树状结构与深度自然语言处理

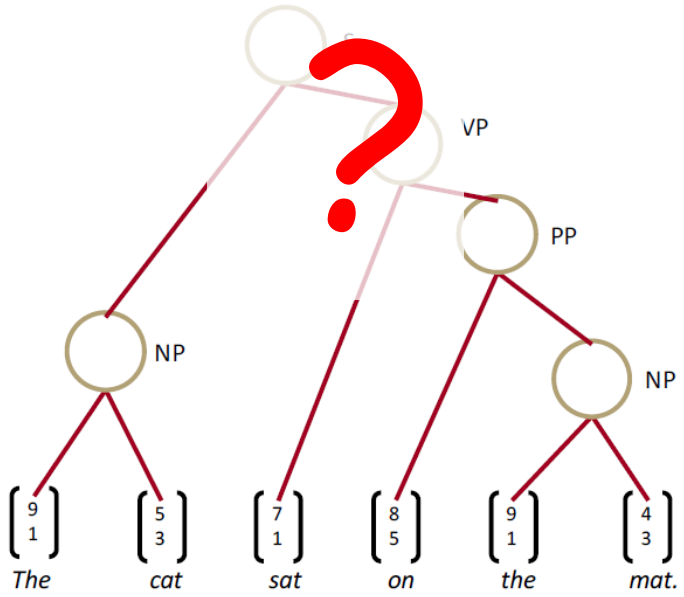
孙栩
信息科学技术学院
xusun@pku.edu.cn
http://xusun.org

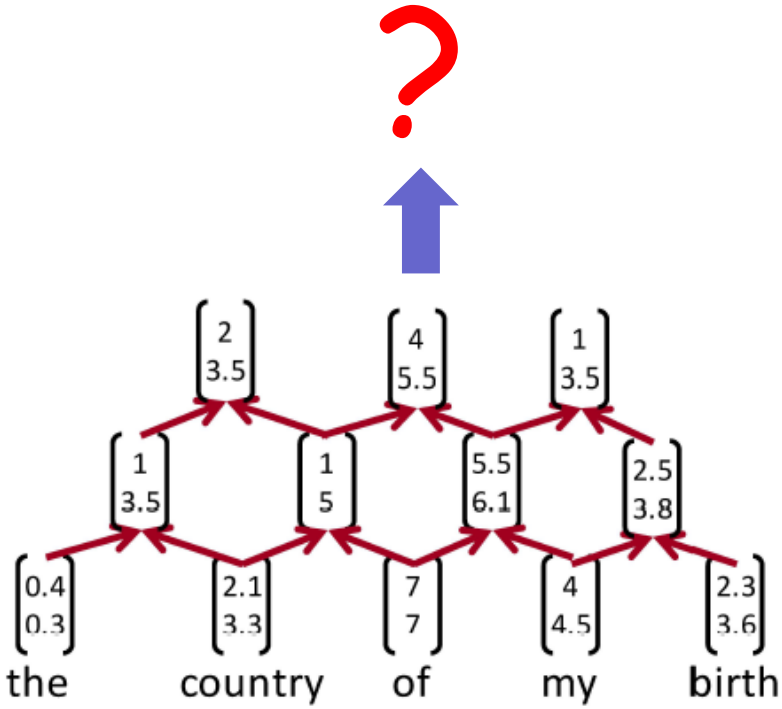
大纲

- **树状结构与深度自然语言处理**
  - 预测树结构：句法分析
  - 利用树结构：文档分类
- **句法分析**
  - 递归神经网络 (Recursive Neural Network)
- **文档分类**
  - 卷积神经网络 (Convolutional Neural Network)

树状结构 与 深度自然语言处理

- **预测树结构**
  - 以短语结构句法分析
    - 传统方法：CFG/PCFG等
    - 介绍：递归神经网络
- **利用树结构**
  - 以文档分类为例
    - 传统方法：词袋/决策树等
    - 介绍：卷积神经网络





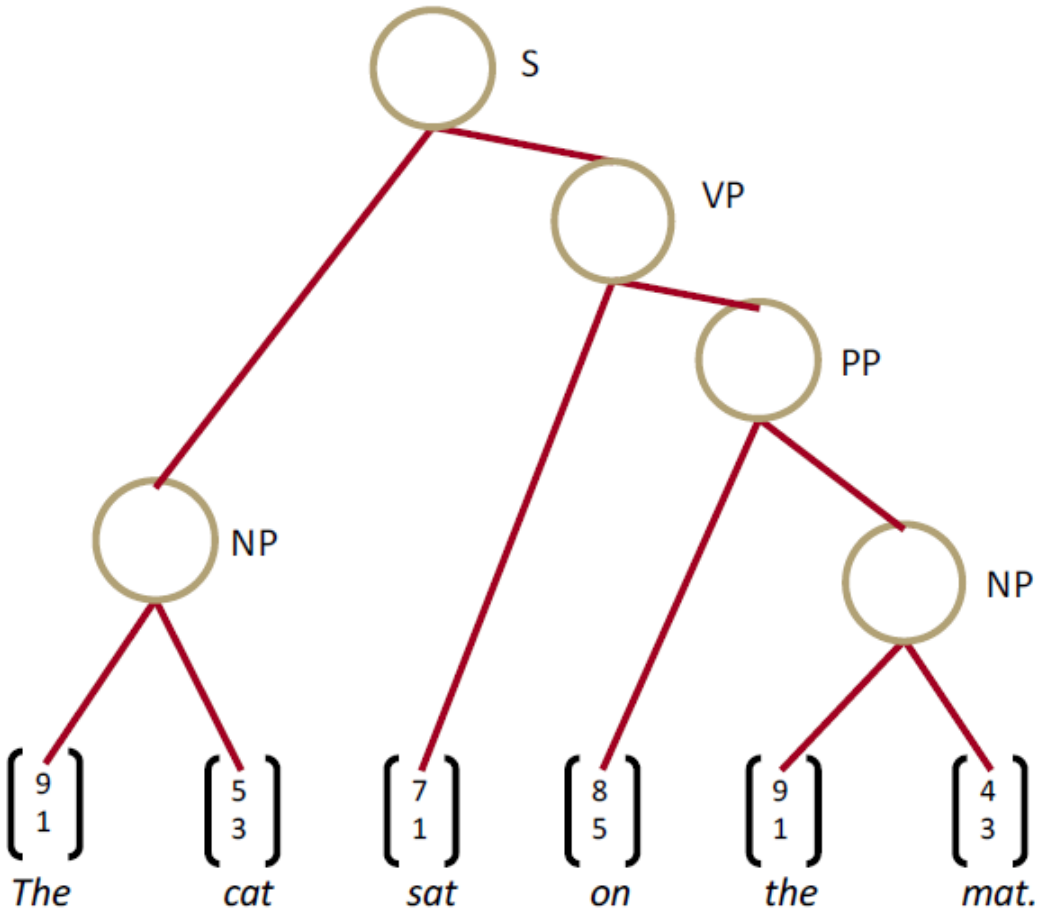
大纲

- **树状结构与深度自然语言处理**
  - 预测树结构：句法分析
  - 利用树结构：文档分类
- **句法分析**
  - 递归神经网络 (Recursive Neural Network)

- **文档分类**
  - 卷积神经网络 (Convolutional Neural Network)

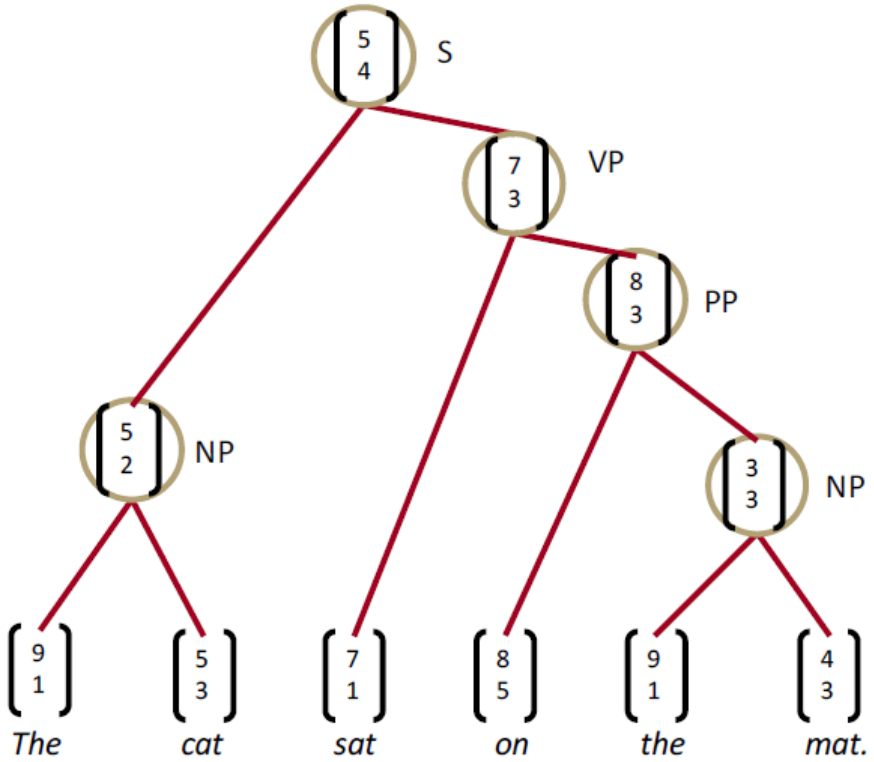
递归神经网络：句法分析

- **句法分析（短语结构）的目标**
  - 获得一颗句法树



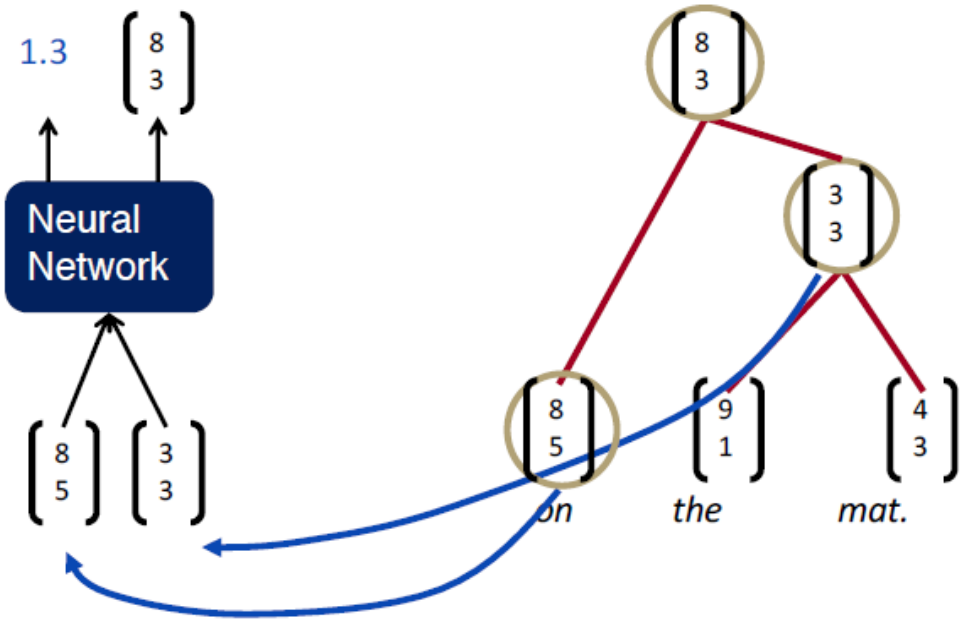
递归神经网络：句法分析

- **递归神经网络的目标**
  - 获得一个句法树的结构
  - 每个结点有对应的向量表示



递归神经网络：句法分析

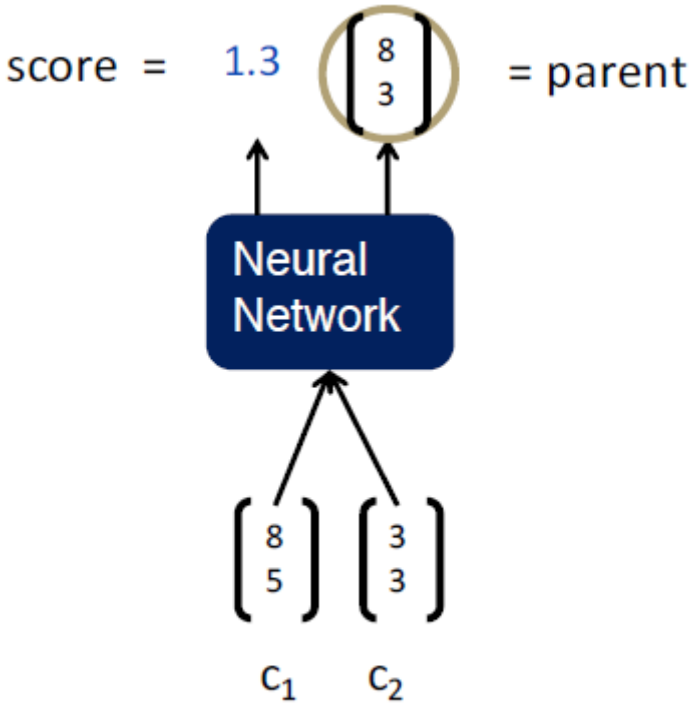
- **输入：两个候选子节点的表示**
- **输出：如果两个节点合并**
  - 语义表示
  - 新节点的得分：用来判断是否要合并



