

# CPU9444\_Final3\_yann

**Autor:** Yann

**latest:** Aug 2024

**Reference:**

**University of Melbourne - COMP30024 Lecture slides**

**USYD - COMP5329 Lecture slides**

**UNSW - COMP9414 Lecture slides**

**UNSW - COMP9814 Lecture slides**

Convolutional Neural Network

Image Modality

CNNs Components

Convolutional Layer

Pooling Layer

Fully Connected Layer

Convolutional Operation

Stride

Padding

Receptive Field

Dilated Convolution

CNN VS MLP

Parameter Efficiency

Inductive Bias

CNN Architecture

Image Net

LeNet — 1998

Alex Net — 2012

ZFNet - 2013

More Filters

VGG — 2014

Smaller Kernel

GoogLeNet — 2014

Inception Module

Global Pooling

Auxiliary Loss

Res Net — 2015

Residual Connection

Deeper Structure

Computer Vision Tasks

Vision Tasks

Classification + Localisation

Object Detection & Segmentation

Multi-task Dataset - COCO

Region CNN (R-CNN)

Selective Search

Feature Extraction

Bounding Box Regression

Fast R-CNN

Shared Feature Map

Region of Interest Pooling

Faster RCNN

Region Proposal Network

Masked R-CNN

ROIAlign

From R-CNN to Mask R-CNN

R-CNN (Region-based Convolutional Neural Networks)

Fast R-CNN

Faster R-CNN

Mask R-CNN

Recurrent Neural Network

Text Modality

One-Hot Encoding

Bag of Words

词嵌入 (Word Embeddings)

Recurrent Neural Network

Time Dependency

Vanishing Gradient

Memory of RNN

LSTM

Gates

Additive Gradient

Gated Recurrent Unit

Transformer

Transformer Architecture

Attention Mechanism

Self-Attention

Transformer architecture

Multi-head attention

Positional Encoding

Analysis

BERT

Masked Language Model

CLS Token

InstructGPT

AutoRegressive Loss

RLHF (Reinforcement Learning from Human Feedback)

Vision Transformer

Exercise

Ex 1

Ex 2

Ex 3

Ex 4

Ex 5

Ex 6

Ex 7

Ex 8

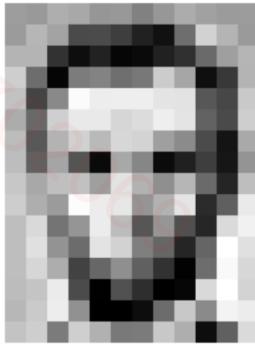
# Convolutional Neural Network

## Image Modality

图像作为二维或三维数据结构，能够以数值形式表示光的亮度和颜色信息。图片（2D）本质上是一个由光亮度表示的矩阵。从物理角度来看，图片是对现实世界中光的一种抽象表示。具体来说，当你看到一张图片时，你实际上看到的是一个二维矩阵，其中每个元素（或像素）代表了一个特定位置的光亮度。

### Images are represented as matrices of pixel values

- Black and white images: a value from 0 (black) to 255 (white) – 1 matrix
- Color images: the same but for 3 channels: red, green and blue, so 3 matrices



157	153	174	164	150	182	129	151	172	161	155	156
155	182	163	74	75	62	33	17	110	210	180	154
180	180	50	14	54	6	10	53	48	106	159	181
206	109	5	124	131	111	120	204	166	15	56	180
194	68	137	251	237	239	239	238	227	87	71	201
172	106	297	232	239	214	220	239	228	91	74	206
188	88	179	209	185	218	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	164	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	34	218	241
190	224	147	106	227	210	127	102	36	101	255	224
190	214	173	64	103	143	96	50	2	109	249	215
187	196	239	75	1	81	47	0	6	217	256	211
183	202	237	148	0	9	12	108	200	138	243	236
195	204	132	207	177	121	123	209	175	13	95	218

167	153	174	184	160	192	129	181	172	161	186	156
155	182	163	74	76	62	33	17	110	210	180	154
180	180	50	14	34	6	10	33	48	106	159	181
206	109	5	124	131	111	120	204	166	16	56	180
194	68	137	251	237	239	239	238	227	87	71	201
172	105	207	233	233	214	220	239	228	98	74	206
188	88	179	209	185	218	211	158	139	75	20	169
189	97	165	84	10	168	134	11	31	62	22	148
199	164	191	193	158	227	178	143	182	106	36	190
205	174	155	252	236	231	149	178	228	43	95	234
190	216	116	149	236	187	85	150	79	34	218	241
190	224	147	106	227	210	127	102	36	101	255	224
190	214	173	64	103	143	96	50	2	109	249	215
187	196	239	75	1	81	47	0	6	217	256	211
183	202	237	148	0	9	12	108	200	138	243	236
195	204	132	207	177	121	123	209	175	13	95	218

Image ref: <https://ai.stanford.edu/~syyeung/cvweb/tutorial1.html>

USYD COMP5329 Lecture slides - Week5

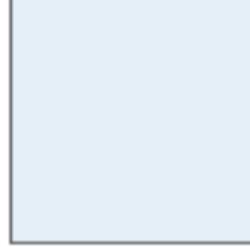
在这个二维矩阵中，每一个小方格就是一个像素。灰度图像中每个像素通常用一个介于 0 到 255 之间的整数来表示，其中 0 表示黑色，255 表示白色，其他值表示不同的灰度。尽管 0-255 是最常见的像素值范围，但这并不是唯一的选择。根据需要，这个范围可以被扩展。例如，在医学成像或卫星成像中，可能需要更高的精度。像素值范围越大，每个像素能表示的信息就越多。这意味着图像可以捕获更多关于光的细节（例如，亮度、颜色等）。灰度图像可以表示为一个二维矩阵！：

$$I = \begin{bmatrix} I_{11} & I_{12} & \cdots & I_{1n} \\ I_{21} & I_{22} & \cdots & I_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ I_{m1} & I_{m2} & \cdots & I_{mn} \end{bmatrix}$$

其中  $I_{ij}$  表示像素  $i, j$  位置的灰度值，范围通常在 0 到 255 之间。

彩色图像中每个像素通常由三个数值组成，分别代表红、绿和蓝（RGB）三种颜色的亮度。RGB 图片可以从一个“3D”的角度来理解，因为它实际上是由三个单色（即单通道）图像组合而成的。这三个单色图像分别表示红色（Red）、绿色（Green）和蓝色（Blue）三个颜色通道。这三个通道的图像具有相同的尺寸，并且它们是按照深度方

向（第三个维度）堆叠在一起的。彩色图像有红、绿和蓝这三个通道，分别是一个 2D 图像，它们堆叠在一起就形成了一个 3D 数据体。在每个像素位置，你都会有三个数值，分别对应红、绿和蓝通道。可以想象在每个像素位置都有一个小小的三维向量。

White	Black	Pale Blue	Gold
			
RGB 255, 255, 255	RGB 0, 0, 0	RGB 255, 239, 248	RGB 228, 189, 79
			

<https://www.datarobot.com/blog/introduction-to-computer-vision-what-it-is-and-how-it-works/>

彩色图像可以表示为一个三维矩阵！：

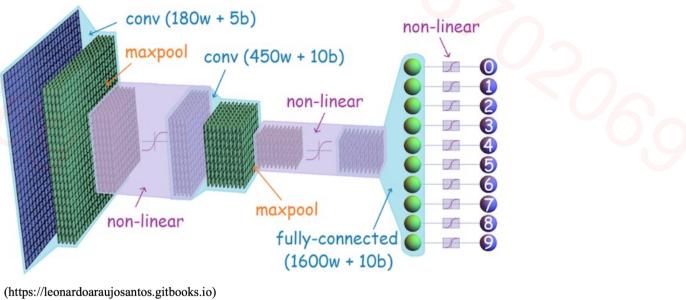
$$I = \begin{bmatrix} [R_{11} & G_{11} & B_{11}] & [R_{12} & G_{12} & B_{12}] & \cdots & [R_{1n} & G_{1n} & B_{1n}] \\ [R_{21} & G_{21} & B_{21}] & [R_{22} & G_{22} & B_{22}] & \cdots & [R_{2n} & G_{2n} & B_{2n}] \\ \vdots & \vdots & \ddots & \vdots \\ [R_{m1} & G_{m1} & B_{m1}] & [R_{m2} & G_{m2} & B_{m2}] & \cdots & [R_{mn} & G_{mn} & B_{mn}] \end{bmatrix}$$

其中  $R_{ij}$ ,  $G_{ij}$ ,  $B_{ij}$  分别表示像素  $i, j$  位置的红、绿、蓝三通道的亮度值。

## CNNs Components

卷积神经网络（Convolutional Neural Networks，简称CNN）是一种专门用于处理具有网格状拓扑结构数据（如图像）的深度神经网络。CNN的主要组成部分包括卷积层、池化层、全连接层。

- Convolutional Layer
- Pooling
- Fully-connected Layer



(<https://leonardoaraujosantos.gitbooks.io>)

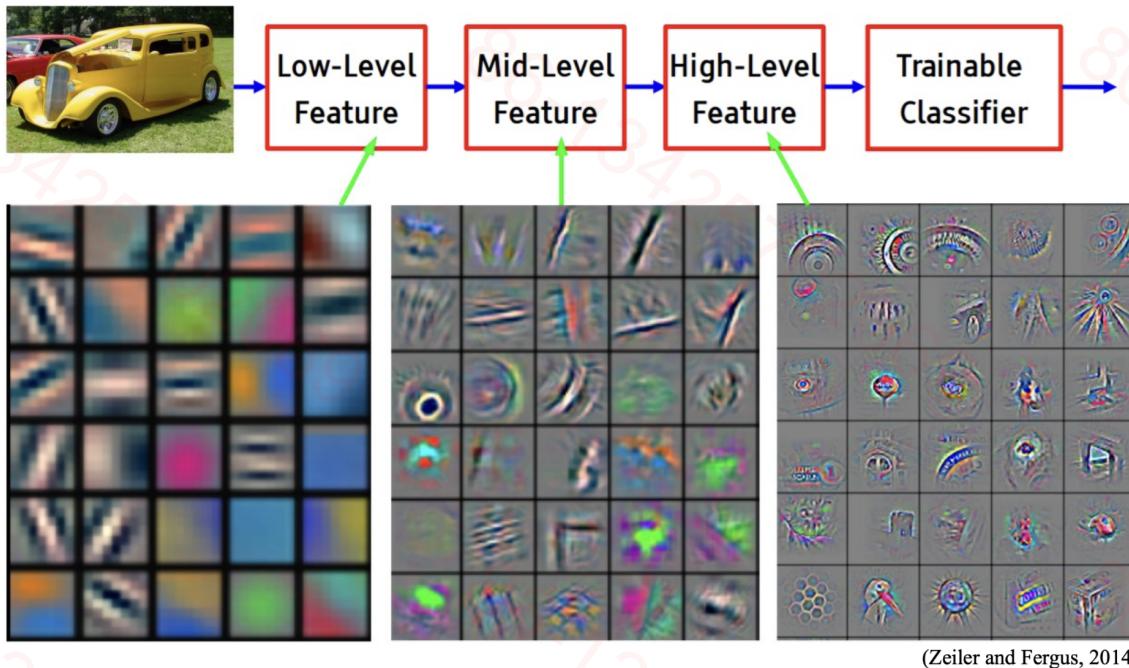
The University of Sydney

Page 8

USYD COMP5329 Lecture slides - Week5

CNN通过多层结构提取图像中的特征，每一层的特征图（Feature Map）都基于前一层的输出。随着网络层数的增加，特征图所表示的特征也变得越来越复杂和抽象。以下详细介绍CNN中各层特征图的生成及其复杂性的演变。前几层卷积层主要提取图像的低级特征，如边缘、线条和简单的纹理。这些特征图主要关注图像的局部区域，反映了基本的几何形状和简单模式。中间层的卷积层将前几层提取到的低级特征组合起来，形成更加复杂的模式，如角点、圆形和曲线。这些特征图开始捕捉到图像中的结构性信息，并逐渐能够识别出一些部分物体的轮廓和形状。深层卷积层进一步组合中级特征，提取出更高级的抽象特征，如物体的局部和整体形状。这些特征图能够捕捉到整个图像中的复杂模式和物体，具有很强的辨识能力。例如，在人脸识别任务中，后面的特征图能够识别人脸的整体结构和细节特征。

- Give insight into the function of intermediate feature layers and the operation of the classifier



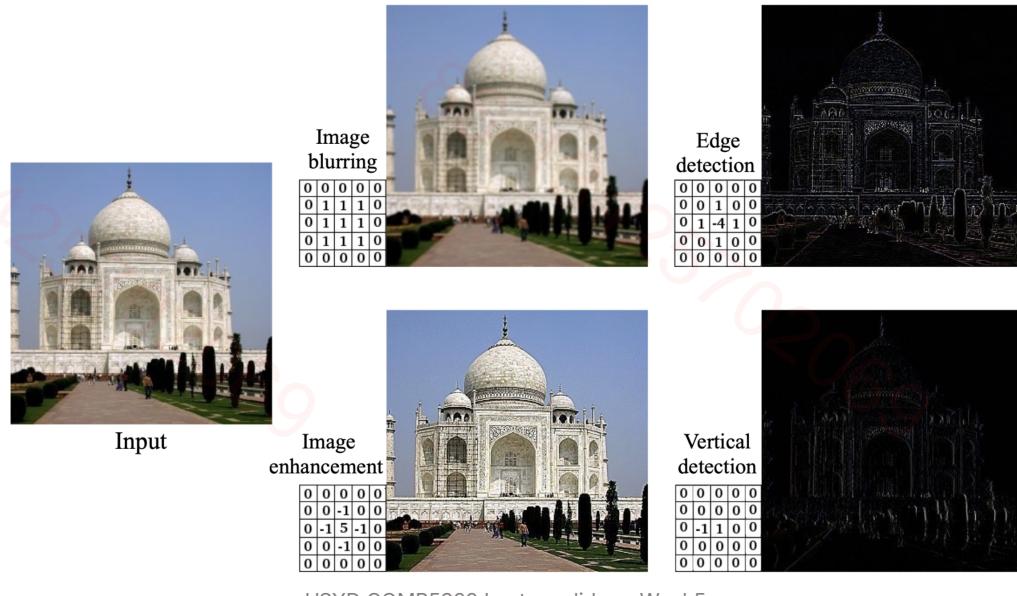
(Zeiler and Fergus, 2014)

USYD COMP5329 Lecture slides - Week5

- **卷积层**：主要负责提取输入数据（如图像）中的特征。
- **池化层**：池化层通常位于连续的卷积层之间，用来降低特征图的空间尺寸。
- **全链接层**：负责将前面卷积层和池化层提取并压缩的特征图转化为最终的输出，如分类标签。

## Convolutional Layer

卷积层通过与输入数据进行卷积运算来提取局部特征。卷积运算是通过卷积核（filter）在输入数据上滑动，并计算局部区域内加权和的过程。在这一层中，通过一系列的滤波器（或称为卷积核）对输入数据进行卷积操作。每个滤波器负责从输入数据中提取一种特定的特征，比如边缘、角点或者纹理等。滤波器在输入数据上滑动，计算滤波器与其覆盖的局部区域的点积，生成特征图（feature map），这个过程称为特征提取。



USYD COMP5329 Lecture slides - Week5

设输入数据为  $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$ ，卷积核为  $\mathbf{K} \in \mathbb{R}^{k_H \times k_W \times C}$ ，其中  $H$  和  $W$  分别是输入数据的高度和宽度， $C$  是输入数据的通道数， $k_H$  和  $k_W$  分别是卷积核的高度和宽度。输出特征图  $\mathbf{Y}$  计算如下：

$$\mathbf{Y}_{i,j} = \sum_{w=0}^{k_H-1} \sum_{c=0}^{k_W-1} \sum_{c=0}^{C-1} \mathbf{X}_{i+h, j+w, c} \cdot \mathbf{K}_{h, w, c} + b$$

其中， $b$  是偏置项， $\mathbf{Y}_{i,j}$  是输出特征图在位置  $(i, j)$  的值。

1	2	0		1	0	1
2	1	1	0	0	0	1
1	0	0	2	1	0	
2	0	0	0	2	1	
0	1	1	2	0	2	
1	0	1	0	1	1	

1	0	-1	
-1	0	0	
0	0	1	

Filter 1

-1	2	0	0
0	1	3	-2
0	0	-1	4
3	-1	-2	-2

0	2	1	
0	1	-1	
-1	1	0	

Filter 2

3			

⋮

Page 23

The University of Sydney

USYD COMP5329 Lecture slides - Week5

## Pooling Layer

池化层通过对输入数据进行下采样操作来减小数据的空间尺寸，从而降低计算量，并且让CNN具有平移不变性的特点，有助于提取更加鲁棒的特征，并且提高模型的泛化能力。常见的池化操作包括最大池化（Max Pooling）和平均池化（Average Pooling）。

max pooling



average pooling



<https://blog.paperspace.com/pooling-in-convolutional-neural-networks/>

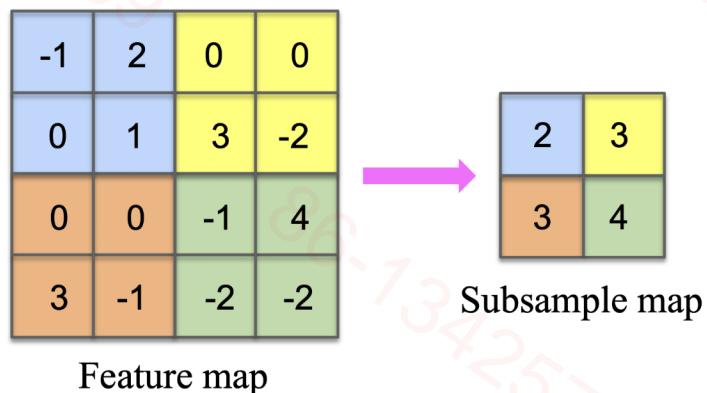
设输入数据为  $\mathbf{X} \in \mathbb{R}^{H \times W \times C}$ , 池化窗口大小为  $p_H \times p_W$ , 池化操作的输出  $\mathbf{Y}$  计算如下:

最大池化 :

$$\mathbf{Y}_{i,j,c} = \max_{0 \leq h < p_H, 0 \leq w < p_W} \mathbf{X}_{i \cdot p_H + h, j \cdot p_W + w, c}$$

## Max pooling

- Filter size: (2,2)
- Stride: (2,2)
- Pooling ops:  $\max(\cdot)$



USYD COMP5329 Lecture slides - Week5

平均池化：

$$\mathbf{Y}_{i,j,c} = \frac{1}{p_H \cdot p_W} \sum h = 0^{p_H-1} \sum_{w=0}^{p_W-1} \mathbf{x}_{i \cdot p_H + h, j \cdot p_W + w, c}$$

## Average pooling

- Filter size: (2,2)
- Stride: (2,2)
- Pooling ops: mean( $\cdot$ )

-1	4	1	2
0	1	3	-2
1	5	-2	6
3	-1	-2	-2

Feature map

4	3
5	6

Max pooling

1	1
2	0

Average pooling

USYD COMP5329 Lecture slides - Week5

## Fully Connected Layer

全连接层将输入数据展平成一个向量，并通过一个线性变换将其映射到输出空间。全连接层通常用于网络的最后几层，以生成分类器或回归器的输出。在全连接层，网络将学习到的特征组合和映射到目标输出，如分类或回归任务中的预测结果。设输入向量为  $\mathbf{x} \in \mathbb{R}^d$ ，权重矩阵为  $\mathbf{W} \in \mathbb{R}^{n \times d}$ ，偏置向量为  $\mathbf{b} \in \mathbb{R}^n$ ，则输出  $\mathbf{y} \in \mathbb{R}^n$  计算如下：

$$\mathbf{y} = \mathbf{W}\mathbf{x} + \mathbf{b}$$

## Convolutional Operation

卷积层是卷积神经网络（CNN）通过在输入图像上应用多个滤波器（卷积核）来提取特征。滑动窗口是卷积运算中卷积核在输入图像上移动的过程。每次移动时，卷积核与局部区域进行卷积计算，得到特征图（feature map）的一个值。

### Stride

步长定义了滤波器在输入特征图上的移动间隔，通常用于调整特征图的大小和控制模型的计算复杂度。当我们在卷积层或池化层应用一个滤波器时，滤波器不一定每次都移动一个像素点。步长是控制滤波器移动几个像素点的参数。例如，如果步长为1 (stride=1)，那么滤波器每次向右或向下移动一个像素点。如果步长为2 (stride=2)，滤波器每次移动两个像素点，这样它覆盖的区域间会有跳跃。不同的stride将会影响特征图的尺寸，通常一个越大的stride就会导致一个更小的feature map size。如果我们希望output更小一些，那么就可以选择大的stride，如果需要保留更多的信息，那么stride就需要小一些，stride最小为1。

1	2	0	1	0	1
2	1	1	0	0	1
1	0	0	2	1	0
2	0	0	0	2	1
0	1	1	2	0	2
1	0	1	0	1	1

1	0	-1
-1	0	0
0	0	1

-1	2		

Stride = 1

The *stride size* is defined by how much you want to shift your filter at each step.

USYD COMP5329 Lecture slides - Week5

1	2	0	1	0	1
2	1	1	0	0	1
1	0	0	2	1	0
2	0	0	0	2	1
0	1	1	2	0	2
1	0	1	0	1	1

1	0	-1
-1	0	0
0	0	1

-1	0

Stride = 3

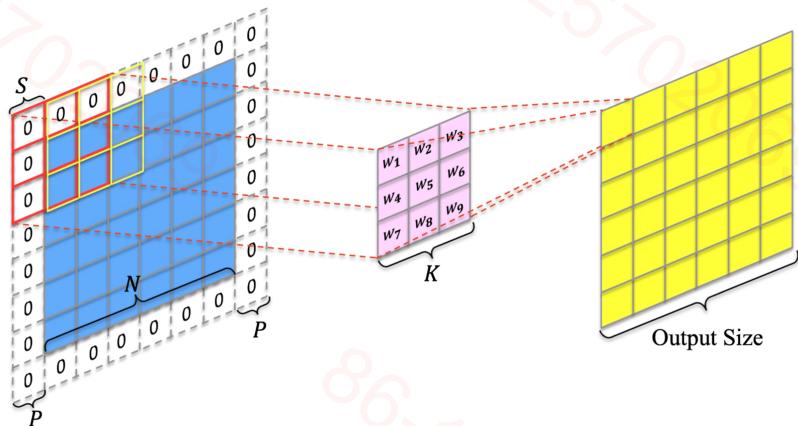
USYD COMP5329 Lecture slides - Week5

## Padding

填充是在输入图像的边缘添加额外的像素，以控制卷积运算后特征图的尺寸并保留边缘信息。设输入图像大小为  $H \times W$ ，卷积核大小为  $k_H \times k_W$ ，步长为  $s$ ，填充大小为  $p$ 。输出特征图的尺寸计算如下：

$$H_{\text{out}} = \left\lceil \frac{H - k_H + 2p}{s} \right\rceil + 1$$

$$W_{\text{out}} = \left\lceil \frac{W - k_W + 2p}{s} \right\rceil + 1$$



The University of Sydney

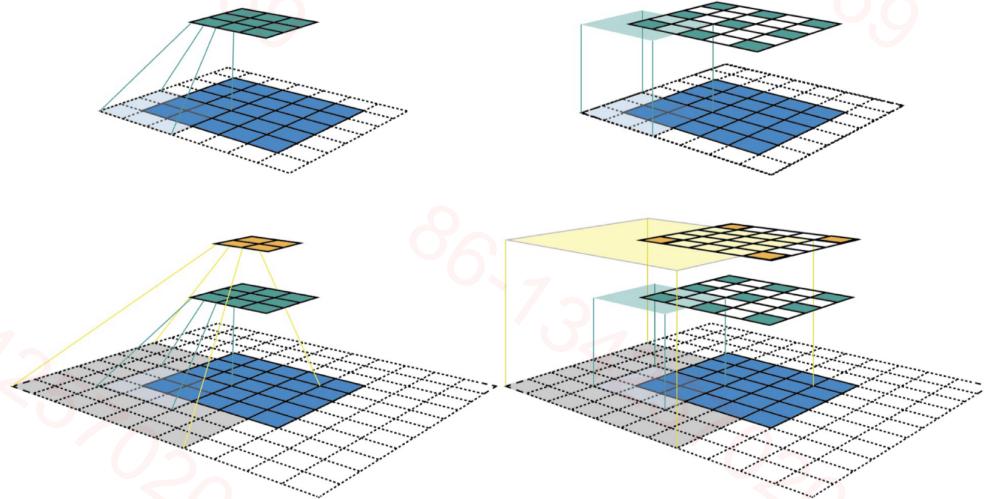
Page 20

USYD COMP5329 Lecture slides - Week5

## Receptive Field

感受野表示特征图中的一个像素能够感受到的原始图像的区域。感受野越大，说明特征图中的像素包含了更多原始图像的上下文信息。但是对于深层的CNN，计算感受野变得复杂，尤其是当网络使用不同大小的滤波器、步长和填充策略时。在深层网络中，感受野可以迅速增大，覆盖大部分甚至全部的输入图像，这意味着网络深层的特征能够捕捉到输入数据中的全局信息。但是感受野增大也会丢失掉局部细节，所以是一个global和local的trade-off。

- The receptive field in Convolutional Neural Networks (CNN) is the **region of the input space that affects a particular unit of the network**.
- In this example, we use the convolution filter  $\mathbf{W}$  with size  $K = 3 \times 3$ , padding  $P = 1$ , stride  $S = 2 \times 2$ .



