常用深度学习模型

前文提到：

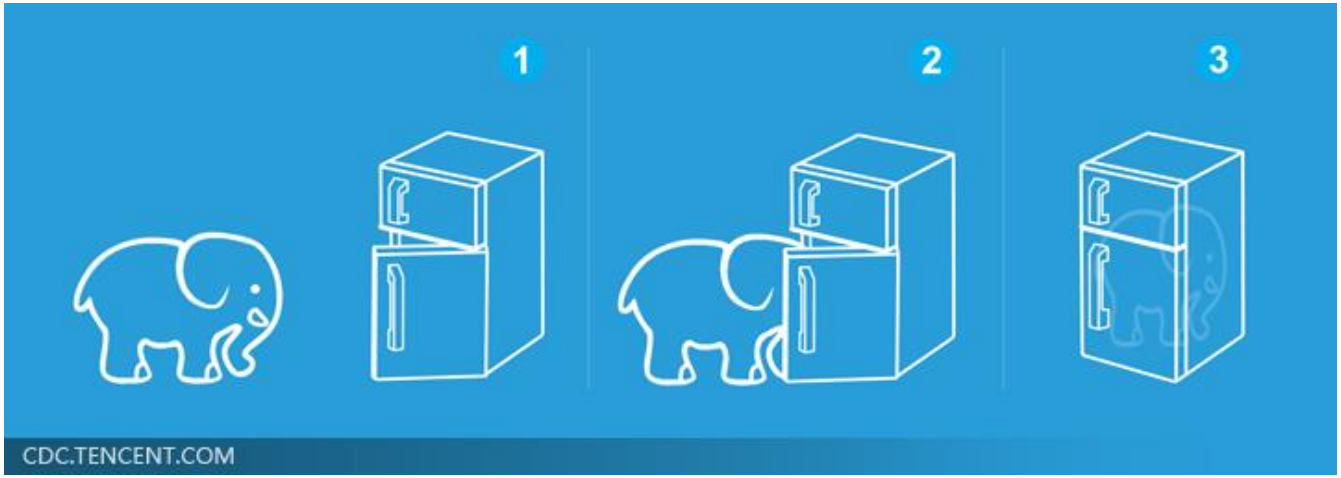
Three Steps for Deep Learning

Step 1: define a set of function

Step 2: goodness of function

Step 3: pick the best function

Deep Learning is so simple ……



主流深度学习模型

根据深度神经网络结构差异，代表性的有

- 卷积神经网络（Convolution Neural Network，CNN）
- 循环神经网络（Recurrent Neural Network，RNN）
  - 长短期记忆网络（Long-short Term Memory，LSTM）
  - 门控循环单元（Gate Recurrent Unit，GRU）