递归神经网络：句法分析

- **模型：**
  - 表示：$p = \tanh(W[c_1, c_2]^T + b)$
  - 分数：$score = u^T p$

- **每个结点的计算都使用相同的W和u**



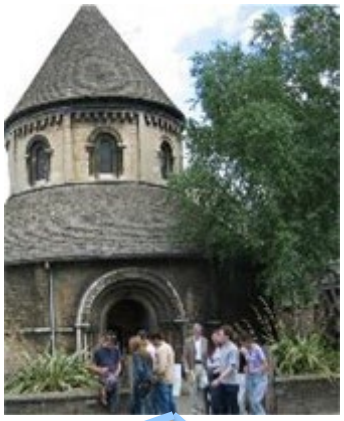
递归神经网络：句法分析

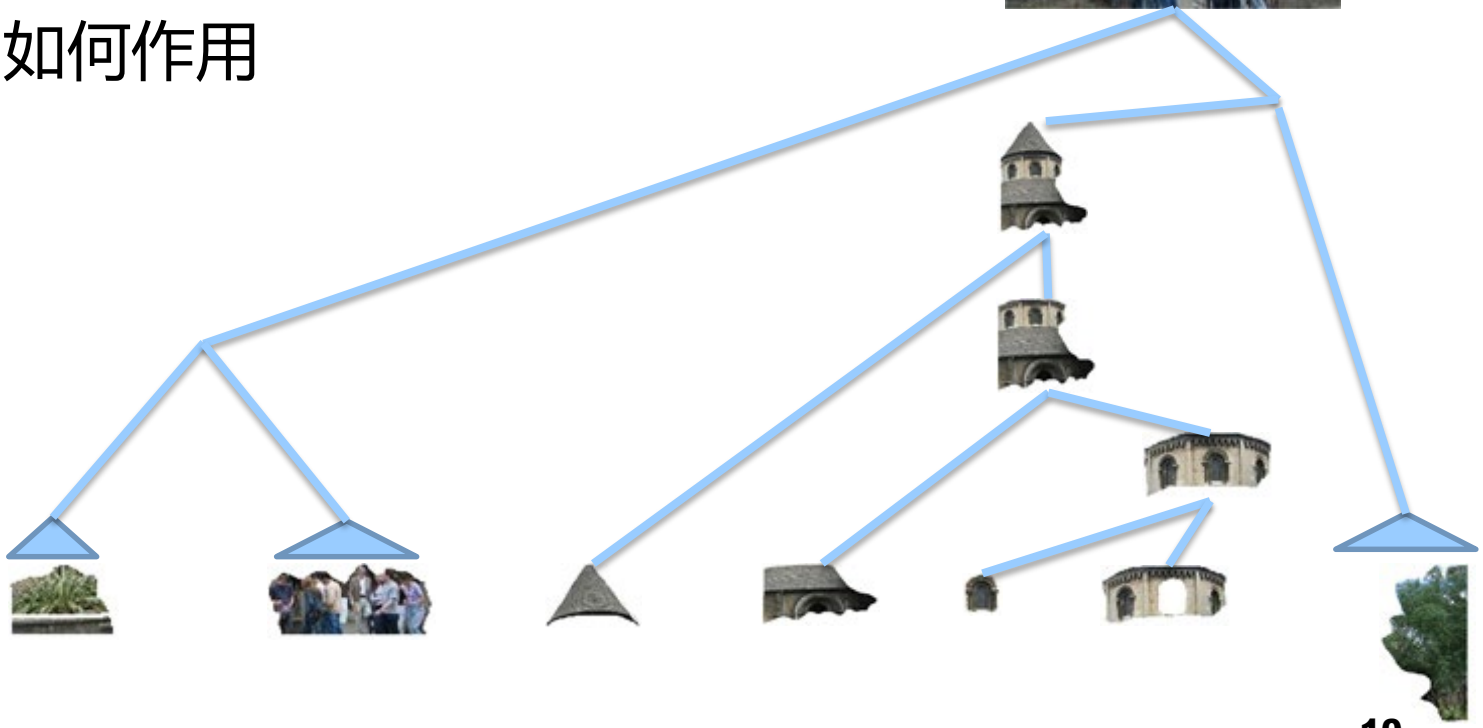
- **实验结果**
  - 标准WSJ分割，labeld F1
  - 基于一个简单PCFG，且使用更少状态
  - 搜索空间Fast pruning，减少矩阵向量乘积
  - 比Stanford factored parser提高3.8%，快20%

Parser	Test, All Sentences
Stanford PCFG, (Klein and Manning, 2003a)	85.5
Stanford Factored (Klein and Manning, 2003b)	86.6
Factored PCFGs (Hall and Klein, 2012)	89.4
Collins (Collins, 1997)	87.7
SSN (Henderson, 2004)	89.4
Berkeley Parser (Petrov and Klein, 2007)	90.1
CVG (RNN) (Socher et al., ACL 2013)	85.0
CVG (SU-RNN) (Socher et al., ACL 2013)	90.4
Charniak - Self Trained (McClosky et al. 2006)	91.0
Charniak - Self Trained-ReRanked (McClosky et al. 2006)	92.1

递归神经网络：场景分析

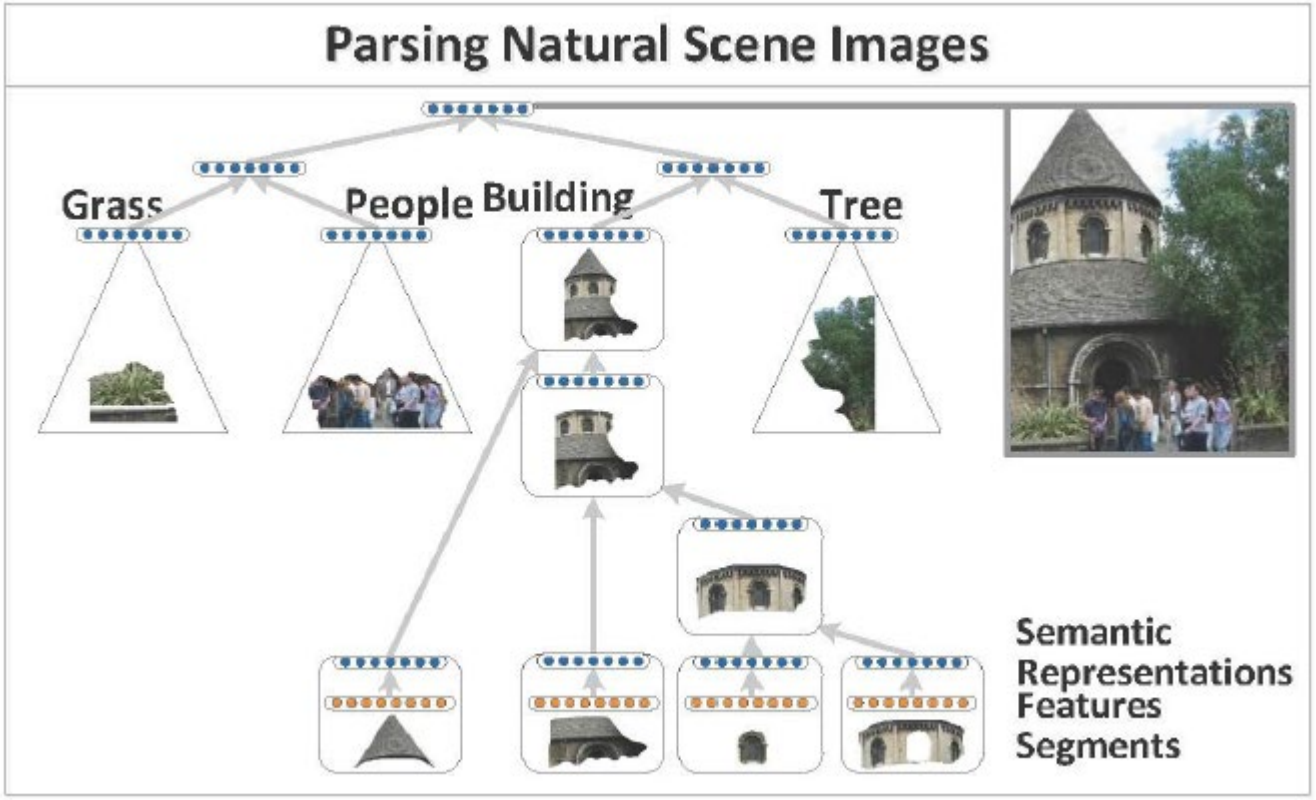
- 场景分析(Scene Parsing)
- 同样是复合性原理
- 一个场景图片的含义是以下的函数
  - 更小的子区域
  - 如何组合成更大的对象
  - 对象间如何作用