The University of Sydney

Page 37

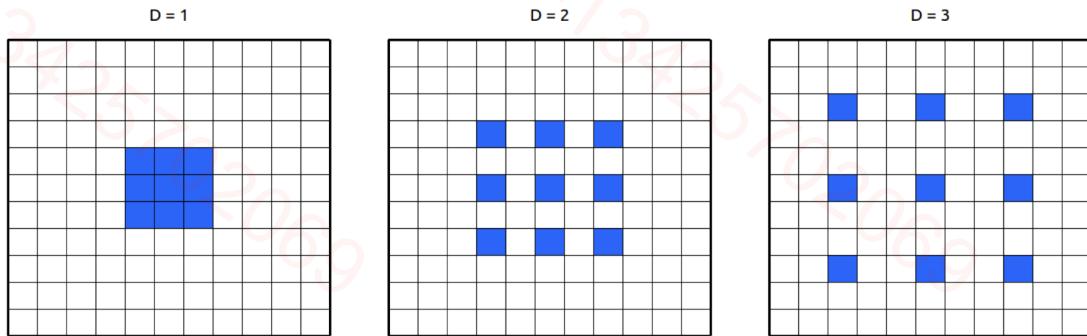
USYD COMP5329 Lecture slides - Week5

设第  $l$  层卷积层的感受野为  $R_l$ , 滤波器大小为  $k$ , 步长为  $s$ , 填充大小为  $p$ , 则有:

$$R_l = R_{l-1} + (k-1) \prod_{i=1}^{l-1} s_i$$

### Dilated Convolution

我们之前的卷积核是 $3 \times 3$ 紧密连接在一块的, 然而空洞卷积在卷积核上做文章。空洞卷积通过在卷积核的元素之间引入间隙 (扩张率 Dilation rate) 来增加感受野。由于不需要增加卷积核的尺寸, 参数数量保持不变, 不增加卷积核的参数数量, 这使得空洞卷积在增加深度和复杂性时保持较高的参数效率。因为我们一次看到的信息更多了, 但是同时也丢失了局部信息。



(Yu et al, 2015)

扩张率决定了卷积核元素之间的距离。扩张率等于1是普通卷积，卷积核的每个元素都紧密相连，没有间隔。等于2的时候，卷积核的每个元素之间跳过一个像素点。这意味着 $3 \times 3$ 的卷积核，实际上会覆盖一个 $5 \times 5$ 的区域，但只使用 $3 \times 3$ 个参数。设扩张率为  $d$ ，输入图像为  $\mathbf{X}$ ，卷积核为  $\mathbf{K}$ ，空洞卷积的输出  $\mathbf{Y}$  计算如下：

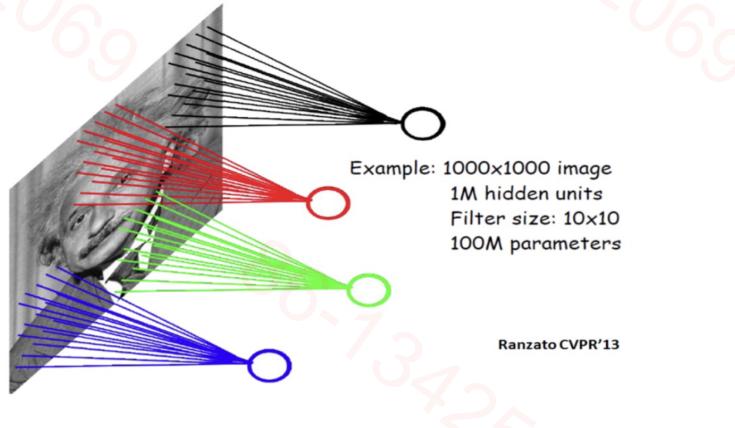
$$\mathbf{Y}_{i,j} = \sum h = 0^{k_H-1} \sum_{w=0}^{k_W-1} \sum_{c=0}^{C-1} \mathbf{X}_{i+d \cdot h, j+d \cdot w, c} \cdot \mathbf{K}_{h,w,c} + b$$

## CNN VS MLP

### Parameter Efficiency

**CNN（卷积神经网络）**通过局部连接和权重共享显著减少了参数量。卷积层中的每个卷积核只与局部区域连接，并且在整个图像上共享相同的权重。卷积层中的卷积核具有权重共享，在不同位置应用相同的权重，这不仅减少了参数数量，还提高了模型的参数效率。池化层通过下采样操作进一步减少了特征图的尺寸，从而减少了参数和计算量。但是**MLP（多层感知器）**中的每个神经元与上一层的所有神经元全连接，这导致参数数量随着输入维度和网络深度的增加迅速增长。MLP没有局部连接的概念，每个神经元都连接到所有输入，这导致参数冗余和计算复杂度高。

- Locally connected neural net
  - Sparse connectivity: a hidden unit is only connected to a local patch
  - It is inspired by biological systems, where a cell is sensitive to a small sub-region, called a receptive field.
  - Here, the receptive field can be called as filter or kernel



The University of Sydney

Page 50

USYD COMP5329 Lecture slides - Week5

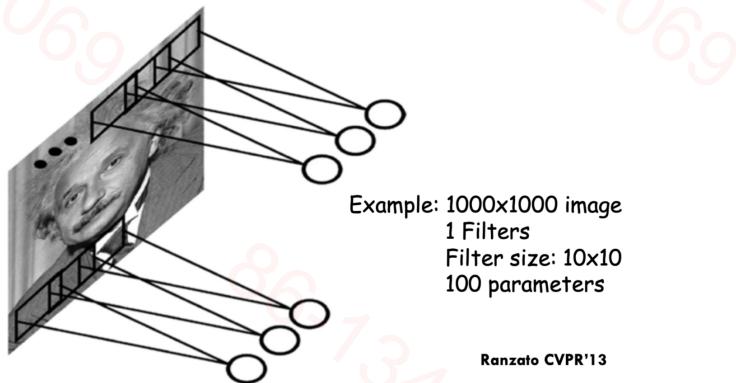
设输入图像尺寸为  $H \times W$ ，通道数为  $C$ ，卷积核尺寸为  $k \times k$ ，卷积核个数为  $N$ 。CNN参数量为  $k^2 \cdot C \cdot N$ ，MLP参数量为  $H \cdot W \cdot C \cdot$  神经元个数。显然，CNN的参数量远小于MLP。

## Inductive Bias

CNN通过局部连接和权重共享假设图像的局部特征在整个图像中是相似的。这种归纳偏置使得CNN能够有效地学习到图像的局部特征。通过池化层，CNN能够实现平移不变性，即网络对输入图像的平移变化具有鲁棒性。这使得CNN在处理图像时更加有效。MLP没有对输入数据的结构作出特殊假设，因此缺乏局部特征提取和平移不变性的能力。

- Shared weights

- Translation invariance: capture statistics in local patches and they are independent of locations
- Hidden nodes at different locations share the same weights. It greatly reduces the number of parameters to learn



The University of Sydney

Page 52

USYD COMP5329 Lecture slides - Week5

## CNN Architecture

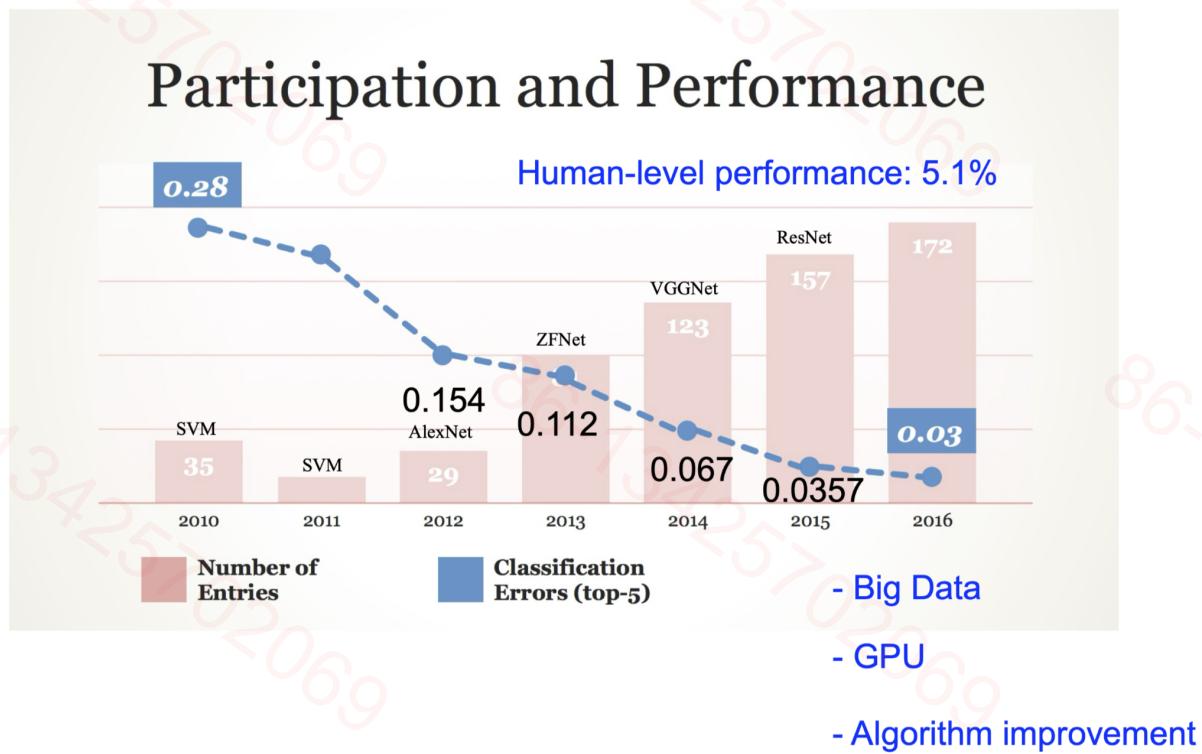
### Image Net



USYD COMP5329 Lecture slides - Week6

**ImageNet** 是机器学习和计算机视觉领域中一个大规模视觉数据库（benchmark dataset），包含超过 1400 万张标注的图像，覆盖了 20000 多个类别。这些图像中的大部分都有准确的标注，确保了数据的质量和可信度。ImageNet 主要用于图像分类任务。最广为人知的是 ImageNet 大规模视觉识别挑战赛（ILSVRC），该挑战赛要求参赛者在 1000 个类别上对图像进行分类，并且每个类别有大约 1000 张图像用于训练。ImageNet 的出现极大地推动了深度学习，特别是卷积神经网络（CNN）的发展。通过在 ImageNet 上进行训练，研究人员可以测试和比较不同模型的性能，从而

不断改进模型结构和训练方法。作为一个标准的基准数据集，ImageNet 为不同的模型提供了一个公平的测试平台。研究人员可以在相同的数据集上比较不同算法和模型的性能，确保实验结果的可重复性和可比性。



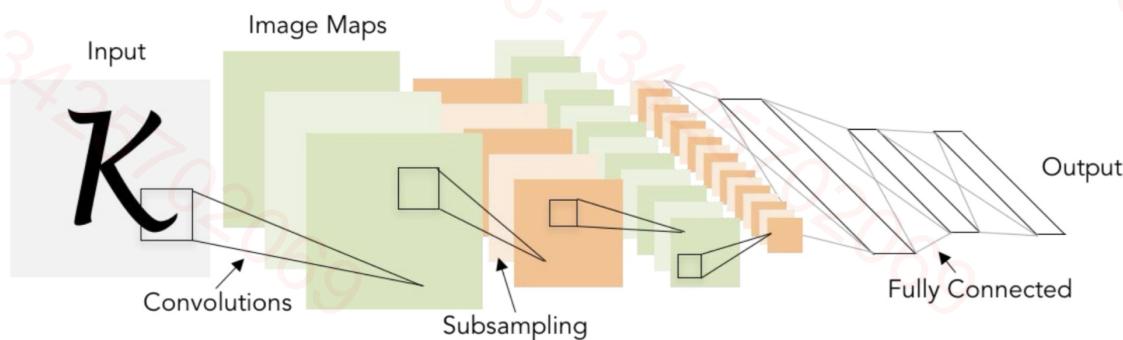
Model	AlexNet	ZF Net	GoogLeNet	Resnet
Year	2012	2013	2014	2015
#Layer	8	8	22	152
Top 5 Acc	15.4%	11.2%	6.7%	3.57%
Data augmentation	✓	✓	✓	✓
Dropout	✓	✓		
Batch normalization				✓

ILSVRC 比赛对推动深度学习的发展起到了至关重要的作用。2012 年，AlexNet 模型在 ILSVRC 比赛中取得了突破性的成绩，其大幅度降低了错误率，引发了学术界和工业界对深度学习的广泛关注。随后，VGG、GoogLeNet、ResNet 等一系列优秀的深度学习模型相继问世，进一步提升了图像识别的性能。

## LeNet — 1998

第一个卷积神经网络被设计出来是为了识别图像信息，采用了backpropagation算法。也就是我们在第一周讲到的后传算法。这个算法能够在手写数据集，也就是著名的MINIST上取得SOTA（State-of-the-art）的分数。这篇文章有三个作者，分别是LeCun，Hinton和Bendigo。这个算法发布后，一作LeCun获得了图灵奖，Hinton是公认的神经网络之父，Bengio是GAN的作者。LeNet 的设计奠定了卷积神经网络的基础，卷积层、池化层和全连接层成为现代卷积神经网络的标准组件。

### □ LeNet [LeCun et al., 1998]



- Architecture is [CONV-POOL-CONV-POOL-FC-FC]
- CONV: 5x5 filter, stride=1
- POOL: 2x2 filter, stride=2

USYD COMP5329 Lecture slides - Week6

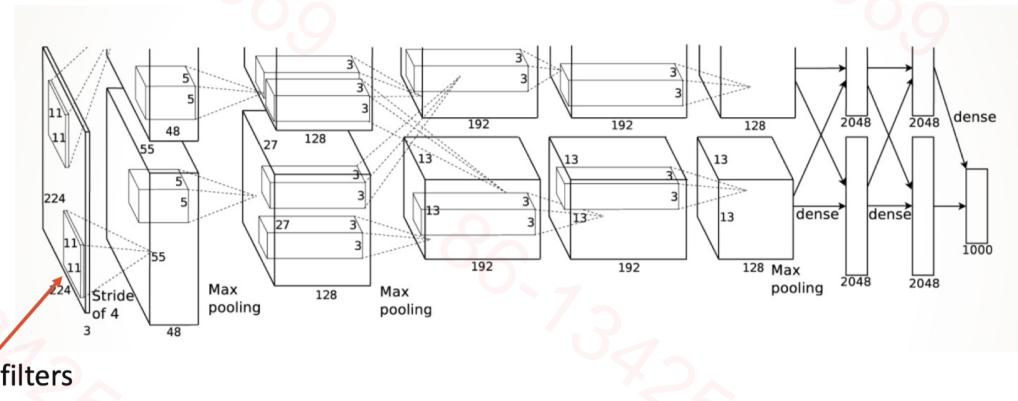
这些创新使得 LeNet 能够在图像识别任务中取得卓越的效果，并为后续的深度学习研究提供了重要的启示。LeNet 的关键创新点：

- **卷积层 (Convolutional Layer)** : LeNet 是第一个将卷积层引入神经网络的模型。卷积层通过卷积操作在输入图像上滑动滤波器，从局部区域提取特征。这种方法不仅减少了模型参数，还提高了模型对图像平移、旋转等变换的鲁棒性。
- **池化层 (Pooling Layer)** : LeNet 采用了池化层（平均池化），通过对卷积层输出进行降采样，进一步减少特征图的尺寸和计算量。池化层有助于提取主要特征，同时减少过拟合的风险。
- **全连接层 (Fully Connected Layer)** : LeNet 将卷积层和池化层提取的特征展平，并通过全连接层进行分类。全连接层类似于多层感知机（MLP），能够学习到输入特征与输出类别之间的复杂非线性关系。
- **反向传播算法 (Backpropagation)** : LeNet 成功应用了反向传播算法，通过梯度下降优化模型参数。这种训练方法使得深度神经网络能够高效地学习，并在手写数字识别任务（如 MNIST 数据集）上取得了优异的表现。

## Alex Net — 2012

**AlexNet** 是由 Alex Krizhevsky、Ilya Sutskever 和 Geoffrey Hinton 在 2012 年提出的，并在当年的 ImageNet 大规模视觉识别挑战赛 (ILSVRC) 中取得了突破性的成绩。这一模型可以说是现代深度学习历史上的一个里程碑，因为它首次证明了深度神经网络能在大规模数据集上实现高效、准确的视觉识别。

## □ AlexNet [Krizhevsky et al. 2012] - 5 conv layers, 3 fully connected layers.



11x11 filters

- Activation function: ReLU
- Data Augmentation
- Dropout (drop rate=0.5)
- Local Response Normalization
- Overlapping Pooling

The University of Sydney

Page 8

USYD COMP5329 Lecture slides - Week6

主要特点为：

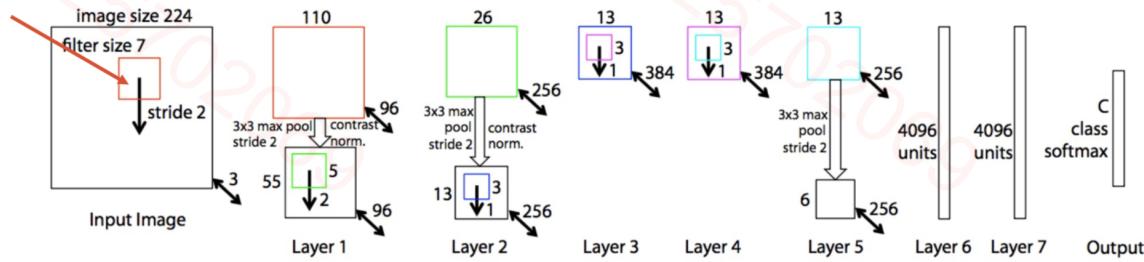
- 使用 **ReLU 激活函数**：AlexNet 使用了 ReLU（Rectified Linear Unit）作为激活函数，相较于传统的 sigmoid 和 tanh 函数，ReLU 可以加速收敛速度，缓解梯度消失问题。
- **数据增强 (Data Augmentation)**：为了防止过拟合，AlexNet 采用了数据增强技术，包括随机裁剪、镜像翻转和颜色抖动等。这些技术增加了训练数据的多样性，提高了模型的泛化能力。
- **Dropout 正则化**：为了进一步防止过拟合，AlexNet 在全连接层中引入了 Dropout 正则化技术。在训练过程中，随机丢弃部分神经元，以减少神经元间的依赖，从而提高模型的泛化能力。
- **重叠池化 (Overlapping Pooling)**：AlexNet 采用了重叠池化（即池化窗口之间有部分重叠），与传统的不重叠池化相比，重叠池化可以提高模型的鲁棒性，减轻过拟合现象。
- **使用分布 GPU 加速**：AlexNet 是第一个广泛使用 GPU 进行训练的深度神经网络。由于深度神经网络的计算量巨大，GPU 的高并行计算能力显著加速了模型训练过程。由于当时的硬件限制，AlexNet 将模型分布在两块 GPU 上进行训练。模型的某些层被切分开来，分别放置在不同的 GPU 上进行并行计算，然后再将结果整合起来。这种方法有效地利用了 GPU 的计算能力，解决了内存瓶颈问题。

## ZFNet - 2013

**ZFNet** (由 Matthew D. Zeiler 和 Rob Fergus 提出，也称为 Zeiler and Fergus Net) 是 AlexNet 的一个改进版本，主要通过增加卷积层的滤波器数量和调整网络结构，进一步提升了模型在图像分类任务中的性能。ZFNet 在 2013 年的

ImageNet挑战赛中表现出色，展示了其强大的特征提取能力和分类能力。

### 7x7 filters



- An improved version of AlexNet: top-5 error from 16.4% to 11.7%
- First convolutional layer: 11x11 filter, stride=4 -> 7x7 filter, stride=2
- The number of filters increase as we go deeper.