◆ 注意力机制

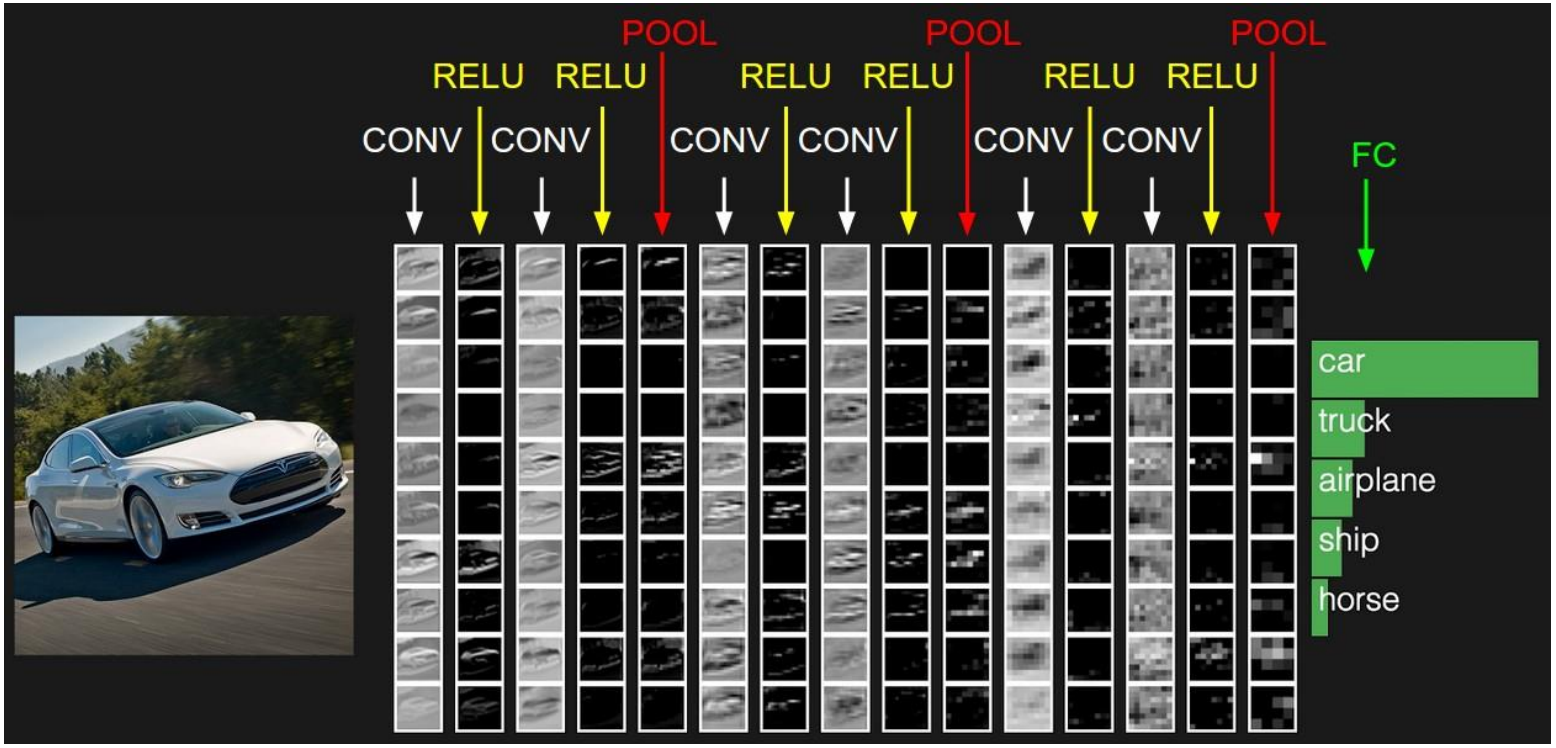
- Transformer

主要内容

- **卷积神经网络（Convolution Neural Network，CNN）**
- 循环神经网络（Recurrent Neural Network，RNN）
  - 长短期记忆网络（Long-short Term Memory，LSTM）
  - 门控循环单元（Gate Recurrent Unit，GRU）
- 注意力机制（Attention）
- Transformer