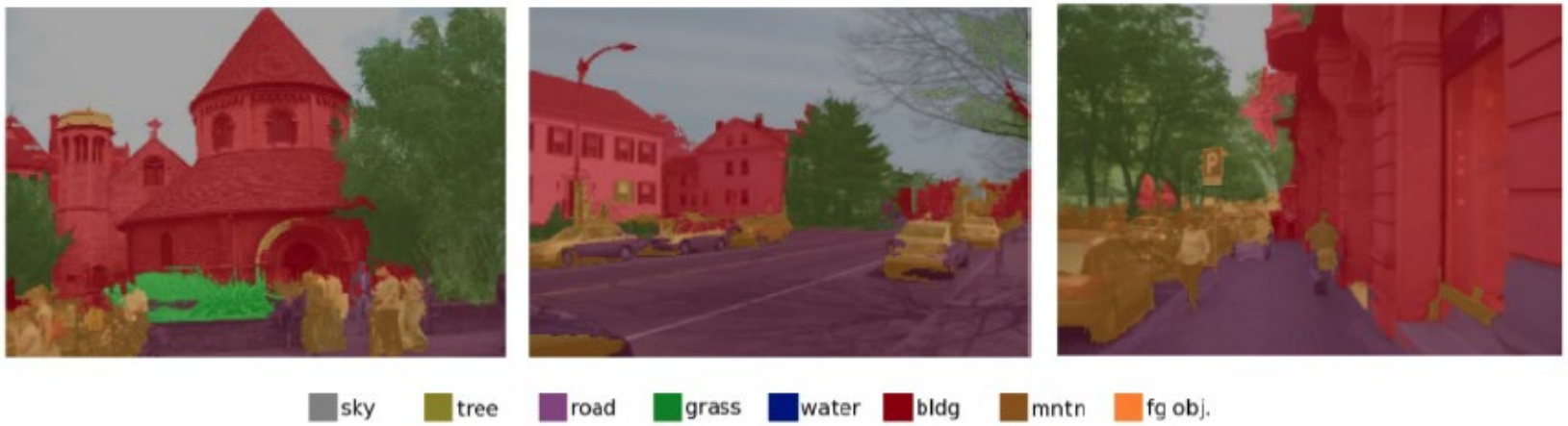
递归神经网络：场景分析

- **算法：**
  - 与自然语言分析同样使用Recursive Neural Network



递归神经网络：场景分析

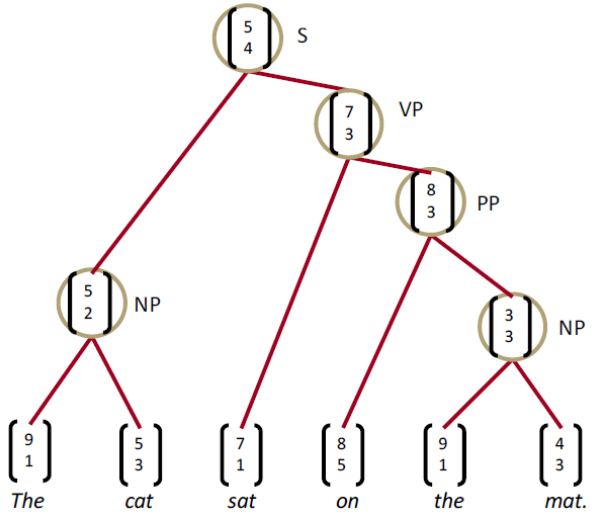
- **图片的多分类划分**
  - 与自然语言分析同样使用Recursive Neural Network

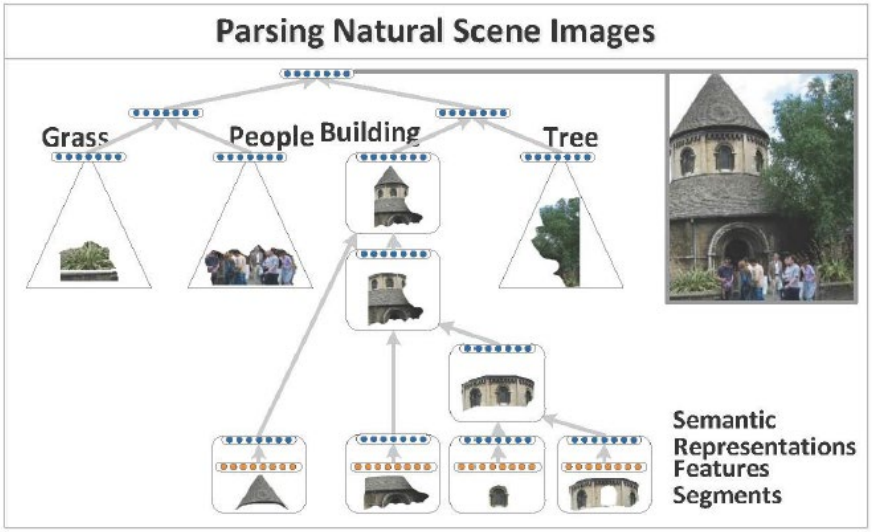


Method	Accuracy
Pixel CRF (Gould et al., ICCV 2009)	74.3
Classifier on superpixel features	75.9
Region-based energy (Gould et al., ICCV 2009)	76.4
Local labelling (Tighe & Lazebnik, ECCV 2010)	76.9
Superpixel MRF (Tighe & Lazebnik, ECCV 2010)	77.5
Simultaneous MRF (Tighe & Lazebnik, ECCV 2010)	77.5
Recursive Neural Network	**78.1**

递归神经网络：小结

- **递归神经网络**
  - 在树结构的不同局部使用相同的神经网络参数
  - 可以用来执行句法分析任务
  - 需要预先定义好的树结构





大纲

- **树状结构与深度自然语言处理**
  - 预测树结构：句法分析
  - 利用树结构：文档分类
- **句法分析**
  - 递归神经网络 (Recursive Neural Network)
- **文档分类**
  - 卷积神经网络 (Convolutional Neural Network)

卷积神经网络

- **文档分类的目标**
  - 给定一个文档、判断文档类型
  - 比如情感极性分析，分为积极、消极等类别

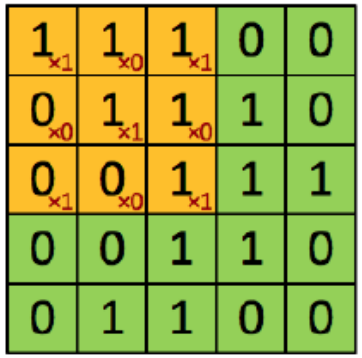
- **RecursiveNN**
  - 递归神经网络可以获得句子的向量表示
    - 可以利用该向量进行分类
  - 但需要预先定义的树结构
    - 训练数据标注
    - 测试数据预先预测树结构、需要句法分析器
  - 只获取合理短语的复合向量

- **NLP中的CNN**
  - 计算所有可能的片段的向量、无论是否是合理的
    - 不需要句法分析器
  - 例子: “the country of my birth”
    - the country, country of, of my, my birth, the country of, country of my, of my birth, the country of my, country of my birth
  - 在语言学或认知学上看似并不可行?

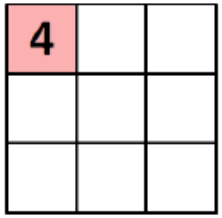
- **卷积运算：对输入重复应用加权求和**

- **一般的1维离散变量卷积的公式**
  - $(f * g)[n] = \sum_{m=-M}^{M} f[n-m]g[m]$
  - 以M为窗口内两个函数的复合
    - f，与相关的词有关
    - g，与位置有关

- **卷积特别擅长从图像中提取特征**
- **2维例子**
  - 黄色: filter weights
  - 绿色：inputs



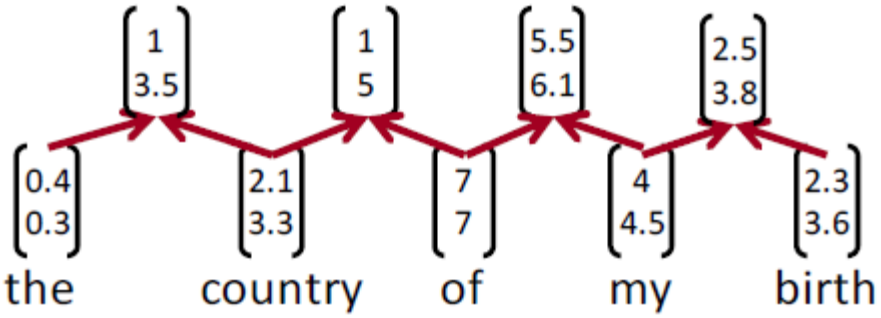
Image



Convolved Feature

Stanford UFLDL wiki

- 第一层：计算所有bigram的向量

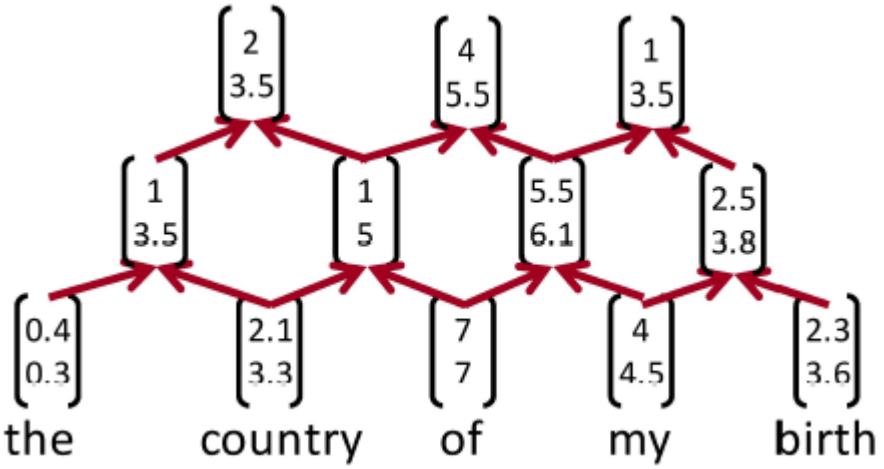


- 局部的计算模式与RecursiveNN中相同

$$p = \tanh\left(W \begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b\right)$$

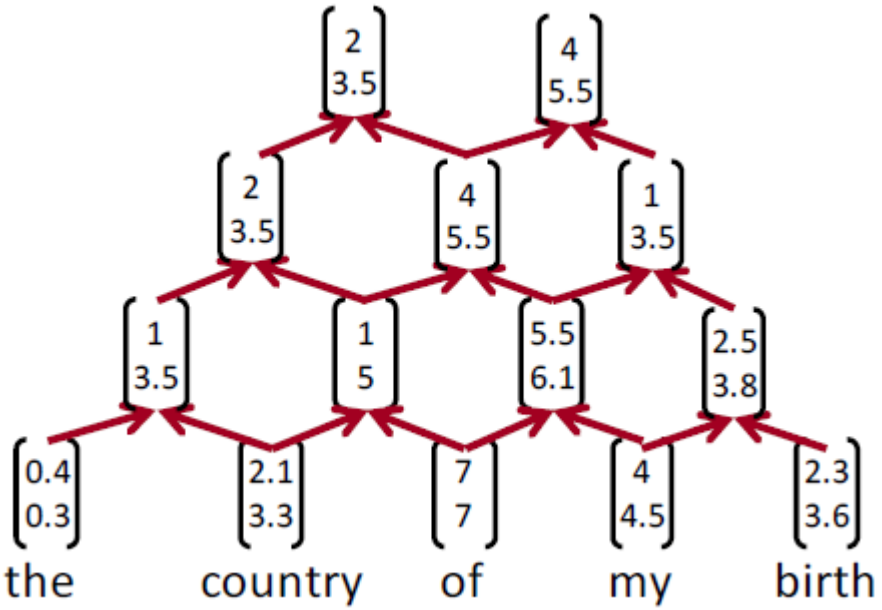
  - RecursiveNN可以解释为对于词向量的特殊卷积

- **更高层：多种计算方式可以选择**
- **选择1(naïve的选择)：**
  - 便于理解但不一定是最好的
  - 重复第一层，但使用不同的权重



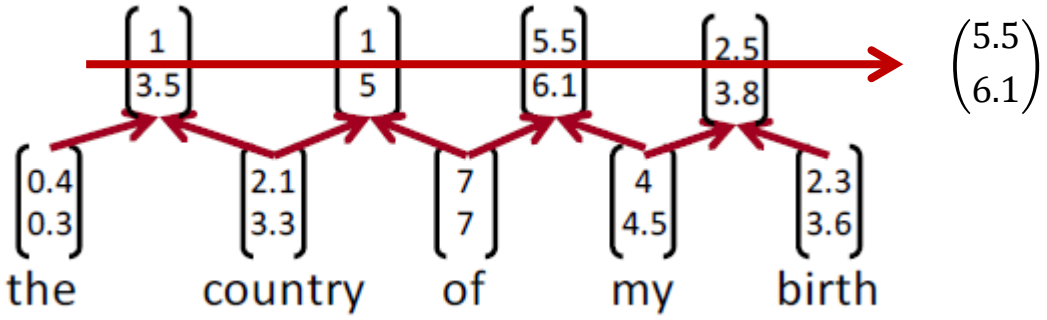
$$p = \tanh\left(W^{(2)}\begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b\right)$$

