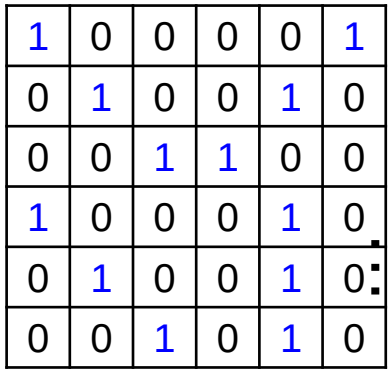


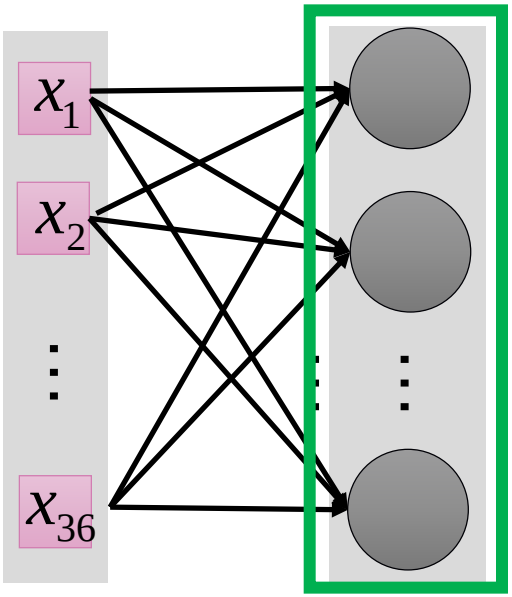


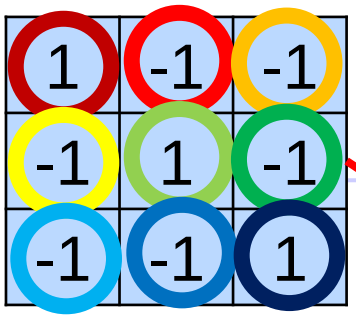
Convolution v.s. Fully Connected



Fully-connected



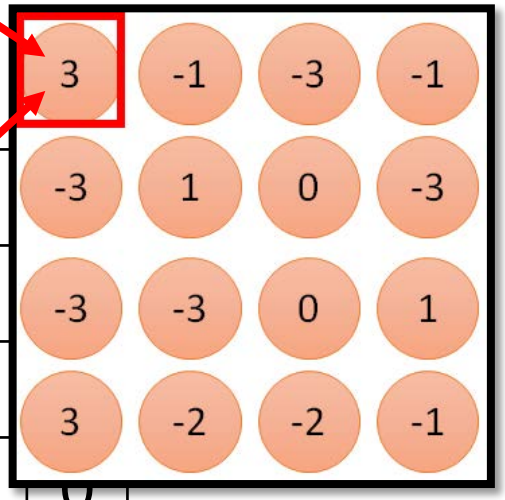




Filter 1

1	0	0	0	0	
0	1	0	0	1	
0	0	1	1	0	
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

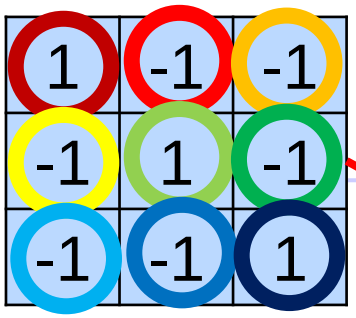


1: 1
2: 0
3: 0
4: 0
⋮
7: 0
8: 1
9: 0
10: 0
⋮
13: 0
14: 0
15: 1
16: 1
⋮

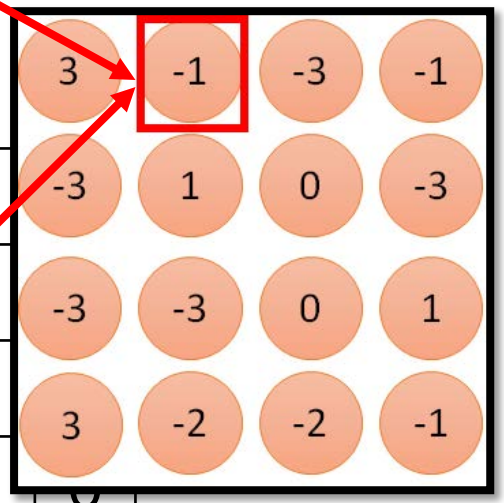
3

Less parameters!

Only connect to 9 input, not fully connected



Filter 1

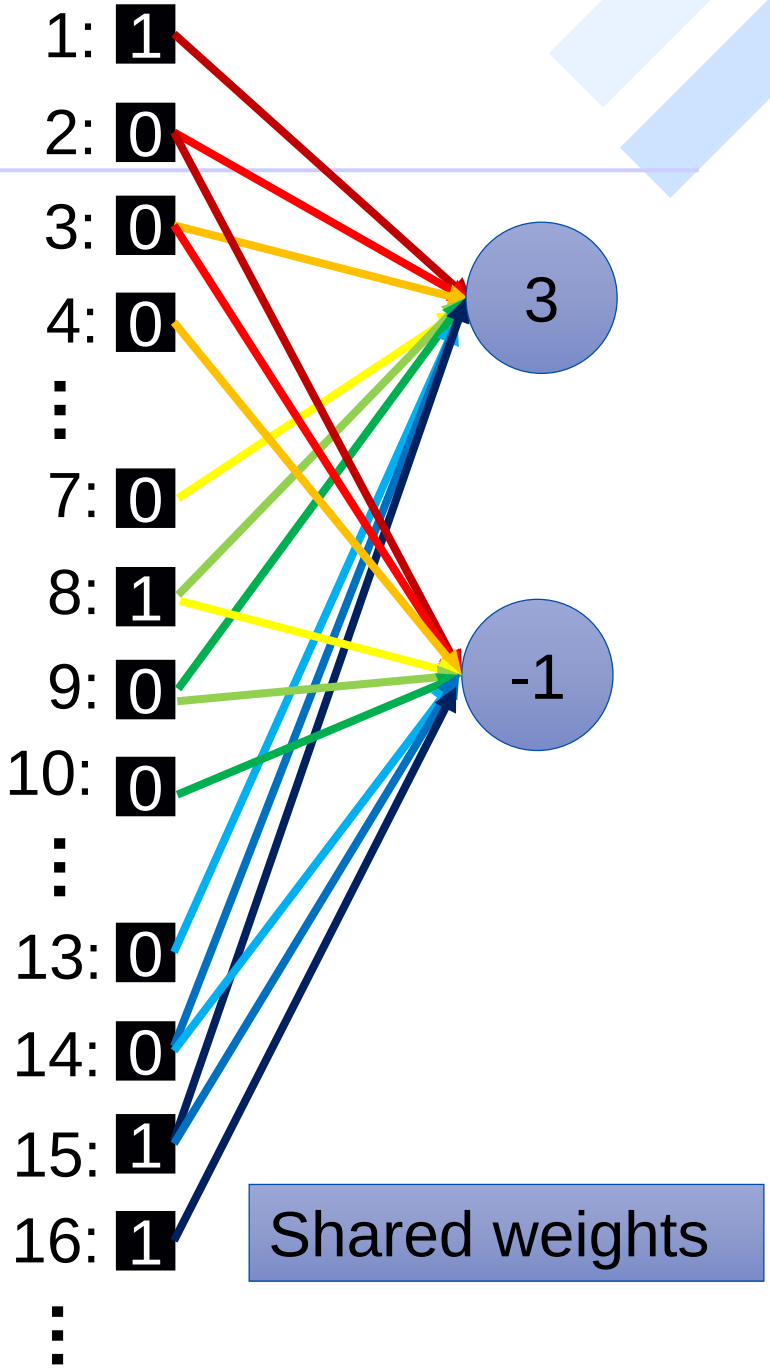


1	0	0	0	0	
0	1	0	0	1	
0	0	1	1	0	
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image

Less parameters!

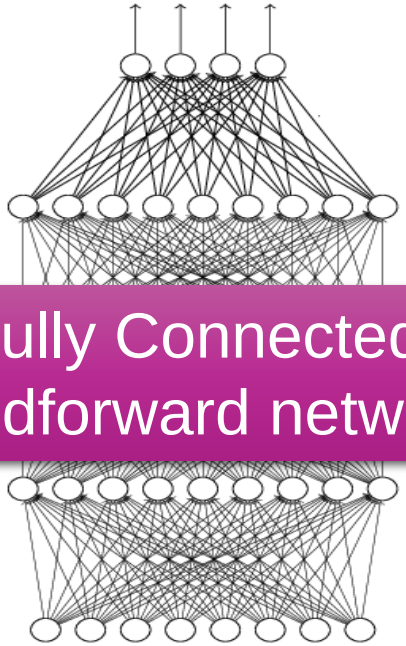
Even less parameters!