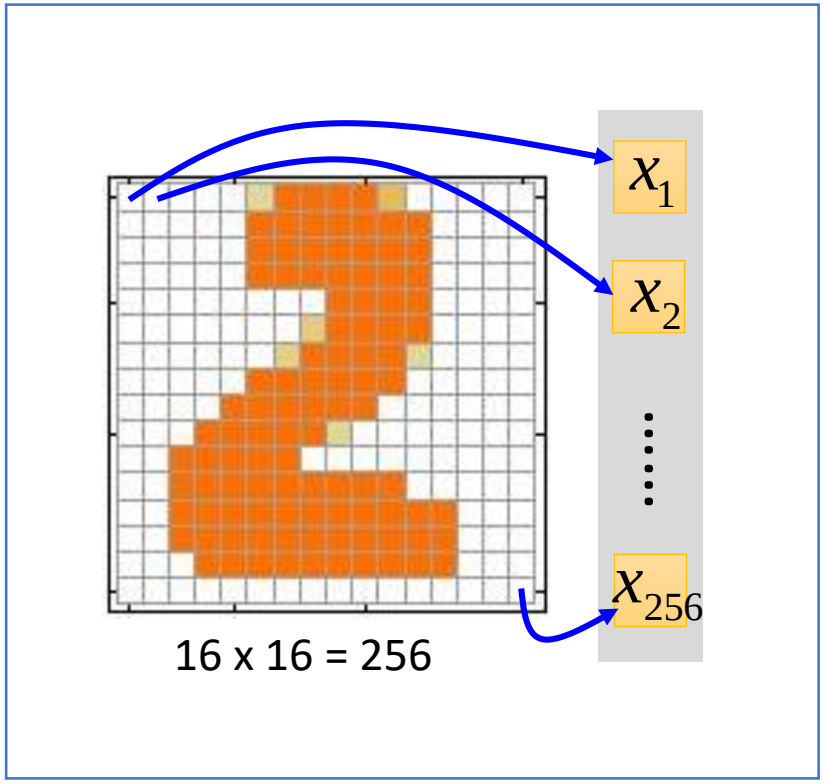
卷积神经网络CNN

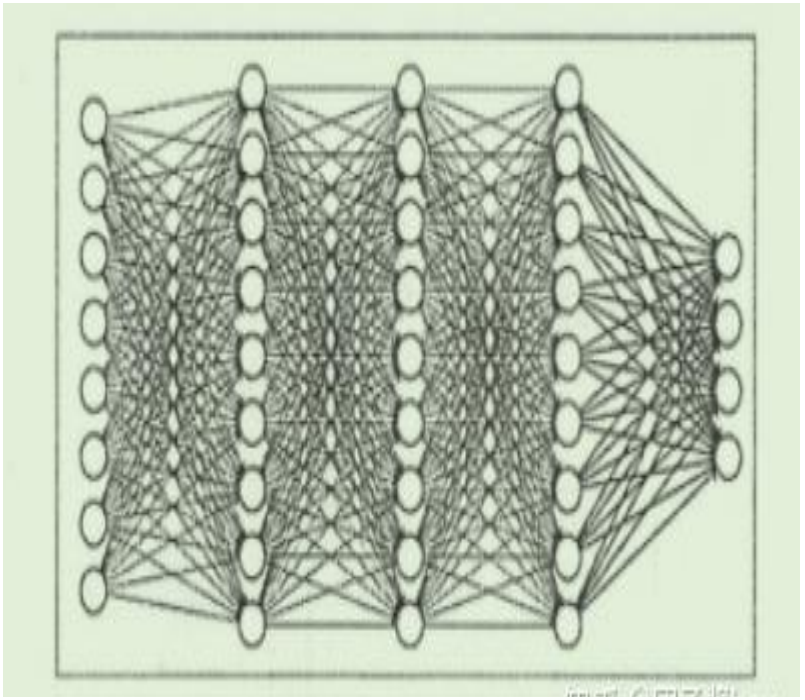
- 对输入进行卷积操作
- 主要应用于图像的处理



卷积神经网络CNN的提出

CNN之前：这种向量没有保留图像中的子结构