- **更高层：多种计算方式可以选择**
- **选择1(naïve的选择)：**
  - 便于理解但不一定是最好的
  - 重复第一层，但使用不同的权重



$$p = \tanh\left(W^{(2)}\begin{bmatrix} c_1 \\ c_2 \end{bmatrix} + b\right)$$

- **选择2(主流的选择)：**
  - 池化操作(pooling)
  - 将得到多个向量合并为一个向量

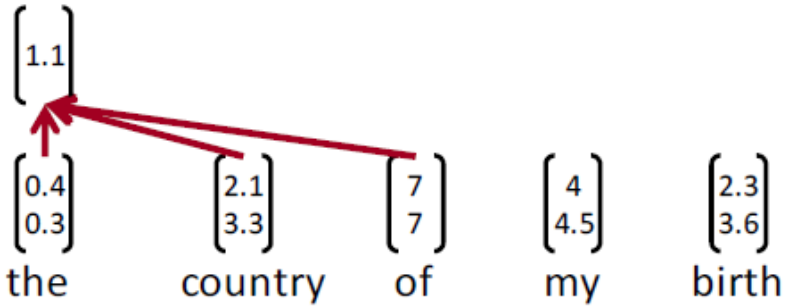


- **由Collobert and Weston 2011和Kim 2014首先应用于NLP**
  - Convolutional Neural Networks for Sentence Classification

卷积神经网络

- **模型：**
  - 一层CNN+一个池化层

- **词向量：** $x_i \in \mathbb{R}^k$
- **句子：** $x_{1:n} = x_1 \oplus x_2 \ldots \oplus x_n$ **（向量拼接）**
  - $x_{i:i+j}$是词的拼接
- **卷积核：** $w \in \mathbb{R}^{hk}$，**窗口为h**
  - h大于2或更高，如3
  - 注意：这里卷积核是向量
- **计算CNN层的特征**
  - $c_i = f(w^T x_{i:i+h-1} + b)$



卷积神经网络

- 池化层(Pooling layer)
- 这里介绍的是最大池化层(max pooling)

- 想法：捕捉最重要的activation

- feature map $c = [c_1, c_2, \dots, c_{n-h+1}] \in \mathbb{R}^{n-h+1}$取最大值
- Pooled：$\hat{c} = \max\{c\}$

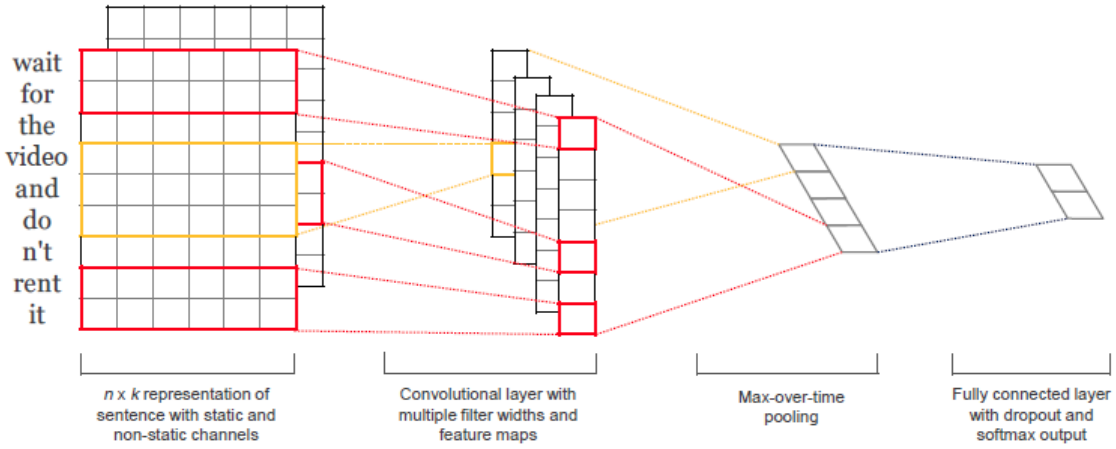
- 但我们想要更多的特征！

卷积神经网络

- **使用多个卷积核**
  - 窗口大小相同的h可以组合为一个矩阵
- **可以使用不同大小的窗口**
- **因为有池化层，窗口的大小并不影响下一层的输入维度**
  - 下一层输入的维度就是卷积核的个数
- **我们可以用不同的卷积核来观察unigrams, bigrams, trigrams, 4-grams, …**

卷积神经网络

- **CNN应用于文档分类**
  - 首先一个卷积，然后一个最大池化
  - 得到最后的特征向量$z = [\widehat{c_1}, \ldots, \widehat{c_m}]$假设有m个filter w
- **如何进行分类?**
  - 简单的softmax层 $y = softmax\left(W^{(S)}z + b\right)$



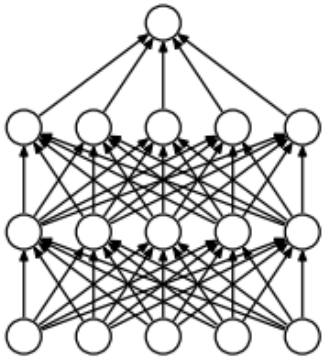
n words (possibly zero padded) and each word vector has k dimensions

- 在分类任务上的结果
  - Kim 2014

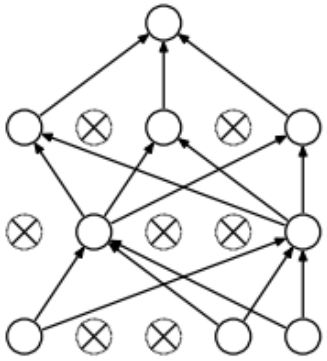
Model	MR	SST-1	SST-2	Subj	TREC	CR	MPQA
CNN-rand	76.1	45.0	82.7	89.6	91.2	79.8	83.4
CNN-static	81.0	45.5	86.8	93.0	92.8	84.7	**89.6**
CNN-non-static	**81.5**	48.0	87.2	93.4	93.6	84.3	89.5
CNN-multichannel	81.1	47.4	**88.1**	93.2	92.2	**85.0**	89.4
RAE (Socher et al., 2011)	77.7	43.2	82.4	–	–	–	86.4
MV-RNN (Socher et al., 2012)	79.0	44.4	82.9	–	–	–	–
RNTN (Socher et al., 2013)	–	45.7	85.4	–	–	–	–
DCNN (Kalchbrenner et al., 2014)	–	48.5	86.8	–	93.0	–	–
Paragraph-Vec (Le and Mikolov, 2014)	–	**48.7**	87.8	–	–	–	–
CCAE (Hermann and Blunsom, 2013)	77.8	–	–	–	–	–	87.2
Sent-Parser (Dong et al., 2014)	79.5	–	–	–	–	–	86.3
NBSVM (Wang and Manning, 2012)	79.4	–	–	93.2	–	81.8	86.3
MNB (Wang and Manning, 2012)	79.0	–	–	**93.6**	–	80.0	86.3
G-Dropout (Wang and Manning, 2013)	79.0	–	–	93.4	–	82.1	86.1
F-Dropout (Wang and Manning, 2013)	79.1	–	–	**93.6**	–	81.9	86.3
Tree-CRF (Nakagawa et al., 2010)	77.3	–	–	–	–	81.4	86.1
CRF-PR (Yang and Cardie, 2014)	–	–	–	–	–	82.7	–
SVM_S (Silva et al., 2011)	–	–	–	–	**95.0**	–	–

技巧：Dropout

- 想法：随机选取一些特征z设为0
- 建立一个mask向量r，元素为Bernoulli随机变量，为1的概率为p(超参数)
- 在训练中删除特征
  - $y = softmax\left(W^{(S)}(r \circ z) + b\right)$
- 原因：预防co-adaptation
  - 避免特定的特征组合导致过拟合



(a) Standard Neural Net

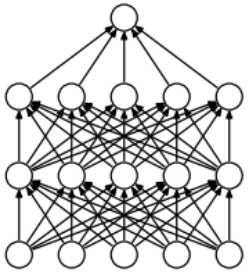


(b) After applying dropout.

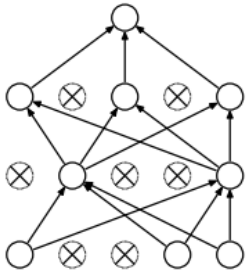
技巧：Dropout

- $y = softmax\left(W^{(S)}(r \circ z) + b\right)$
- **在训练中，反向传播只经过z中那些为r为1的元素**

- **在测试时，没有dropout，z变大**
- **因而，我们缩放最终的向量p倍**
  - $\widehat{W} = pW$
  - 实际中也可在训练中前向计算时将输出变为p倍，测试时不变



(a) Standard Neural Net



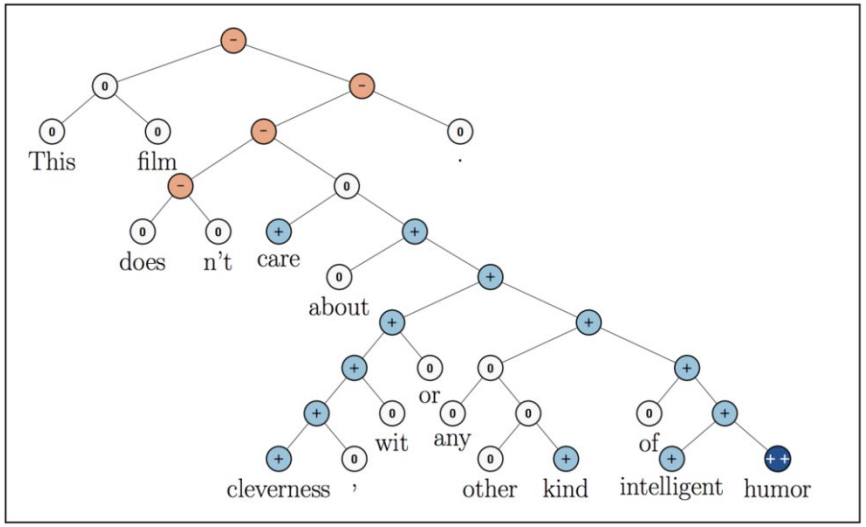
(b) After applying dropout.