单击此处添加标题

The whole CNN

cat dog ……



Fully Connected Feedforward network

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Flatten

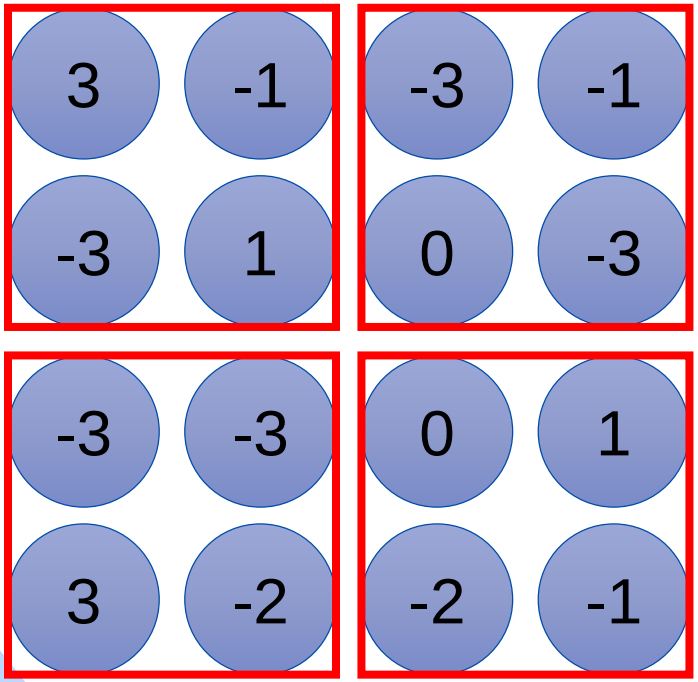
CNN – Max Pooling

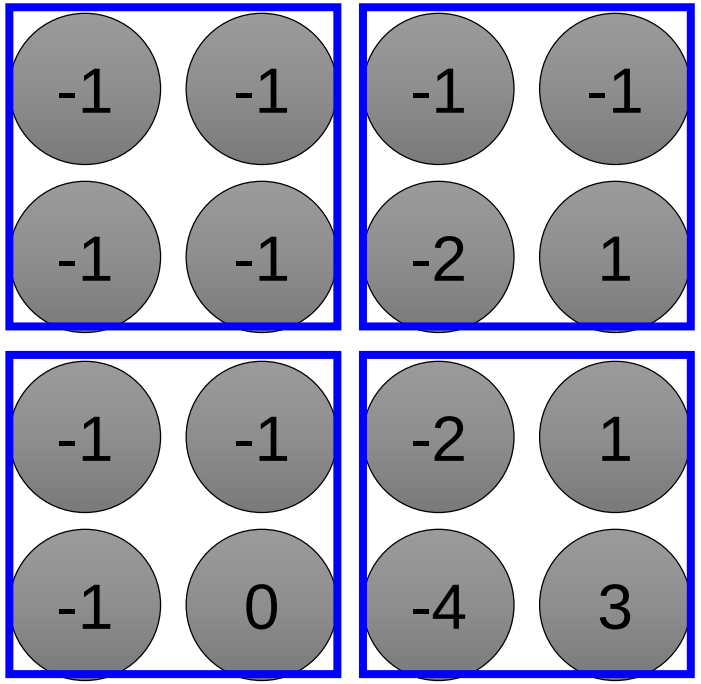
1	-1	-1
-1	1	-1
-1	-1	1

Filter 1

-1	1	-1
-1	1	-1
-1	1	-1

Filter 2

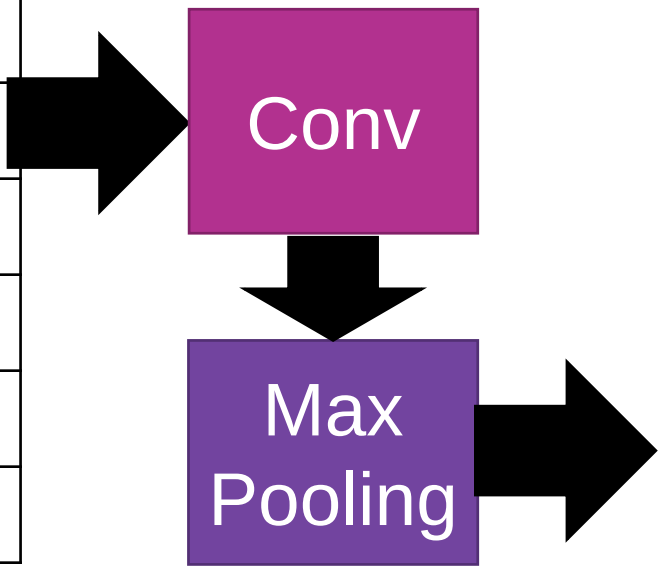




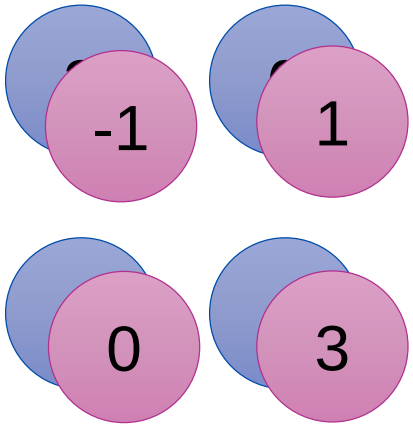
CNN – Max Pooling

1	0	0	0	0	1
0	1	0	0	1	0
0	0	1	1	0	0
1	0	0	0	1	0
0	1	0	0	1	0
0	0	1	0	1	0

6 x 6 image



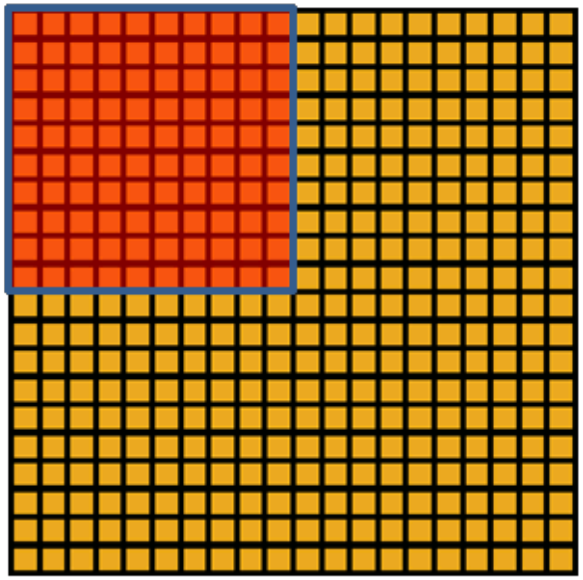
New image but smaller

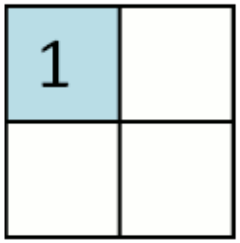


2 x 2 image

Each filter is a channel

CNN – Max Pooling



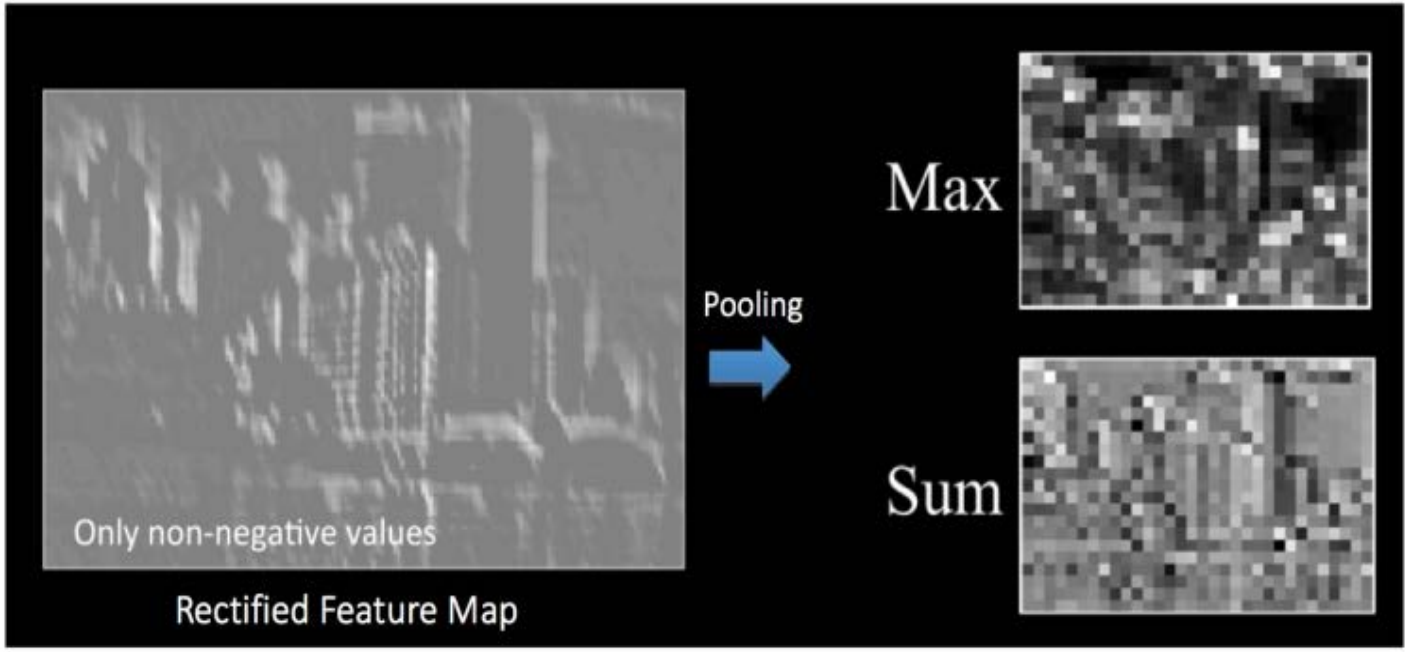


Convolved feature

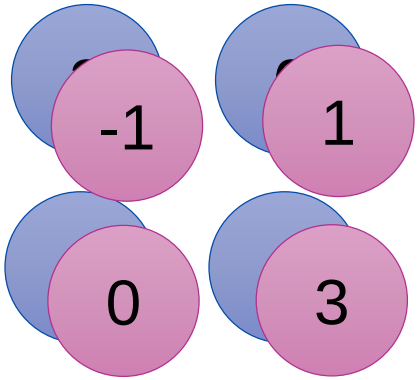
Pooled feature

Other pooling methods

- Averagy pooling
- Sum pooling



The whole CNN



A new image

Convolution

Max Pooling

Convolution

Max Pooling

Can repeat many times

Smaller than the original image

The number of the channel is the number of filters

The whole CNN

cat dog ……

Convolution

Max Pooling

A new image

Convolution

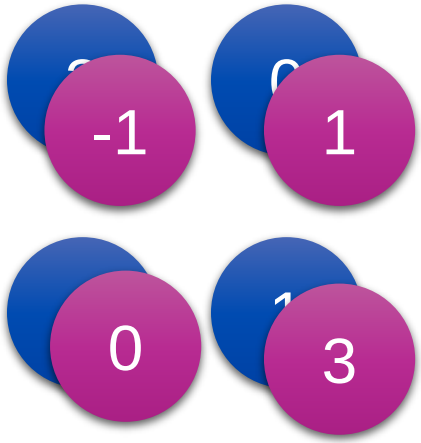
Max Pooling

A new image

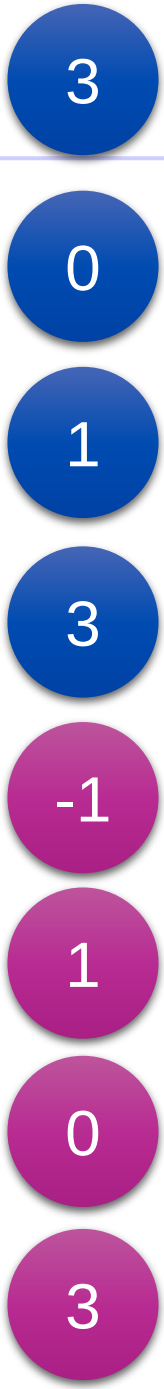
Flatten

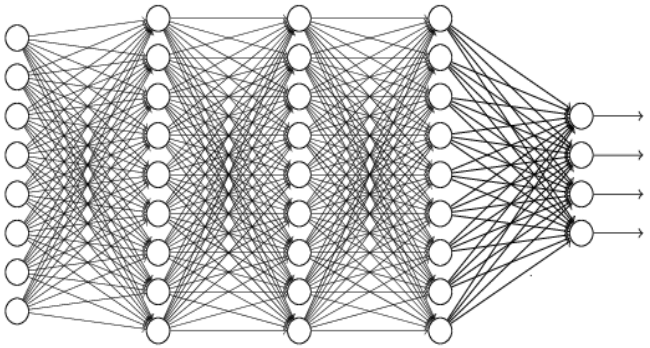
Fully Connected Feedforward network

Flatten



Flatten





Fully Connected Feedforward network

Deeper is Better?

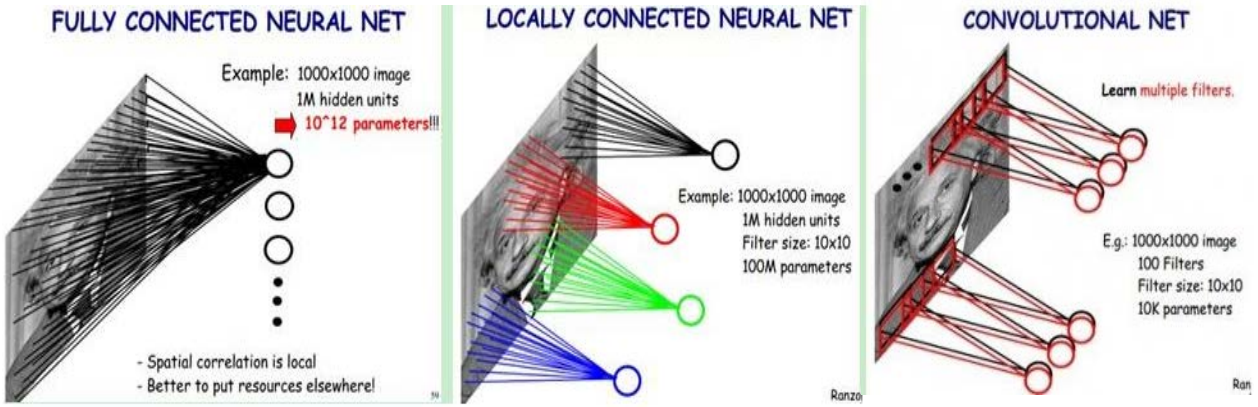
Layer X Size	Word Error Rate (%)
1 X 2k	24.2
2 X 2k	20.4
3 X 2k	18.4
4 X 2k	17.8
5 X 2k	17.2
7 X 2k	17.1

Not surprised, more parameters, better performance

Seide, Frank, Gang Li, and Dong Yu. "Conversational Speech Transcription Using Context-Dependent Deep Neural Networks." *Interspeech*. 2011.

CNN vs. DNN

- CNN相比于DNN优势
  - 更少的参数

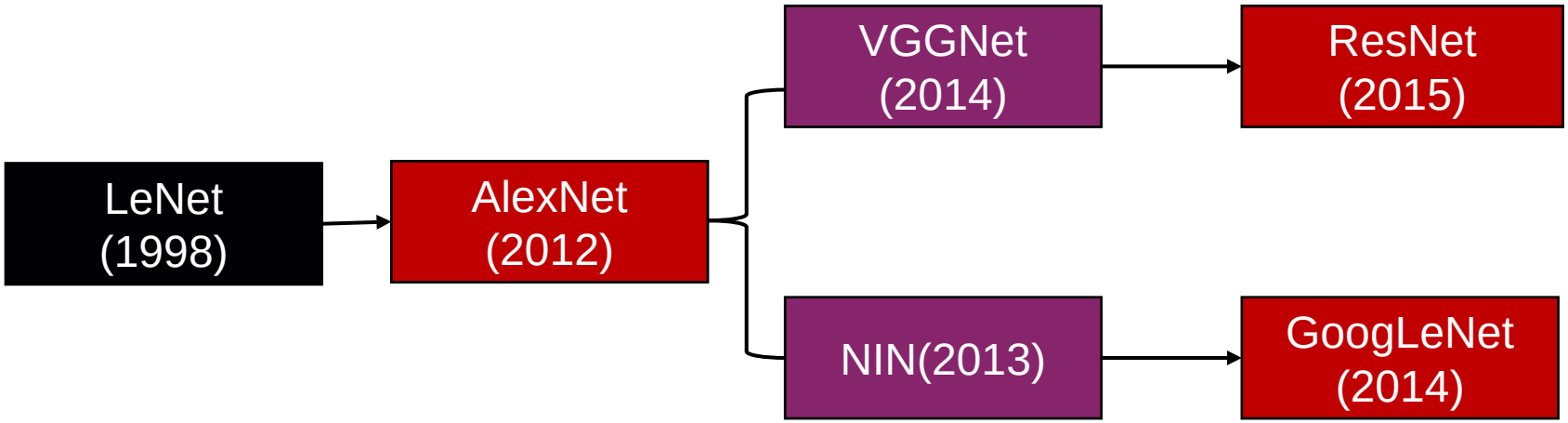


  - 局部感受野与局部相关性 (**important!**)

03 网络结构演化

单击此处添加文本具体内容

现代CNN模型结构

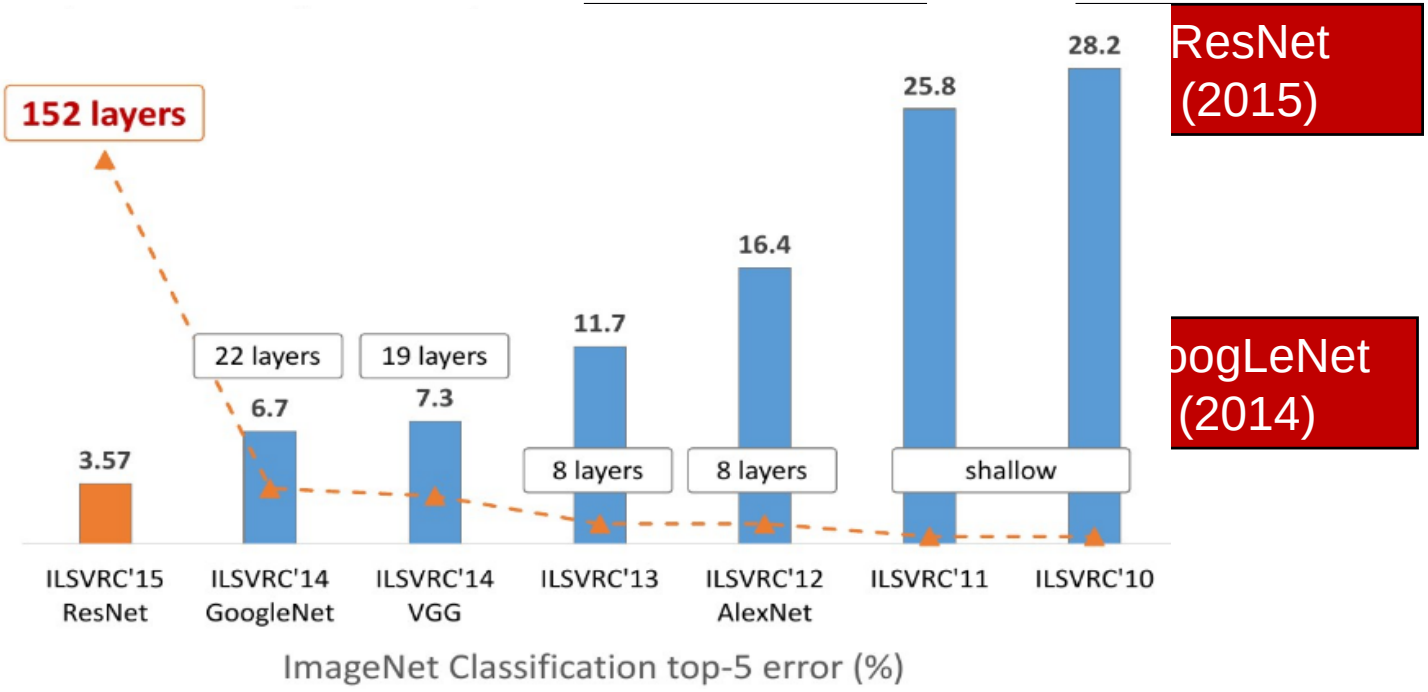

LeNet
(1998)
AlexNet
(2012)
VGGNet
(2014)
ResNet
(2015)
NIN(2013)
GoogLeNet
(2014)

现代CNN模型结构

LeNet (1998)

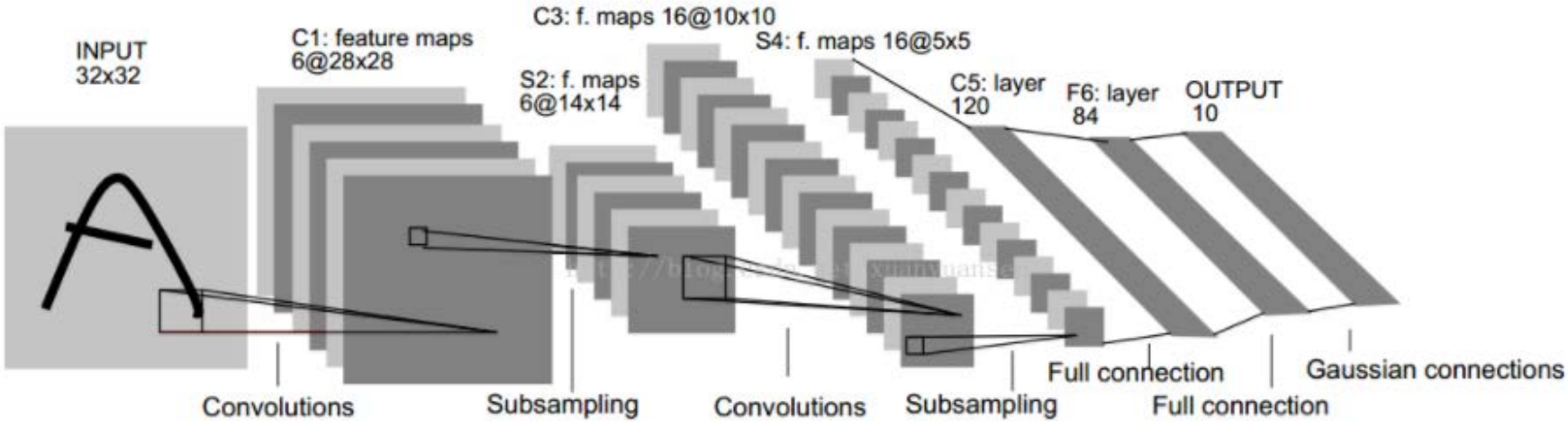
ResNet (2015)

oogLeNet (2014)



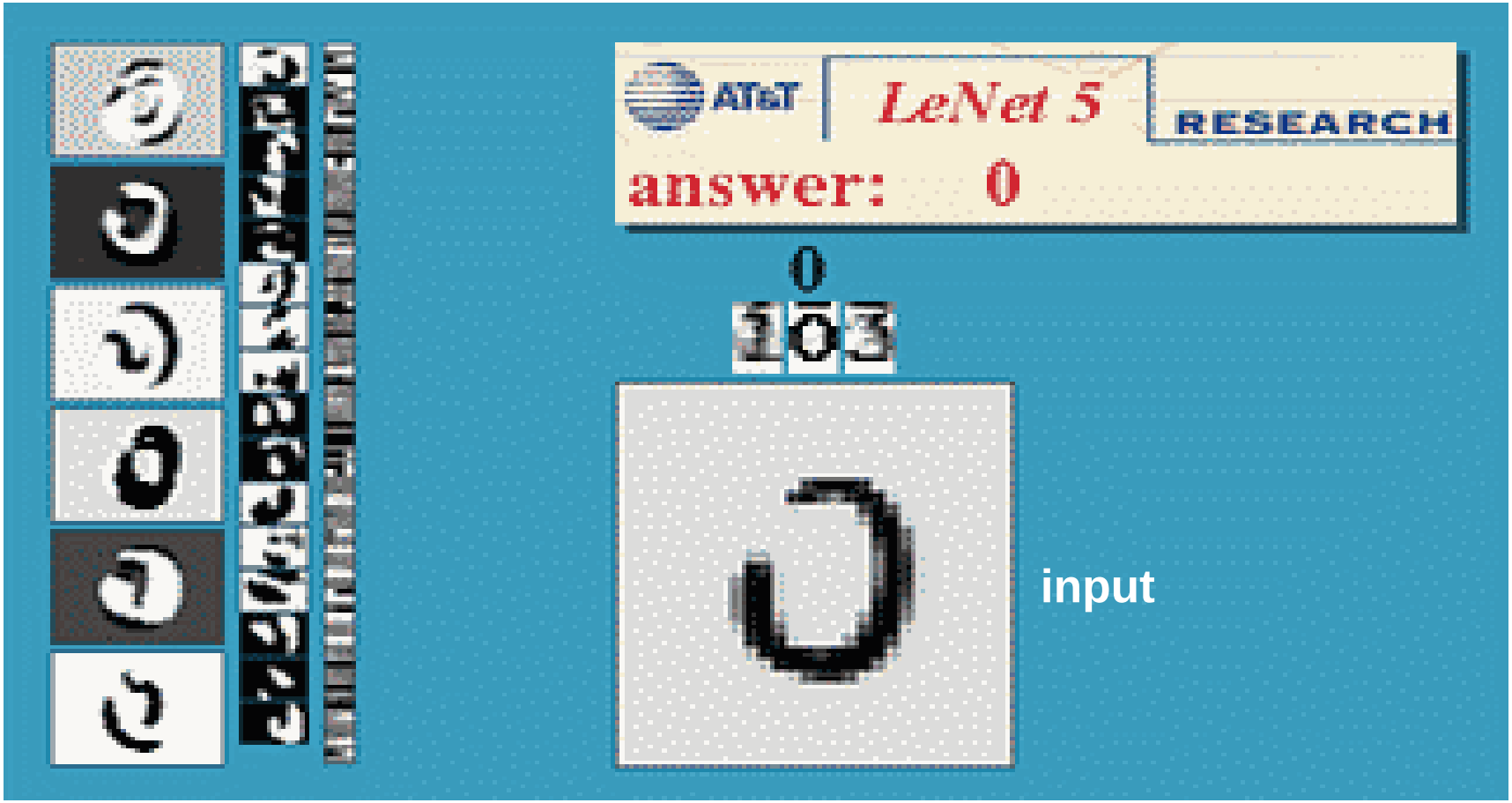
LeNet5

- LeNet-5, convolutional neural Networks
  http://yann.lecun.com/exdb/lenet/index.html