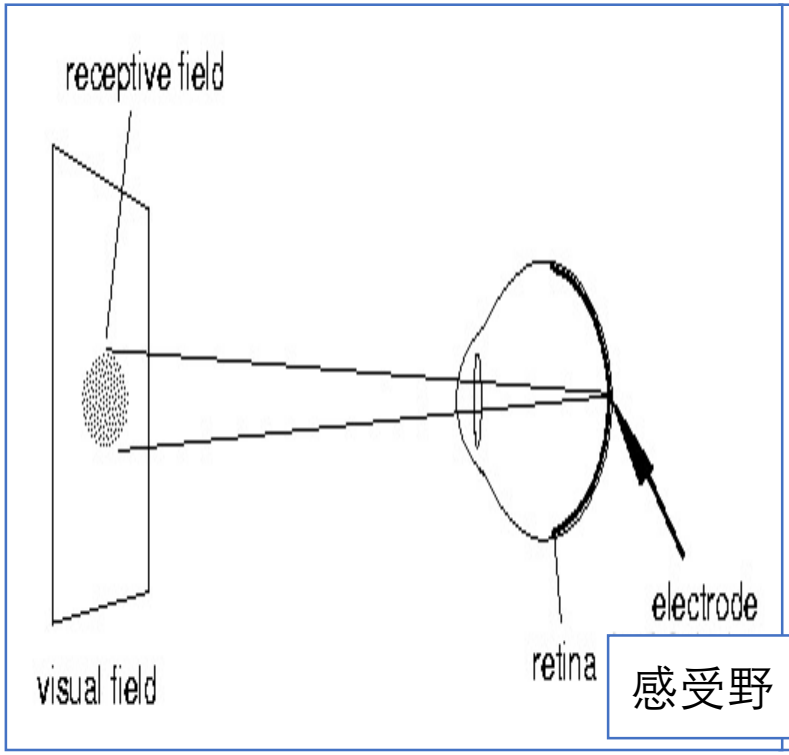




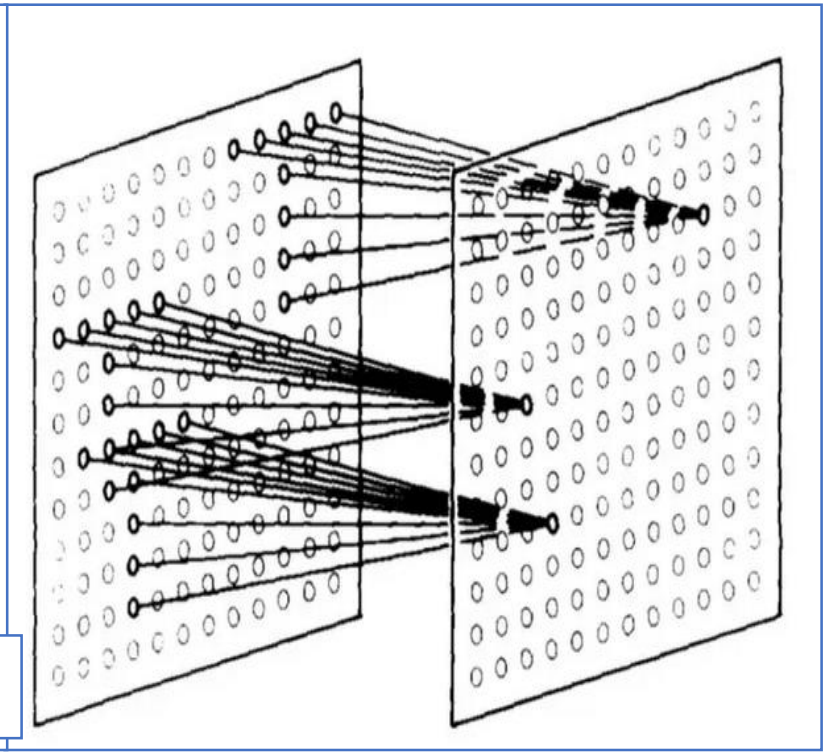
全连接前馈神经网络

卷积神经网络CNN的提出

视觉分层理论：初级(特征提取)-中级(目标识别)-高级(三位重构)