USYD COMP5329 Lecture slides - Week6

### More Filters

ZFNet在每一层中使用了更多的滤波器（filters），增加了网络的特征提取能力。具体来说，ZFNet在前几层中增加了卷积核的数量，从而能够捕捉到更多的图像细节和特征。这一改进使得模型在处理复杂图像时表现得更加出色。

ZFNet的一大贡献在于引入了一种新的可视化技术，使得研究人员能够更直观地理解卷积神经网络内部的工作原理。通过可视化每一层的激活特征图，研究人员可以看到网络如何逐层提取图像特征，从而更好地设计和优化网络结构。

## VGG — 2014

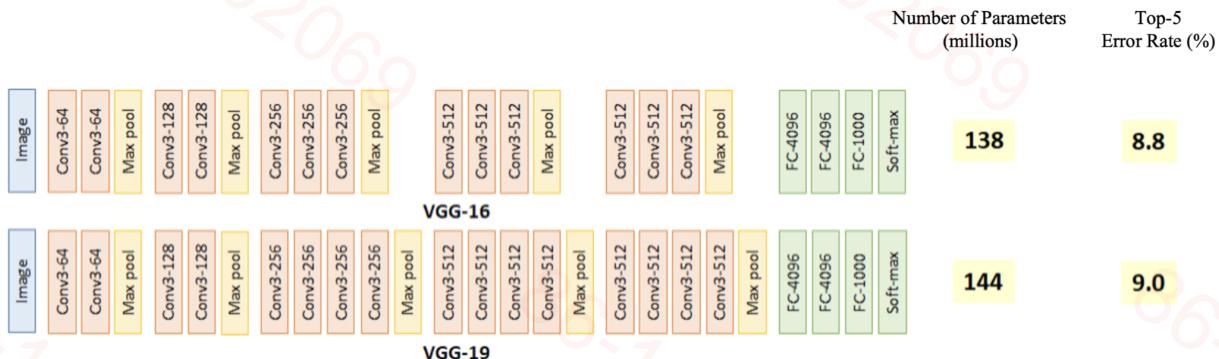
**VGGNet**是由牛津大学视觉几何组（Visual Geometry Group，简称VGG）提出的卷积神经网络模型。它在2014年的ImageNet大规模视觉识别挑战赛（ILSVRC）中表现出色，成为了深度学习领域的一个重要里程碑。

### Small filters:

- 3x3 convolutional layers (stride 1, pad 1)
- 2x2 max-pooling, stride 2

### Deeper networks:

- AlexNet: 8 layers
- VGGNet: 16 or 19 layers



USYD COMP5329 Lecture slides - Week6

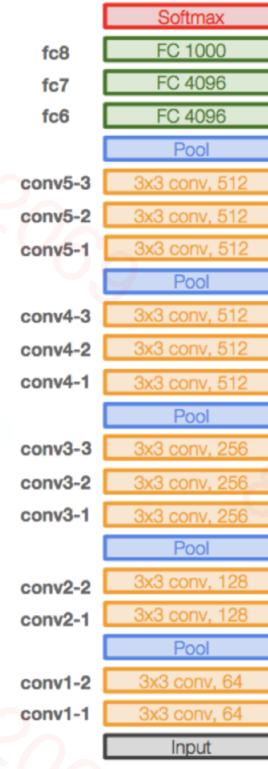
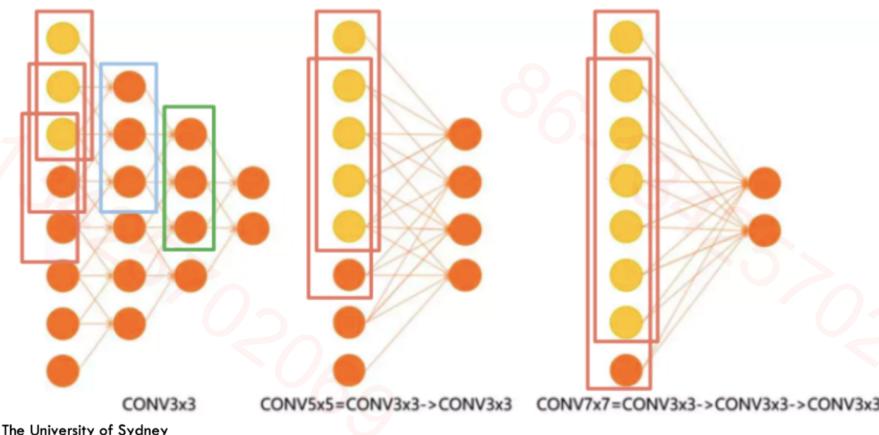
### Smaller Kernel

VGGNet的一个显著特点是广泛使用了3x3的卷积核。相比于5x5或7x7的卷积核，3x3的卷积核包含更少的参数。例如，一个5x5的卷积核有25个参数，而两个连续的3x3卷积核总共只有18个参数（每个9个），显著减少了参数数量。每一层卷积后都可以跟一个非线性激活函数（如ReLU），这样可以引入更多的非线性变换，从而使网络能够学习到更复杂的特征。

## Why small filters?

- Two 3x3 layer = 5x5 layer
- Three 3x3 layer = 7x7 layer
- Deeper, more non-linearity
- Less parameters

**Receptive Field (RF):** the region in the input space that a particular CNN's feature is looking at (i.e. be affected by).



VGG16

Page 17

Image credit to mc.ai

USYD COMP5329 Lecture slides - Week6

VGGNet通过堆叠更多的卷积层来加深网络结构，最深的VGGNet（VGG-19）包含了19个权重层（16个卷积层和3个全连接层）。增加网络的深度，使得VGGNet能够学习到更丰富和抽象的特征表示，从而提高了模型的性能。从VGGNet开始，卷积神经网络普遍采用了统一的3x3卷积核尺寸。这种统一的设计简化了网络结构的设计和实现。VGGNet在每一组卷积层后面都使用了2x2的最大池化层。这些池化层通过下采样减少了特征图的尺寸，同时保留了最重要的特征，进一步减少了参数和计算量。

## GoogLeNet — 2014

**GoogLeNet**, 也称为 **Inception v1**, 是由Google的研究人员在2014年提出的，并在当年的ImageNet竞赛中获得了第一名。GoogLeNet引入了一种名为“Inception”的新型架构，使其在不增加计算复杂性的情况下具有更大的深度和宽度。

# CNN Architecture: Part I

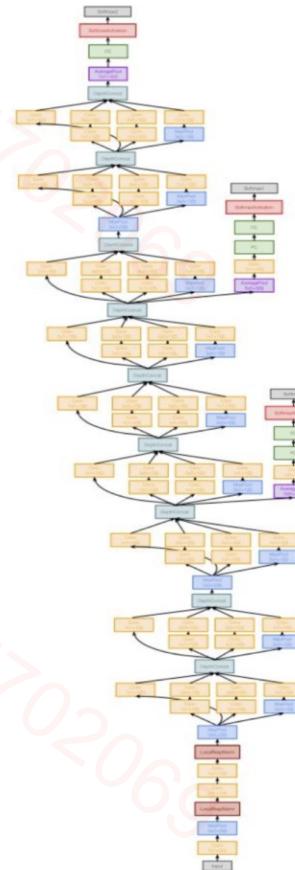
## □ GoogLeNet [Szegedy et al., 2014]

### Deeper networks

- 22 layers.
- Auxiliary loss

### Computational efficiency

- Inception module
- Remove FC layer, use global average pooling
- 12x less parameters than AlexNet

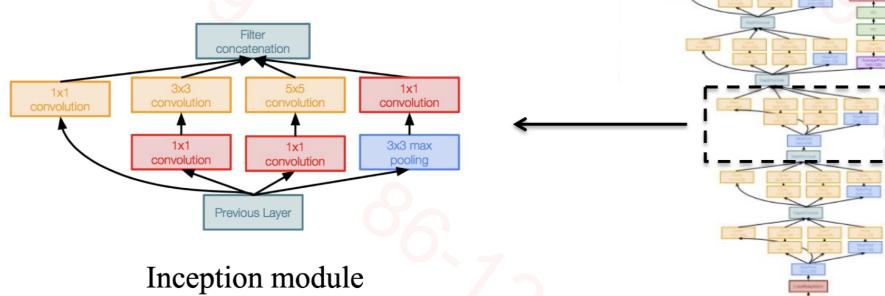


USYD COMP5329 Lecture slides - Week6

### Inception Module

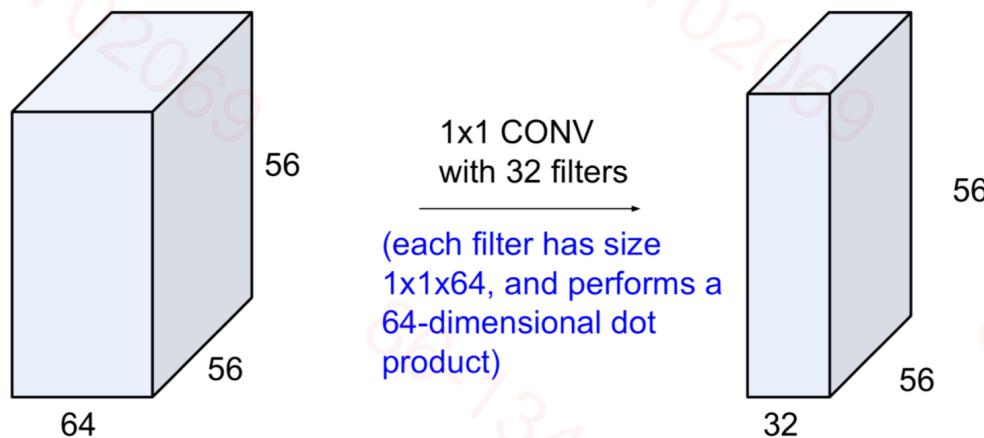
GoogLeNet包含多个Inception模块，具有较高的深度（总共22层）和较大的宽度。每个Inception模块内并行处理不同尺度的特征，并将结果拼接在一起，从而在保证计算效率的同时提高了模型的表达能力。GoogLeNet展示了深度和宽度都可以通过合适的架构来扩展，而不仅仅是通过堆叠更多的层。这一模型开启了系列后续研究和改进，包括多个Inception v2、v3、v4和Inception-ResNet等变体。GoogLeNet在计算效率和模型性能之间取得了良好的平衡，成为了深度学习领域的重要基石。

Design a good local network topology (NIN) and then stack these modules on top of each other.



USYD COMP5329 Lecture slides - Week6

Inception Module 使用  $1 \times 1$  的卷积核进行卷积操作，这不仅可以帮助减少维度，还能在不改变空间尺寸的情况下增加网络的深度和复杂性。然后接一个  $3 \times 3$  的卷积。这样做的目的是在不大幅增加计算量的前提下，增加更大的感受野。最后使用  $1 \times 1$  卷积进行特征变换。池化操作有助于提高模型的空间不变性，而后续的  $1 \times 1$  卷积用于恢复通道数。



### 1x1 convolutional layer

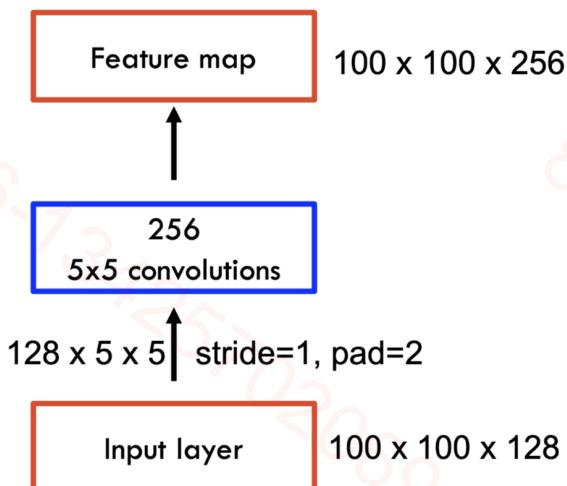
- preserve spatial dimensions (56x56)
- reduces depth (64 -> 32)

USYD COMP5329 Lecture slides - Week6

# CNN Architecture: Part I

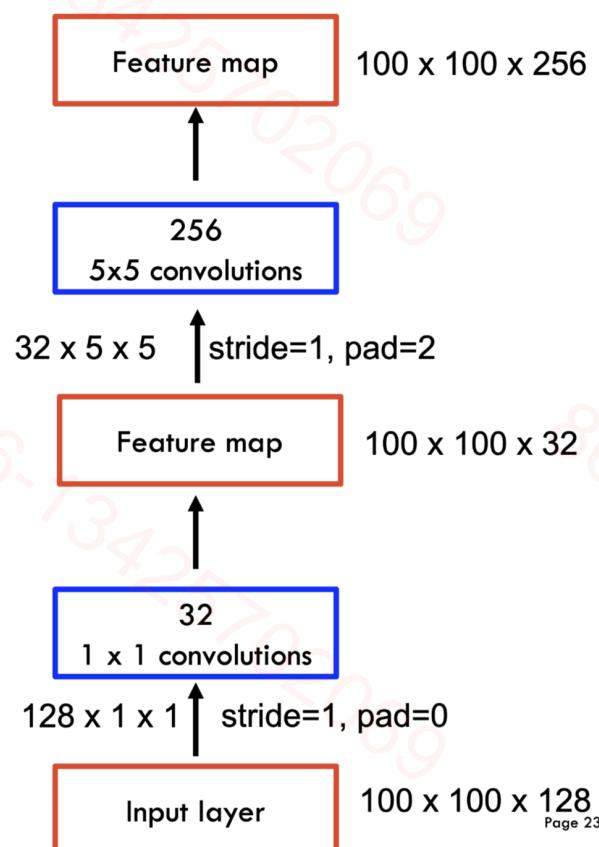
## □ GoogLeNet [Szegedy et al., 2014]

#parameter: 128x5x5x256



The University of Sydney

#parameter: 128x1x1x32 + 32x5x5x256



USYD COMP5329 Lecture slides - Week6

## Global Pooling

全局池化是一种通过对每个特征图的所有值取平均值或最大值来生成输出的技术。全局平均池化层通过对每个特征图的所有值取平均值来生成输出，代替了传统的全连接层，从而减少了模型的参数数量：

- **减少参数数量**：全局池化层不需要学习额外的参数，因此可以显著减少模型的参数数量。这不仅降低了计算复杂度，还减少了过拟合的风险。
- **缓解过拟合**：由于参数数量减少，全局池化层降低了模型的复杂度，从而缓解了过拟合现象，提高了模型在测试数据上的泛化能力。

设输入特征图为  $X$ ，其维度为  $H \times W$ ，则全局平均池化的输出为：

$$y = \frac{1}{H \times W} \sum_{i=1}^H \sum_{j=1}^W X(i, j)$$

## Auxiliary Loss

辅助损失是在网络的中间层添加辅助分类器，用于生成额外的损失信号。这些辅助分类器在训练过程中提供额外的梯度信号，帮助更深层的网络部分进行有效的学习。辅助损失的引入有以下优点：

- **解决梯度消失问题**：在深层神经网络中，梯度在反向传播过程中可能会逐渐减小至接近零，导致网络难以训练。辅助分类器通过提供额外的梯度信号，帮助更深层的网络部分进行有效的学习，缓解了梯度消失问题。

- **稳定训练过程**：辅助分类器在中间层生成的损失信号可以帮助稳定训练过程，提高模型的收敛速度和性能。

辅助损失的计算公式为：

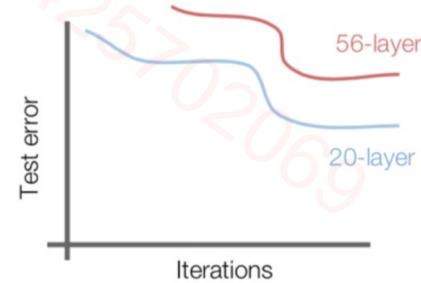
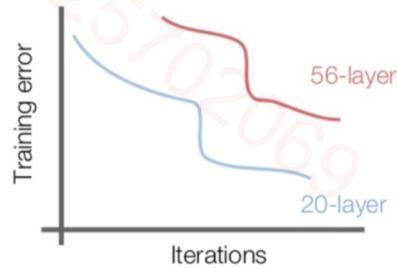
$$L_{\text{total}} = L_{\text{main}} + \alpha L_{\text{aux}}$$

其中： $L_{\text{main}}$  是主分类器的损失， $L_{\text{aux}}$  是辅助分类器的损失。 $\alpha$  是权重参数，用于平衡主损失和辅助损失的贡献。

## Res Net — 2015

**ResNet**，全称Residual Network，由何凯明等人在2015年提出，并获得了CVPR最佳论文奖。ResNet首次引入了残差连接（Residual Connection）的概念，有效解决了深度神经网络在训练时遇到的梯度消失问题，使得训练非常深的网络成为可能。

Stacking deeper layers on a “plain” convolutional neural network  
=> the deeper model performs worse, but not overfitting.



Hypothesis: deeper models are harder to optimize.

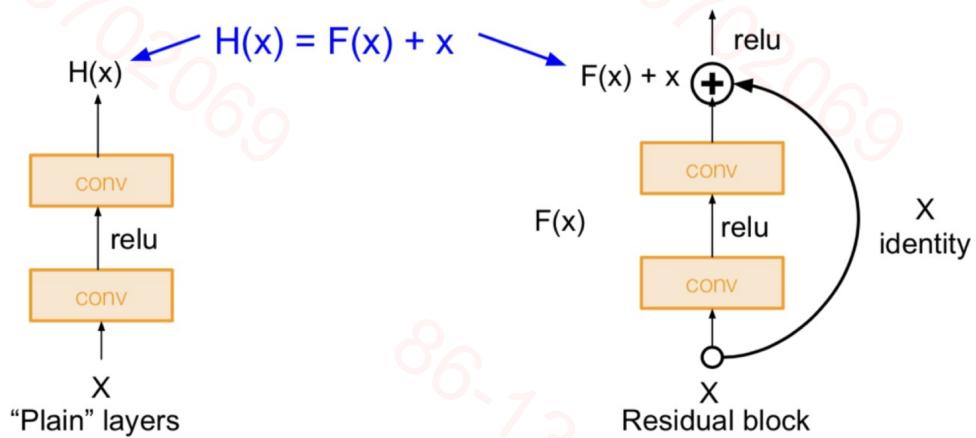
USYD COMP5329 Lecture slides - Week6

### Residual Connection

残差连接的核心思想是允许网络的一层直接将其输入不经修改地传递给更深的层。在传统的深度神经网络中，每一层的输出都是作为下一层的输入；而在残差网络中，通过添加一个“短路”连接，即直接将输入加到层的输出上。

在一个典型的残差模块中，输入首先通过两个（或更多）卷积层进行处理，得到特征转换后的输出。然后，这个输出会与原始输入直接相加（前提是它们的维度相同；如果不同，通常使用一个 $1 \times 1$ 的卷积来调整维度）。这样，模块的输出不仅包含了从输入到输出的变化量（即“残差”），也直接包括了原始输入本身。

**Solution:** use network layers to fit a residual mapping instead of directly trying to fit a desired underlying mapping



USYD COMP5329 Lecture slides - Week6

设输入为  $x$ ，经过两个卷积层后的输出为  $F(x)$ 。在残差块中，输出为：

$$y = F(x) + x$$

如果输入和输出的维度不同，则需要使用1x1卷积来调整维度：

$$y = F(x) + W_s x$$

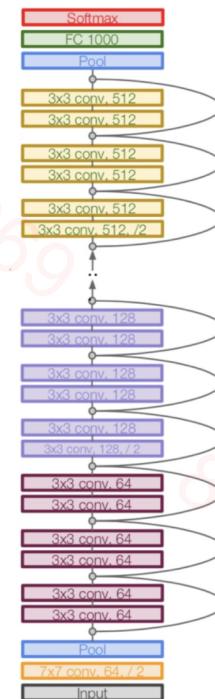
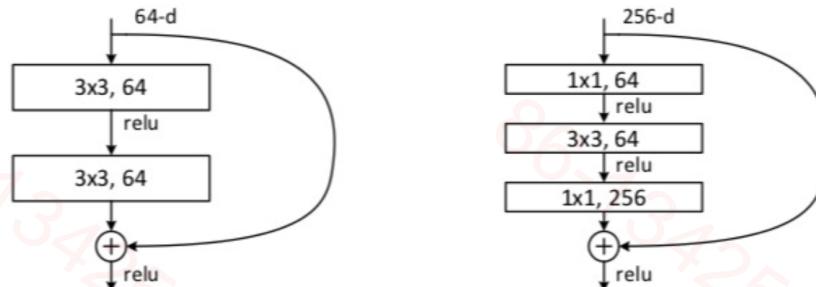
其中， $W_s$  为1x1卷积的权重矩阵。

### Deeper Structure

在传统的深度神经网络中，当网络很深时，梯度在反向传播过程中可能会逐渐减小至接近零，导致网络难以训练。通过残差连接，梯度可以直接流过跳跃连接，保持较强的梯度，从而有效地训练深层网络。这使得ResNet能够训练非常深的网络，如ResNet-50、ResNet-101甚至ResNet-152。

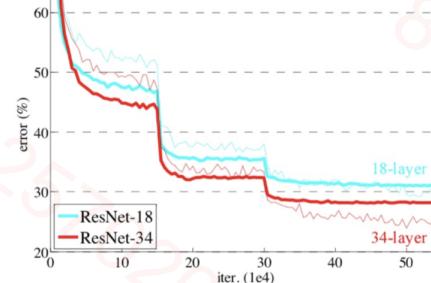
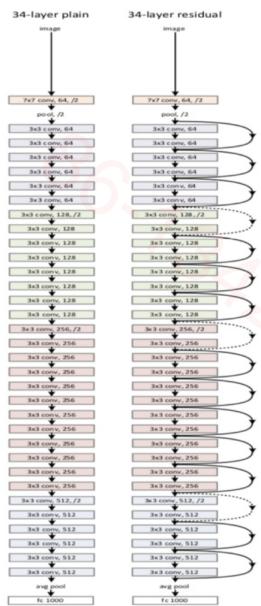
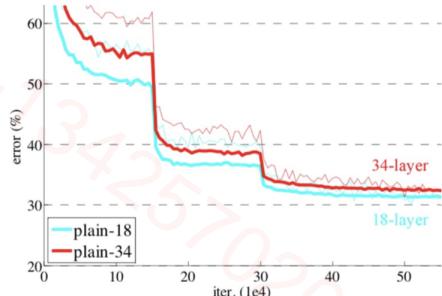
## □ ResNet [He et al., 2015]

Use “**bottleneck**” layer to improve efficiency (similar to GoogLeNet ).