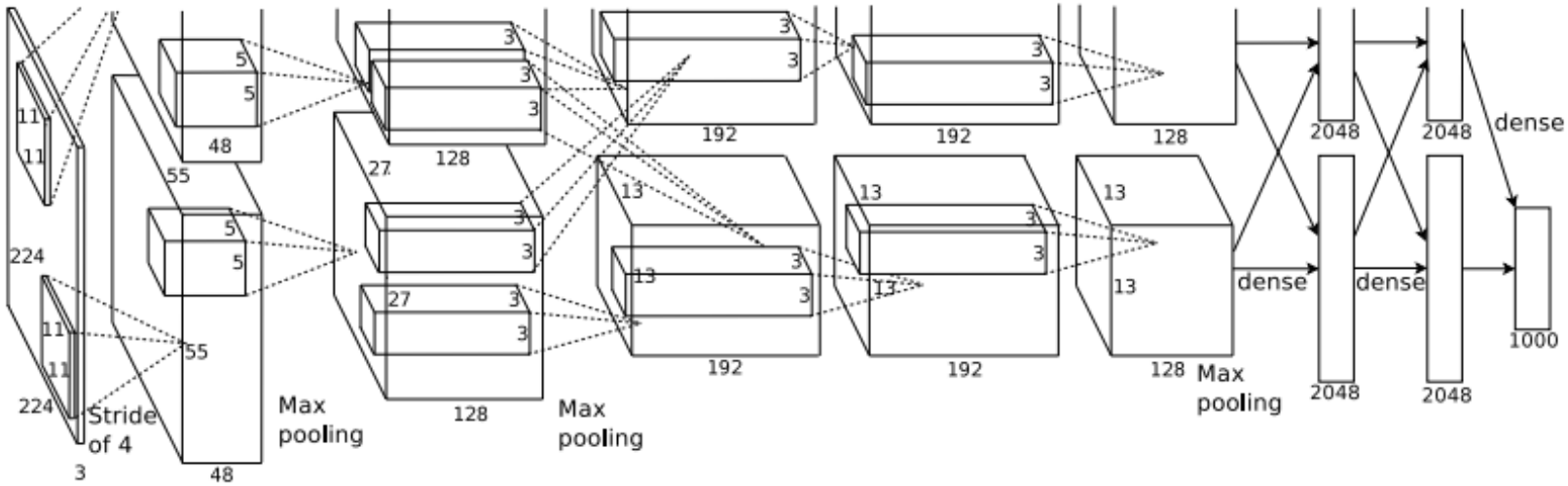
LeNet5

- MNIST数据集



AlexNet(2012)

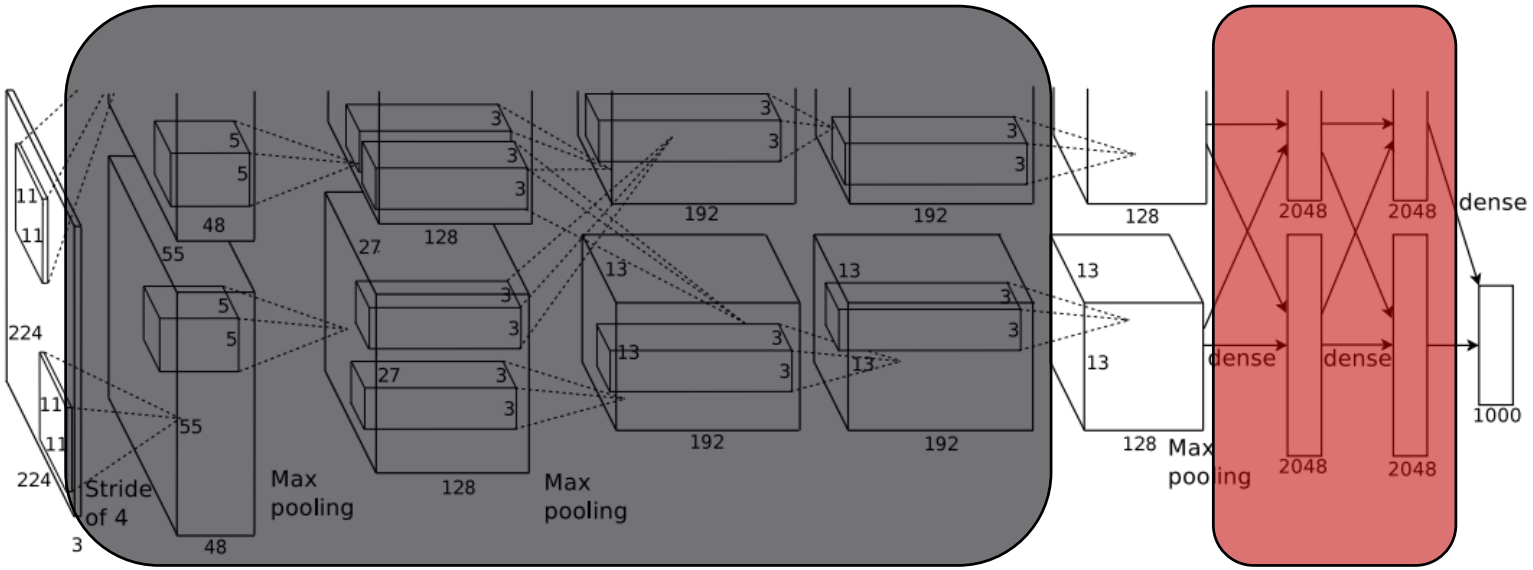
- 使用百万规模完全标准尺寸的图像数据集ImageNet训练
- 设计大规模CNN网络从原始图像训练端到端分类模型
- 使用GPU大大加速CNN模型训练
- ImageNet数据集上top-5正确率比之前最好方法提升9%



Alexnet(2012)

- 模型总参数量：6千万参数

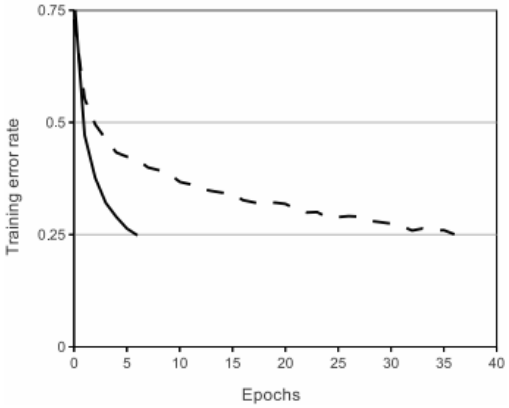
卷积层（5%的参数，
95%的计算量）

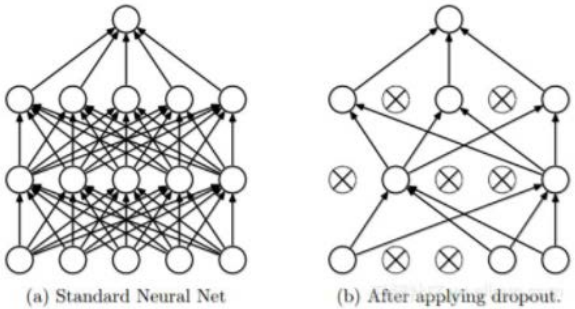


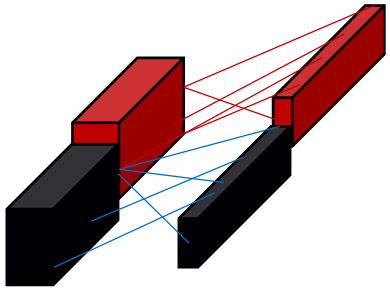
全连接层
（95%的参数，
5%的计算量）

AlexNet(2012)

- 关键技术
  - 使用ReLU激活函数代替传统sigmoid和tanh
    - 相比tanh，收敛速度提升6倍
  - 引入Dropout层防止过拟合
    - 在每个mini-batch的前向传播过程中，将全连接层的输出特征以一定概率将特征随机置为0
  - 引入Group convolution减少计算量和显存占用
    - 卷积层之间输入输出通道之间局部连接





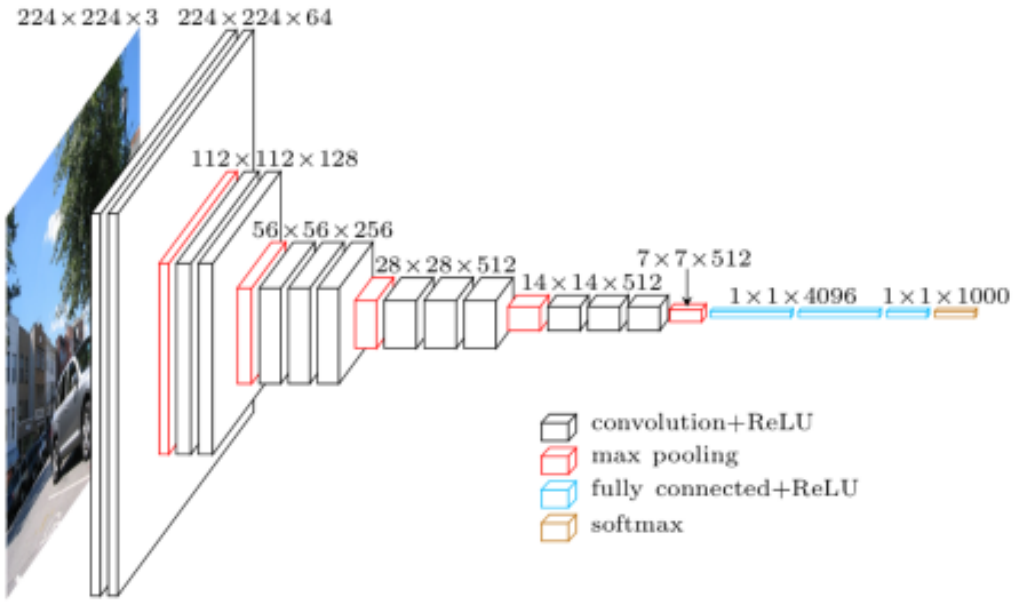


NIN(2013)

- **提出一种Network in Network的思想，使用非线性的MLP卷积代替传统卷积**
  - 实现中使用1x1 conv + ReLU+conv来实现MLPconv。1x1 conv后来成为CNN中标准模块
- **提出使用global average pooling代替fc层，大大减少参数量**
  - AlexNet中全连接层占总参数量95% (6千万)
  - NiN参数量：750万，约为AlexNet的1/8

VGGNet(2014)

- ImageNet竞赛 2014年亚军方案
- 使用连续的3x3小卷积代替AlexNet中7x7的大卷积，控制参数量的同时网络加深。3x3卷积成为后来CNN结构中的标配
- 设计16和19层的VGG-16和VGG-19网络，成为后来的各种计算机视觉任务的基准模型



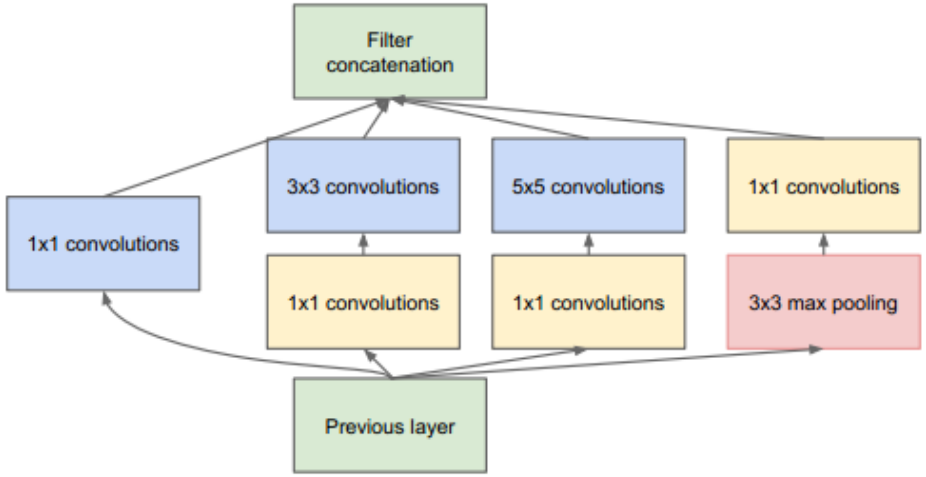
VGGNet(2014)

- ImageNet竞赛 2014年亚军方案
- 使用连续的3x3小卷积代替AlexNet中7x7的大卷积，控制参数量的同时网络加深。3x3卷积成为后来CNN结构中的标配
- 设计16和19层的VGG-16和VGG-19网络，成为后来的各种计算机视觉任务的基准模型

ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

GoogLeNet(2014)

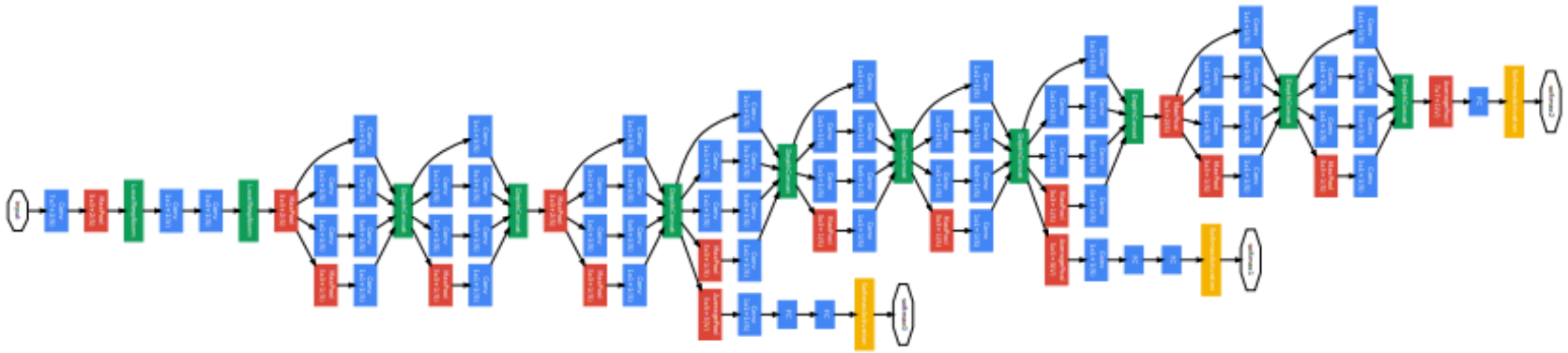
- ImageNet竞赛 2014年冠军方案，来自google团队，致敬LeNet
- 23层网络，深度上超过之前所有模型，单参数相比VGG模型小很多 (5 million vs. 144 million)
- Inception模块化设计，不同尺度特征图串联，形成更强的表达能力
  - 引入NIN 中1x1 conv，用来降维，降低计算量