感受野



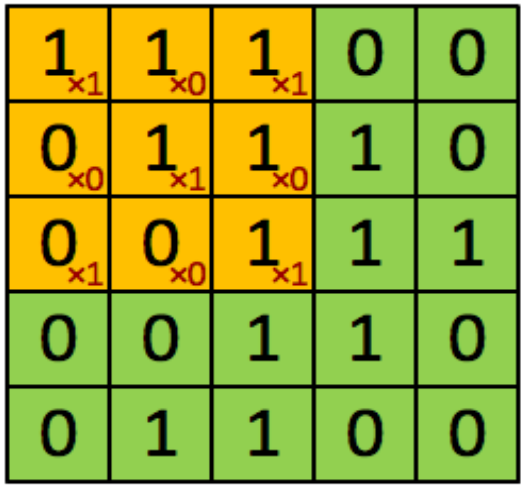
卷积操作（Convolution）

- 卷积核：

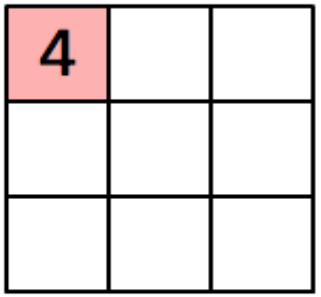
1	0	1
0	1	0
1	0	1

二维卷积核可利用二维结构
适用于处理图像矩阵

- 操作过程：
  - 点积求和



Image



Convolved Feature

理解卷积操作

- 卷积可以对图像模式进行提取
- 例子：找出图像中的垂直边缘

10	10	0	0
10	10	0	0
10	10	0	0
10	10	0	0

*

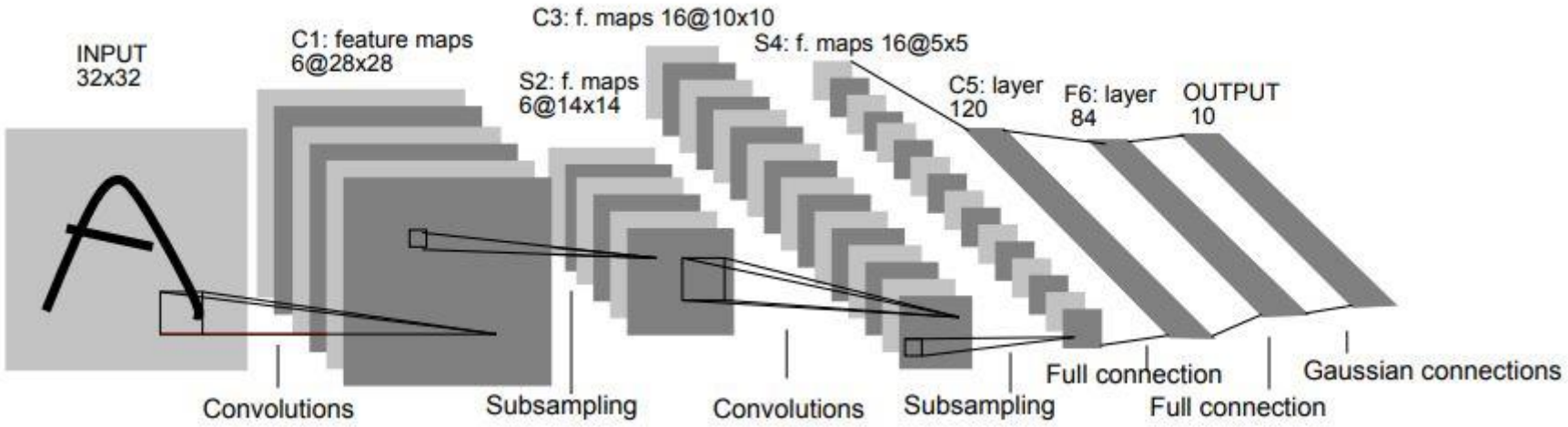
1	-1
1	-1

=

0	20	0
0	20	0
0	20	0

深度CNN网络