ResNet

USYD COMP5329 Lecture slides - Week6



USYD COMP5329 Lecture slides - Week6

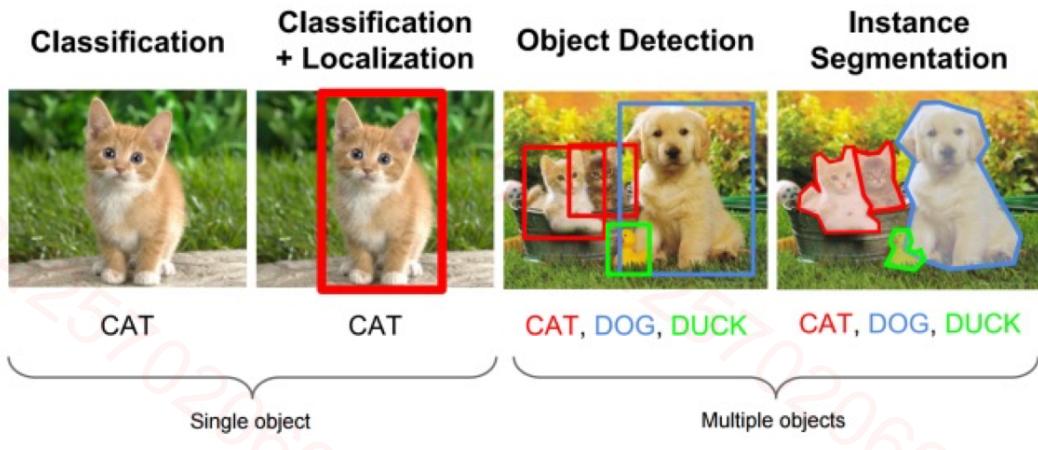
ResNet的成功展示了深层神经网络的强大能力，并为后续的研究和应用提供了重要的启示。自ResNet提出以来，残差连接成为深度学习中的标准组件，被广泛应用于各类模型中，如自然语言处理中的Transformer和医学图像分割中的U-Net等。通过引入残差连接，ResNet有效地解决了深度神经网络的梯度消失问题，使得训练非常深的网络成为可能。

能，并显著提升了模型的性能。这一创新为深度学习的发展开辟了新的道路，对计算机视觉和其他领域的研究产生了深远的影响。

## Computer Vision Tasks

### Vision Tasks

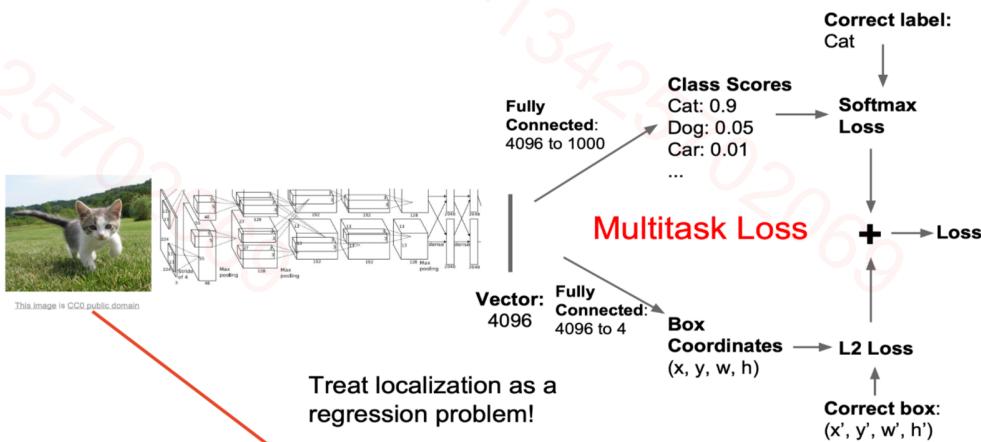
计算机视觉领域包含许多不同的任务，每个任务都旨在从图像或视频数据中提取特定的信息。



USYD COMP5329 Lecture slides - Week11

### Classification + Localisation

不仅对输入图像进行分类，还要在图像中确定目标物体的位置。输出包括物体的类别和边界框（bounding box），表示物体在图像中的位置。在某些情况下，**分类和定位任务可以相互促进**。例如，边界框的精确预测可以帮助改进分类的准确性，因为更准确的边界框能够提供更集中的图像区域，提高分类的可靠性。反之亦然，准确的分类信息可以引导模型更好地学习边界框的位置。通过联合优化分类和定位任务，模型能够更全面地理解图像中的信息，从而在实际应用中取得更好的效果。



It is assumed that there is only one object in an input image.

USYD COMP5329 Lecture slides - Week11

在训练过程中，设计一个组合损失函数，同时考虑分类和定位的性能。分类通常使用交叉熵损失（Cross-Entropy Loss），而定位则可能使用均方误差（Mean Squared Error, MSE）或交并比（Intersection over Union, IoU）等指标来计算边界框的准确性：

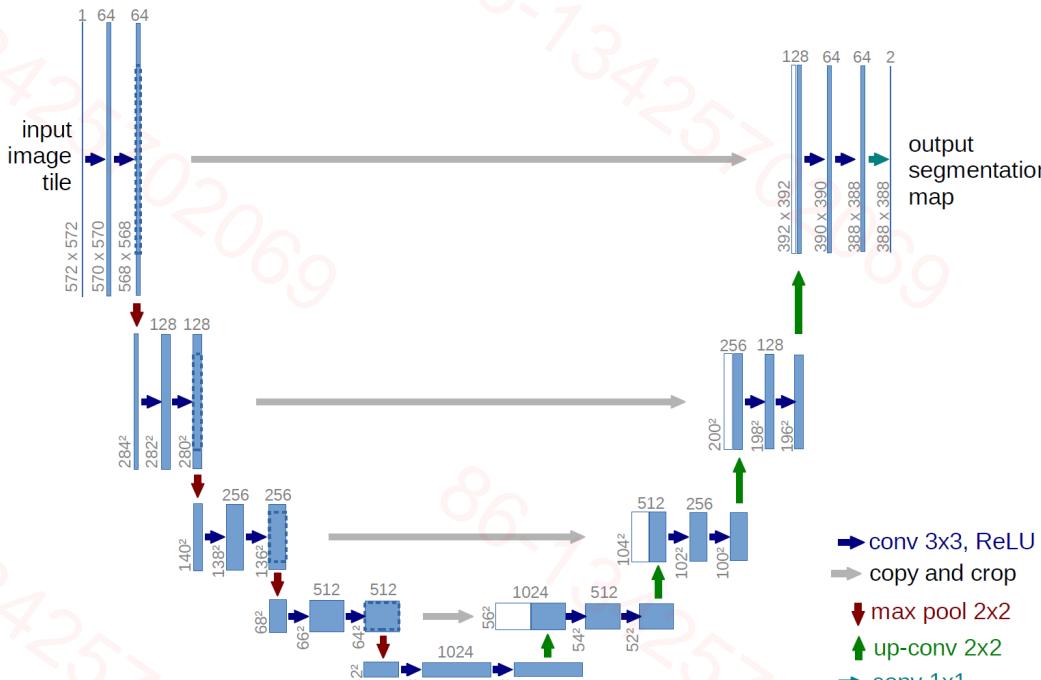
- **分类损失（Classification Loss）**：使用交叉熵损失函数计算预测类别与真实类别之间的差异。
- **定位损失（Localization Loss）**：使用均方误差或IoU计算预测边界框与真实边界框之间的差异。
- **组合损失（Combined Loss）**：将分类损失和定位损失加权求和，得到总损失，用于指导模型的训练。

$$\text{Total Loss} = \alpha \times \text{Classification Loss} + \beta \times \text{Localization Loss}$$

其中， $\alpha$  和  $\beta$  是用于平衡分类和定位损失的权重参数。

## Object Detection & Segmentation

目标检测的主要目标是在图像中检测并定位多个物体。输出包括每个检测到的物体的类别和边界框（bounding box），表示物体在图像中的位置。图像分割的目标是将图像中的每个像素分类到特定的类别。虽然这是两种不同的任务，但是它们之间存在一些共性，都可以看作是视觉下游任务。



<https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/>

上游任务通常是指在大规模数据集上进行的预训练任务，其目的是学习通用的特征表示。这些任务通常包括图像分类等，用于训练一个强大的特征提取器（encoder）。编码器负责提取图像的高层特征。通常由多个卷积层和池化层组成，逐步减少空间分辨率，同时增加特征的抽象程度。下游任务指在特定应用场景中使用预训练的特征提取器，并进一步训练以完成具体任务的过程。下游任务可以是目标检测、图像分割等。通过使用上游任务训练好的encoder，下游任务只需从头学习一个decoder，并在特定数据集上进行微调，从而有效利用预训练模型的强大特征表示能力。解码器负责将编码器提取的高层特征恢复到原始图像的空间分辨率。通常由多个反卷积层（转置卷积层）或上采样层组成，以逐步增加空间分辨率，并输出分割掩码。解码器的结构可以从头学习，并在下游任务数据集上进行微调（fine-tuning）。如果是定位任务，Decoder被称作定位头（Localization Head）。作用是将特征映射到边界框（bounding box），用于确定图像中目标物体的位置和大小。分割任务被称作分割头（Segmentation Head），作用是将特征映射到分割掩码（segmentation mask），该掩码对图像的每个像素进行分类。

## Multi-task Dataset - COCO

COCO (Common Objects in Context) 是一个广泛使用的多任务数据集，适用于计算机视觉任务。COCO数据集由 Microsoft 发布，包含各种图像和标注，涵盖以下几种主要任务：

- **图像分类 (Image Classification)**：将每张图像分配给一个或多个预定义的类别。
- **物体检测 (Object Detection)**：识别图像中的物体，并为每个物体生成一个边界框（bounding box）。`"bbox"` 字段表示边界框的坐标。
- **实例分割 (Instance Segmentation)**：为每个物体生成精确的像素级别的分割掩码。标注格式：`"segmentation"` 字段表示分割掩码的多边形坐标。
- **关键点检测 (Keypoint Detection)**：识别人体姿态的关键点（如关节、鼻子、眼睛等）。
- **语义分割 (Semantic Segmentation)**：将图像中的每个像素分配给一个类别。

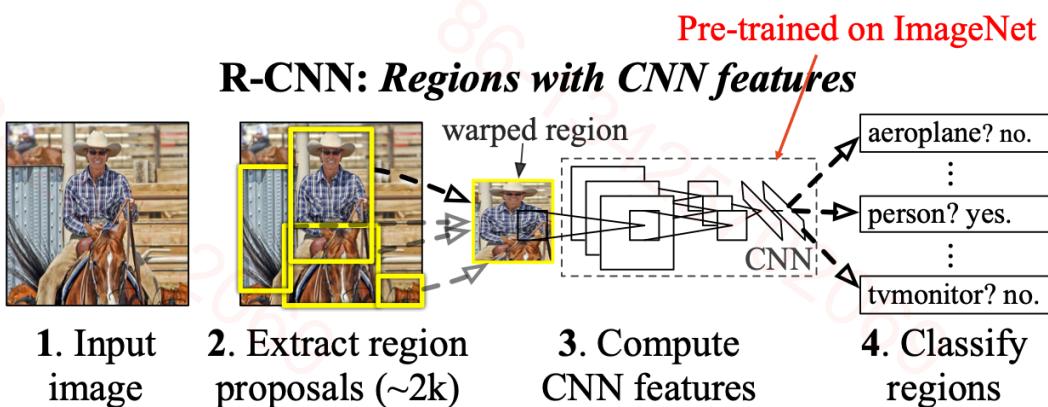
# COCO DATASET FORMAT

USYD COMP5329 Lecture slides - Week11

COCO数据集支持多种计算机视觉任务，包括图像分类、物体检测、实例分割和关键点检测等。并且提供了详细且多样的标注，包括边界框、分割掩码和关键点等，包含数十万张标注图像和数百万个标注对象，提供了充足的数据量，适合深度学习模型的训练和评估。

## Region CNN (R-CNN)

R-CNN (Region-based Convolutional Neural Networks) 是由Ross Girshick等人于2014年提出的一种用于目标检测的方法。它的主要创新点在于将区域提议（region proposals）和卷积神经网络（CNN）结合起来，通过生成候选区域并对每个区域进行特征提取和分类，实现目标检测。



USYD COMP5329 Lecture slides - Week11

## Selective Search

R-CNN首先使用选择性搜索算法在输入图像中生成约2000个区域建议。这些区域是图像中可能包含目标的候选区域。选择性搜索通过智能分割图像来生成区域提案。图像最初被分割成数百个大小的、一致的区域（称为超像素），然后根据某些相似性标准合并这些区域。这个过程是层次性的，从这些超像素开始，逐步合并成越来越大的区域。

- Motivation
  - Sliding window approach is not feasible for object detection with convolutional neural networks.
  - We need a more faster method to identify object candidates.
- Finding object proposals
  - Greedy hierarchical superpixel segmentation
  - Diversification of superpixel construction and merge
    - Using a variety of color spaces
    - Using different similarity measures
    - Varying staring regions

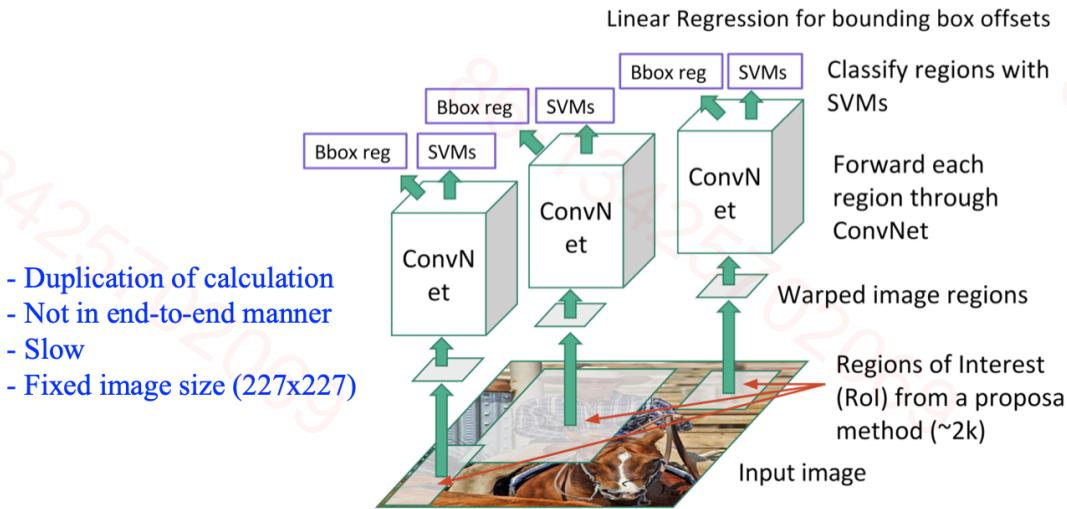


[Uijlings13] J. R. R. Uijlings, K. E. A. van de Sande, T. Gevers, A. W. M. Smeulders: **Selective Search for Object Recognition**. IJCV 2013

USYD COMP5329 Lecture slides - Week11

## Feature Extraction

对每个区域建议，R-CNN使用预训练的卷积神经网络（如AlexNet）来提取特征。原始的R-CNN将每个提议的区域缩放（或扭曲）到固定大小，以符合CNN输入的要求。提取的特征被送入一组分类器（通常是支持向量机，SVM）中，每个分类器负责判断一个类别是否存在于该区域中。

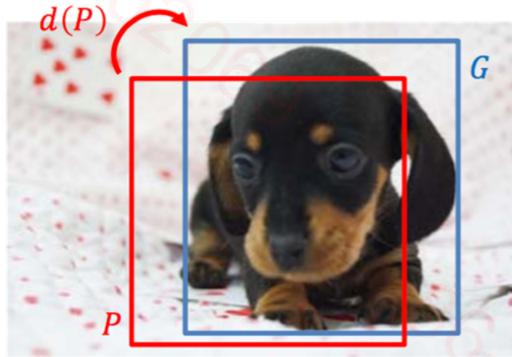


USYD COMP5329 Lecture slides - Week11

## Bounding Box Regression

边界框回归是一种在目标检测任务中常用的方法，主要用于精确定位图像中对象的位置。其目的是通过对初始的区域建议（region proposals）进行修正和优化，使得边界框能够更准确地包围实际的目标对象。在目标检测任务中，初始的区域建议可能来自选择性搜索、滑动窗口或其他方法，这些区域建议可能并不完全准确。边界框回归的作用就是对这些初始的区域建议进行调整，以获得更精确的边界框。

- Region proposal:  $P = (P_x, P_y, P_w, P_h)$
- Ground-truth:  $G = (G_x, G_y, G_w, G_h)$
- Transformation:  $d(P) = (t_x, t_y, t_w, t_h)$



$$\begin{aligned}\hat{G}_x &= P_w d_x(P) + P_x \\ \hat{G}_y &= P_h d_y(P) + P_y \\ \hat{G}_w &= P_w \exp(d_w(P)) \\ \hat{G}_h &= P_h \exp(d_h(P))\end{aligned}$$

$$d_i(P) = \mathbf{w}_i^T \phi_5(P)$$

CNN pool5 feature

$$\mathbf{w}_i^* = \underset{\mathbf{w}_i}{\operatorname{argmin}} \sum_{k=1}^N \left( t_i^k - \mathbf{w}_i^T \phi_5(P^k) \right)^2 + \lambda \|\mathbf{w}_i\|^2$$

USYD COMP5329 Lecture slides - Week11

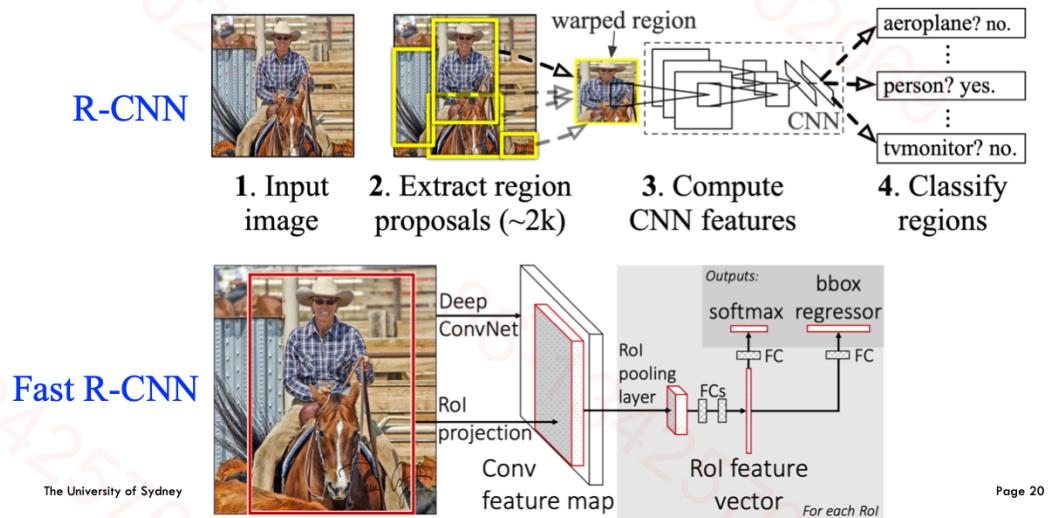
- R-CNN通过使用预训练的CNN（如AlexNet）进行特征提取，显著提高了目标检测的准确性。R-CNN的结构分为多个独立的步骤（区域建议、特征提取、分类、边界框回归），使得每个模块都可以独立优化和替换。但是R-CNN需要对每个区域建议进行单独的特征提取和分类，这导致了高昂的计算成本和低效的推理速度。对每个区域建议提取的特征需要存储，这增加了内存使用量和I/O开销。R-CNN的训练过程分为多个阶段（预训练CNN、训练SVM、训练边界框回归），这增加了训练复杂性和时间。

### Fast R-CNN

Fast R-CNN 是 Ross Girshick 在 2015 年提出的目标检测模型，旨在解决 R-CNN 在速度和训练效率上的问题，同时保持高精度。与传统的 R-CNN 相比，Fast R-CNN 有几个显著的创新和改进。

Drawback of R-CNN and the modification:

1. Multi-stage training. → End-to-end joint training.
2. Expensive in space and time. → Convolutional layer sharing.
3. Test-time detection is slow. → Single scale testing

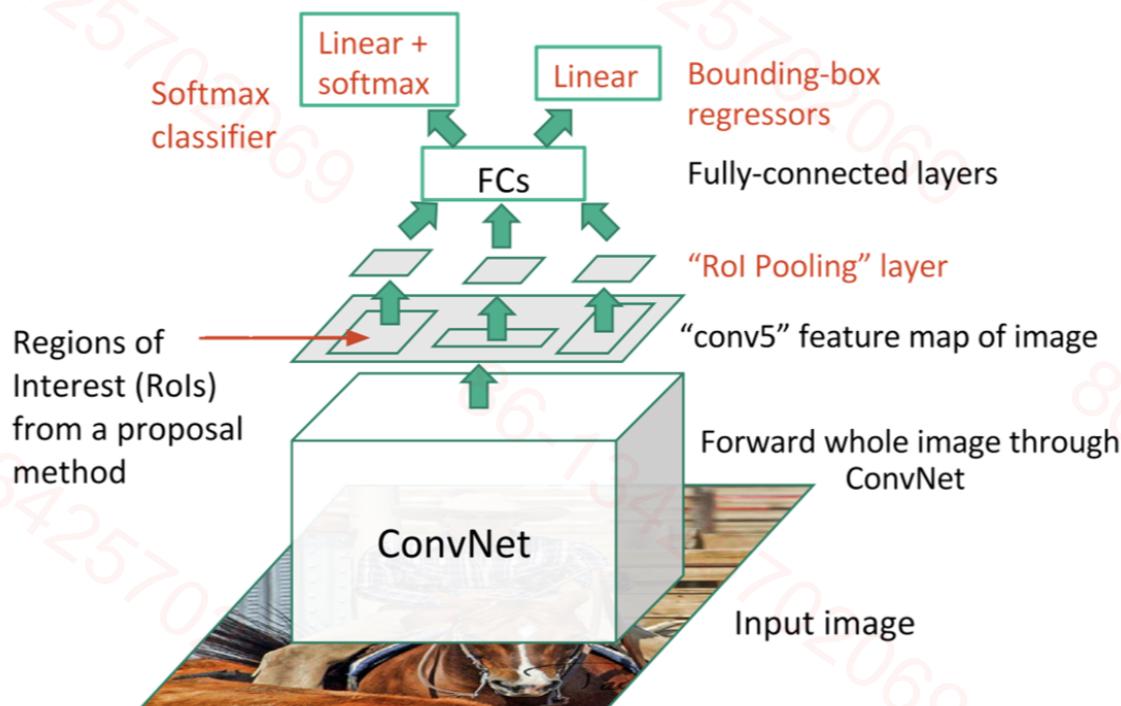


USYD COMP5329 Lecture slides - Week11

### Shared Feature Map

Fast R-CNN 将整个输入图像一次性输入到卷积神经网络中，得到一个共享的特征图。这与 R-CNN 的方法不同，后者对每个区域提案单独提取特征。通过共享特征图，大幅减少了重复计算，提高了计算效率。

## Fast R-CNN



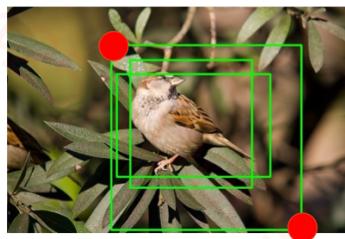
USYD COMP5329 Lecture slides - Week11

并且Fast R-CNN 使用一个多任务损失函数，同时训练分类（类别预测）和边界框回归（位置微调）两项任务。通过一次训练同时优化分类和定位性能，提高了模型的训练效率和检测精度。整个网络可以通过反向传播算法和随机梯度下降（SGD）一次性更新权重，实现了**End-to-End Training**。简化了训练过程，避免了像 R-CNN 那样的多步骤训练流程。

### Region of Interest Pooling

ROI Pooling 层的目的是处理来自预先定义区域（ROI）的特征，这些区域通常由前一阶段生成。因为这些**区域的大小和比例各不相同**，而传统的全连接层需要**固定大小的输入**，从共享的特征图上裁剪出每个区域提案对应的特征，通过**ROI Pooling 层将不同大小的区域调整为相同大小的特征映射**。解决了输入到全连接层的特征尺寸不一致的问题，使得网络能够处理不同大小和比例的区域提案。

A type of max-pooling. The output always has the same size.