(b) Inception module with dimension reductions

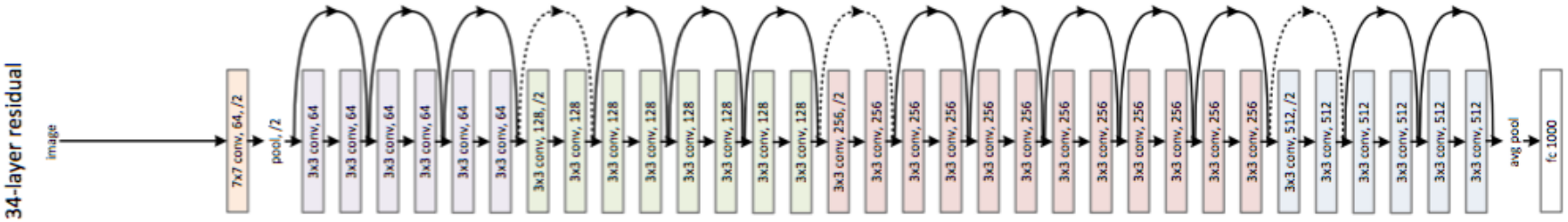
GoogLeNet(2014)

- ImageNet竞赛 2014年冠军方案，来自google团队，致敬LeNet
- 23层网络，深度上超过之前所有模型，单参数相比VGG模型小很多(5 million vs. 144 million)
- Inception模块化设计，不同尺度特征图串联，形成更强的表达能力
  - 引入NIN 中1x1 conv，用来降维，降低计算量



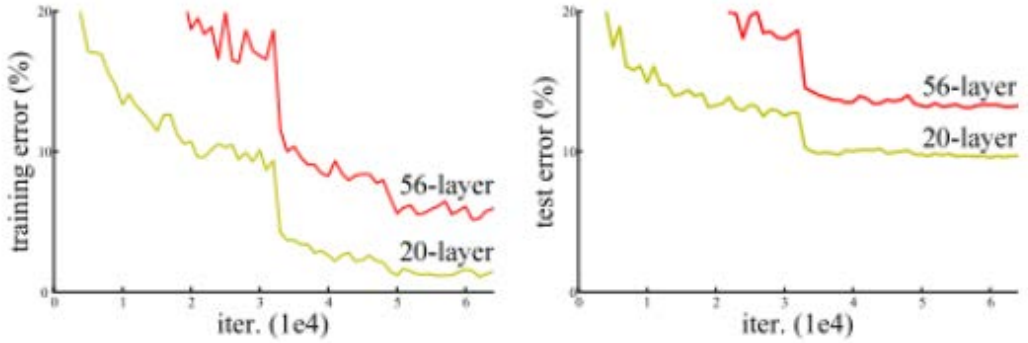
ResNet (2015)

- ImageNet竞赛 2015年冠军方案，CVPR 2016 best paper。图像分类识别率超过人类水平
- 提出152层的CNN结构，大大超越之前最深的23层GoogLeNet网络
- 提出“残差学习”思想，使用简单的“跳层”连接（shortcut）大大缓解深度网络训练时的梯度消失问题

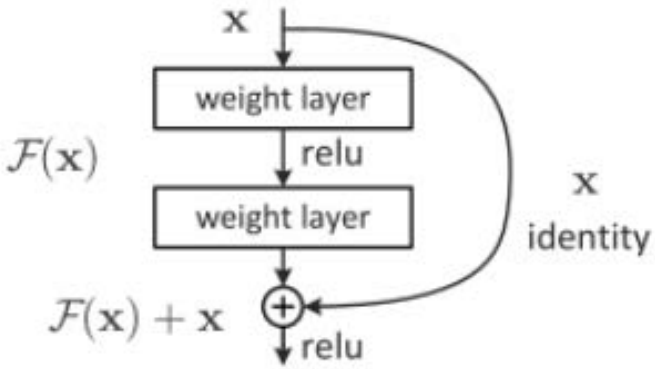


ResNet (2015)

- 动机：深度对识别效果影响很大，理论上越深的网络效果应该越好。实际上简单加深时效果反而变差，主要原因是梯度消失（弥散）



- 解决方案：使用跨层的连接，方向传播时部分梯度可以绕过residual path，直接沿着identity path跳过若干层，回到输入层，从而在网络很深时也可以避免梯度消失



04 理解CNN

单击此处添加文本具体内容

深度学习的解释性——让模型说话

- “Facebook聊天发明自己的语言”——关于对AI的信任问题
- 优步自动驾驶汽车撞死行人——关于AI的安全性问题

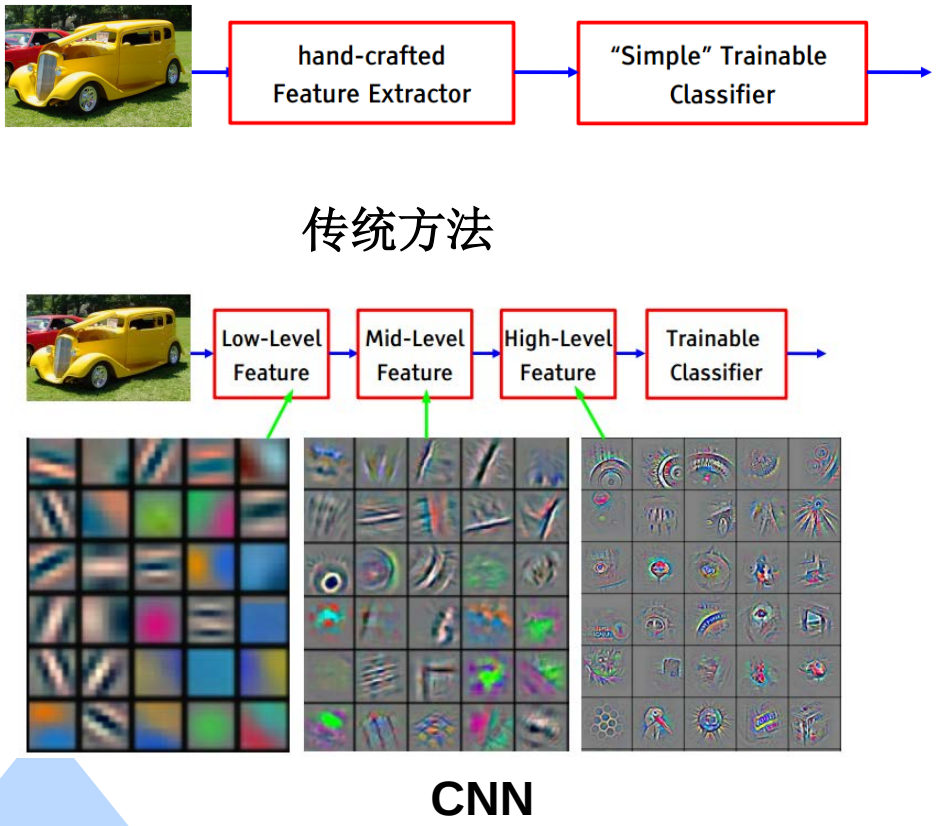
- 深思：深度学习在诸多应用中表现出高性能，但在不同应用场景中鲁棒性不尽如人意，深度学习的可解释性是否是解决这一问题的关键？

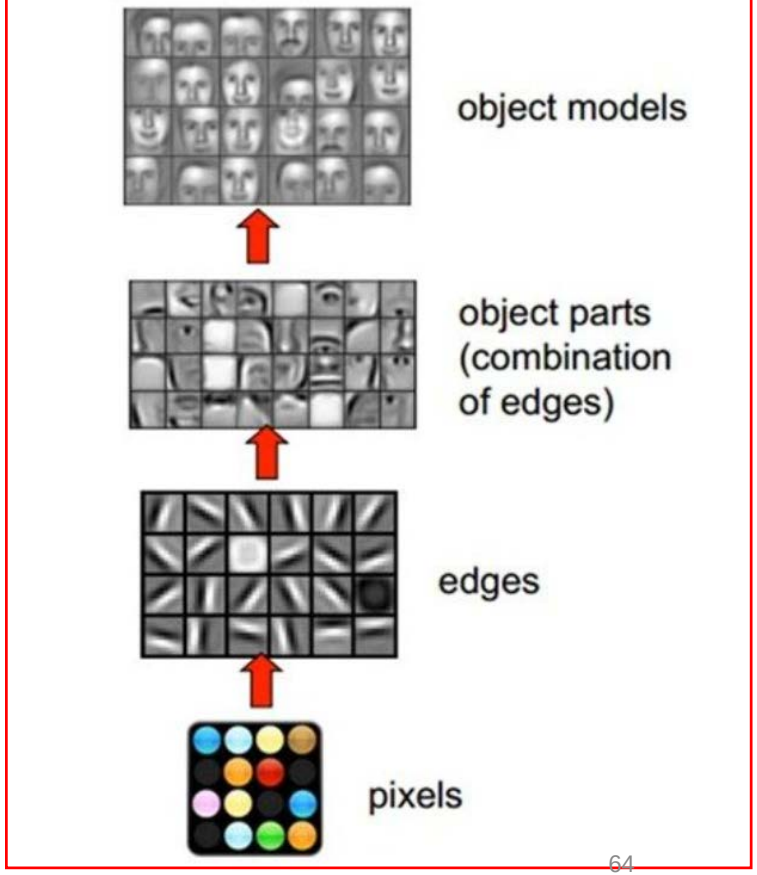
➢男性&未婚&博士&秃头的条件对应「不感兴趣」

➢CNN可视化——更加直观

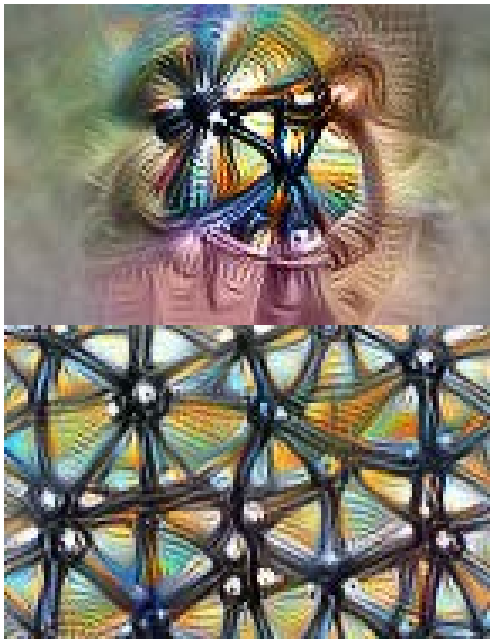
理解CNN

- CNN以图像的原始像素作为输入，使用**多层的**可学习的卷积层和池化层逐层对图像学习复杂的非线性变换，基于输出层定义的损失函数使用反向传播算法**端到端(End-to-end)**学习，从而自动学习得到图像**底层到高层的层次化语义表达.**



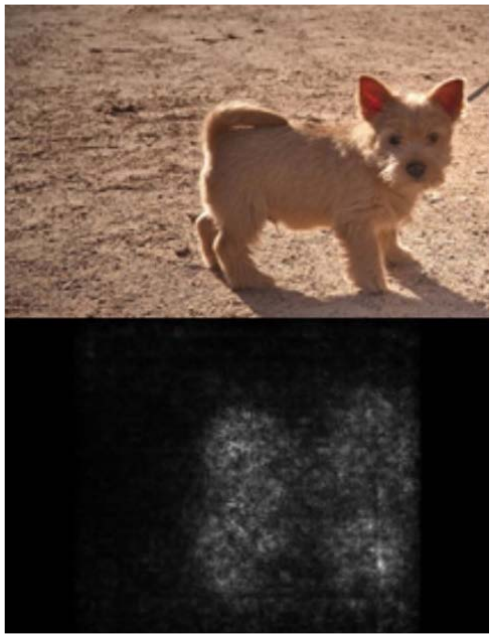


CNN可视化



Feature visualization

通过生成样本
确定神经网络对什么产生响应

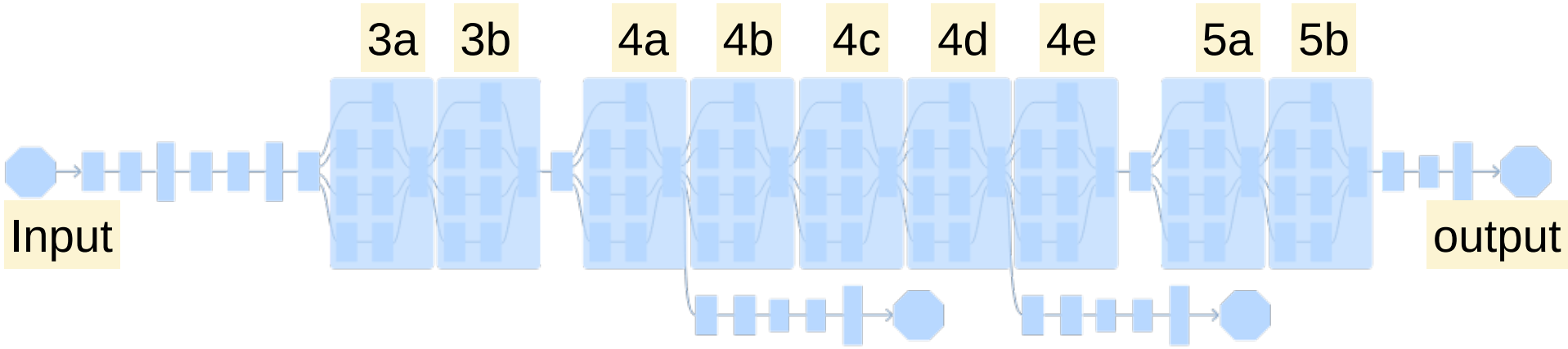


Attribution

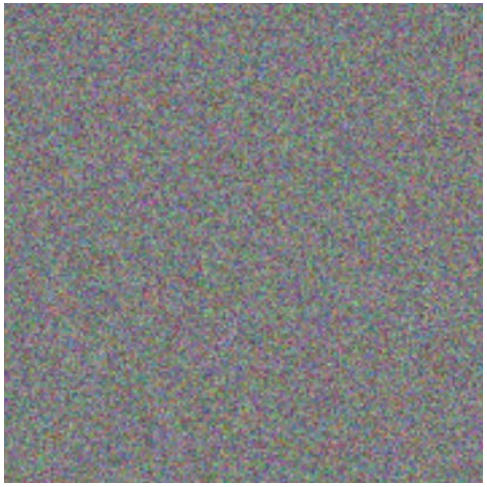
研究样本中的哪一部分
会令神经网络产生响应

feature visualization

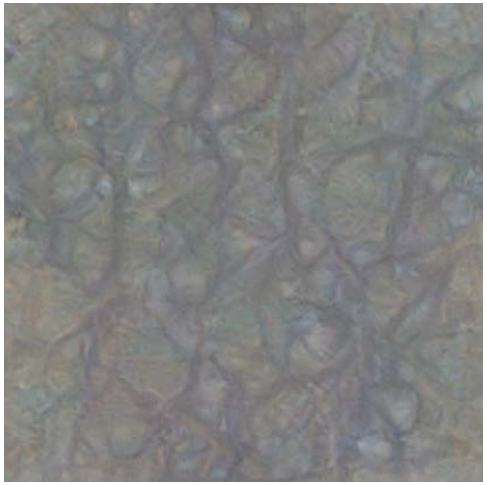
- 谷歌GoogLeNet可视化
  - https://**distill.pub**/2017/feature-visualization/