- 示例：LeNet-5



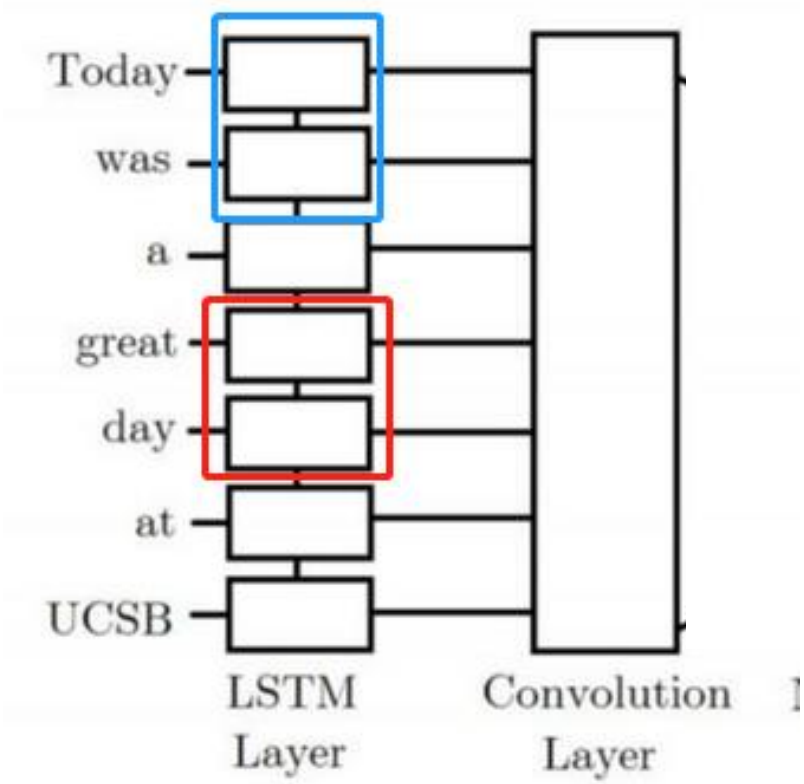
- 深度CNN网络一般由多层卷积层堆叠，卷积层之间插入池化层，最后由全连接层完成分类任务

主要内容

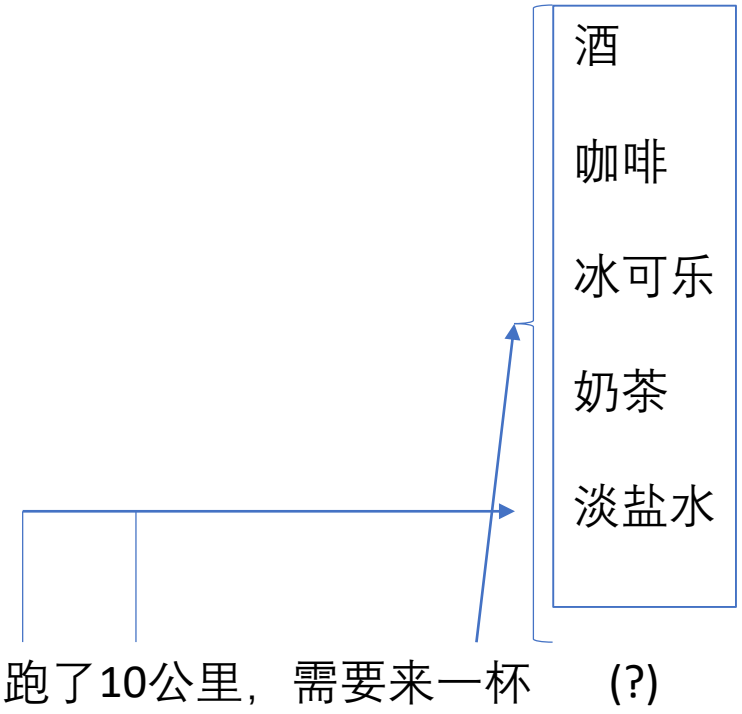
- 卷积神经网络（Convolution Neural Network，CNN）
- **循环神经网络（Recurrent Neural Network，RNN）**
  - 长短期记忆网络（Long-short Term Memory，LSTM）
  - 门控循环单元（Gate Recurrent Unit，GRU）
- 注意力机制（Attention）
- Transformer

循环神经网络RNN的提出

CNN对句子卷积：==n-gram?