Input to ROI pooling layer:

1. A fixed-size feature map
2. A list of regions of interest

0.85	0.84
0.97	0.96

a region proposal

$8 \times 8$  feature map  $\longrightarrow$   $2 \times 2$  output

input															
0.88	0.44	0.14	0.16	0.37	0.77	0.96	0.27								
0.19	0.45	0.57	0.16	0.63	0.29	0.71	0.70								
0.66	0.26	0.82	0.64	0.54	0.73	0.59	0.26								
0.85	0.34	0.76	0.84	0.29	0.75	0.62	0.25								
0.32	0.74	0.21	0.39	0.34	0.03	0.33	0.48								
0.20	0.14	0.16	0.13	0.73	0.65	0.96	0.32								
0.19	0.69	0.09	0.86	0.88	0.07	0.01	0.48								
0.83	0.24	0.97	0.04	0.24	0.35	0.50	0.91								

The University of Sydney

region proposal															
0.88	0.44	0.14	0.16	0.37	0.77	0.96	0.27								
0.19	0.45	0.57	0.16	0.63	0.29	0.71	0.70								
0.66	0.26	0.82	0.64	0.54	0.73	0.59	0.26								
0.85	0.34	0.76	0.84	0.29	0.75	0.62	0.25								
0.32	0.74	0.21	0.39	0.34	0.03	0.33	0.48								
0.20	0.14	0.16	0.13	0.73	0.65	0.96	0.32								
0.19	0.69	0.09	0.86	0.88	0.07	0.01	0.48								
0.83	0.24	0.97	0.04	0.24	0.35	0.50	0.91								

pooling sections															
0.88	0.44	0.14	0.16	0.37	0.77	0.96	0.27								
0.19	0.45	0.57	0.16	0.63	0.29	0.71	0.70								
0.66	0.26	0.82	0.64	0.54	0.73	0.59	0.26								
0.85	0.34	0.76	0.84	0.29	0.75	0.62	0.25								
0.32	0.74	0.21	0.39	0.34	0.03	0.33	0.48								
0.20	0.14	0.16	0.13	0.73	0.65	0.96	0.32								
0.19	0.69	0.09	0.86	0.88	0.07	0.01	0.48								
0.83	0.24	0.97	0.04	0.24	0.35	0.50	0.91								

Page 22

USYD COMP5329 Lecture slides - Week11

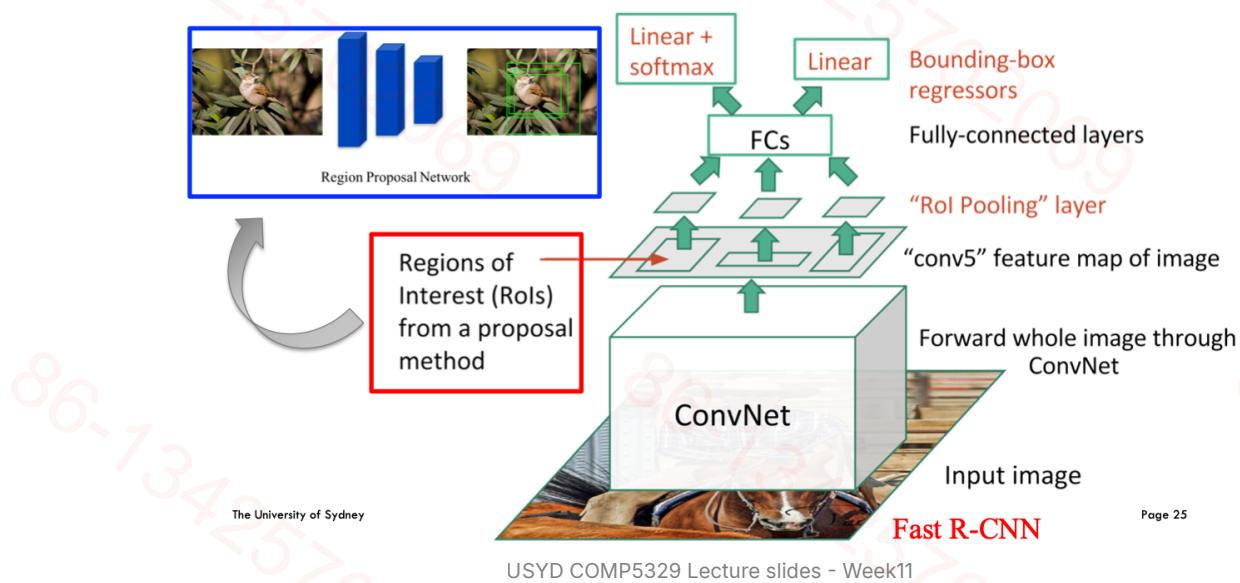
ROI pooling接受一个固定大小的特征图和一个ROI列表，每个ROI定义了特征图中的一个子区域。然后将每个ROI划分为固定数量的子网格（例如 $7 \times 7$ ）。对每个子网格进行最大池化操作，将不同大小的ROI转换为固定大小的输出特征。

由于共享特征图和ROI Pooling的使用，**Fast R-CNN在检测速度上显著快于R-CNN**。通过多任务损失函数，能够在单一阶段内同时训练分类和边界框回归，简化了训练流程。尽管加快了速度，**Fast R-CNN依然保持了高精度的检测性能**。但是**ROI Pooling层在处理尺寸变化时可能会导致特征的量化误差**，影响检测精度。Fast R-CNN仍然需要依赖选择性搜索等方法生成区域提案，这部分计算相对耗时，未能实现完全端到端的检测。

## Faster RCNN

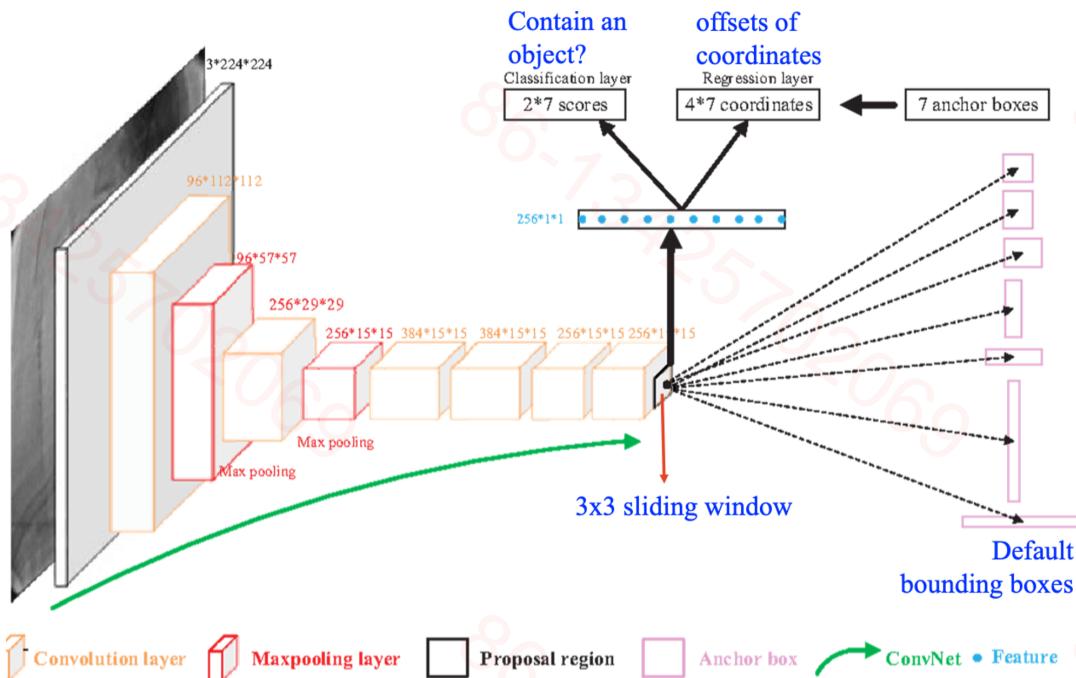
Faster R-CNN是Shaoqing Ren等人在2015年提出的一种目标检测模型，它在Fast R-CNN的基础上进一步提升了检测速度和精度。Faster R-CNN的核心创新在于引入了区域提案网络（Region Proposal Network, RPN），实现了端到端的目标检测。

Replace the slow selective search algorithm with a fast neural net  
- region proposal network (RPN).



## Region Proposal Network

RPN 直接在共享的特征图上滑动，使用一组固定大小的锚点框（anchor boxes）来预测边界框和对象分数。这些锚点框覆盖了不同的尺度和长宽比，确保能够捕捉到各种可能的对象。RPN 可以实时生成高质量的区域提案，显著提升了提案生成的效率和准确性，避免了选择性搜索等传统方法的计算瓶颈。



区域提案网络（RPN）的核心思想是通过滑动窗口机制生成锚点框，这些锚点框用于预测可能包含对象的区域。在特征图的每个位置生成多个锚点框，这些锚点框具有不同的尺度和长宽比。RPN 使用小的网络来预测每个锚点框的对象性分数（即框中是否包含对象）和边界框调整参数（即对锚点框进行微调以更好地拟合对象）。对生成的区域提案进行非极大值抑制，去除冗余的提案，保留最有可能包含对象的区域。

由于 RPN 的引入，Faster R-CNN 能够实时生成高质量的区域提案，显著提升了目标检测的速度。同时生成更加精确的区域提案，提高了目标检测的整体准确性。通过端到端的训练方式，Faster R-CNN 能够更好地协调各个部分的性能，提升模型的一致性和效果。但是尽管 RPN 提升了性能，但它也增加了模型的复杂性，需要更多的计算资源和更长的训练时间。RPN 的性能在很大程度上依赖于锚点框的设计，锚点框的选择和比例可能需要根据不同的数据集进行调整和优化。

## Masked R-CNN

Mask R-CNN 是在 Faster R-CNN 基础上发展而来的目标检测和实例分割模型，由 Kaiming He 等人在 2017 年提出。它不仅能够识别图像中的对象及其精确位置（边界框），还可以为每个实例生成高质量的分割掩模（mask）。Mask R-CNN 在 Faster R-CNN 的基础上增加了一个并行的分支，用于生成对象的分割掩模。这个分支通过全卷积网络（Fully Convolutional Network, FCN）来预测每个对象的像素级掩模。这使得 Mask R-CNN 不仅能够检测对象，还能在像素级别进行实例分割，实现更高质量的对象分割。

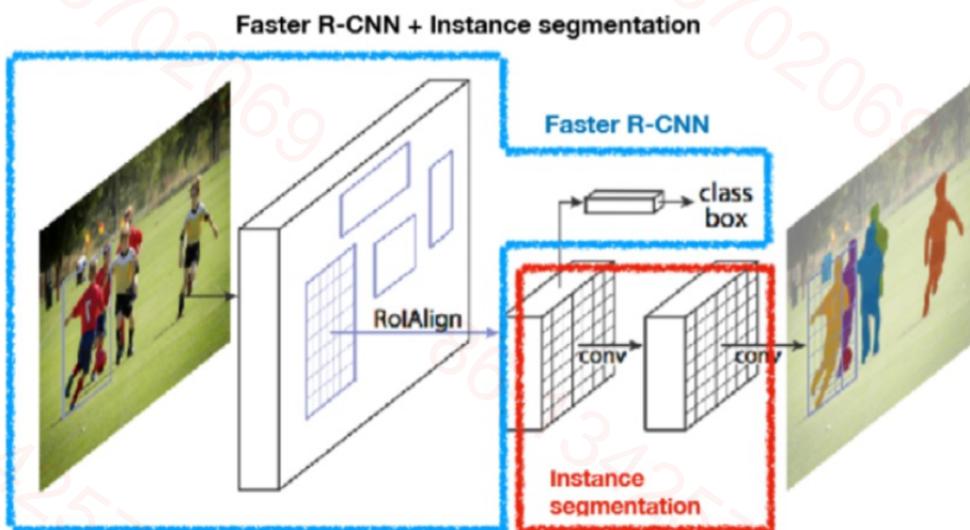
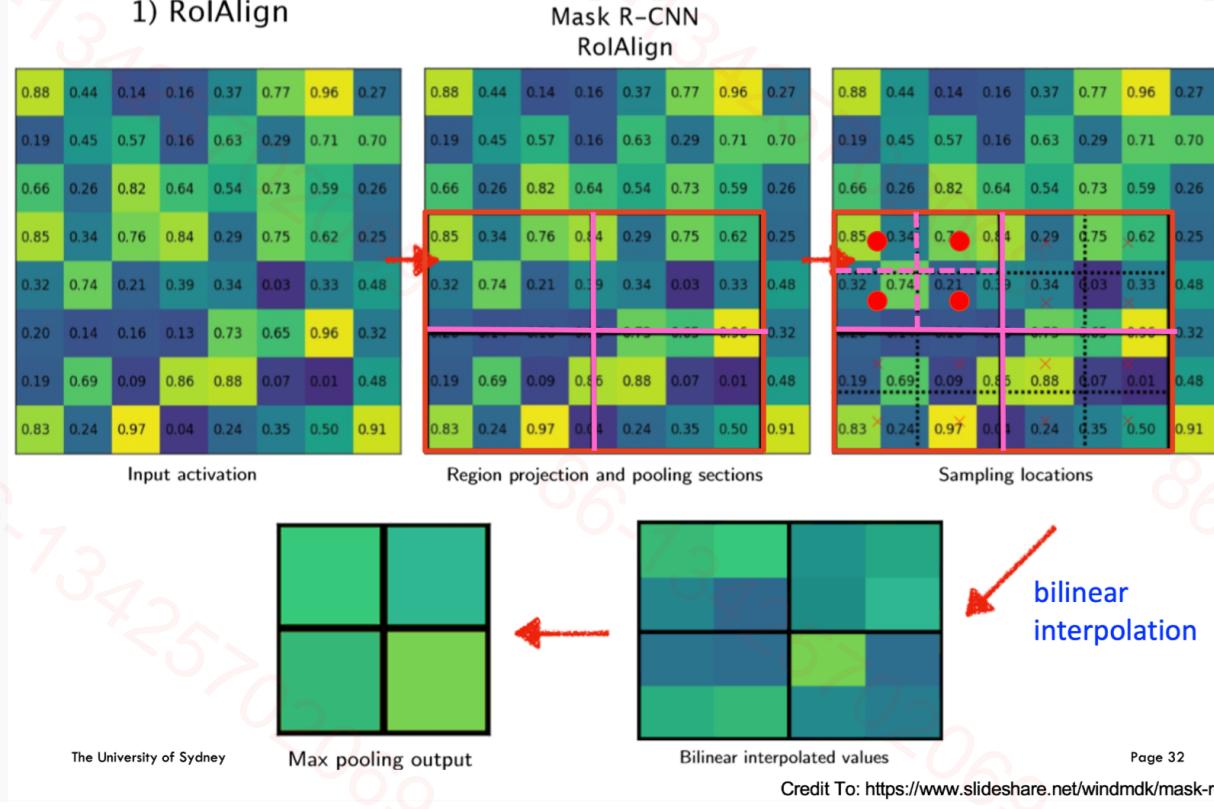


Figure 1. The Mask R-CNN framework for instance segmentation.

USYD COMP5329 Lecture slides - Week11

### ROIAlign

Mask R-CNN 引入了 ROIAlign 方法，解决了 ROI Pooling 在从特征图到原始图像进行量化时可能引入的不精确对齐问题。ROIAlign 通过双线性插值精确计算输入特征图上任意位置的值，确保了特征和输入图像之间的精确空间对应。ROIAlign 保证了特征和输入图像的精确对齐，避免了因量化而导致的误差，提高了分割掩模的精度和目标检测的准确性。

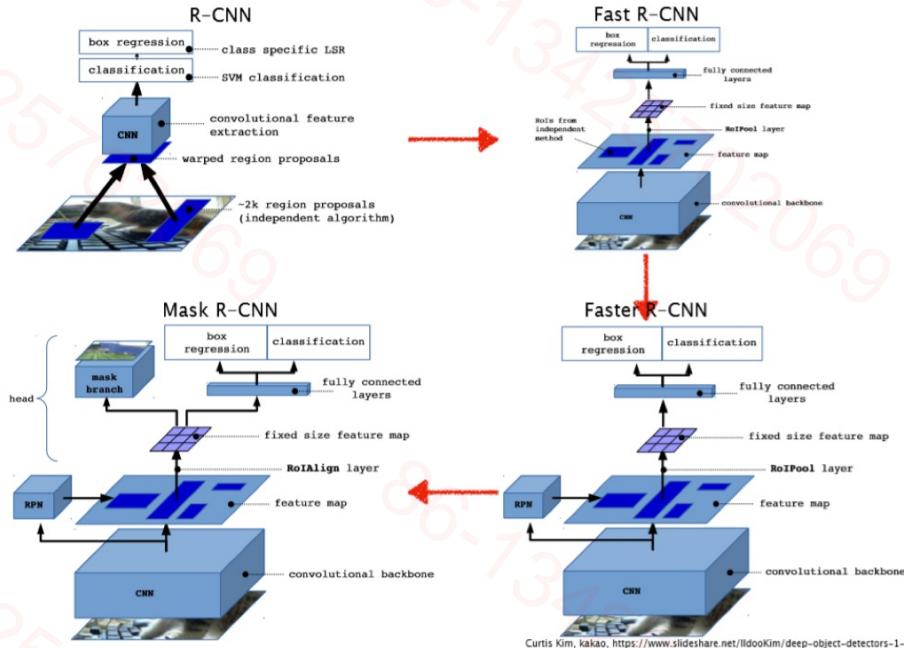


Mask R-CNN 能够在进行目标检测的同时，生成高质量的对象分割掩模，实现了像素级的精确分割。通过引入 ROIAlign，Mask R-CNN 解决了 ROI Pooling 的量化误差问题，提高了分割和检测的精度。Mask R-CNN 的结构可以方便地扩展到其他任务，如人体姿态估计等。但是相比于 Faster R-CNN，Mask R-CNN 增加了分割掩模的分支，计算复杂度和内存消耗较高，训练时间较长。

## From R-CNN to Mask R-CNN

从 R-CNN 到 Mask R-CNN 的演变体现了计算机视觉技术的快速进步。每一步的改进都基于解决前一版本中的缺陷和提高模型的效率：

- 效率提升：**从 R-CNN 的选择性搜索到 Faster R-CNN 的 RPN，再到 Mask R-CNN 的全局处理，目标检测模型的速度和效率显著提高。
- 精度提高：**从最初的区域建议到精确的边界框预测，再到像素级的分割，模型的精度和适用性不断增强。
- 复杂度管理：**每一代模型在增加功能和精度的同时，也面临着计算复杂度和训练难度的增加，需要更强的硬件支持和优化策略。



USYD COMP5329 Lecture slides - Week11

## R-CNN (Region-based Convolutional Neural Networks)

- 使用选择性搜索（Selective Search）生成约 2000 个区域建议（region proposals）。
- 对每个区域建议使用卷积神经网络（如 AlexNet）提取特征。
- 使用支持向量机（SVM）进行分类，并用线性回归模型进行边界框回归。

首次将 CNN 引入目标检测，显著提高了检测精度。但是计算量大，速度慢。每个区域建议都需要单独提取特征，训练和推理时间较长。

## Fast R-CNN

- 将整个图像一次性输入卷积神经网络，生成共享特征图（feature map）。
- 利用 ROI Pooling 层，从共享特征图上提取固定大小的区域特征。
- 使用多任务损失函数同时训练分类和边界框回归。

显著提高了训练和推理速度，避免了重复计算特征。但是仍依赖选择性搜索生成区域建议，区域建议生成过程仍然较慢。

## Faster R-CNN

- 引入区域提案网络（Region Proposal Network, RPN），实时生成高质量的区域建议。
- RPN 使用锚点框（anchor boxes）来预测边界框和对象分数，直接在特征图上滑动。

区域建议生成速度大幅提升，实现了端到端的目标检测。但是RPN 和目标检测网络共享特征图，但训练过程更加复杂。

## Mask R-CNN

- 在 Faster R-CNN 基础上增加了一个并行的分支，用于生成对象的分割掩模（mask）。

- 引入 ROIAlign 方法，解决 ROI Pooling 在量化过程中引入的不精确对齐问题。

实现了像素级的实例分割，ROIAlign 提高了分割和检测的精度。但是计算复杂度进一步增加，对硬件要求更高。

## Recurrent Neural Network

### Text Modality

自然语言处理（NLP）是计算机科学与人工智能的一个分支，旨在使计算机能够理解、解释和生成人类语言。NLP的应用范围广泛，包括机器翻译、情感分析、文本生成和信息检索等。在进行NLP任务时，首先需要将文本数据转换为计算机可以处理的数值数据，这通常涉及文本的表示与编码。

Image credit to <https://twitter.com/SamsungMobile/status/967807667463958531>

The University of Sydney

Page 2

USYD COMP5329 Lecture slides - Week8

文本数据本质上是离散的字符和单词序列，直接进行数学运算是不现实的。因此，我们需要将文本转换为数值表示，常见的方法包括独热编码（One-Hot Encoding）、词袋模型（Bag of Words）、词嵌入（Word Embeddings）等。

### One-Hot Encoding

独热编码是最简单的文本表示方法之一。它将每个单词或字符表示为一个仅在对应位置为1、其余位置为0的二进制向量。独热编码直接将文本转换为可运算的向量，每个类别被平等且独立地视为一个特征。通过等距表示所有类别，消除了原始编码中可能存在的数值大小关系。例如，将类别编码为1, 2, 3可能暗示 $1 < 2 < 3$ 的顺序关系，这在没有自然顺序的分类任务中是不合适的。

## □ One hot encoding

- The length of vector is lexicon size
  - Each dimension corresponds to a word in the lexicon
  - The dimension for the word is 1, and others are 0
- lexicon = {apple, bag, cat, dog, elephant}

Word	D_1	D_2	D_3	D_4	D_5
Apple	1	0	0	0	0
Bag	0	1	0	0	0
Cat	0	0	1	0	0
Dog	0	0	0	1	0
Elephant	0	0	0	0	1

但是如果分类变量的类别数非常多，**独热编码会导致数据集的特征维度大幅增加**。这不仅增加了模型的计算负担，也可能增加模型过拟合的风险。由于独热编码中大部分位置都是0，这会产生大量的稀疏矩阵。**稀疏矩阵不仅占用存储空间，而且可能增加某些算法处理这些数据的计算复杂度**。对于有序分类数据，**独热编码会丢失类别间的序信息**。例如，“低”、“中”、“高”这类有序分类数据，独热编码无法表示其顺序关系。

## Bag of Words

词袋模型将文本表示为一个词汇表中的词频向量。每个维度表示一个词在文本中出现的次数。虽然词袋模型简单有效，但它忽略了词的顺序和语法关系。构建方式是**收集所有文本中的所有不同单词**，构建一个词汇表。**词汇表的每个单词都有一个唯一的索引**。对于每个文本，生成一个与词汇表等长的向量。向量的每个位置对应一个单词，表示该单词在文本中出现的次数。假设我们有两个简单的句子构建词汇表：

```
["猫", "坐", "在", "椅子", "上", "狗", "趴", "地板"]
```

接着，生成词频向量：

```
"猫坐在椅子上" : [1, 1, 1, 1, 1, 0, 0, 0]  
"狗趴在地板上" : [0, 0, 1, 0, 1, 1, 1, 1]
```

I love this movie! It's sweet, but with satirical humor. The dialogue is great and the adventure scenes are fun... It manages to be whimsical and romantic while laughing at the conventions of the fairy tale genre. I would recommend it to just about anyone. I've seen it several times, and I'm always happy to see it again whenever I have a friend who hasn't seen it yet!