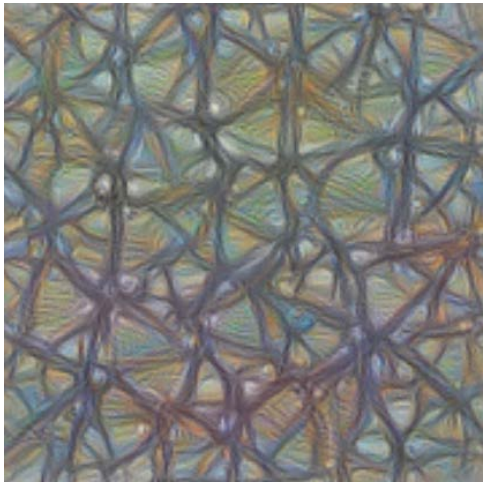
feature visualization



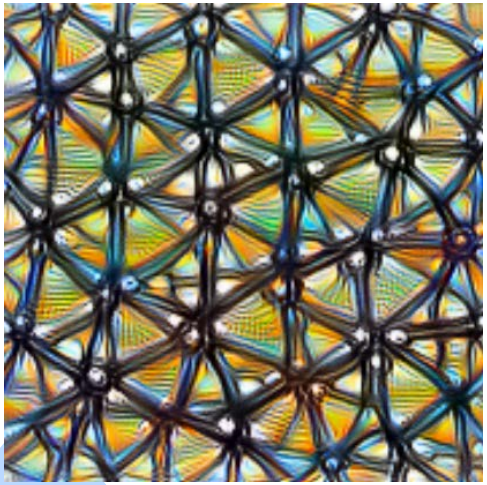
Step 0



Step 4



Step 48

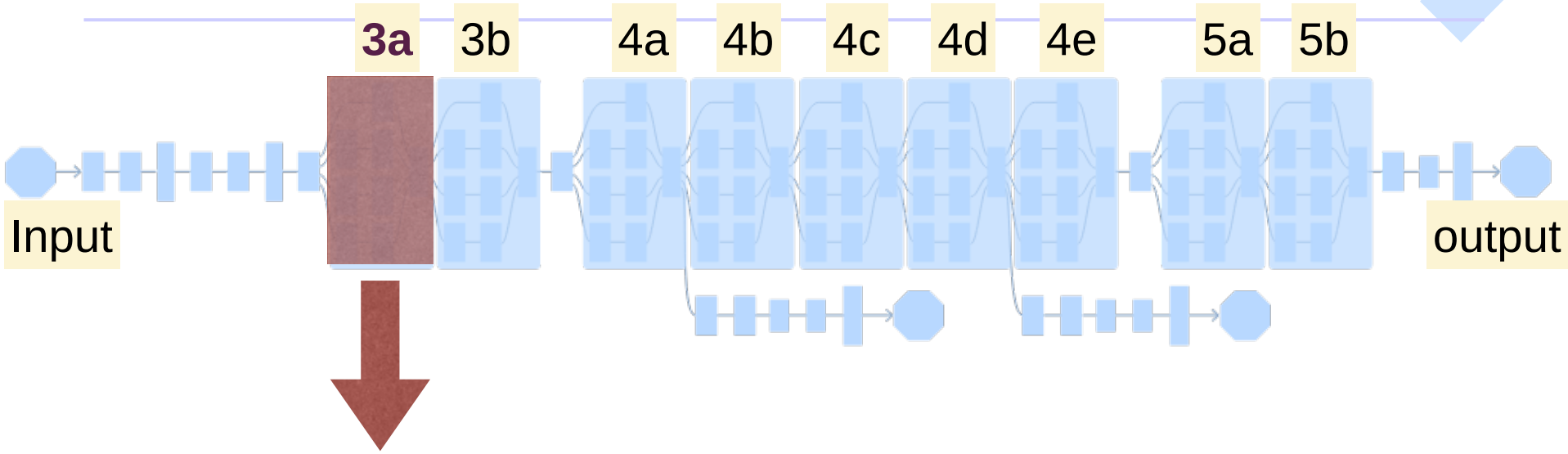


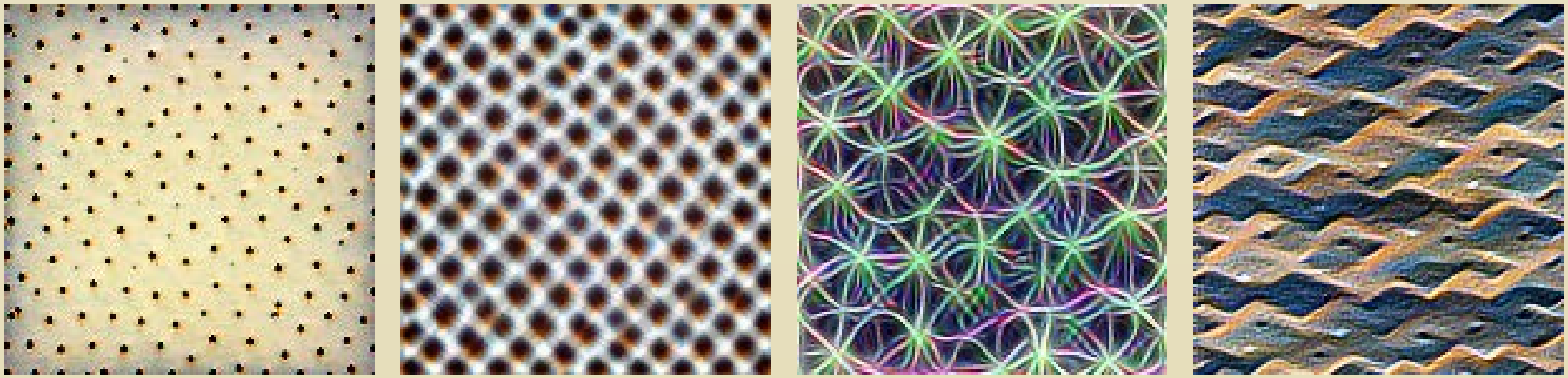
Step 2048

从随机 noise 开始，
优化 (optimize) 图片以激活特定的神经元

Olah, et al., "Feature Visualization", Distill, 2017.