- **Kim (2014) reports 2 – 4% improved accuracy and ability to use very large networks without overfitting**

卷积神经网络：小结

- **卷积神经网络**
  - 利用卷积操作
  - 不需要预先定义好的树结构
  - 自底向上构建完全树、包含所有可能的短语片段
  - 可能包含非常多噪音短语，需要考虑抗过拟合

- **Recursive CNN**
  - 只在预先定义好的树上执行操作
  - 但每次操作使用卷积而不是全连接
  - Best of Both Worlds
    - 卷积操作的表达能力、树形式的结构限定，被广泛使用



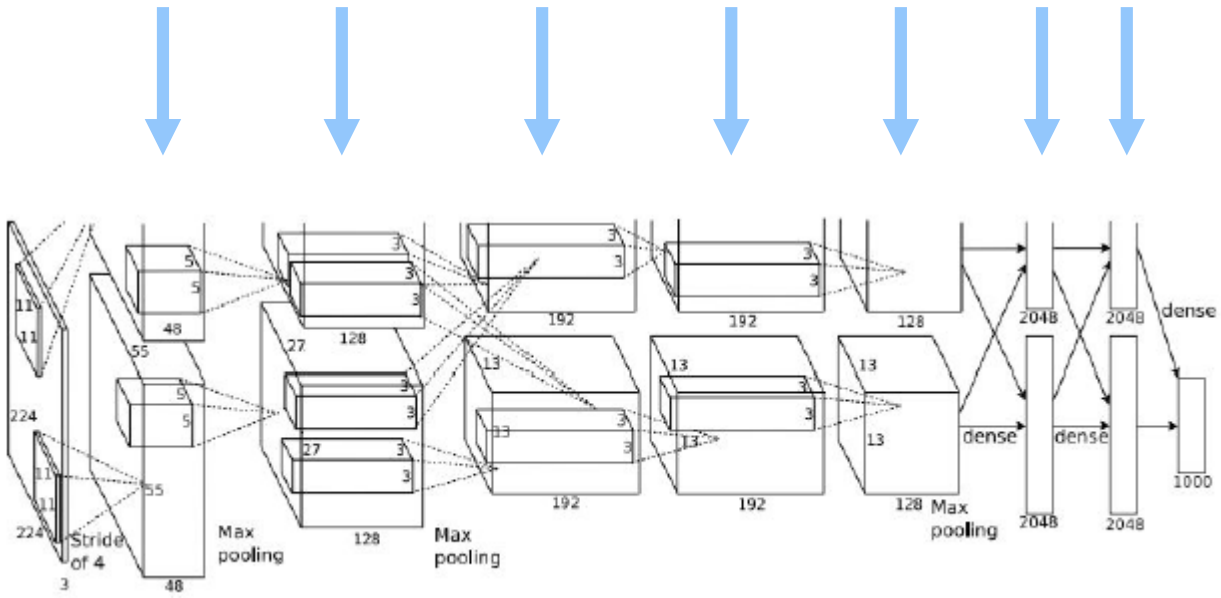
大纲

- **树状结构与深度自然语言处理**
  - 预测树结构：句法分析
  - 利用树结构：文档分类
- **句法分析**
  - 递归神经网络 (Recursive Neural Network)
- **文档分类**
  - 卷积神经网络 (Convolutional Neural Network)
  - CNN的可视化解释（from CV）
    - 类似的，也可以解释文档分类模型

CNN中发生了什么

- 大量工作表明擅长捕捉适用于图像分类的模式
- 但CNN究竟在寻找怎样的模式?

输入
3 x 244 x 244

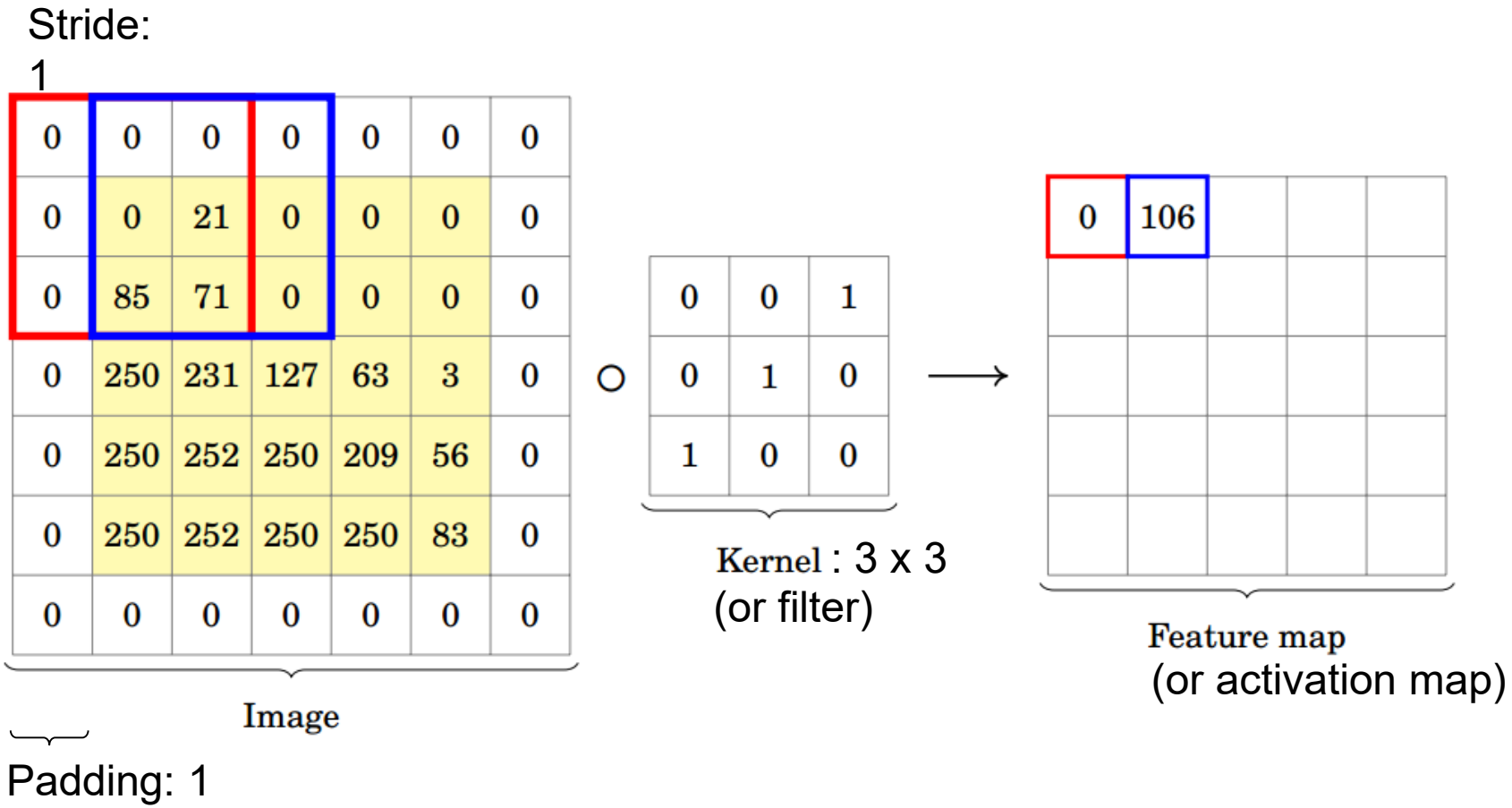


输出
1000

CNN中发生了什么

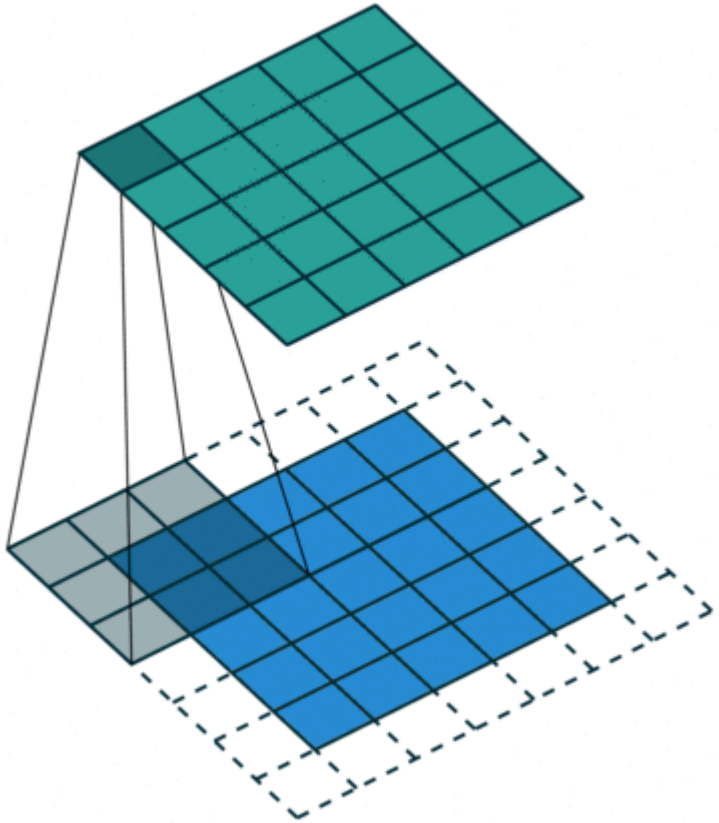
- 可视化理解CNN
- 通过可视化方法，将CNN的各个组件以直观的形式展示出来
- Kernel/Filter
- Feature Map

CNN简介: convolution



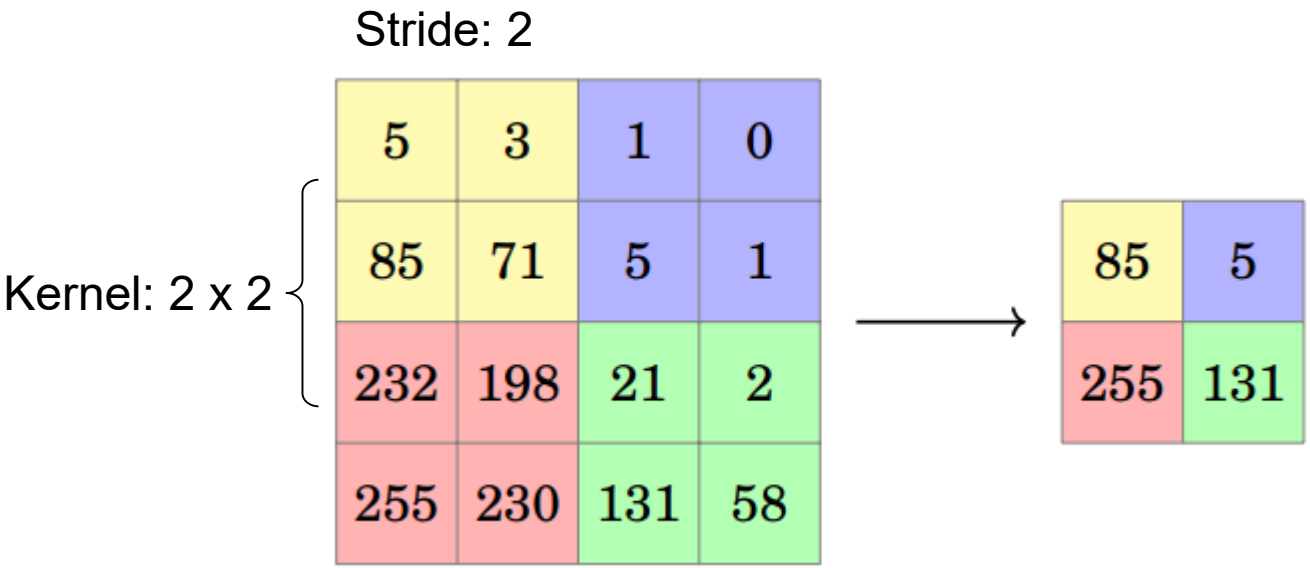
https://www.vaetas.cz/blog/intro-convolutional-neural-networks/

CNN简介: convolution



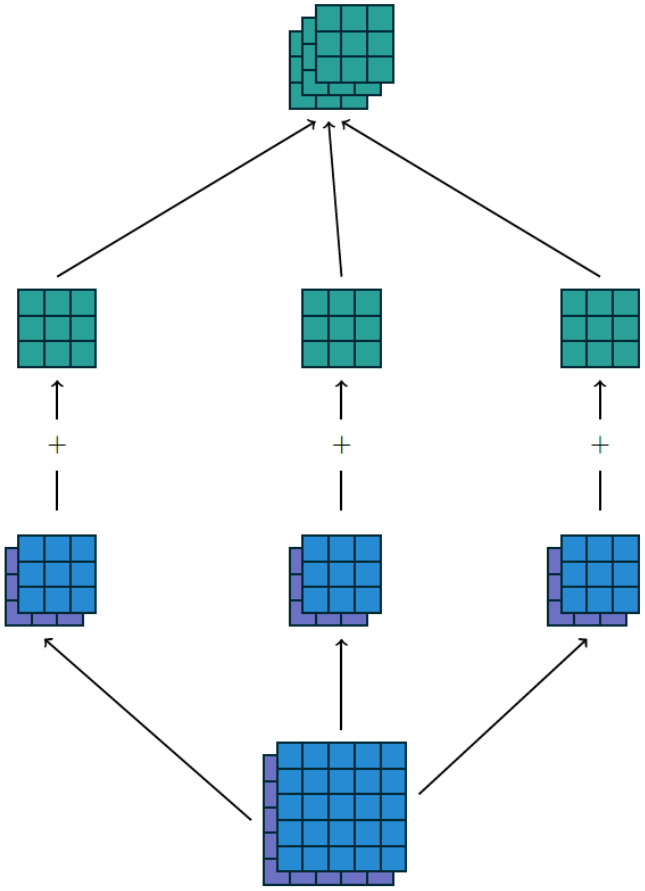
https://github.com/vdumoulin/conv_arithmetic