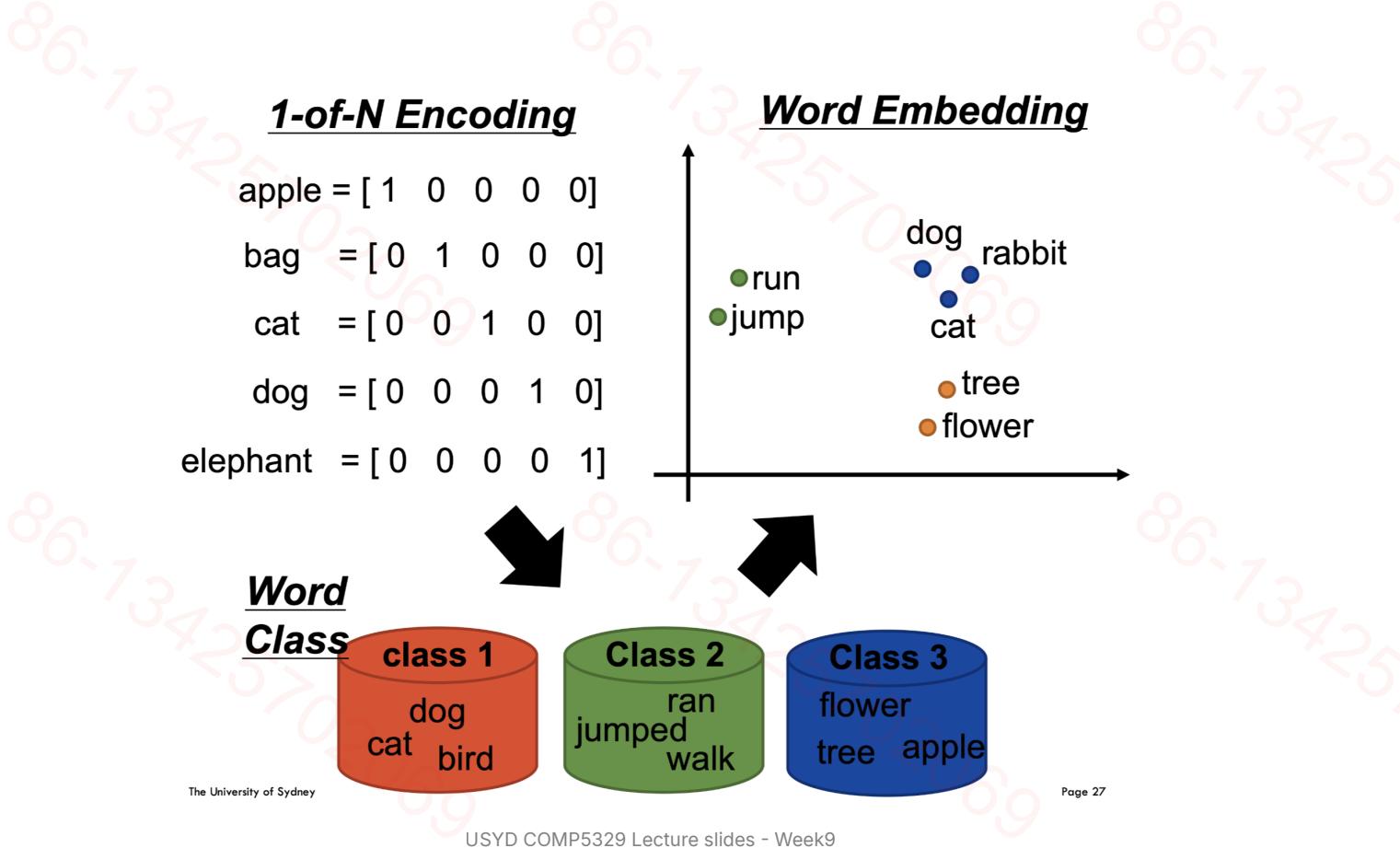
<https://ogre51.medium.com/nlp-explain-bag-of-words-3b9fc4f211e8>

词袋模型的实现和理解都非常简单，**只需统计词频即可**。在处理大规模文本时，**词袋模型可以快速生成词频向量**，便于进一步的文本分析和建模。但是**词袋模型不考虑单词在文本中的顺序，也不考虑语法结构**。这意味着“猫在椅子上”和“椅子在猫上”会被表示为相同的词频向量，尽管它们的语义完全不同。同时词汇表中的单词数量可能非常大，这会导致**词频向量的维度非常高**，从而增加计算复杂度和存储需求。由于每个文本中只包含词汇表中的一小部分单词，**生成的词频向量通常是稀疏的**。这会在某些情况下影响算法的性能。在实际应用中，词袋模型常常与其他文本表示方法（如TF-IDF、词嵌入）结合使用，以提高文本分析和建模的效果。

## 词嵌入 (Word Embeddings)

词嵌入是一种将单词映射到低维连续向量空间的技术，用于捕捉单词之间的语义关系。这种方法解决了独热编码和词袋模型的许多不足，能够更有效地表示文本数据。常见的词嵌入技术包括Word2Vec、GloVe和BERT。词嵌入的核心思想是通过训练，**将每个单词表示为一个实数向量**，这些向量位于一个低维连续空间中。与高维、稀疏的独热编码不同，词嵌入的向量是低维且密集的。这种表示方式能够捕捉到单词之间的语义关系，即**语义相似的单词在向量空间中的距离也更近**。特点为：

- 语义关系**：词嵌入能够捕捉单词之间的语义关系，相似词语的向量距离更近。
- 低维表示**：将高维的独热编码转换为低维的连续向量，有效降低了计算复杂度和存储需求。
- 稠密表示**：词嵌入向量是稠密的，这使得它们能够更高效地参与后续的计算和模型训练。



假设我们使用Word2Vec训练了一个模型，得到了以下单词的词向量：

- "猫" : [0.1, 0.2, 0.3]
- "狗" : [0.1, 0.2, 0.35]
- "桌子" : [0.4, 0.5, 0.6]

在这个例子中，"猫"和"狗"的向量在空间中距离较近，表明它们在语义上是相似的，而"桌子"的向量则远离"猫"和"狗"，表明它们在语义上不相似。

## Recurrent Neural Network

循环神经网络（RNN）是一类能够处理序列数据的神经网络模型，其关键特征是能够记住先前输入的信息，并使用这些信息来影响当前的输出。这使得RNN在自然语言处理等需要处理时间序列数据的任务中非常有效。

### Time Dependency

在处理自然语言时，我们的输入是一串序列，其中的每个单词叫做一个token。token之间的位置关系被视为时序关系，因为输入的顺序会影响其含义。例如：

```
"I saw a saw"
"我看到了一把锯子"
```

在这个例子中，虽然"saw"在句子中出现了两次，但由于其在句子中的位置不同，它们的含义完全不同。第一个"saw"在"I"的后面，作为动词使用，意思是"看见"。第二个"saw"跟在量词"a"后面，作为名词使用，意思是"锯子"。

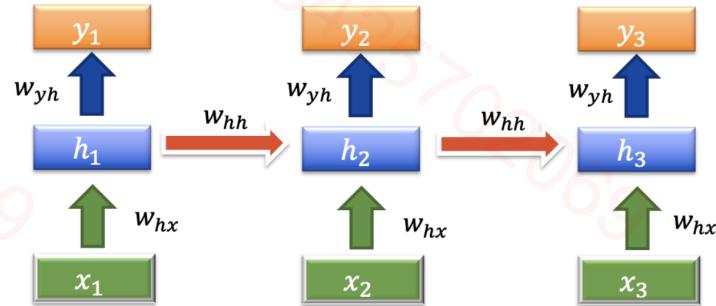
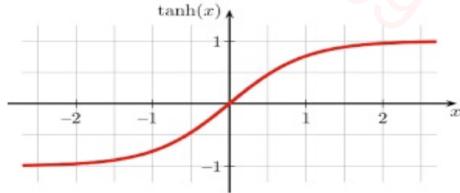
RNN通过其循环结构能够处理这种时序依赖关系。

## Vanishing Gradient

RNN的核心特征是其循环结构，这种结构允许信息从一个时间步骤传递到下一个时间步骤。具体来说，RNN的每个单元不仅接收当前时间步骤的输入，还接收前一个时间步骤的隐藏状态。这使得RNN能够“记住”前面的信息，并将其用于当前的计算。

### Formally

- $x_t$  is the input
- $h_t$  is the hidden state
- $y_t$  is the output
- $h_0 = \vec{0}$
- $h_t = \tanh(w_{hh}h_{t-1} + w_{hx}x_t + b_h)$
- $y_t = w_{yh}h_t$



The University of Sydney

Page 13

USYD COMP5329 Lecture slides - Week8

在训练RNN时，我们需要对时间步进行展开，这意味着RNN在每个时间步都有两个输入：当前时间步的输入  $x_t$  和前一时间步的隐藏状态  $h_{t-1}$ 。具体的计算如下：

$$h_t = \tanh(W_{hh}h_{t-1} + W_{hx}x_t + b_h)$$

这种递归关系可以展开成前面的多个时间步，比如：

$$h_t = \tanh(W_{hh} \cdot \tanh(W_{hh} \cdot \tanh(\cdots \tanh(W_{hh}h_0 + W_{hx}x_1 + b_h) + W_{hx}x_2 + b_h) + \cdots) + W_{hx}x_t + b_h)$$

在反向传播中，我们需要计算损失函数相对于网络参数的梯度。对于RNN，这涉及到对多个时间步展开的求导。具体来说，损失函数  $L$  对隐藏状态  $h_t$  的梯度计算涉及链式法则：

$$\frac{\partial L}{\partial W_{hh}} = \sum_{t=1}^T \frac{\partial L}{\partial h_t} \frac{\partial h_t}{\partial W_{hh}}$$

由于  $h_t$  是通过  $h_{\{t-1\}}$  递归计算得到的，链式法则将引入多次相乘的梯度项：

$$\frac{\partial h_t}{\partial W_{hh}} = \frac{\partial h_t}{\partial h_{t-1}} \frac{\partial h_{t-1}}{\partial h_{t-2}} \cdots \frac{\partial h_1}{\partial W_{hh}}$$

这种链式的梯度计算导致了梯度的指数级变化。当权重矩阵  $W_{hh}$  的特征值大于1时，梯度会呈指数级增长，导致梯度爆炸；当特征值小于1时，梯度会呈指数级衰减，导致梯度消失。具体来说：

- **梯度爆炸 (Gradient Explosion)**：如果权重矩阵的特征值大于1，梯度会随着时间步的增加呈指数级增长。这会导致权重更新幅度过大，使得模型参数不稳定，甚至发生数值溢出。
- **梯度消失 (Gradient Vanishing)**：如果权重矩阵的特征值小于1，梯度会随着时间步的增加呈指数级衰减。这会导致梯度逐渐变得非常小，使得模型几乎无法更新参数，训练过程停止。

### □ Vanishing gradient problem

$$\square h_t = \tanh(w_{hh}h_{t-1} + w_{hx}x_t + b_h)$$

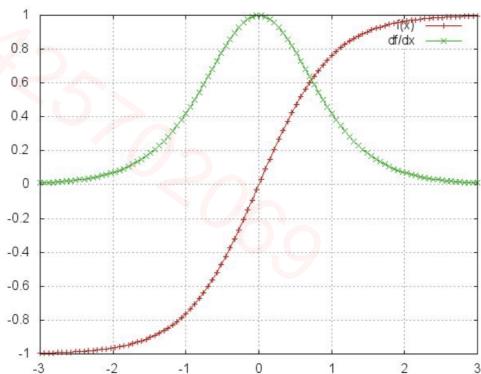
$$\square y_t = w_{yh}h_t$$

$\square$  For every step, its loss is  $E_n$

$$\frac{\partial E_3}{\partial w_{hh}} = \frac{\partial E_3}{\partial y_3} \frac{\partial y_3}{\partial h_3} \frac{\partial h_3}{\partial w_{hh}}$$

$0.9^{1000} \approx 0$
$1.01^{1000} \approx 21000$

$$(\tanh)'w_{hh}(\tanh)'w_{hh}(\tanh)'w_{hh} \dots$$



$$(\tanh)' \leq 1$$

If  $0 < w_{hh} < 1$ , Vanishing Gradients

$$(\tanh)'w_{hh}(\tanh)'w_{hh}(\tanh)'w_{hh} \dots \rightarrow 0$$

If  $w_{hh}$  is larger, Exploding Gradients

$$(\tanh)'w_{hh}(\tanh)'w_{hh}(\tanh)'w_{hh} \dots \rightarrow \infty$$

The University of Sydney

Page 15

USYD COMP5329 Lecture slides - Week8

在训练长序列时，RNN可能会遇到梯度消失或梯度爆炸问题。这使得模型难以学习到长距离的依赖关系。RNN的计算是顺序的，这使得并行化处理困难，训练时间较长。RNN难以捕捉到长序列中的长期依赖关系，虽然LSTM和GRU等变体在一定程度上缓解了这个问题，但仍然存在挑战。

### Memory of RNN

尽管RNN没有明确的内存模块，但它通过其循环连接能够记住一些信息，并在后续的时间步骤中使用这些信息。例如，当我们说“故人西辞黄鹤楼”，RNN可能会预测下一句是“烟花三月下扬州”。这种记忆是通过模型的权重和结构涌现出来的。

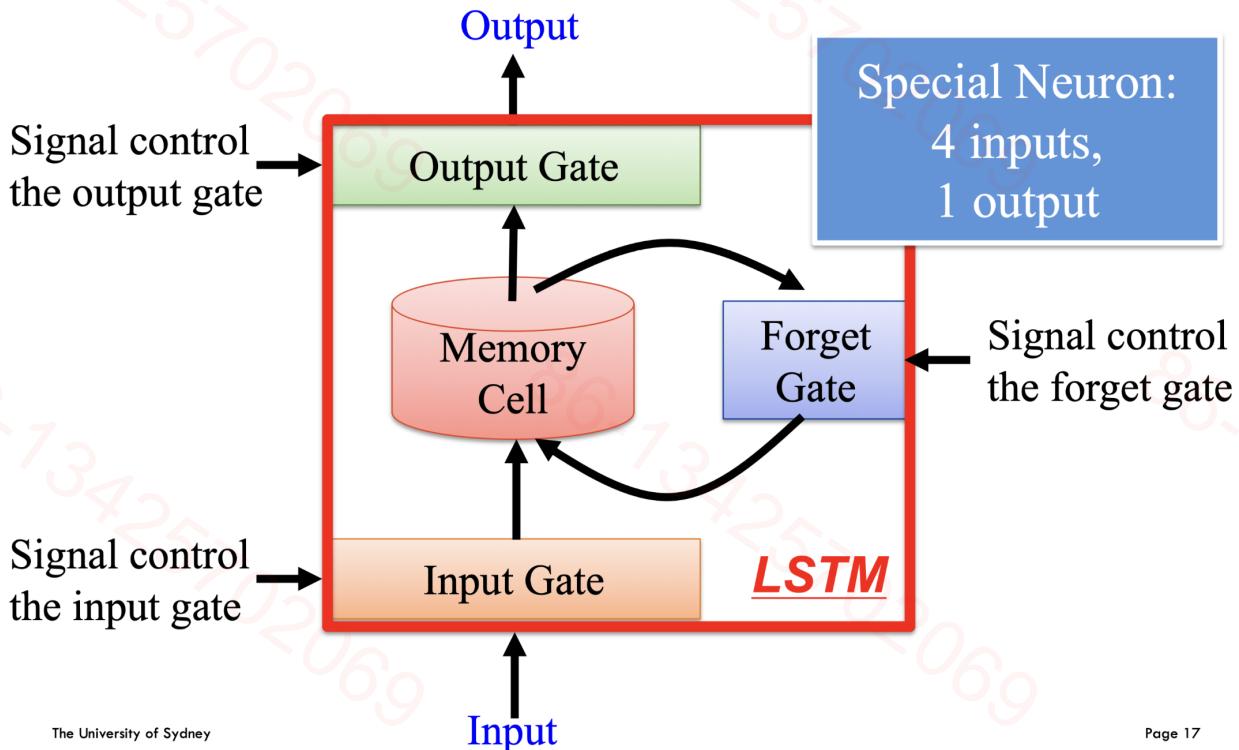
“涌现现象”是复杂系统理论中的一个概念，指从简单的规则中自然产生复杂的行为。在RNN的上下文中，尽管每个单独的神经元功能相对简单，整个网络却能通过这些简单的交互产生复杂的行为模式，如语言中的语法结构和长期依赖关系的学习。这种从简单到复杂的行为是涌现的一种形式。

### LSTM

LSTM (Long Short-Term Memory) 网络是一种能够有效解决RNN梯度消失问题的变体。通过引入门控机制，LSTM在处理长序列时表现出色。以下是对LSTM的详细介绍，特别是门控特性和如何解决RNN梯度消失问题。

### Gates

LSTM包含三个主要的门：输入门、遗忘门和输出门。这些门通过控制信号（通常使用sigmoid函数）来调节信息流，从而实现对信息的选择性记忆和遗忘。



The University of Sydney

Page 17

USYD COMP5329 Lecture slides - Week8

- **输入门 (Input Gate)** : 决定哪些新信息需要存储到单元状态。

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$

- **遗忘门 (Forget Gate)** : 决定哪些信息需要从单元状态中删除。

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$$

- **输出门 (Output Gate)** : 基于当前的单元状态和输入，决定最终的输出。

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$$

- **单元状态更新**：单元状态 (Cell State) 存储跨时间步的信息，并在每个时间步进行更新。

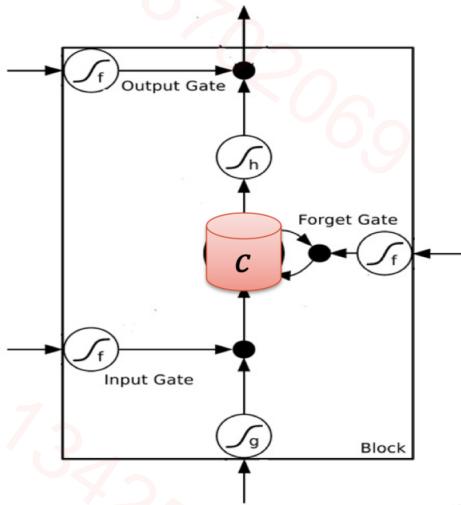
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$$

$$h_t = o_t \cdot \tanh(C_t)$$

- Candidate cell state
  - $\tilde{c}_t = \tanh(w_{hh}h_{t-1} + w_{hx}x_t + b_h)$

- Update cell state
  - $c_t = f_t * c_{t-1} + i_t * \tilde{c}_t$
- Output state
  - $h_t = o_t * \tanh(c_t)$



The University of Sydney

• 28

USYD COMP5329 Lecture slides - Week8

## Additive Gradient

LSTM通过加性梯度而非乘性梯度解决了RNN的梯度消失问题。具体来说，LSTM的单元状态  $C_t$  直接通过遗忘门和输入门进行加性更新，而不是像RNN那样逐步通过非线性激活函数传播，这使得梯度不会在长序列中指数级衰减。

- RNN的梯度传播：

$$\frac{\partial L}{\partial h_t} = \frac{\partial L}{\partial h_{t+1}} \cdot \frac{\partial h_{t+1}}{\partial h_t} + \frac{\partial L}{\partial y_t} \cdot \frac{\partial y_t}{\partial h_t}$$

由于激活函数的存在， $\frac{\partial h_{t+1}}{\partial h_t}$  会导致梯度逐步衰减或爆炸。

- LSTM的梯度传播：

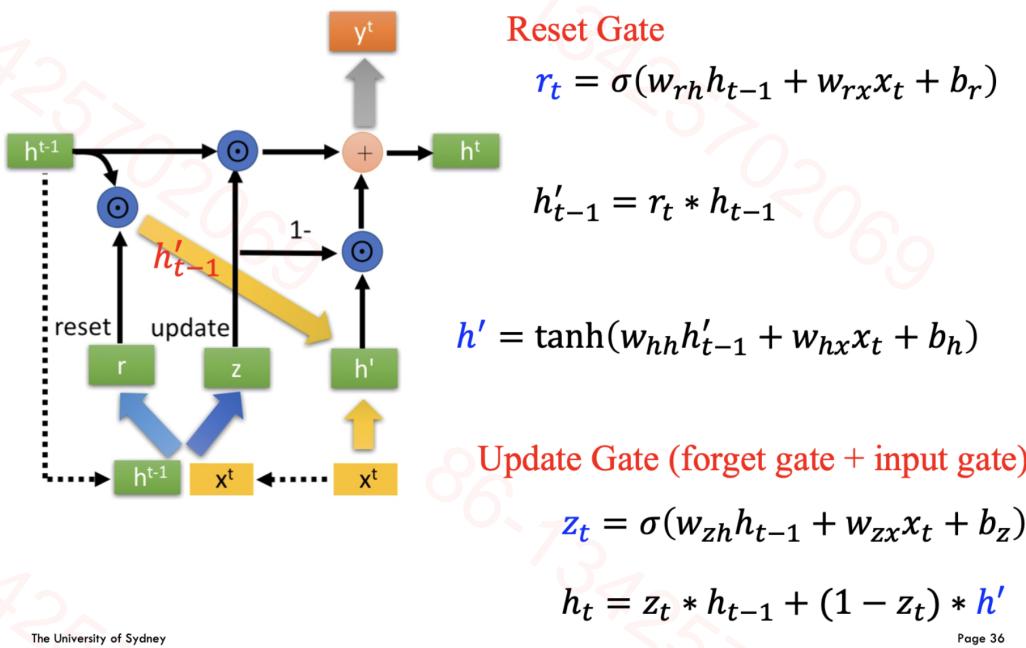
$$\frac{\partial L}{\partial C_t} = \frac{\partial L}{\partial C_{t+1}} \cdot \frac{\partial C_{t+1}}{\partial C_t} + \frac{\partial L}{\partial h_t} \cdot \frac{\partial h_t}{\partial C_t}$$

由于  $C_t$  的更新是加性的 ( $C_t = f_t \cdot C_{t-1} + i_t \cdot \tilde{C}_t$ )，梯度不会在长序列中指数级衰减。

LSTM通过引入输入门、遗忘门和输出门，门控机制允许LSTM选择性地记忆和遗忘信息，有效地解决了RNN的梯度消失问题。其加性梯度更新机制使得LSTM能够处理长序列数据，捕捉长期依赖关系。然而，LSTM也因其复杂性和计算成本较高而存在一定的局限性。通过合理的设计和优化，LSTM在自然语言处理等领域表现出色。

## Gated Recurrent Unit

门控循环单元（Gated Recurrent Unit, GRU）是由Cho等人在2014年提出的一种循环神经网络（RNN）的变体。GRU的设计目的是在保留LSTM性能的同时，简化模型的结构。虽然LSTM在处理长时间依赖方面表现优异，但其复杂的门控机制和较多的参数增加了计算和训练的难度。GRU通过减少门控的数量和参数量，提供了一种计算更高效且易于训练的替代方案。



The University of Sydney

USYD COMP5329 Lecture slides - Week8

Page 36

GRU结合了LSTM的输入门和遗忘门的功能，只使用两个门：重置门和更新门。这两个门可以更简单地控制信息流，从而减少计算复杂度。

- **重置门 (Reset Gate)** : 控制前一时刻的信息在当前时刻的候选隐状态中的影响。

$$r_t = \sigma(W_r \cdot [h^{t-1}, x^t] + b_r)$$

- **更新门 (Update Gate)** : 控制前一时刻的隐状态在当前时刻的隐状态中的影响，决定了多少前一时刻的信息需要保留到当前时刻。

$$z_t = \sigma(W_z \cdot [h^{t-1}, x^t] + b_z)$$

- **候选隐状态 (Candidate Hidden State)** : 综合当前输入和前一时刻的信息生成候选隐状态。

$$\tilde{h}_t = \tanh(W_h \cdot [r_t \odot h^{t-1}, x^t] + b_h)$$

- **隐状态更新** : 结合前一时刻的隐状态和候选隐状态，得到当前时刻的隐状态。

$$h_t = (1 - z_t) \odot h^{t-1} + z_t \odot \tilde{h}_t$$

在上述公式中， $\sigma$  表示sigmoid函数， $\tanh$  表示tanh激活函数， $\odot$  表示逐元素乘积（Hadamard乘积）。