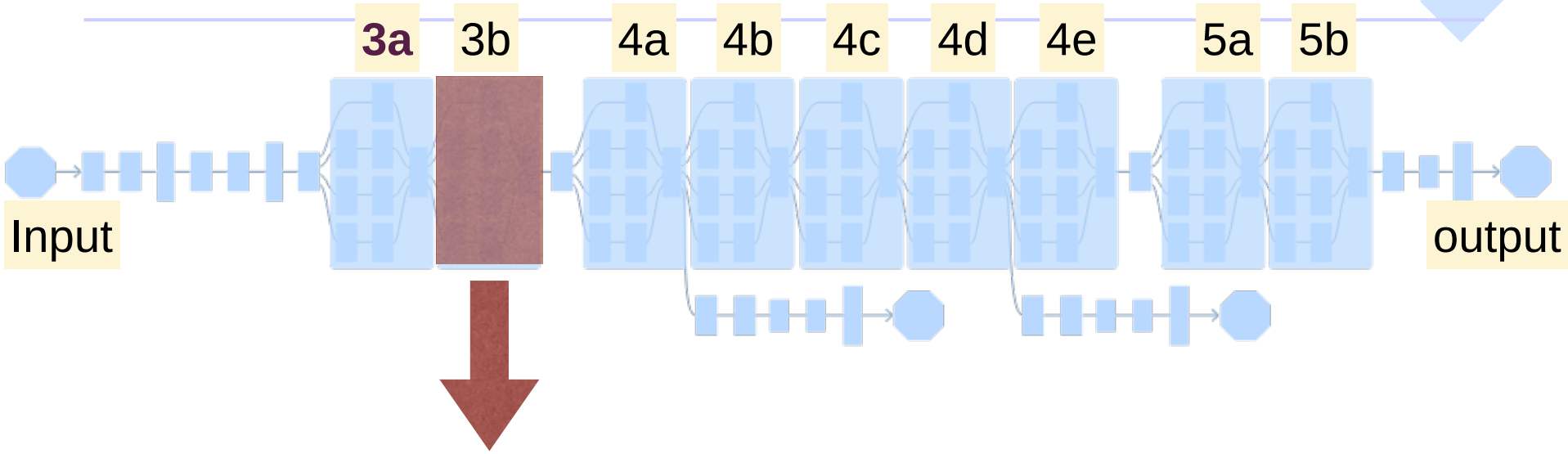
feature visualization

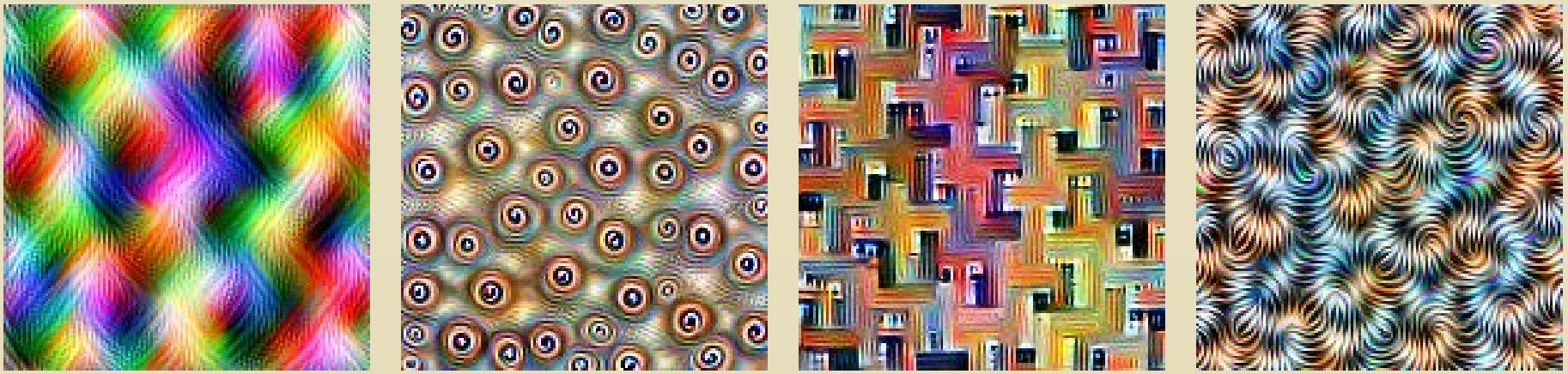




检测出了简单的纹理信息（**texture**），且都是局部（**local**）纹理

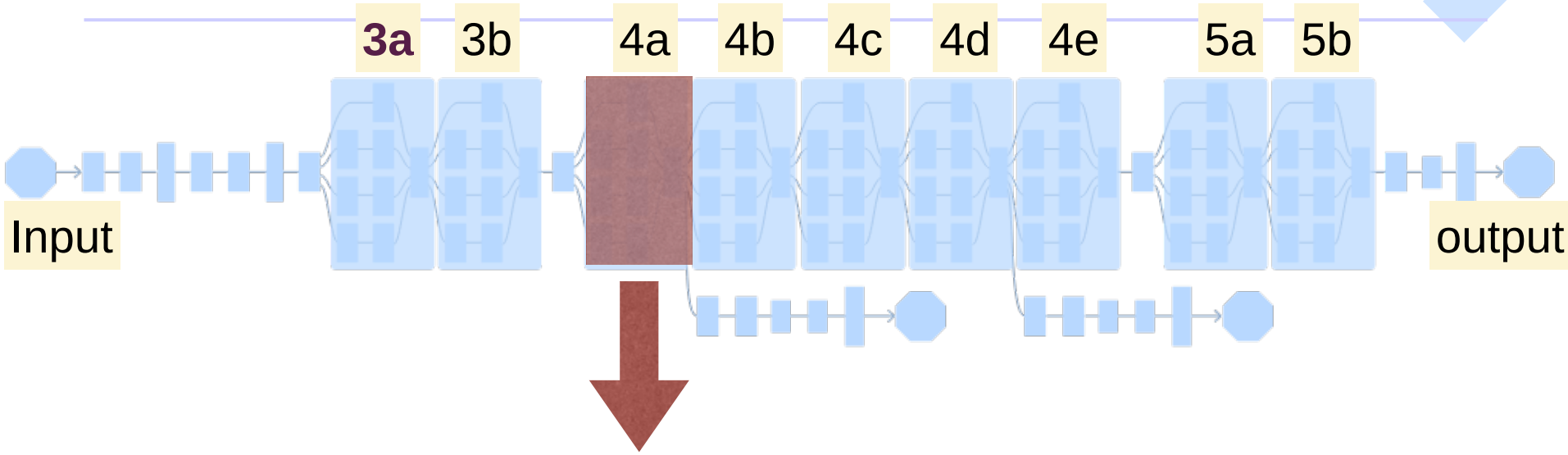
feature visualization





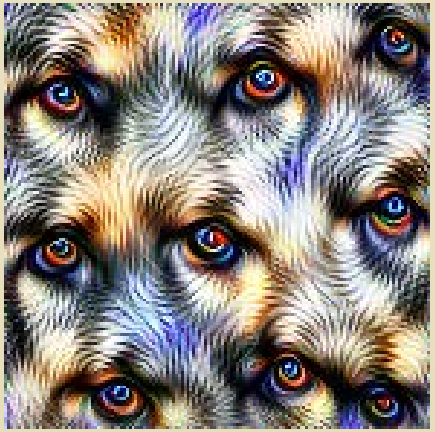
纹理开始变得复杂，仍然都是局部纹理

feature visualization

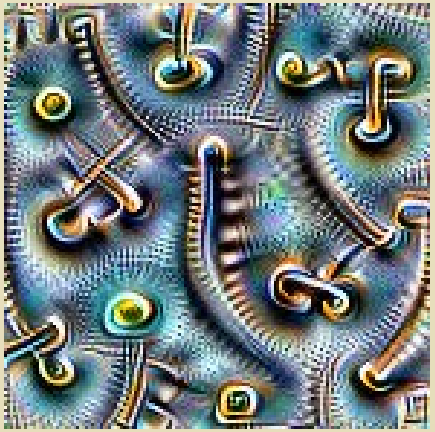


复杂度显著提升，出现了复杂的模式（pattern）

书架



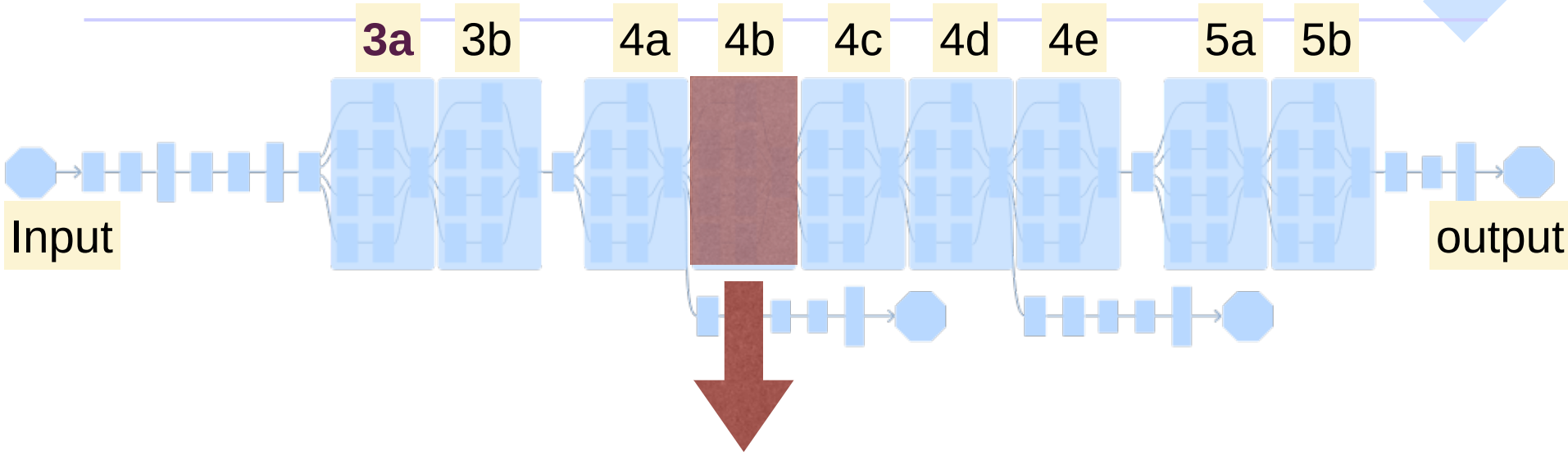
狗眼睛



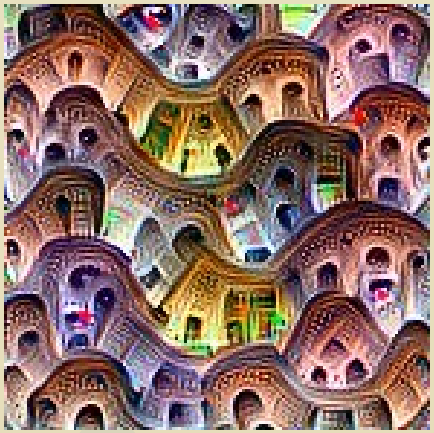
文本，柳钉

鸟

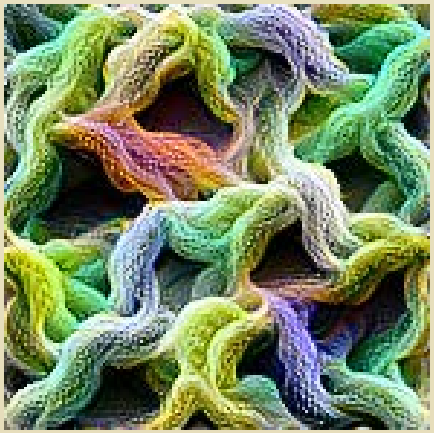
feature visualization



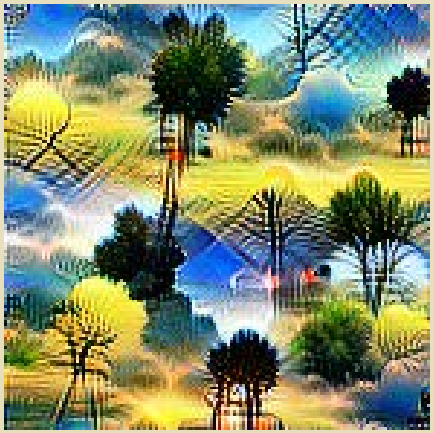
可以辨别出部分物体了，更多的 **context** 开始出现



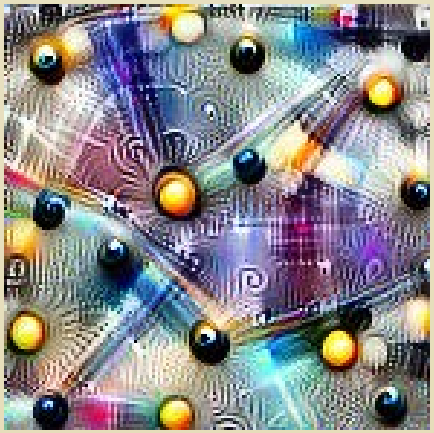
建筑结构



蓬松的绳

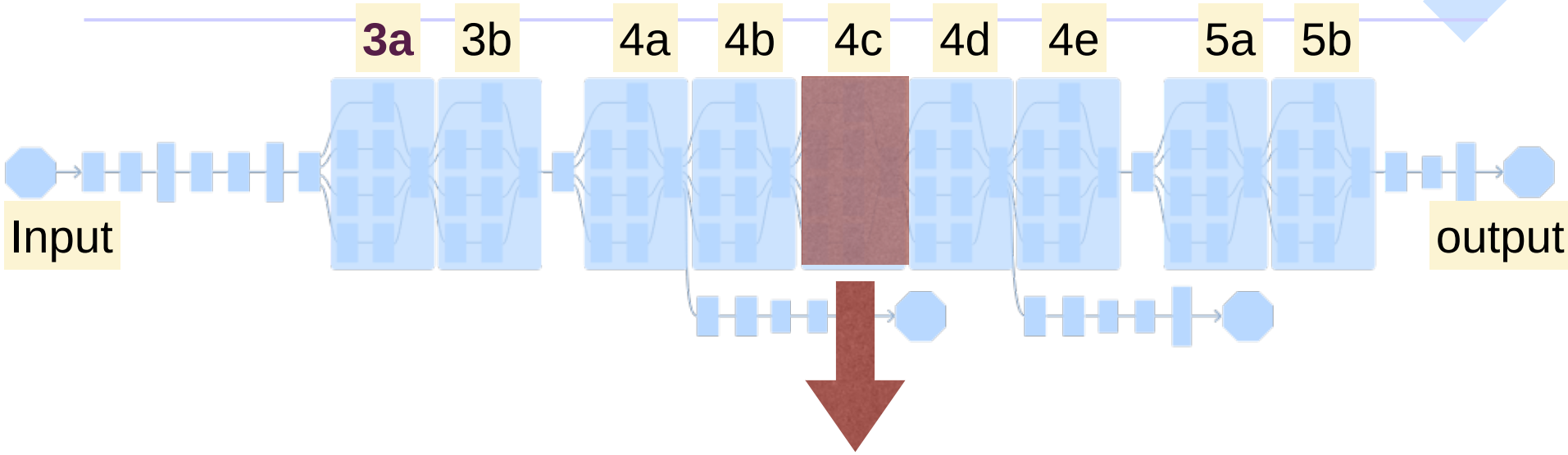


树



台球

feature visualization



网络开始响应具体的对象（这也许是最值得探索的一层）

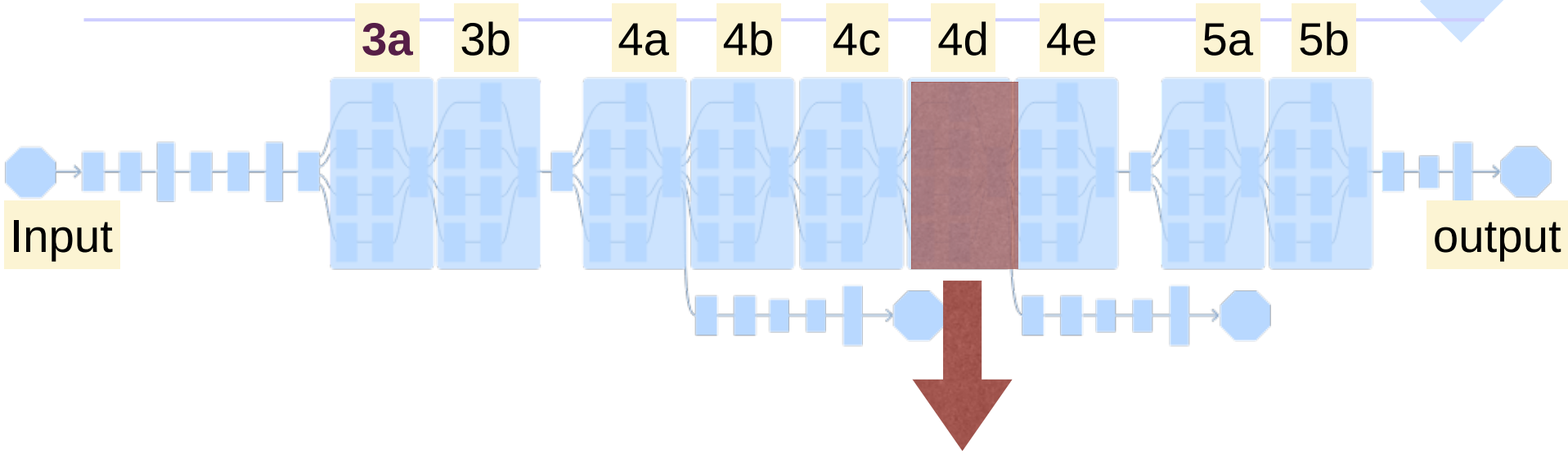
棕榈树

轮子

带项圈的狗

房子

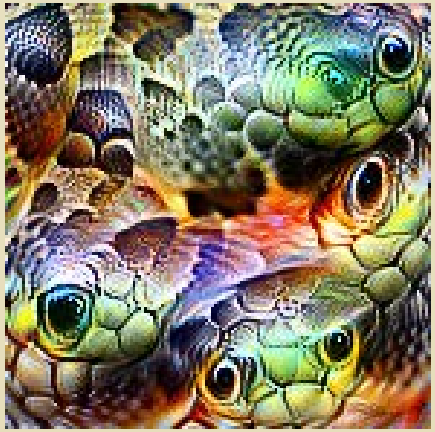
feature visualization



出现更复杂的概念，开始看到神经元对多个不相关的概念产生反应

狗嘴

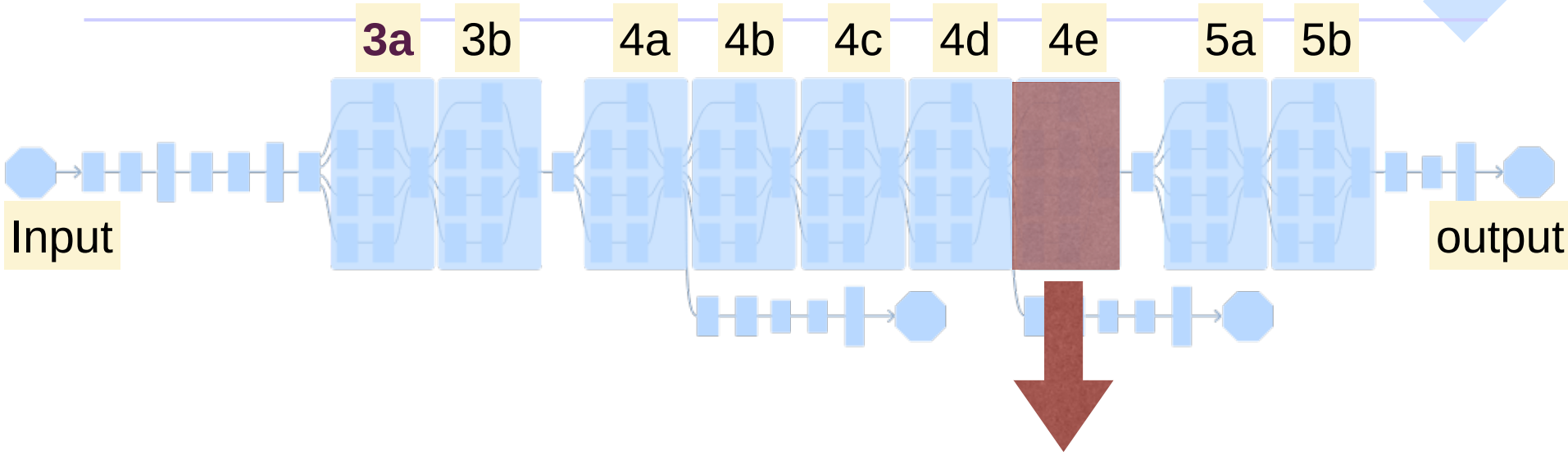
灵长目动物



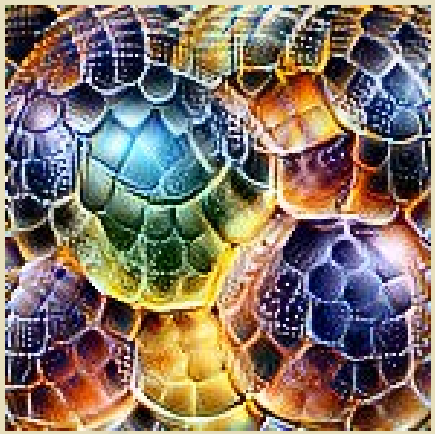
蛇头

餐厅的菜肴

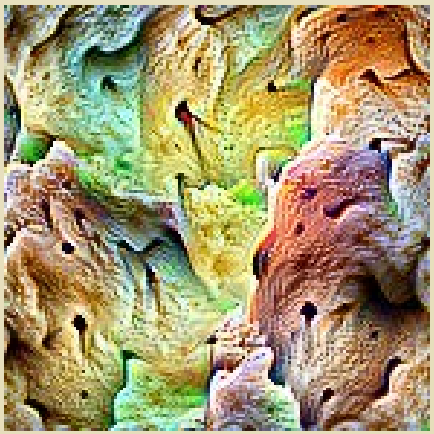
feature visualization



区分特定的物种，对多个视觉相关的概念产生响应



龟壳

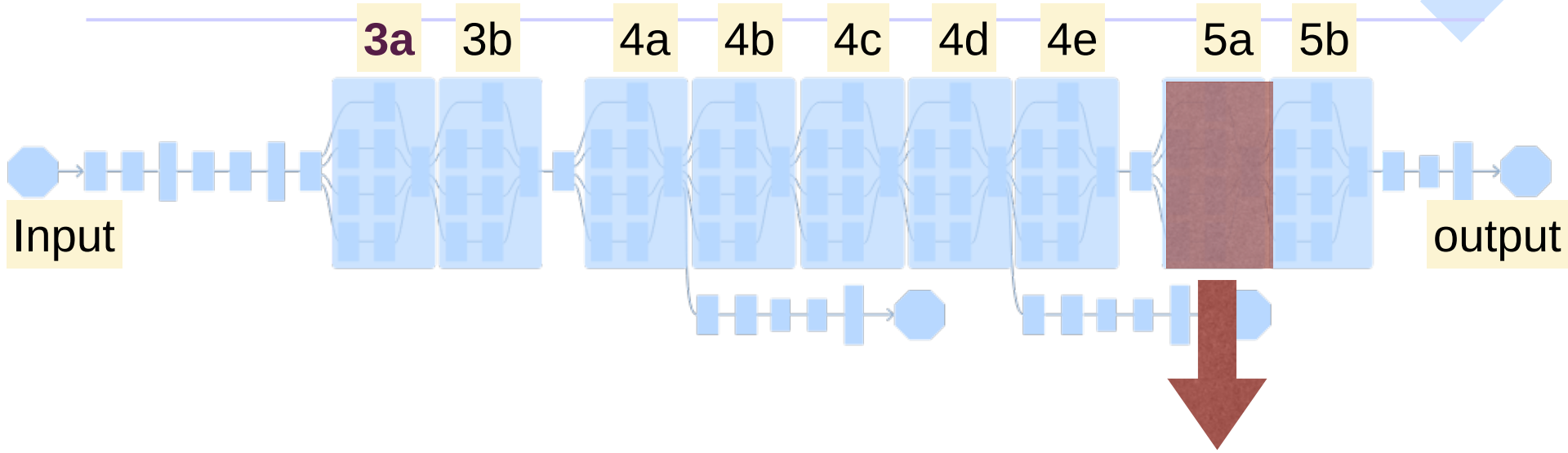


冰淇淋 & 面包

猫的皮肤

宽边帽

feature visualization

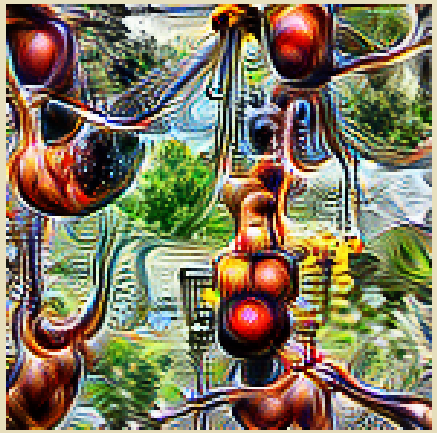


可视化开始变得抽象，但所包含的语义信息还是较为具体

糖果

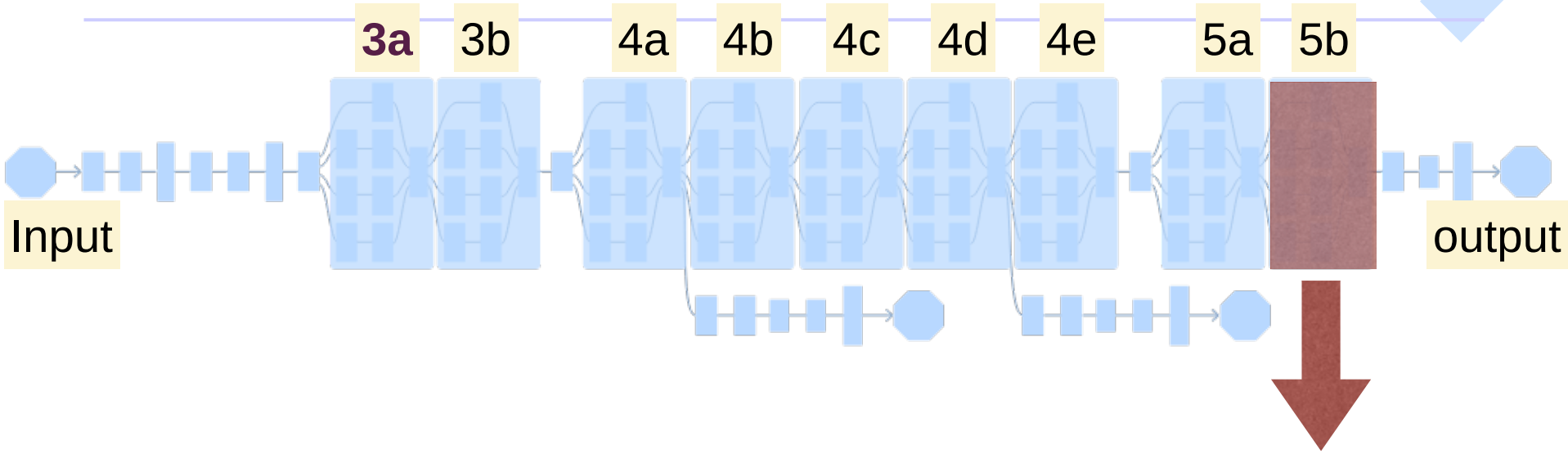
球

铜管乐器

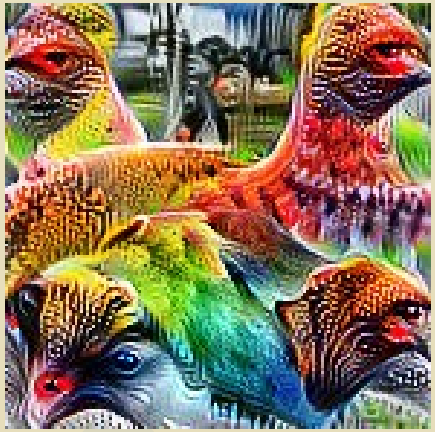
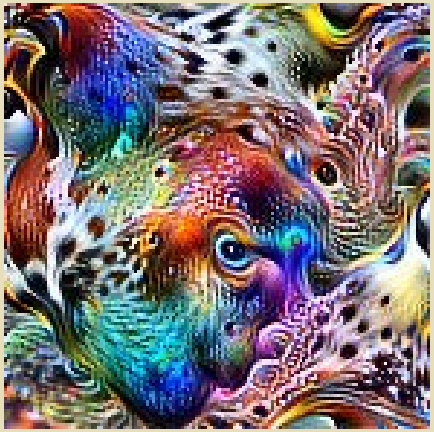