CNN简介: max pooling



https://www.vaetas.cz/blog/intro-convolutional-neural-networks/

CNN简介: conv + pooling



Feature maps: 3 channels

Pooling: across channels

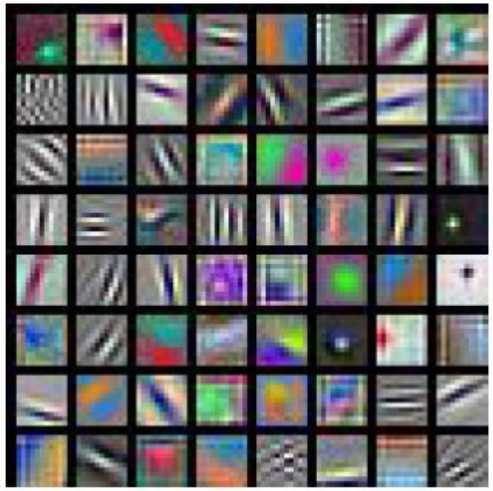
Convolution: 3 kernels per channel

Feature maps: 2 channels

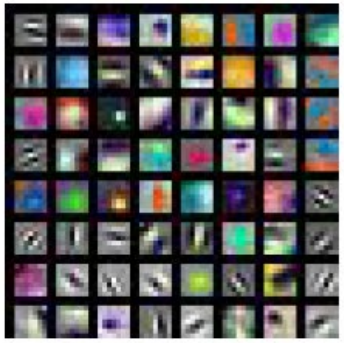
Channel: number of feature maps

https://arxiv.org/abs/1603.07285

Filter可视化：第一层

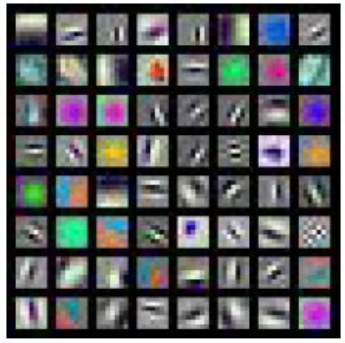


AlexNet:
64 x 3 x 11 x 11



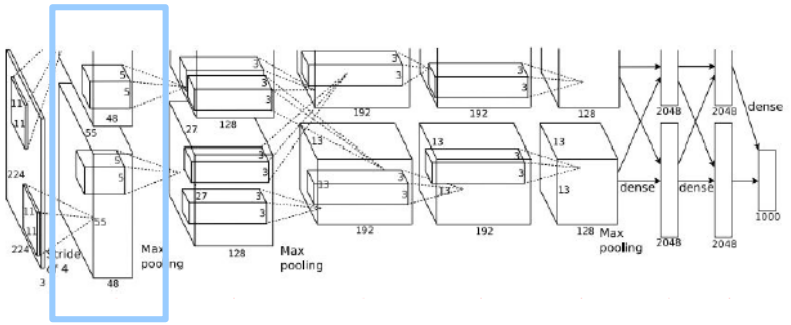
ResNet-18:
64 x 3 x 7 x 7

ResNet-101:
64 x 3 x 7 x 7

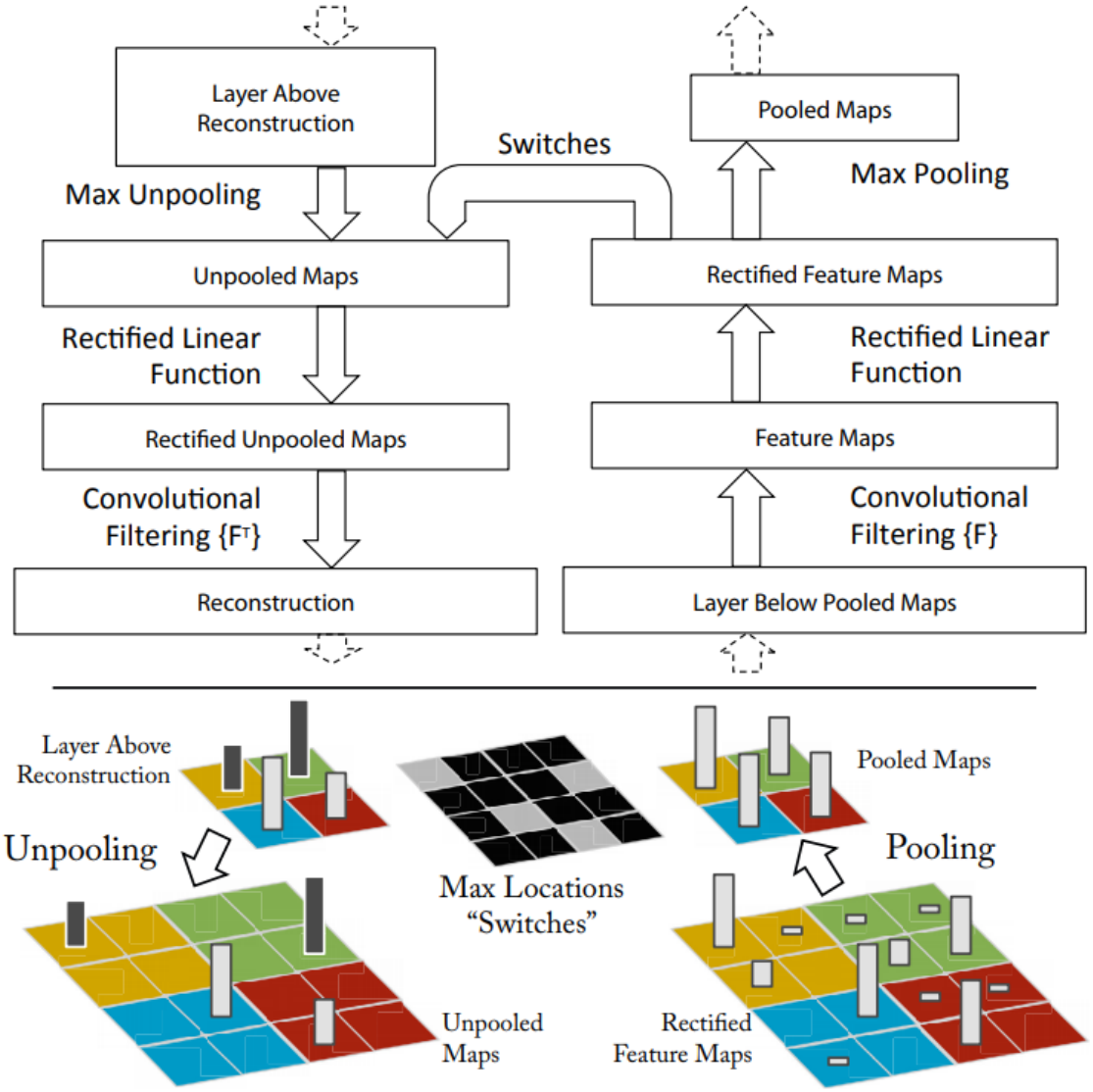


DenseNet-121:
64 x 3 x 7 x 7

第一层普遍捕捉
的是线条

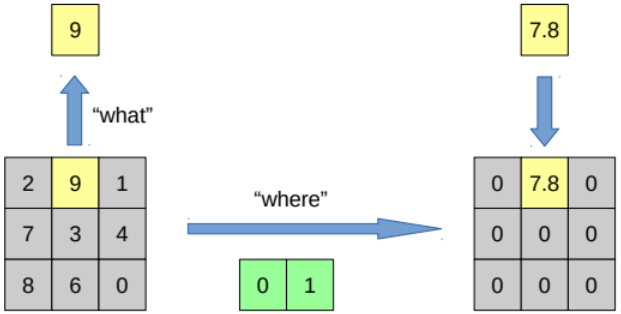


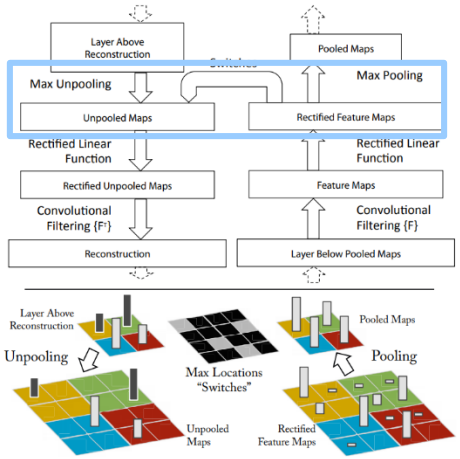