相对于LSTM，**GRU结构更简单**，只包含两个门（重置门和更新门），这减少了模型的参数量和计算复杂度。由于参数量较少，**GRU在训练时的计算效率更高，收敛速度更快**。

## Transformer

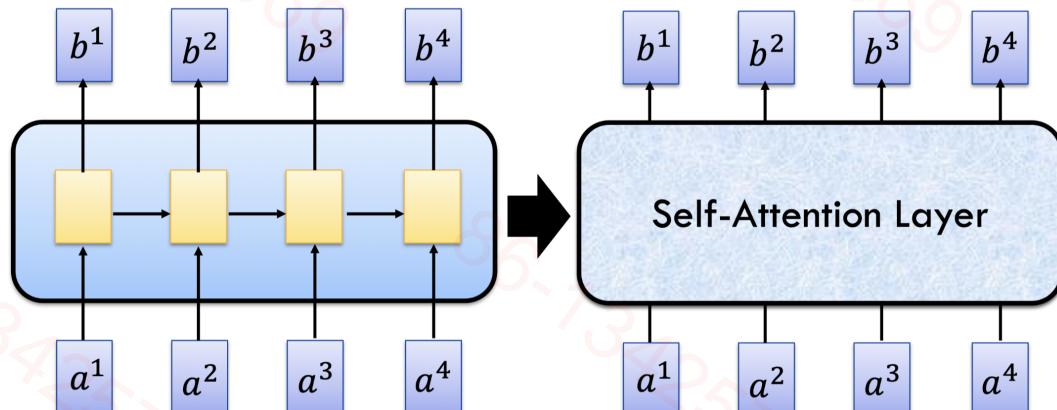
### Transformer Architecture

Transformer模型是由Vaswani等人在2017年提出的一种新型神经网络结构，用于处理序列数据任务，特别是在自然语言处理（NLP）领域表现卓越。与传统的循环神经网络（RNN）和长短时记忆网络（LSTM）不同，Transformer完全基于注意力机制（Attention Mechanism），无需递归或卷积运算，从而大幅提升了并行计算的效率。主要特点为：

- 自注意力机制（Self-Attention Mechanism）**：这是Transformer的核心组件。自注意力机制允许模型在处理某个单词（或位置）时，同时考虑输入序列中所有单词（或位置）的信息。自注意力机制通过计算输入序列中每个元素与其他元素的相似度（即注意力权重），来调整每个元素的表示。
- 并行计算**：由于Transformer没有循环结构，可以在序列的所有位置上并行计算，这极大地提高了计算效率。相比之下，RNN和LSTM必须顺序处理每个时间步。
- 编码器-解码器结构（Encoder-Decoder Architecture）**：Transformer采用编码器-解码器架构，其中编码器将输入序列转换为一系列连续表示，解码器根据这些表示生成输出序列。每个编码器和解码器层都包含两个主要子层：多头自注意力机制和前馈神经网络（Feedforward Neural Network）。
- 多头注意力机制（Multi-Head Attention Mechanism）**：这是自注意力机制的扩展版本，通过并行地计算多个独立的注意力机制（称为“头”），来捕捉不同的子空间表示。多头注意力机制允许模型关注输入序列中的不同部分，并在不同层次上进行信息融合。

## Attention Mechanism

Attention机制的主要动机是通过让模型关注输入序列中的重要部分来提高其性能。与传统的卷积神经网络（CNN）和循环神经网络（RNN）相比，Attention机制能够显著提高并行计算的效率，且更好地捕捉长距离依赖关系。在序列到序列的任务中，Attention机制可以替代隐藏层，实现从输入序列到输出序列的映射。



You can try to replace any thing that has been done by RNN with self-attention.

Attention Is All You Need

The University of Sydney  
6 Dec 2017

Ashish Vaswani<sup>\*</sup>  
Google Brain  
avaswani@google.com

Noam Shazeer<sup>\*</sup>  
Google Brain  
noam@google.com

Niki Parmar<sup>\*</sup>  
Google Research  
niki@google.com

Jakob Uszkoreit<sup>\*</sup>  
Google Research  
use@google.com

Llion Jones<sup>\*</sup>  
Google Research  
llion@google.com

Aidan N. Chang<sup>†</sup>  
University of Texas  
aidan@cs.utexas.edu

Lukasz Kaiser<sup>\*</sup>  
Google Brain  
lukasz Kaiser@google.com

Illia Polosukhin<sup>‡</sup>  
illia.polosukhin@gmail.com



Attention is all  
you need.

Page 3

USYD COMP5329 Lecture slides - Week9

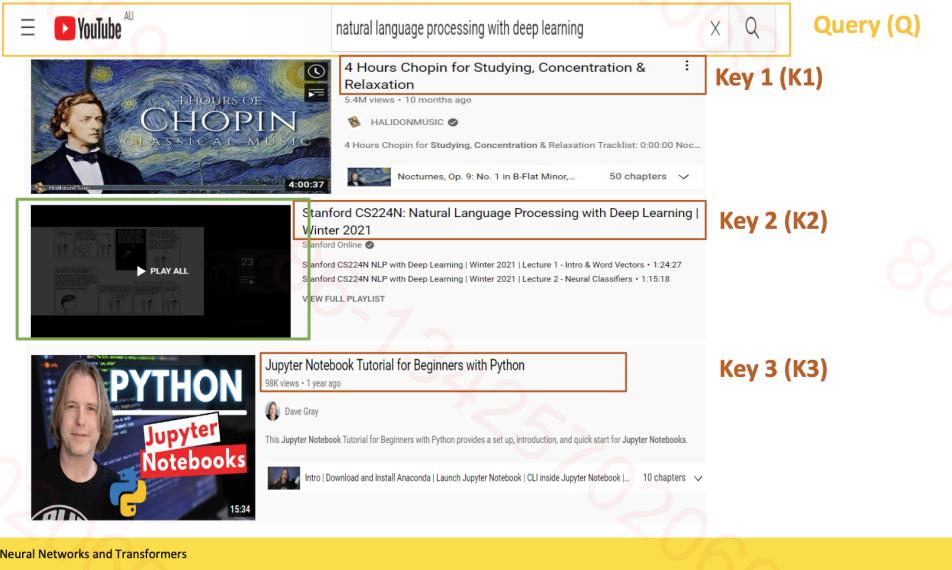
Attention机制模拟了人类的注意力，赋予模型根据任务需求关注不同输入部分的能力。例如，在进行视频搜索时，我们会有个查询（Query），它代表我们感兴趣的内容。通过Query，我们从搜索引擎中找到许多相关信息，这些信息

称为键 (Key)。然后，通过这些Key找到实际的视频 (Value)。因此，Attention机制包括三要素：Query ( $Q$ )、Key ( $K$ ) 和 Value ( $V$ )。

# Self-attention in transformers

- Understanding self-attention with Search
    - Query (Q)

- Query (Q)
  - Key (K)
  - Value (V)



## Self-Attention

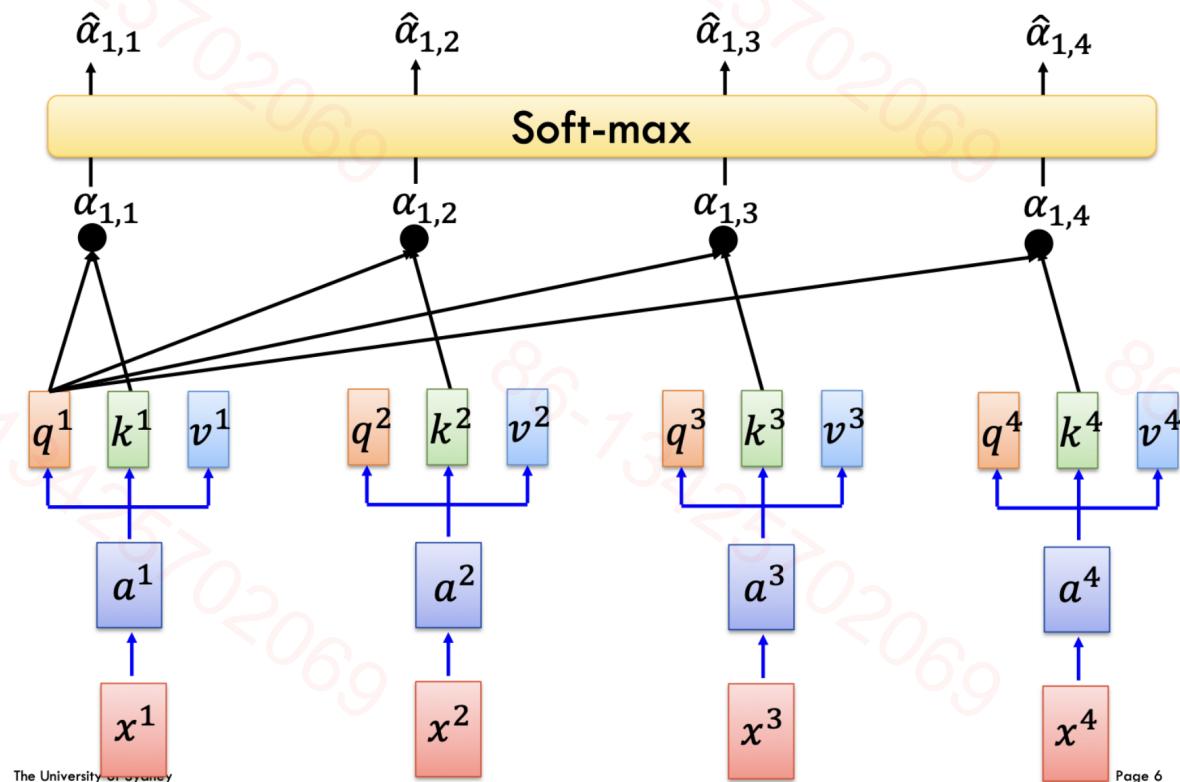
在自注意力机制中，输入序列  $X = [x_1, x_2, \dots, x_n]$  被映射到 Query、Key 和 Value 向量，这些向量用于计算注意力分数并生成输出表示。首先需要计算  $Q = XW_Q$ ,  $K = XW_K$ ,  $V = XW_V$ 。其中， $W_Q$ 、 $W_K$  和  $W_V$  是可学习的权重矩阵。注意力分数为：

$$\text{Attention}(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V$$

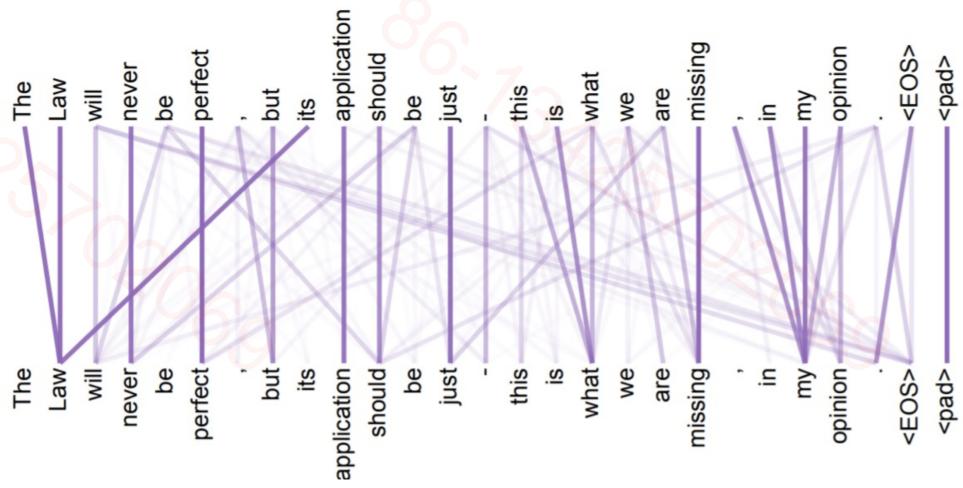
这里， $QK^T$  表示Query和Key之间的点积， $\sqrt{d_k}$  是一个缩放因子，用于避免点积值过大从而导致梯度消失问题。经过 softmax 函数后，注意力分数被归一化为一个概率分布，用于加权 Value。

## Self-Attention

$$\hat{\alpha}_{1,i} = \exp(\alpha_{1,i}) / \sum_j \exp(\alpha_{1,j})$$



USYD COMP5329 Lecture slides - Week9



The University of Sydney

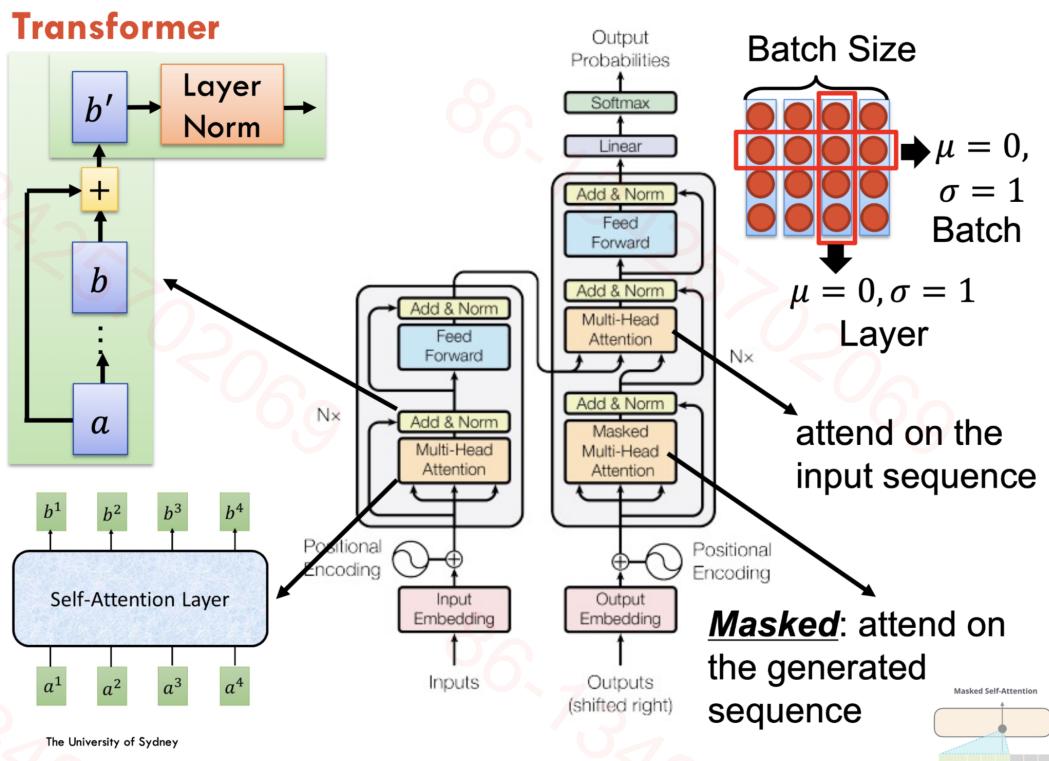
<https://arxiv.org/abs/1706.03762>

USYD COMP5329 Lecture slides - Week9

## Transformer architecture

Transformer一开始被设计出来是按照Encoder-decoder框架设计的，并且采用了残差连接。编码器的任务是压缩输入信息，将其转换为内部表示。编码器由多个相同的编码层（Layer）堆叠而成，每个编码层由一个多头自注意力（Multi-Head Self-Attention）机制和一个前馈神经网络（Feed-Forward Neural Network）组成。每一层的输出通过残差连接（Residual Connection）和层归一化（Layer Normalization）传递给下一层。

解码器的任务是根据编码器的输出生成目标序列，不直接看到输入的input。解码器也由多个相同的解码层堆叠而成，每个解码层由三个部分组成：一个多头自注意力机制，一个多头编码器-解码器注意力机制（Encoder-Decoder Attention），和一个前馈神经网络。解码器同样使用残差连接和层归一化。在训练过程中，解码器输入为部分目标序列，通过掩码机制防止模型看到未来的词。



### Multi-head attention

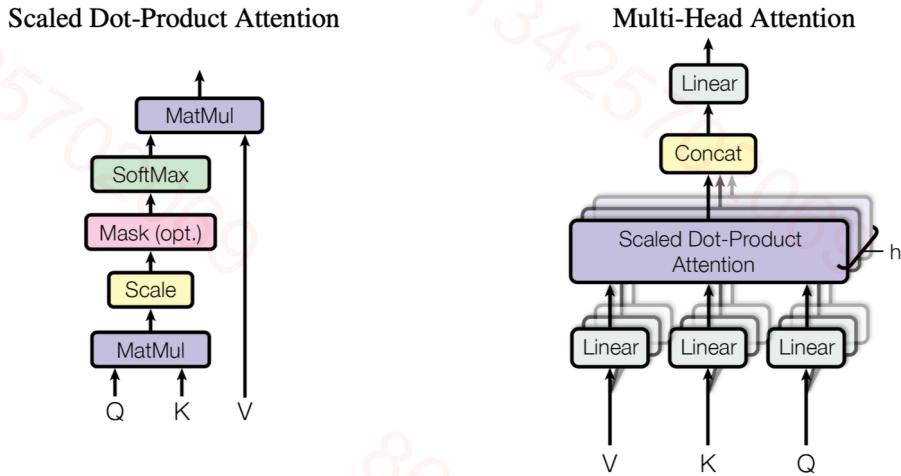


Figure 2: (left) Scaled Dot-Product Attention. (right) Multi-Head Attention consists of several attention layers running in parallel.

<https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fb053c1c4a845aa-Paper.pdf>

多头注意力机制是Transformer架构的核心组件，增强了模型同时关注输入序列不同部分的能力。它允许模型从不同的表示子空间在不同的位置上共同关注信息，提高了注意力机制的性能和鲁棒性。

多头注意力机制通过对查询 ( $Q$ )、键 ( $K$ ) 和值 ( $V$ ) 进行 $h$ 次不同的线性投影，将它们投影到 $d_q$ 、 $d_k$ 和 $d_v$ 维度的子空间中。在这些投影后的查询、键和值的版本上并行执行注意力函数，产生 $d_v$ 维的输出值。这些输出值随后被连接起来，通过最终的线性层投影，得到输出值：

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)W^O$$

其中，每个head代表一个attention， $W^O \in \mathbb{R}^{hd_v \times d_{\text{model}}}$  是最终的投影矩阵。

通过将注意力机制分割成多个头，模型可以同时处理序列的不同部分，增强了并行性和计算效率。每个注意力头可以学习输入序列中的不同模式或特征，类似于卷积神经网络（CNN）中的多个滤波器可以学习不同的空间层次。这有助于模型捕捉数据中的复杂关系。

## Positional Encoding

Transformer模型的自注意力机制允许模型在处理输入时忽略元素的顺序，仅基于元素之间的相似性进行操作。这种无序处理方式有利于并行计算，但它忽略了序列数据（如文本）的时间或位置顺序。为了弥补这一缺陷，我们引入了位置嵌入，确保模型能够捕捉到序列中元素的位置关系。

Transformer论文中采用了正弦 (sin) 和余弦 (cos) 函数来生成位置嵌入。其主要设计思想是利用不同频率的正弦和余弦函数，确保每个位置的嵌入向量在高维空间中是唯一的，并且不同位置之间有规律的变化。具体来说，给定一个序列长度为 $L$ 的输入，位置嵌入使用以下公式生成：

$$\text{PE}(pos, 2i) = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{\text{model}}}}}\right)$$

$$PE(pos, 2i + 1) = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right)$$

其中， $pos$  是序列中的位置。 $i$  是位置嵌入向量的维度索引。 $d_{model}$  是嵌入向量的维度。选择正弦和余弦的原因是：

- **周期性特性**：正弦和余弦函数的周期性使得它们能够自然地表示序列中的位置信息。不同频率的正弦和余弦函数允许位置嵌入在高维空间中形成独特的模式，捕捉到元素之间的相对位置关系。
- **连续性和可微性**：正弦和余弦函数是连续且可微的，这对梯度下降优化过程非常有利。通过这些函数生成的位置嵌入具有平滑的变化，可以帮助模型更好地捕捉到位置之间的关系。
- **无参数设计**：使用正弦和余弦函数生成位置嵌入不需要额外的参数。相比于学习嵌入（learnable embeddings），这种方法更简单，且不需要额外的训练步骤。

## Analysis

在Transformer模型中，Attention机制具有强大的全局信息捕捉能力，能够有效处理长距离依赖的问题。这相较于传统的RNN和CNN有明显的优势。

Table 1: Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types.  $n$  is the sequence length,  $d$  is the representation dimension,  $k$  is the kernel size of convolutions and  $r$  the size of the neighborhood in restricted self-attention.

Layer Type	Complexity per Layer	Sequential Operations	Maximum Path Length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(\log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

<https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fb053c1c4a845aa-Paper.pdf>

- **RNN**: RNN通过递归的方式处理序列数据，依赖于前一个时间步的输出。由于梯度消失和梯度爆炸的问题，RNN在捕捉长距离依赖时效果不佳，并且必须按顺序处理数据，这限制了它的计算可以并行化的程度。
- **CNN**: CNN通过局部感受野捕捉局部特征，但在捕捉全局依赖时可能需要多层堆叠才能完成。这意味着要捕捉到长距离依赖需要更深的网络结构。
- **Self-Attention**: Attention机制可以直接关注序列中的任意两个位置之间的关系，无论它们的距离有多远。通过矩阵计算同时处理所有的序列元素，这使得模型可以充分利用现代硬件架构进行高效的并行计算。在单一操作中就能捕捉全局依赖，不需要多层堆叠，这极大地提高了效率和效果。
- **Restricted Self-Attention**: 这是Self-Attention的一种受限版本，限制了Attention机制的范围，以减少计算复杂度，但在一定程度上牺牲了全局依赖捕捉能力。

## BERT

BERT (Bidirectional Encoder Representations from Transformers) 是一个基于Transformer编码器的双向预训练语言模型，其目标是通过大量无标记文本数据的预训练，生成文本的深层表示。这些表示可以用于多种自然语言处理（NLP）下游任务，如情感分析、问答系统等。BERT的特别之处在于其**双向性**，即模型训练时会同时考虑每个词左右两侧的上下文信息。这与传统的单向语言模型（如GPT）不同。

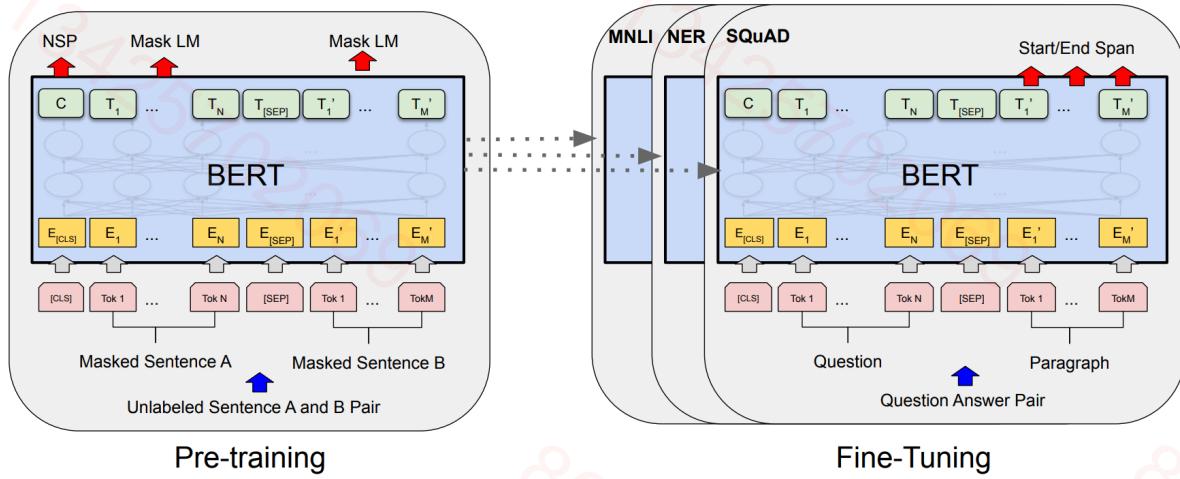


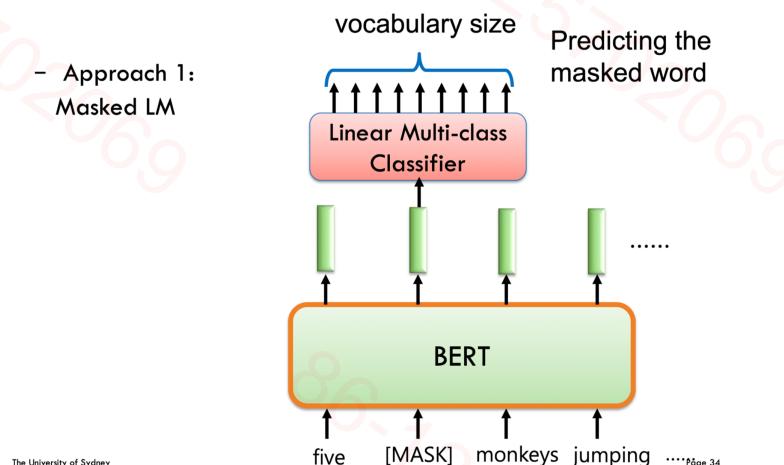
Figure 1: Overall pre-training and fine-tuning procedures for BERT. Apart from output layers, the same architectures are used in both pre-training and fine-tuning. The same pre-trained model parameters are used to initialize models for different down-stream tasks. During fine-tuning, all parameters are fine-tuned. [CLS] is a special symbol added in front of every input example, and [SEP] is a special separator token (e.g. separating questions/answers).