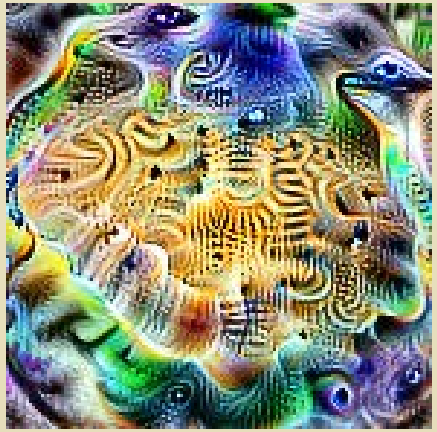
交通灯

feature visualization



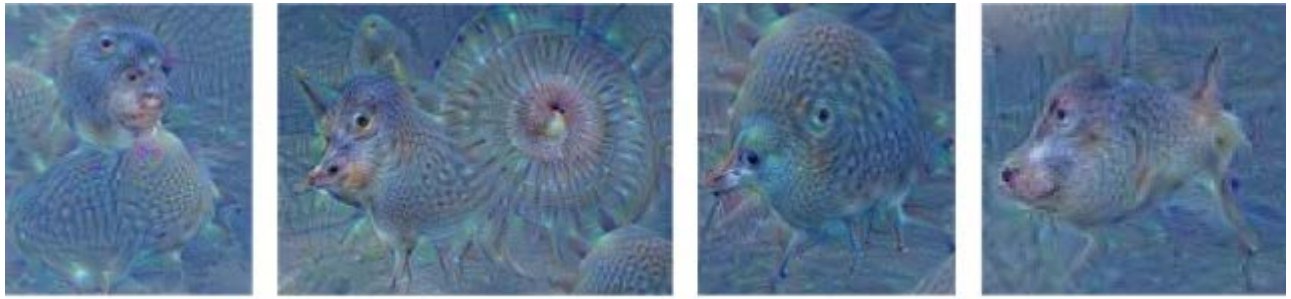
像是没有意义的大杂烩，神经元似乎不再对应特别的语义概念

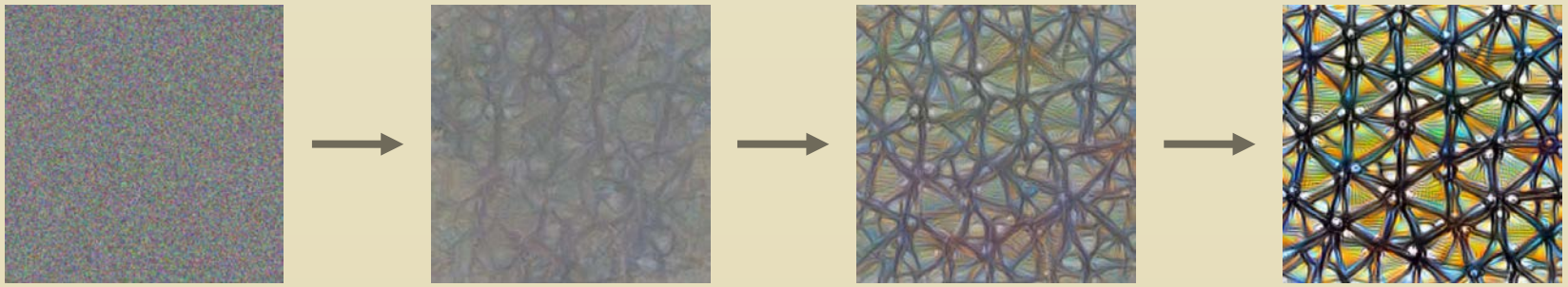





Google deepdream

- 项目地址：https://github.com/google/deepdream





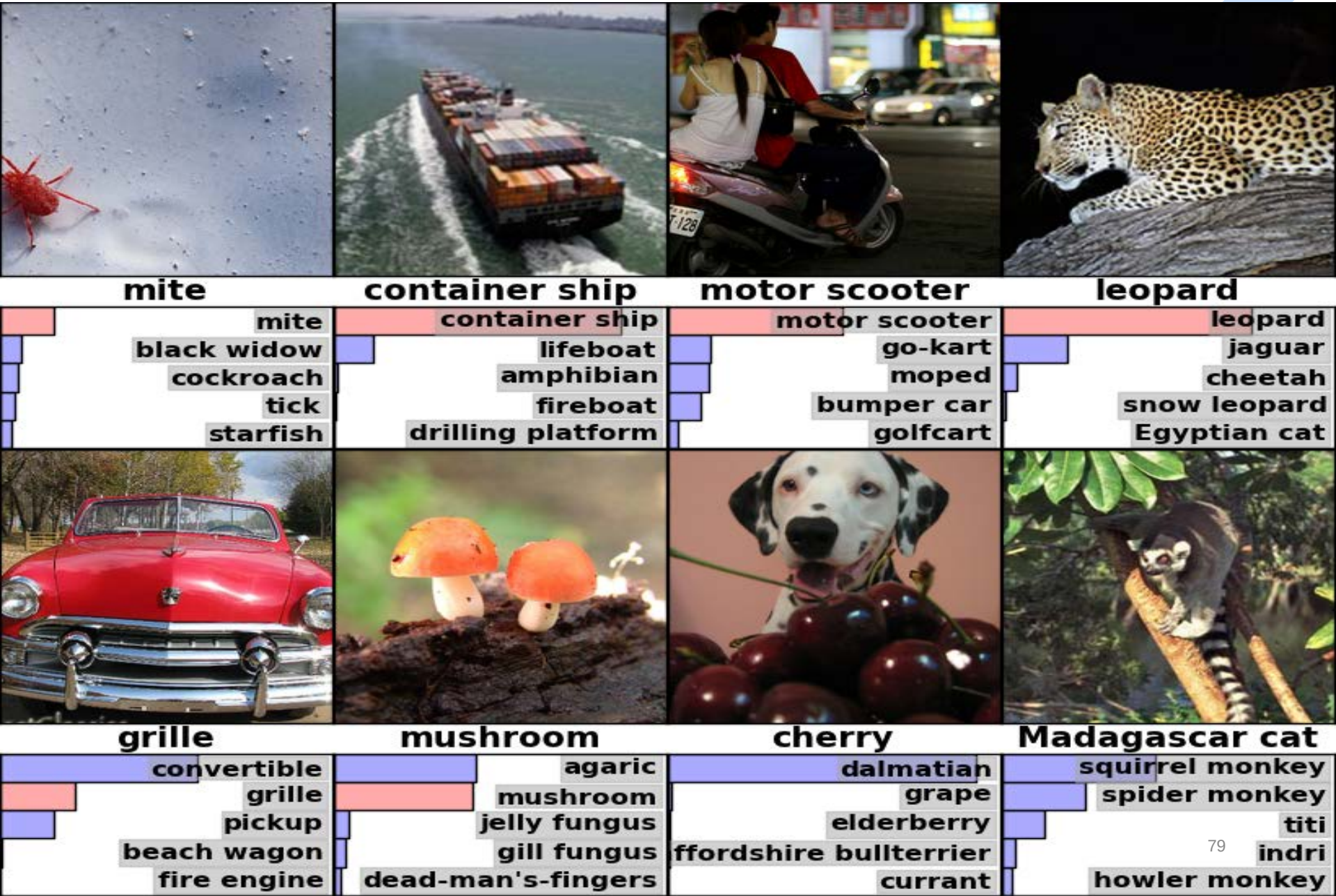
从随机 noise 开始，优化 (optimize) 图片以激活特定的神经元（网络层）

如果我们输入一张正常的图片替代噪声（noise），会发生什么？

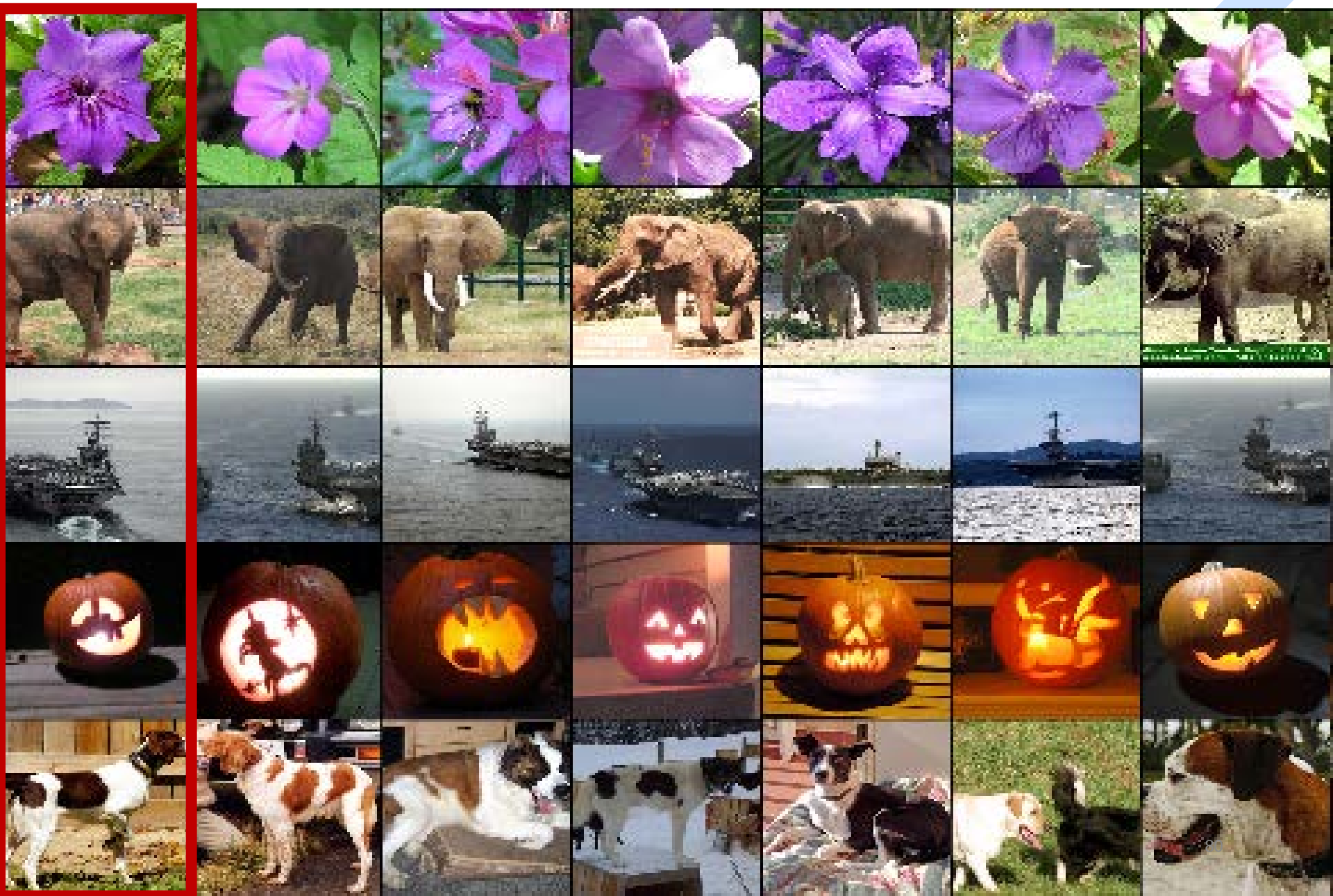
05 应用

单击此处添加文本具体内容

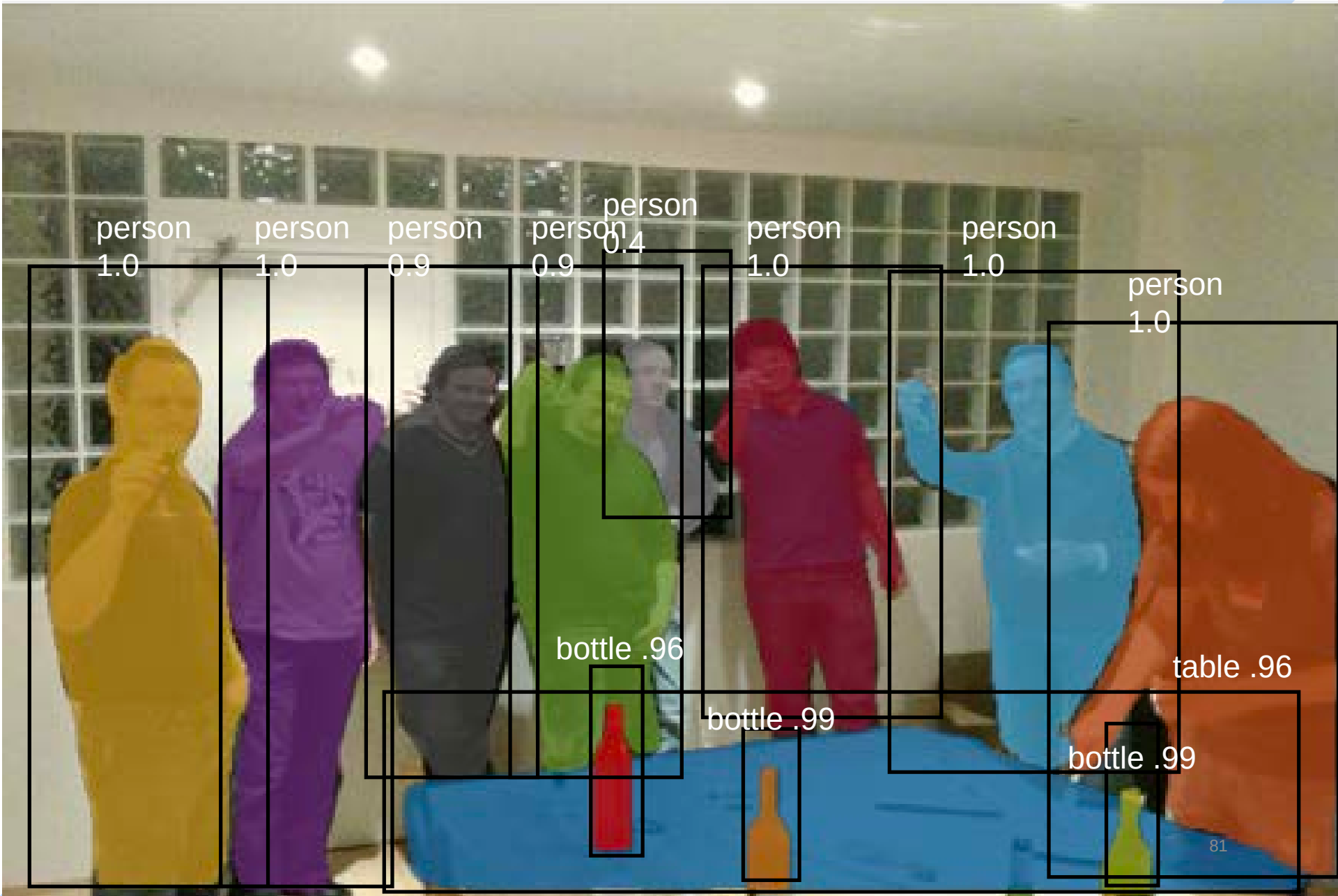
图像分类


mite
container ship
motor scooter
leopard
mite
black widow
cockroach
tick
starfish
container ship
lifeboat
amphibian
fireboat
drilling platform
motor scooter
go-kart
moped
bumper car
golfcart
leopard
jaguar
cheetah
snow leopard
Egyptian cat
grille
mushroom
cherry
Madagascar cat
convertible
grille
pickup
beach wagon
fire engine
agaric
mushroom
jelly fungus
gill fungus
dead-man's-fingers
dalmatian
grape
elderberry
ffordshire bullterrier
currant
squirrel monkey
spider monkey
titi
indri
howler monkey