Deconvnet & Convnet



https://arxiv.org/pdf/1311.2901.pdf

Unpooling

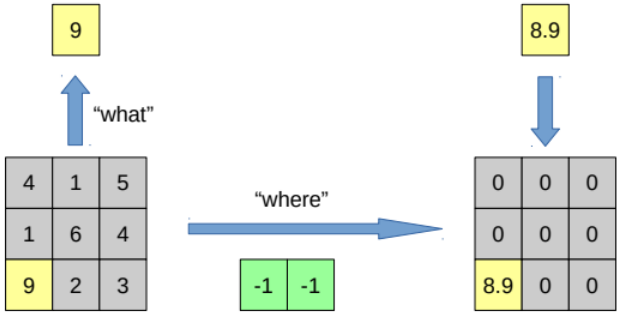




Max pooling不可逆

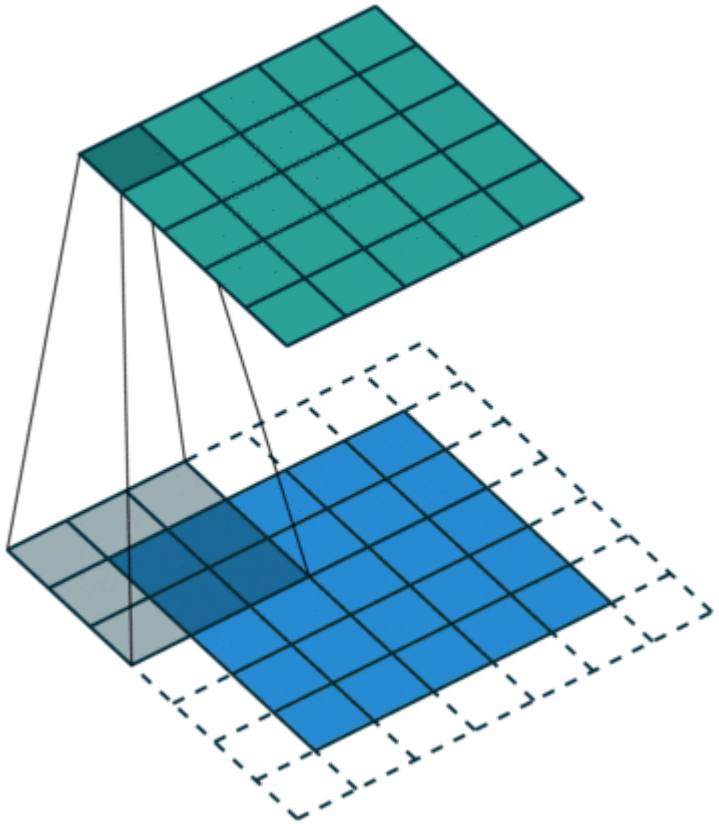
额外记录最大值位置（switches）

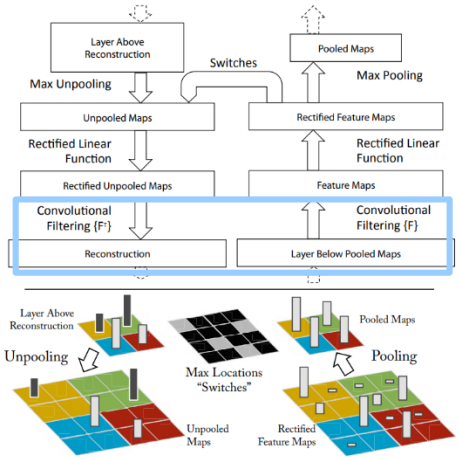
一种近似的反过程



https://arxiv.org/pdf/1506.02351.pdf

Deconvolution





Convolution同样不可逆

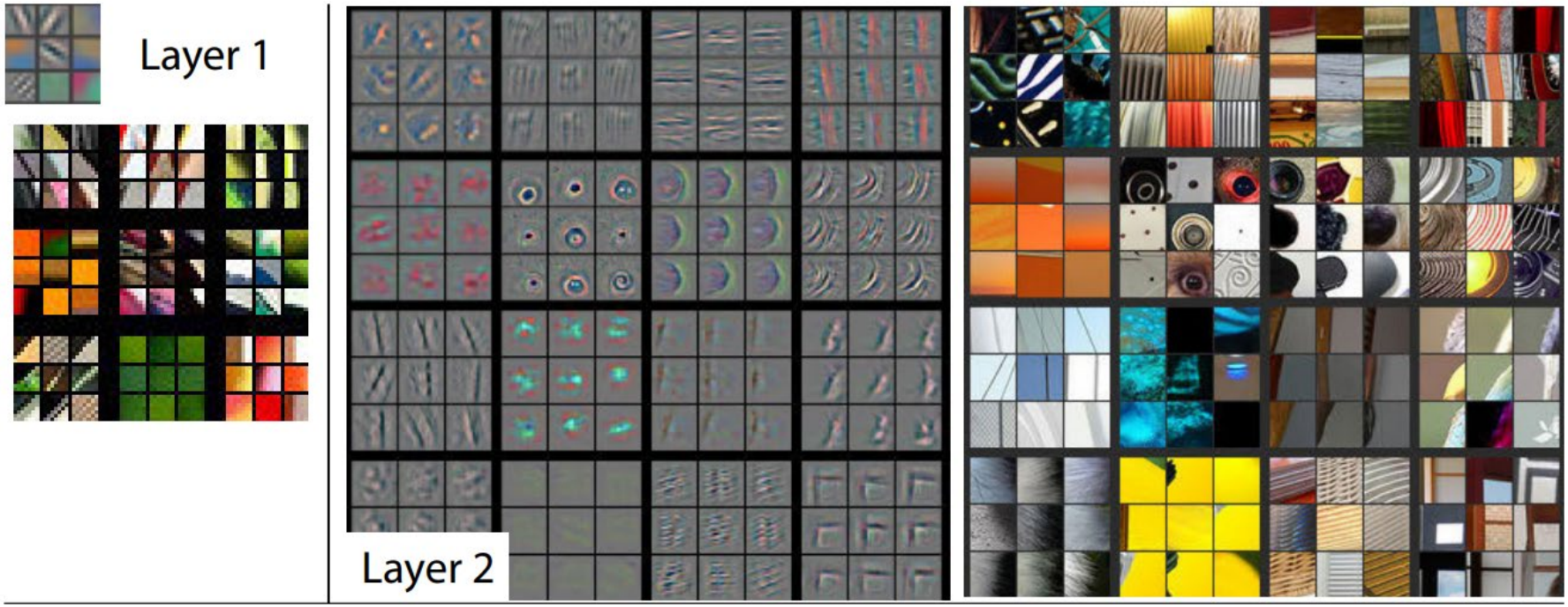
使用Deconvolution，上层feature map每个位置乘转置filter

同样是一种近似的反过程

（由于名称上的误导性，除了论文中会使用deconvolution这一名名词外，其余场景倾向于使用transposed convolution这一名词）

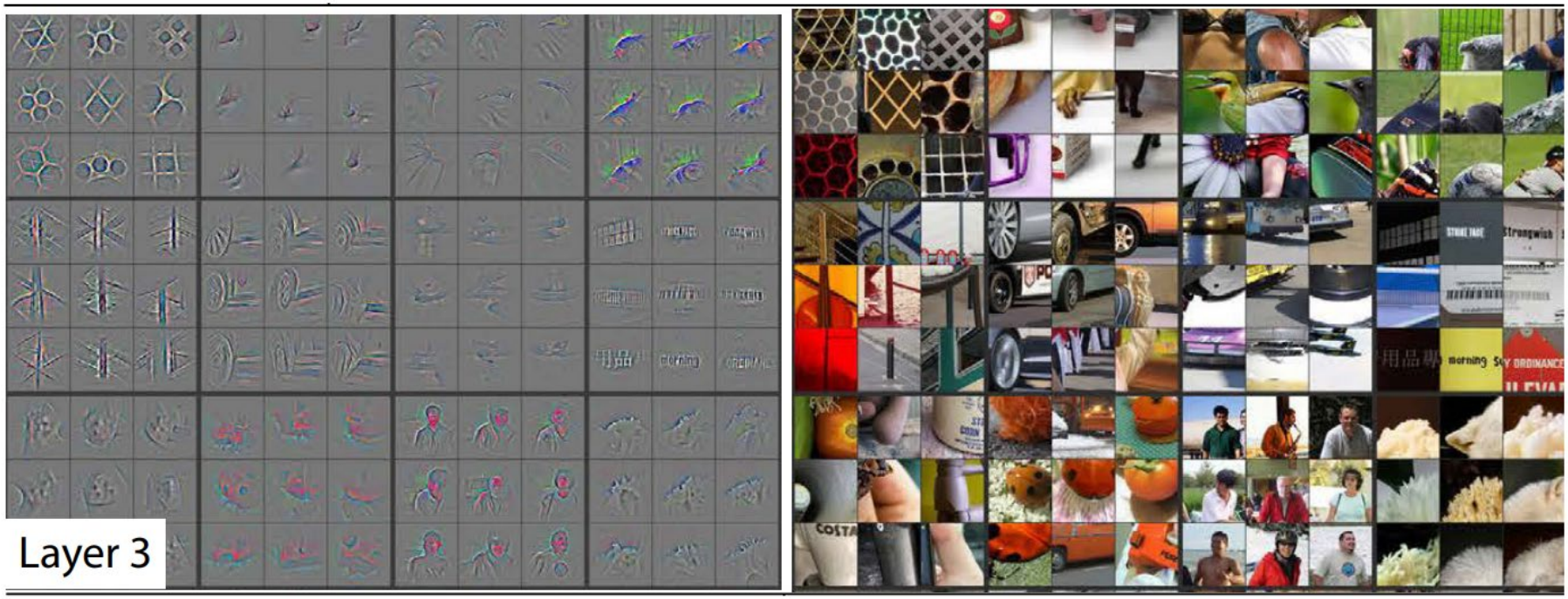
https://github.com/vdumoulin/conv_arithmetic

第一、二层：轮廓线条、配色



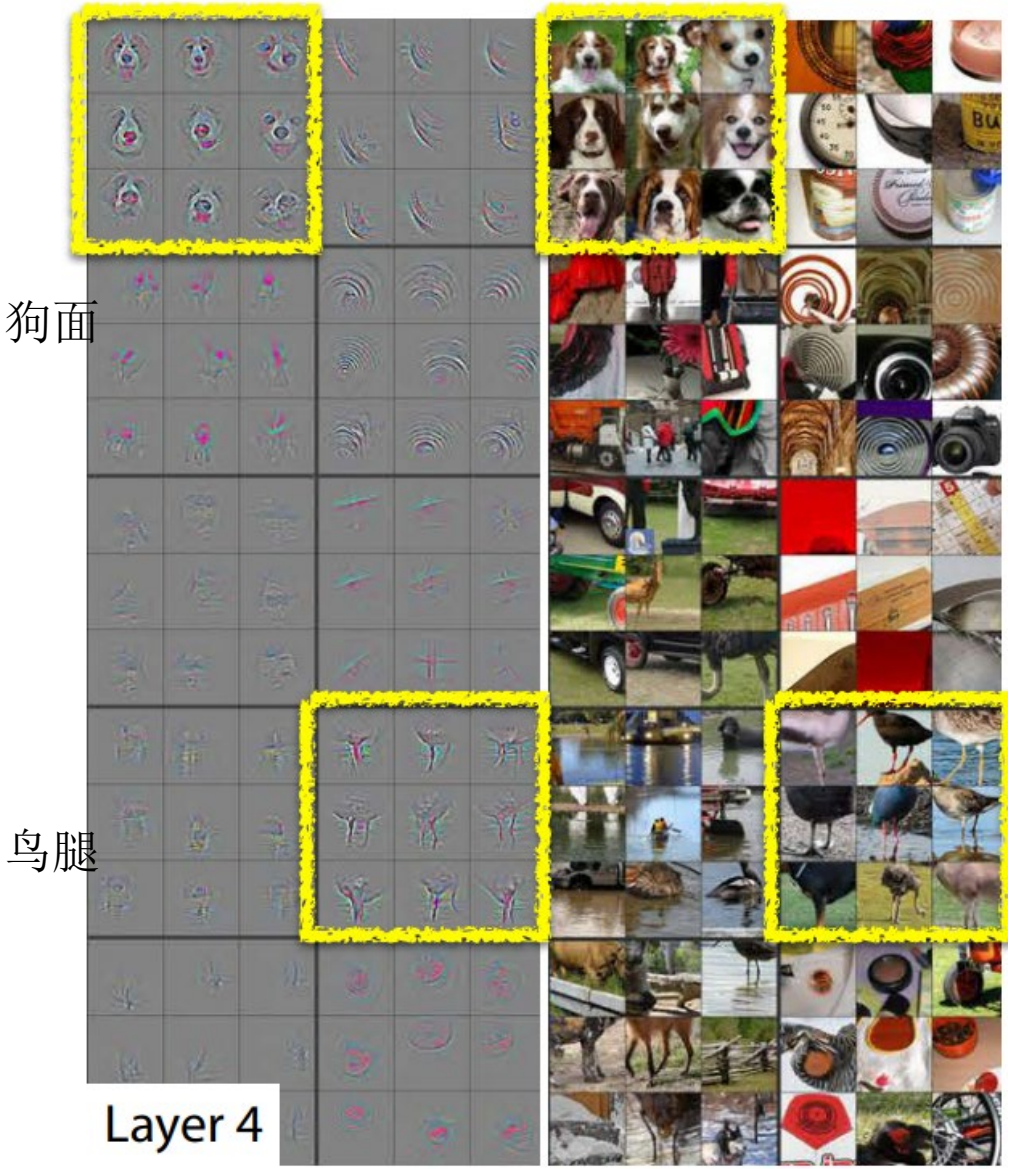
https://courses.cs.washington.edu/courses/cse590v/14au/cse590v_dec5_DeepVis.pdf