<https://arxiv.org/pdf/1810.04805>

## Masked Language Model

BERT (Bidirectional Encoder Representations from Transformers) 的训练采用了掩码语言模型（Masked Language Model, MLM）和下一个句子预测（Next Sentence Prediction, NSP）两种任务。MLM是其核心任务之一，具体步骤如下：

- **掩码过程**：在输入文本序列中随机选择一部分单词进行掩码，通常掩码的比例为15%。对于选择的单词，有80%的几率用特殊的[MASK]标记替换，有10%的几率用随机的其他单词替换，剩下的10%保持不变。这种设计是为了让模型不仅能学习[MASK]标记，还能适应真实词汇的上下文。
- **模型训练**：BERT模型接收掩码后的序列，尝试预测被掩码的单词。模型的目标是最大化被掩码单词的正确预测概率，即学习到输入序列中每个单词的深层表示。



USYD COMP5329 Lecture slides - Week9

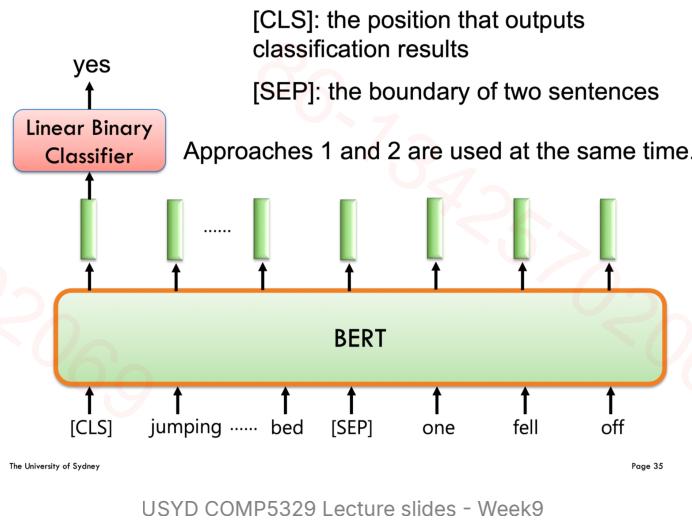
具体来说，给定输入序列  $x_1, x_2, \dots, x_n$ ，其中部分单词被[MASK]替换，模型通过以下方式进行训练：

$$\mathcal{L}_{MLM} = - \sum_{i \in \text{masked indices}} \log P(x_i | x_1, x_2, \dots, x_{i-1}, [\text{MASK}], x_{i+1}, \dots, x_n)$$

通过这种方式，BERT能够同时利用上下文信息进行单词预测，实现了双向建模。

## CLS Token

在BERT模型中，CLS (classification token) 标记是一个特殊的符号，放在每个输入序列的开头。序列标记CLS token用于表示整个输入序列的聚合表示，即整个句子或段落的综合信息。在预训练和微调阶段，CLS标记的输出表示常用于分类任务。例如，在情感分析、文本分类、问答系统等任务中，CLS标记的表示被用来进行分类决策。



BERT同时利用上下文信息，使得模型在理解和生成自然语言时更加准确。BERT的深层表示可以应用于多种下游任务，只需通过微调即可适应具体任务。通过大量无标记文本的预训练，BERT能够在有限的标记数据上表现出色。但是BERT的双向性和大规模预训练使得其计算复杂度和资源需求较高。虽然BERT在捕捉局部和全局上下文信息方面表现出色，但其处理长序列时的效率仍然有限。

## InstructGPT

InstructGPT是OpenAI开发的一种基于GPT-3.5的模型，通过整合“人类反馈强化学习”（Reinforcement Learning from Human Feedback, RLHF）的方法来增强大型语言模型（LLMs）与用户意图的对齐。这种方法解决了之前模型中观察到的一些缺陷，例如生成无关、不真实或甚至有害的输出。

## AutoRegressive Loss

GPT采用自回归模型，即在生成下一个词的预测时，会使用之前所有生成的词作为上下文。具体来说，模型通过最大化每一步生成下一个词的概率来进行训练。假设输入序列为 $x_1, x_2, \dots, x_n$ ，模型的目标是最大化下述似然函数：

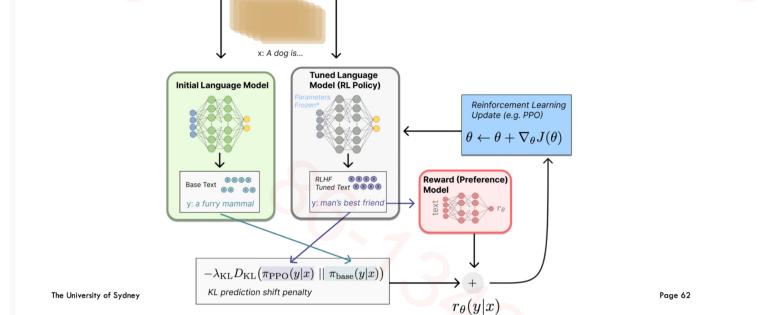
$$P(x_1, x_2, \dots, x_n) = \prod_{t=1}^n P(x_t | x_1, x_2, \dots, x_{t-1})$$

这种自回归损失函数确保了每一步的输出都依赖于之前的输出，从而实现连贯文本的生成。

## RLHF (Reinforcement Learning from Human Feedback)

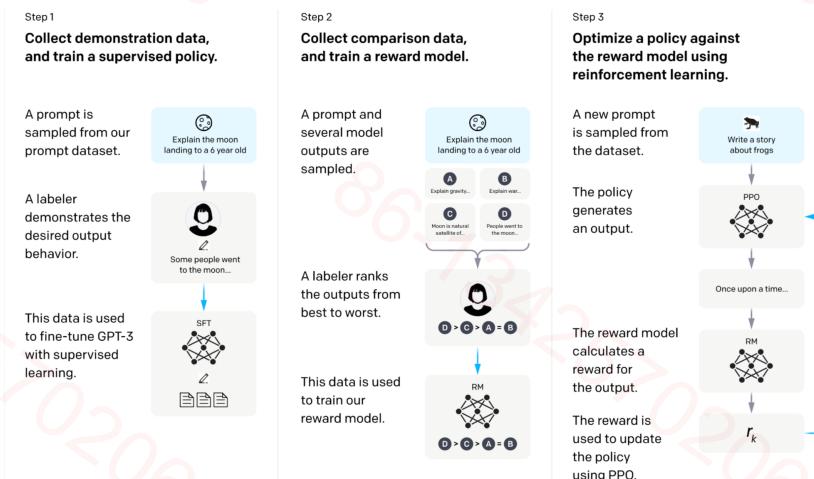
在InstructGPT中，RLHF用于增强模型与用户意图的对齐。首先训练一个**奖励模型（Reward Model, RM）**，RM接收一段文本序列并返回代表人类偏好的标量奖励。训练集通过抽取提示并通过初始语言模型生成新文本，然后由人类注释者对这些输出进行排名。人类注释者对来自语言模型的输出进行排名，而不是直接打分。

- Fine-tuning the LM by reinforcement learning
1. Sample a prompt
  2. Pass it through both initial LM and the RL Policy (Copy of the initial LM)
  3. Pass the output from policy to RM to calculate reward, and use output from both initial LM and policy to calculate shift penalty
  4. Use reward and penalty together to update the policy by PPO (Proximal Policy Optimization)



USYD COMP5329 Lecture slides - Week9

策略更新的方式是，抽取一个提示。将其通过初始语言模型和强化学习策略（初始语言模型的副本）传递。将策略的输出传递给RM以计算奖励，并使用初始语言模型和策略的输出来计算偏移惩罚。使用奖励和惩罚一起通过近端策略优化（Proximal Policy Optimization, PPO）更新策略。



USYD COMP5329 Lecture slides - Week9

通过RLHF，InstructGPT能够更好地理解和响应用户意图，生成更相关、更有用的内容。结合了自回归预训练和强化学习微调双重训练，确保了模型在生成连贯文本的同时，也能更好地符合人类偏好。通过RLHF，**模型能够从人类反馈中持续学习和改进**，使其在各种应用场景中表现出色。

## Vision Transformer

Vision Transformer (ViT) 是Google Brain团队在2020年提出的一种将Transformer架构应用于图像分类任务的新方法。ViT展示了在处理大规模数据集时，其性能可以超越传统的卷积神经网络 (CNN)。

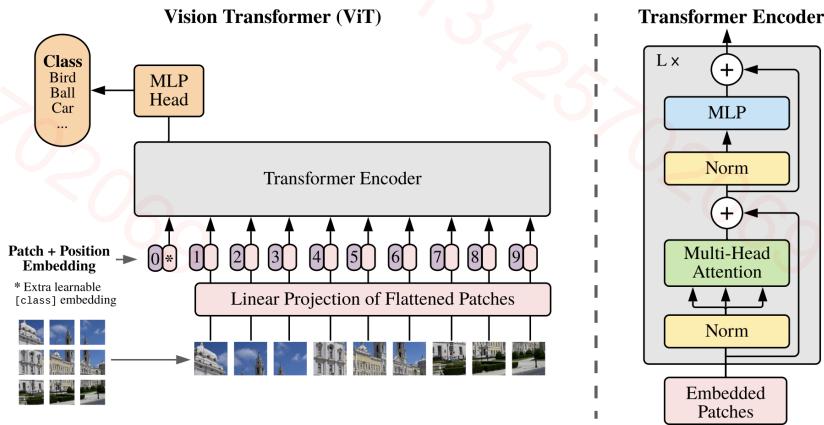


Figure 1: Model overview. We split an image into fixed-size patches, linearly embed each of them, add position embeddings, and feed the resulting sequence of vectors to a standard Transformer encoder. In order to perform classification, we use the standard approach of adding an extra learnable “classification token” to the sequence. The illustration of the Transformer encoder was inspired by Vaswani et al. (2017).

USYD COMP5329 Lecture slides - Week9

ViT分割图像为补丁，将输入图像分割成多个小块（patches），每个补丁通常是正方形，例如 $16 \times 16$ 像素。这与在自然语言处理中将文本分割成词或子词单元类似。每个补丁被展平并通过一个线性层转换成固定大小的向量。由于Transformer模型不处理序列中的位置信息，ViT为每个补丁添加位置嵌入，以保留图像中的空间信息。

嵌入后的补丁和位置嵌入的合并向量被送入标准的Transformer编码器。编码器通过多层自注意力机制和前馈网络处理这些向量，捕获补丁之间的复杂关系。自注意力机制使ViT能够捕捉全局特征，而不仅仅是局部特征。ViT在大规模数据集上表现出色，能够学习更复杂和全局的图像特征。

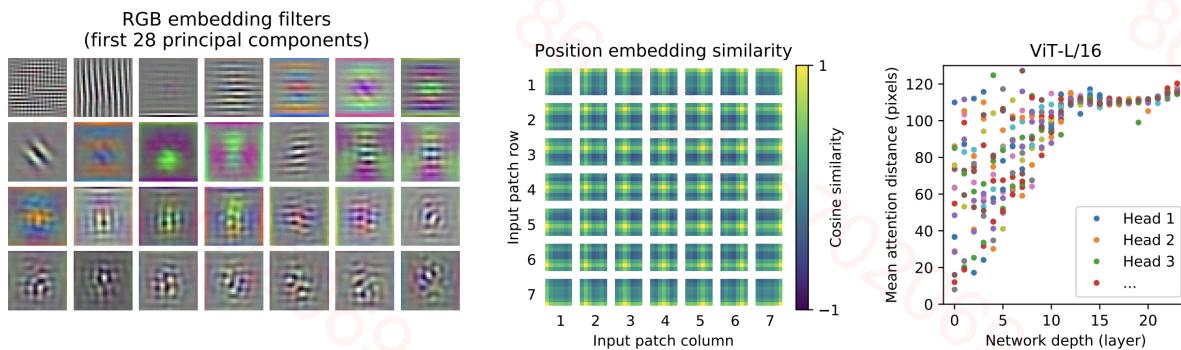


Figure 7: **Left:** Filters of the initial linear embedding of RGB values of ViT-L/32. **Center:** Similarity of position embeddings of ViT-L/32. Tiles show the cosine similarity between the position embedding of the patch with the indicated row and column and the position embeddings of all other patches. **Right:** Size of attended area by head and network depth. Each dot shows the mean attention distance across images for one of 16 heads at one layer. See Appendix D.7 for details.

USYD COMP5329 Lecture slides - Week9

和CNN相比CNN通过局部感受野和卷积层逐步构建特征层次，主要擅长捕捉局部特征。ViT通过自注意力机制直接捕捉全局特征，使得模型能够在处理大规模数据集时表现更好。由于CNN具有归纳偏置，所以在较小的数据集上也能表

现良好，但在大规模数据集上同样需要大量数据来提升性能。ViT需要大量的预训练数据（至少1亿张图片）才能达到令人满意的性能，这可能限制其在数据较少的应用场景中的效用。计算复杂度较低，适合处理高分辨率图像和大规模数据集。自注意力机制在计算上更为昂贵，尤其是在处理高分辨率图像时。

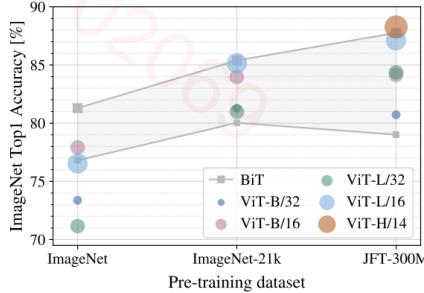


Figure 3: Transfer to ImageNet. While large ViT models perform worse than BiT ResNets (shaded area) when pre-trained on small datasets, they shine when pre-trained on larger datasets. Similarly, larger ViT variants overtake smaller ones as the dataset grows.

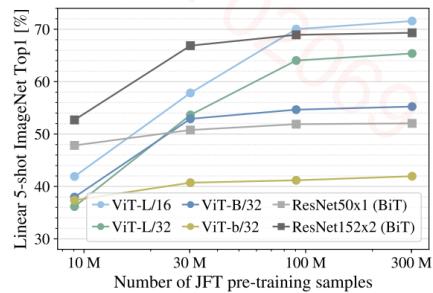


Figure 4: Linear few-shot evaluation on ImageNet versus pre-training size. ResNets perform better with smaller pre-training datasets but plateau sooner than ViT, which performs better with larger pre-training. ViT-b is ViT-B with all hidden dimensions halved.

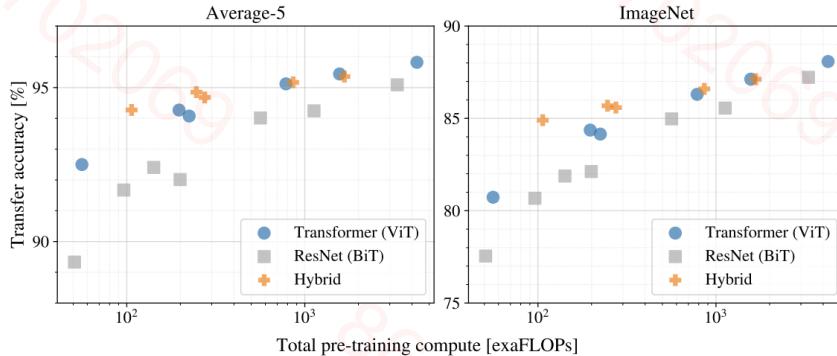


Figure 5: Performance versus pre-training compute for different architectures: Vision Transformers, ResNets, and hybrids. Vision Transformers generally outperform ResNets with the same computational budget. Hybrids improve upon pure Transformers for smaller model sizes, but the gap vanishes for larger models.

## Exercise

### Ex 1

Consider a convolutional neural network which takes as input a  $65 \times 77$  color image (i.e., with three channels R, G, B). The first convolutional layer has 18 filters that are 5-by-5, with stride 3 and no zero-padding.

Compute the number of:

1. Weights per filter in this layer (including bias):
2. Neurons in this layer:
3. Connections into the neurons in this layer:
4. Independent parameters in this layer:

## **Ex 2**

The Context Layer in a Simple Recurrent Network:

- a. is computed from the current input and the previous hidden layer
- b. is comprised of the inputs in a sliding window around the current timestep
- c. is a copy of the hidden layer from the previous timestep
- d. is computed from the current input and the previous output

## **Ex 3**

Which of these architectures would have the best chance of learning long-range dependencies?

- a. Long Short Term Memory
- b. Elman Network
- c. Simple Recurrent Network
- d. Feedforward network with sliding window

## **Ex 4**

Two common methods for unsupervised pre-training of neural networks are:

- a. Deep Boltzmann Machine and Bayesian Inference
- b. Weight Initialization and Autoencoder
- c. Bayesian Inference and Weight Initialization
- d. Autoencoder and Deep Boltzmann Machine

## **Ex 5**

## Singular Value Decomposition

Co-occurrence matrix  $X_{(L \times M)}$  can be decomposed as  $X = U S V^T$  where  $U_{(L \times L)}$ ,  $V_{(M \times M)}$  are unitary (all columns have unit length) and  $S_{(L \times M)}$  is diagonal, with diagonal entries  $s_1 \geq s_2 \geq \dots \geq s_M \geq 0$

$$\begin{matrix} & M \\ L & \boxed{\phantom{000}} \end{matrix} \approx \begin{matrix} & N \\ L & \boxed{\begin{matrix} \underline{\mathbf{u}_1} \\ \underline{\mathbf{u}_2} \\ \vdots \\ \underline{\mathbf{u}_k} \end{matrix}} \end{matrix} \begin{matrix} & N \\ & \boxed{\begin{matrix} s_1 & s_2 \\ & \vdots \\ & s_N \end{matrix}} \end{matrix} \begin{matrix} & M \\ & \boxed{\begin{matrix} | & | \\ \mathbf{v}_1 & \mathbf{v}_2 \\ | & | \end{matrix}} \\ N \end{matrix}$$

$\mathbf{X}$        $\tilde{\mathbf{U}}$        $\tilde{\mathbf{S}}$        $\tilde{\mathbf{V}}^T$

We can obtain an approximation for  $X$  of rank  $N < M$  by truncating  $U$  to  $\tilde{U}_{(L \times N)}$ ,  $S$  to  $\tilde{S}_{(N \times N)}$  and  $V$  to  $\tilde{V}_{(N \times M)}$ . The  $k$ th row of  $\tilde{U}$  then provides an  $N$ -dimensional vector representing the  $k^{\text{th}}$  word in the vocabulary.

Considering a Singular Value Decomposition  $X = USV^T$ , what are the special properties of matrices  $U$ ,  $S$ , and  $V$ ?

- a.  $U$  is orthogonal,  $V$  is upper triangular, and  $S$  is symmetric.
- b.  $U$ ,  $V$  are upper triangular, and  $S$  is diagonal.
- c.  $U$ ,  $V$  are symmetric and  $S$  is orthogonal.
- d.  $U$ ,  $V$  are unitary and  $S$  is diagonal.

### Ex 6

## word2vec (Mikolov et al., 2013)

- **Idea:** predict rather than count
- Instead of counting how often each word  $w$  occurs near "university" train a classifier on a **binary prediction task**:
  - Is  $w$  likely to show up near "university"?
- We don't actually care about this task
  - But we'll take the learned classifier weights as the word embeddings
- Use running text as implicitly supervised training data
- No need for hand-labeled supervision

24



## Word2Vec issues

- Word2Vec is a linear model in the sense that there is no activation function at the hidden nodes
- this 1-word prediction model can be extended to multi-word prediction in two different ways:
  - Continuous Bag of Words
  - Skip-Gram
- need a computationally efficient alternative to Softmax (Why?)
  - Hierarchical Softmax
  - Negative Sampling
- need to sample frequent words less often

27



## Cost Function

Softmax can be used to turn these linear sums  $u_j$  into a probability distribution estimating the probability of word  $j$  occurring in the context of word  $k$

$$\text{prob}(j|k) = \frac{\exp(u_j)}{\sum_{j'=1}^V \exp(u_{j'})} = \frac{\exp(\mathbf{v}_j^T \mathbf{v}_k)}{\sum_{j'=1}^V \exp(\mathbf{v}_{j'}^T \mathbf{v}_k)}$$

We can treat the text as a sequence of numbers  $w_1, w_2, \dots, w_T$  where  $w_i = j$  means that the  $i^{\text{th}}$  word in the text is the  $j^{\text{th}}$  word in the vocabulary.

We then seek to maximize the log probability

$$\frac{1}{T} \sum_{t=1}^T \sum_{-c \leq r \leq c, r \neq 0} \log \text{prob}(w_{t+r} | w_t)$$

where  $c$  is the size of training context (which may depend on  $w_t$ )

26



Which statement about word2vec is FALSE?

- a. Representations for the same word at the input and output layers are different
- b. It aims to maximise the log probability of a word, based on the surrounding words
- c. The tanh activation function is used at the hidden nodes
- d. Performance improves if frequent words are sampled less often

### Ex 7

Base rates of women having breast cancer and having no breast cancer are 0.02% and 99.98% respectively. The true positive rate or sensitivity  $P(\text{positive mammography} | \text{breast cancer}) = 85\%$  and the true negative or specificity  $P(\text{negative mammography} | \sim\text{breast cancer}) = 95\%$ . Compute  $P(C | M)$ .

### Ex 8

Write the formula for Bayes' Rule, in terms of a cause A and an effect B.

### Ex 9

In the context of supervised learning, explain the difference between maximum likelihood estimation and Bayesian inference.