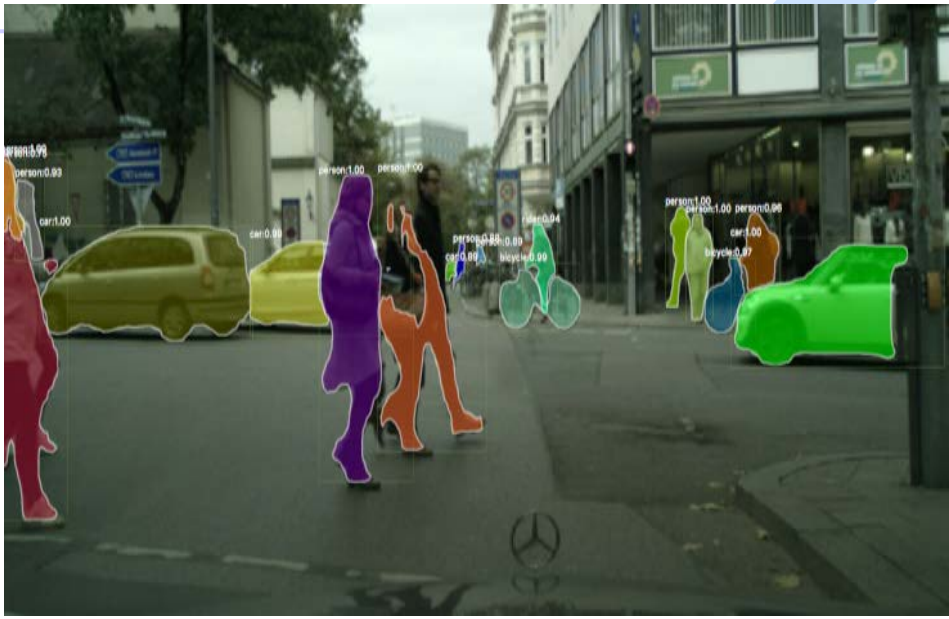
图像检索

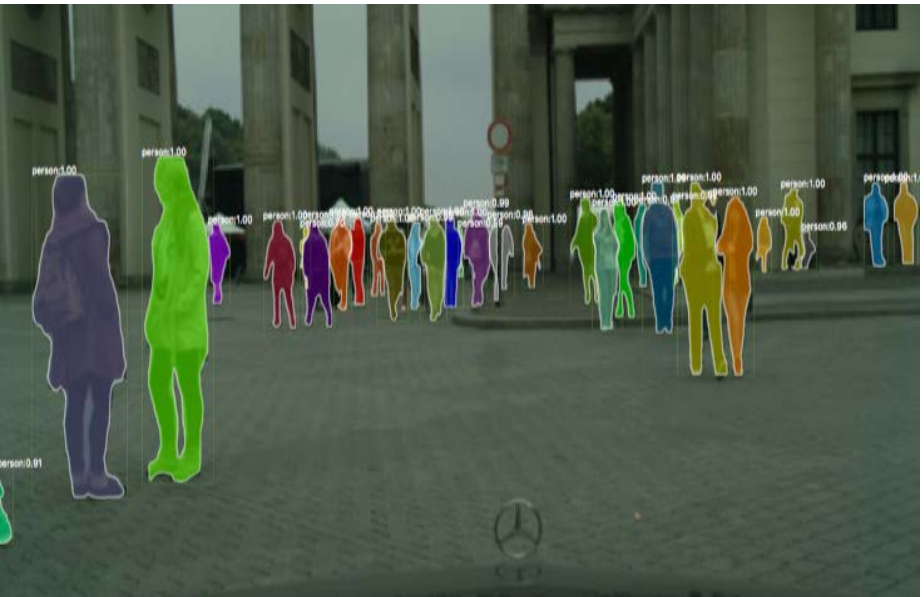


目标检测、分割

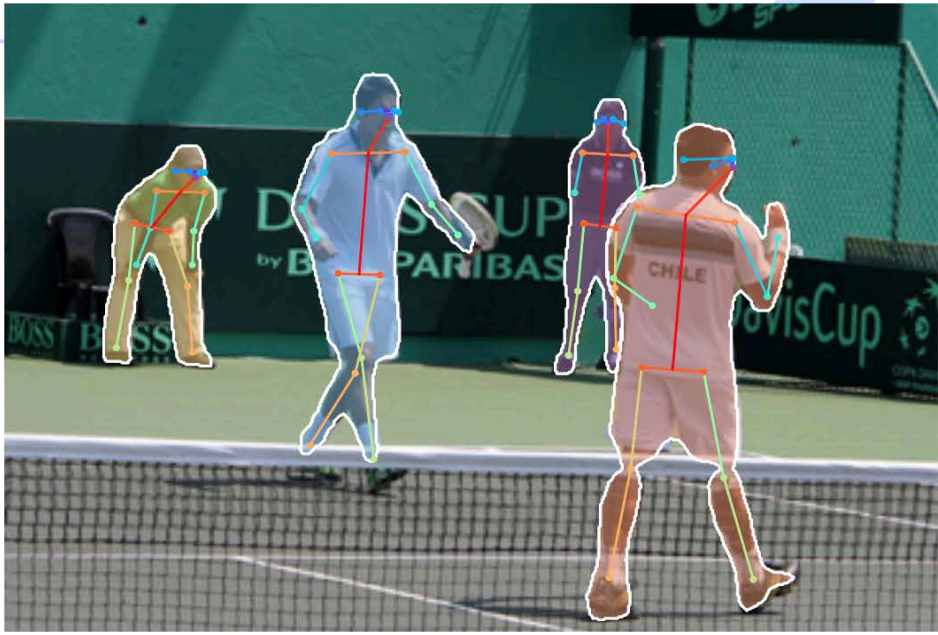

person
1.0
person
1.0
person
0.9
person
0.9
person
0.4
person
1.0
person
1.0
person
1.0
bottle .96
bottle .99
bottle .99
table .96

目标检测、分割





目标分割、关键点检测



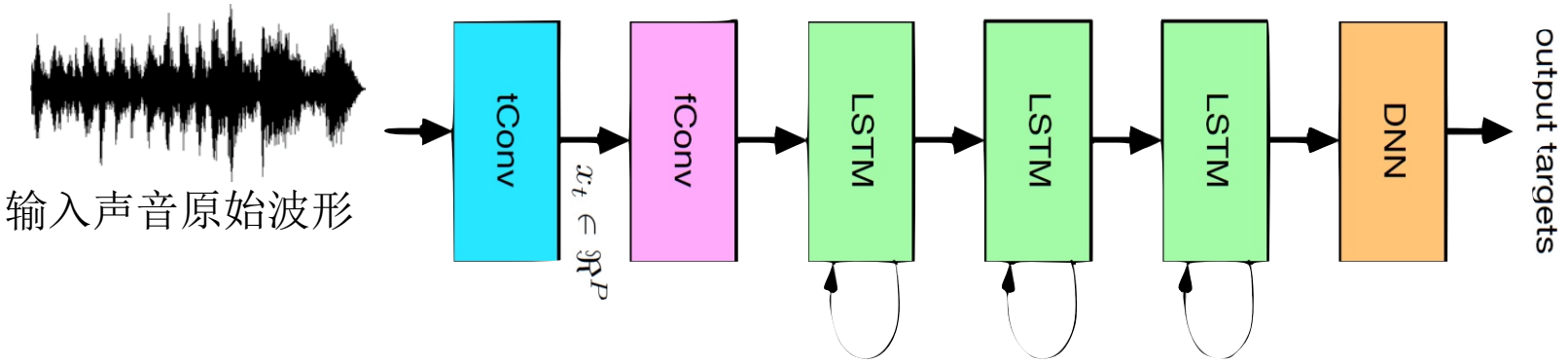
风格转换

图像生成

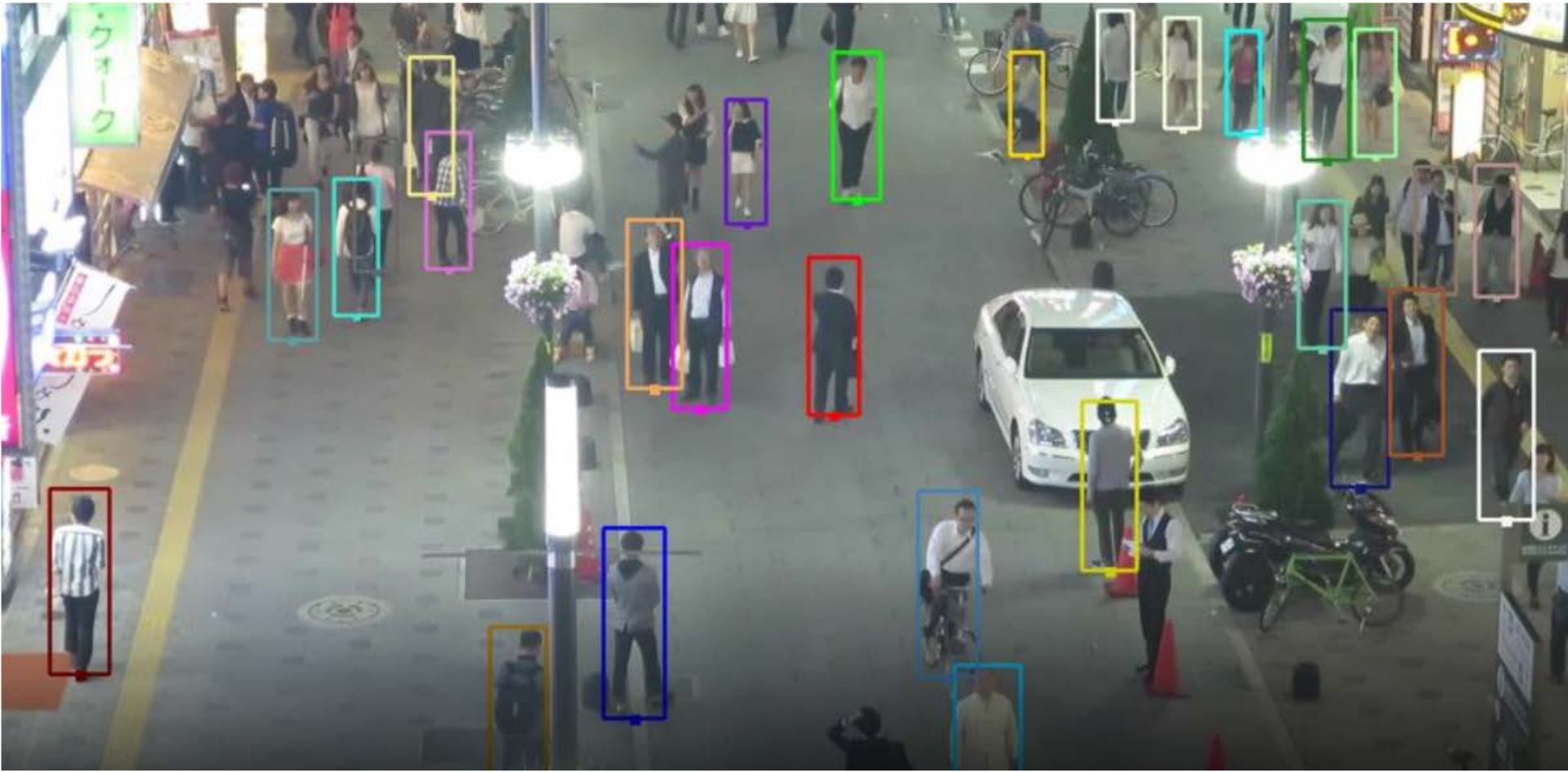
Vision & language

- 构建视觉和自然语言之间的语义“桥梁”
  - 图像描述生成（image captioning）
  - 跨媒体检索（cross-media retrieval）
  - 图像问答（image question answering）

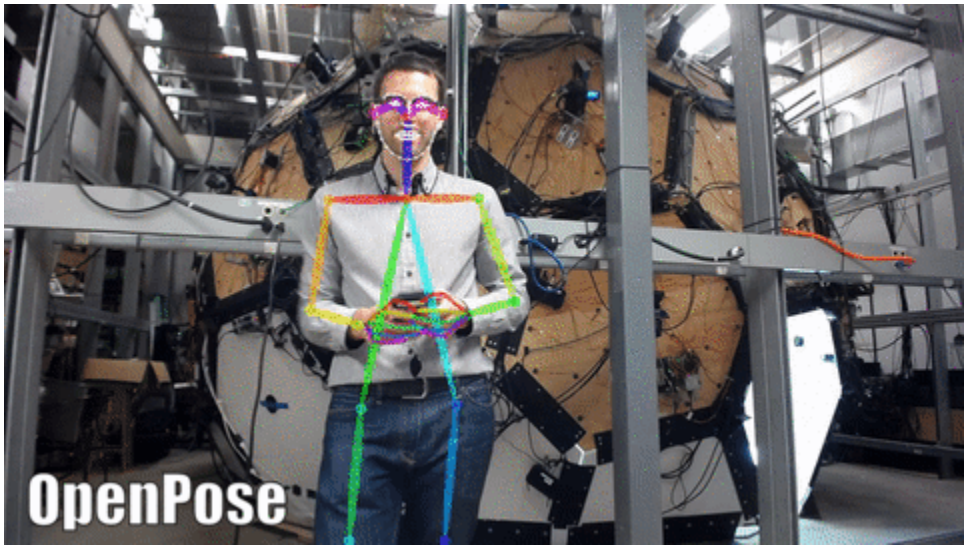
CNN与语音识别

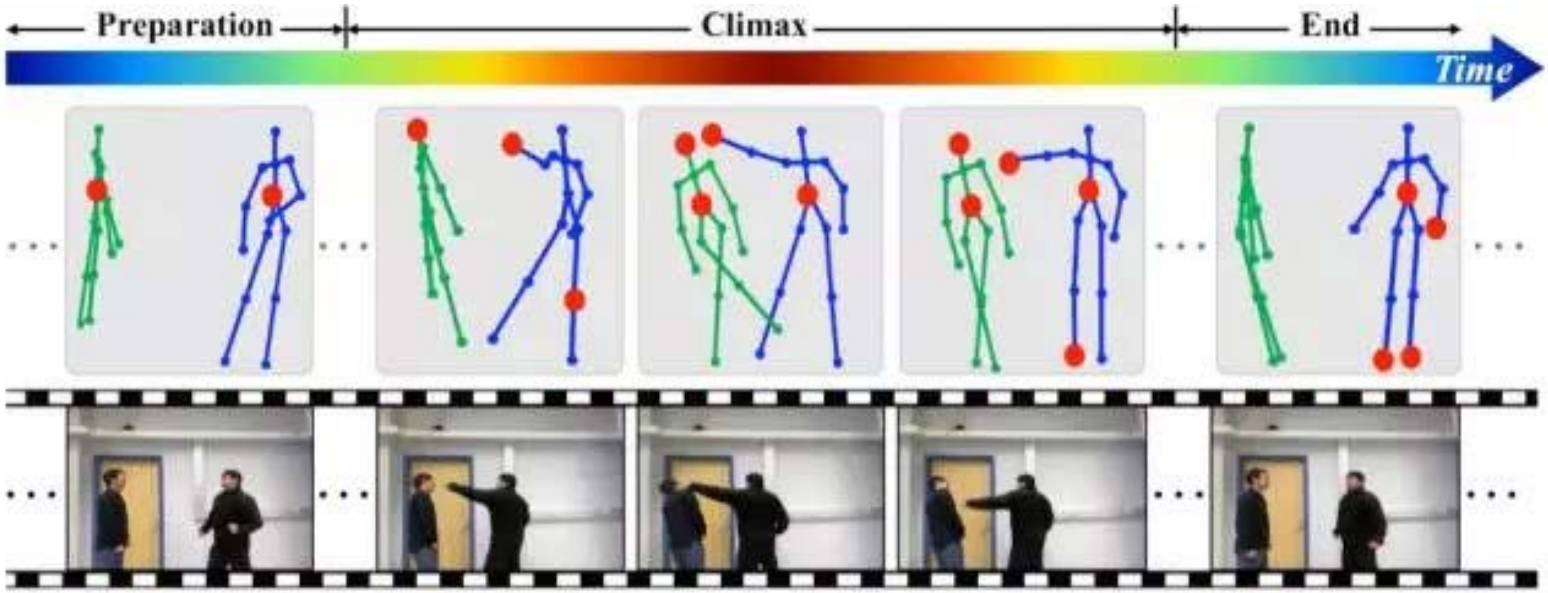

输入声音原始波形
tConv
$x_t \in \mathbb{R}^p$
fConv
LSTM
LSTM
LSTM
DNN
output targets

多目标跟踪 MOT



姿态估计/行为检测/识别





数据集...

- COCO数据集
- activitinet
- HMDB51：来源为YouTube视频，共计51类动作，约7000段视频。数据库主页为：HMDB: a large human motion database
- KTH人体行为数据库：该数据库包括6类行为（walking,jogging, running, boxing, hand waving, hand clapping）,是由25个不同的人执行的，分别在四个场景下，一共有599段视频。背景相对静止。正确率需要达到95.5%以上才能够发文章。下载地址：http://www.nada.kth.se/cvap/actions/
- INRIA XMAX多视角视频库：该数据库从五个视角获得，一共11个人执行14种行为。室内四个方向和头顶一共安装5个摄像头。另外背景和光照基本不变。下载地址：http://4drepository.inrialpes.fr/public/viewgroup/6
- UCF Sports 数据库：该视频包括150段关于体育的视频，一共有13个动作。实验室采用留一交叉验证法。2011年cvpr有几篇都用这个数据库，正确率要达到87%才能发文章。下载地址：http://vision.eecs.ucf.edu/data.html
- Hollywood 人体行为库：该数据库包括8类行为。这些都是电影中的片段。 下载地址：http://www.di.ens.fr/~laptev/actions/hollywood2/
- Olympic sports dataset：该数据库有16种行为，783段视频。现在的正确率大约在75%左右。下载地址：http://vision.stanford.edu/Datasets/OlympicSports/
- 谷歌AVA dataset:是YouTube上提取的被标注的80个原子动作。共5.8万个片段，包含握手、踢腿、拥抱、接吻、喝酒、玩乐器、散步等日常活动。下载地址：https://research.google.com/ava/