第三层：纹理



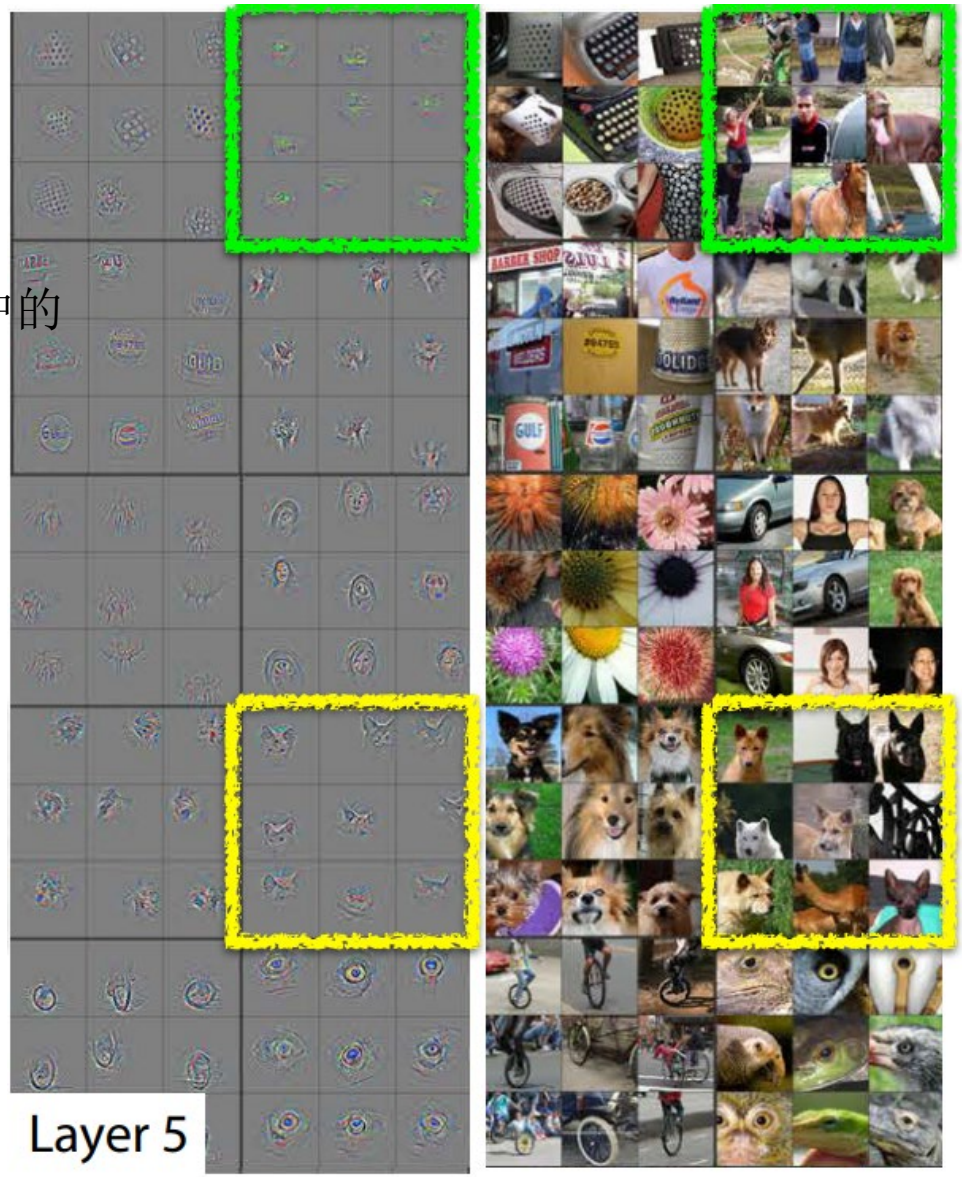
https://courses.cs.washington.edu/courses/cse590v/14au/cse590v_dec5_DeepVis.pdf

第四层：物体的组件



https://courses.cs.washington.edu/courses/cse590v/14au/cse590v_dec5_DeepVis.pdf

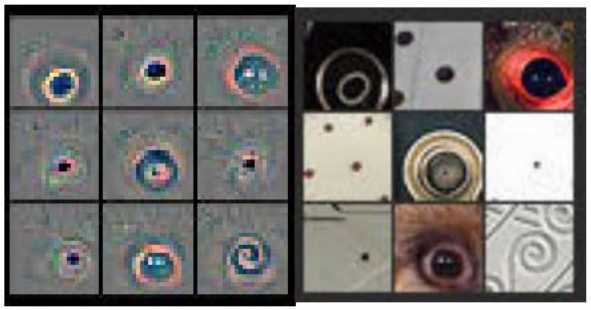
第五层：物体的姿态



https://courses.cs.washington.edu/courses/cse590v/14au/cse590v_dec5_DeepVis.pdf

说明了什么

- 分层表示：从低到高，从简单到复杂
- 单个卷积核的特异性：只针对某类特征
- 选取判别性的特征：与训练目标相关



Layer 2

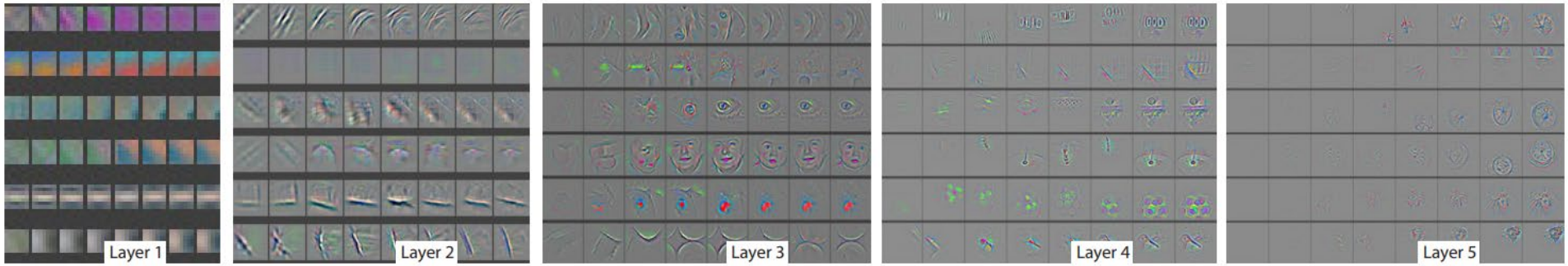
Layer 4

Layer 5

https://courses.cs.washington.edu/courses/cse590v/14au/cse590v_dec5_DeepVis.pdf

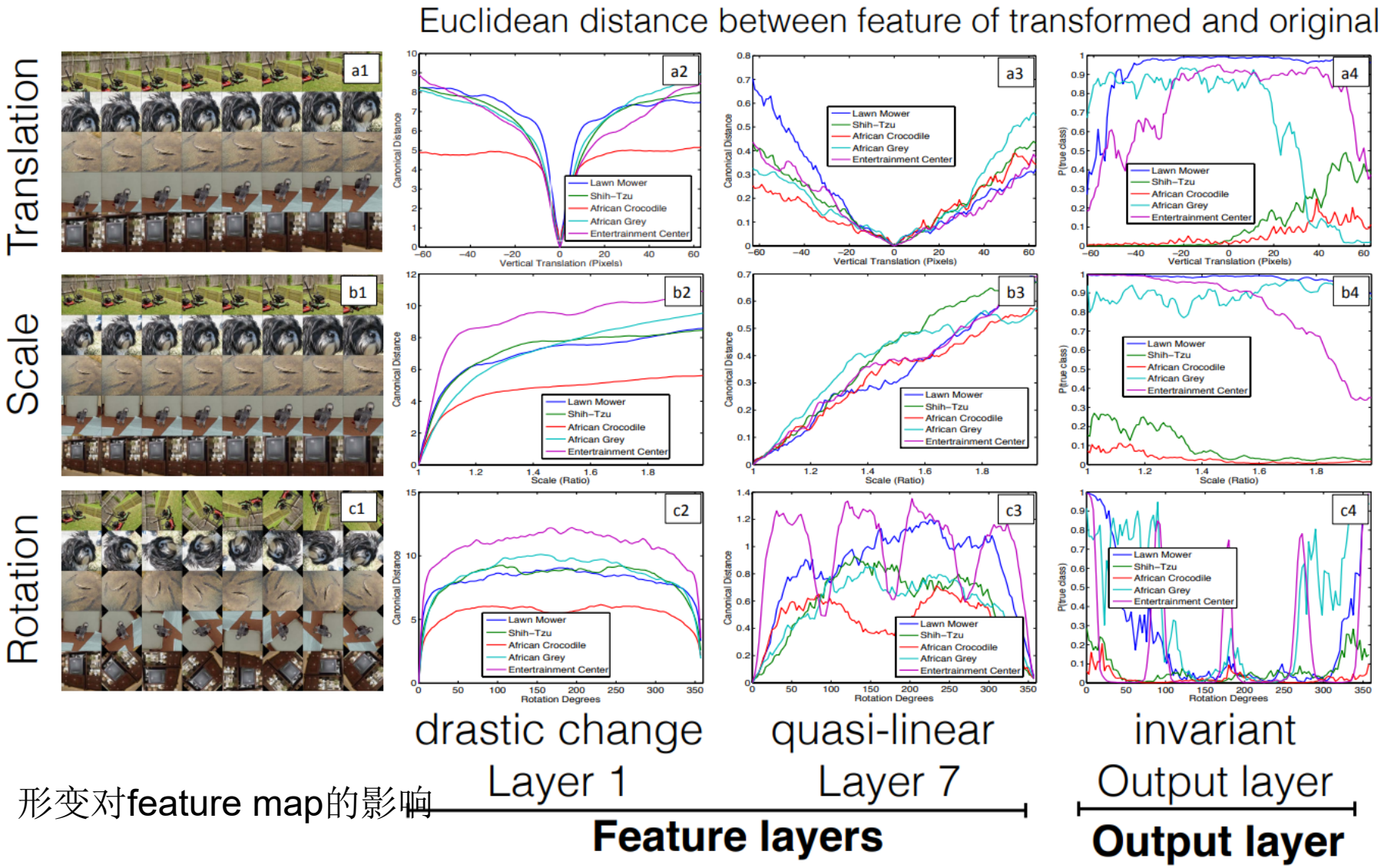
其他：filter的学习过程

- 底层收敛更快
- 高层收敛开始的晚
- 突变：不同图片导致不同的强激活信号



https://courses.cs.washington.edu/courses/cse590v/14au/cse590v_dec5_DeepVis.pdf

其他：高层特征的不变性



形变对feature map的影响

https://courses.cs.washington.edu/courses/cse590v/14au/cse590v_dec5_DeepVis.pdf

其他

- 针对可视化分析的结果

- 可以改进原模型

Error %	Val Top-1	Val Top-5	Test Top-5
(Gunji et al., 2012)	-	-	26.2
(Krizhevsky et al., 2012), 1 convnet	40.7	18.2	– –
(Krizhevsky et al., 2012), 5 convnets	38.1	16.4	16.4
(Krizhevsky et al., 2012)*, 1 convnets	39.0	16.6	– –
(Krizhevsky et al., 2012)*, 7 convnets	36.7	15.4	15.3
Our replication of (Krizhevsky et al., 2012), 1 convnet	40.5	18.1	– –
1 convnet as per Fig. 3	38.4	16.5	– –
5 convnets as per Fig. 3 – (a)	36.7	15.3	15.3
1 convnet as per Fig. 3 but with layers 3,4,5: 512,1024,512 maps – (b)	37.5	16.0	16.1
6 convnets, (a) & (b) combined	**36.0**	**14.7**	**14.8**

- 验证低层CNN的可迁移性

# Train	Acc % 15/class	Acc % 30/class
(Bo et al., 2013)	–	81.4 ± 0.33
(Jianchao et al., 2009)	73.2	84.3
Non-pretrained convnet	22.8 ± 1.5	46.5 ± 1.7
ImageNet-pretrained convnet	$\mathbf{83.8 \pm 0.5}$	$\mathbf{86.5 \pm 0.5}$

Table 4. Caltech-101 classification accuracy for our convnet models, against two leading alternate approaches.

THANKS!