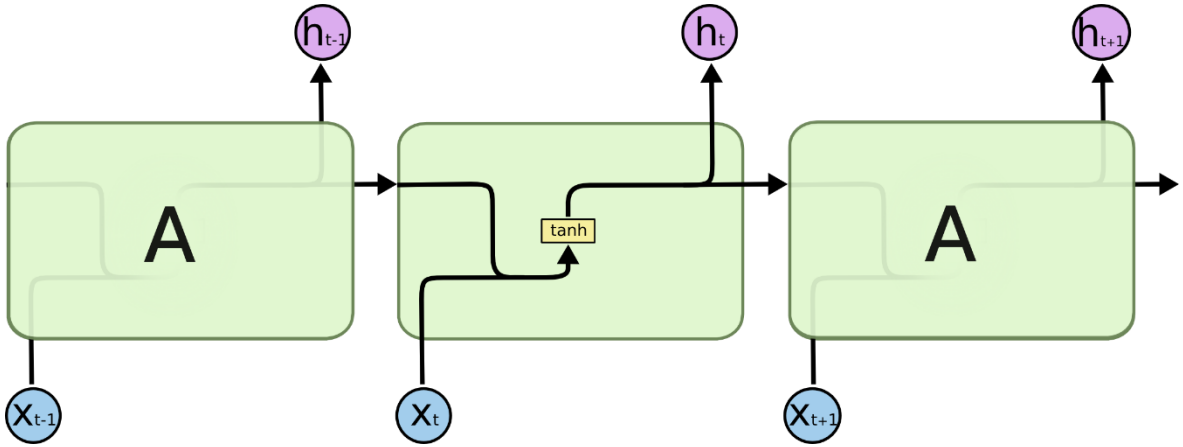


需要比N元文法更好的语言模型



RNN

- 循环神经网络（Recurrent Neural Network）

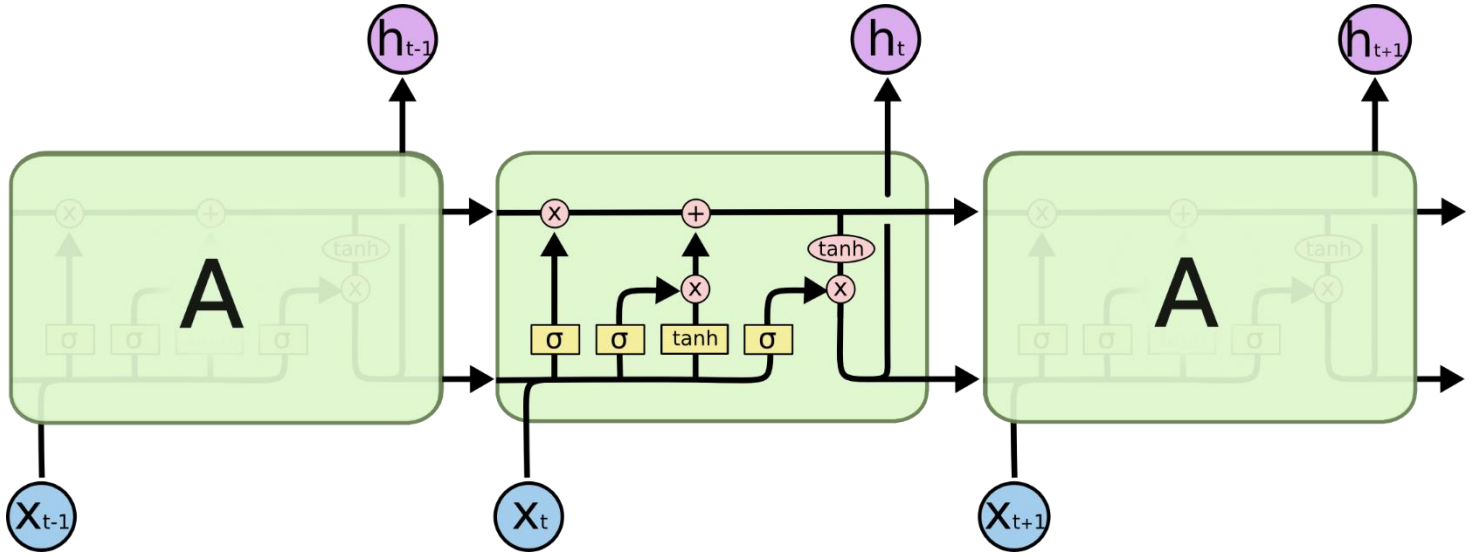


$$h_t = \tanh(Wh_{t-1} + Ux_t + b)$$

- 适合处理序列数据：保留了全部历史信息

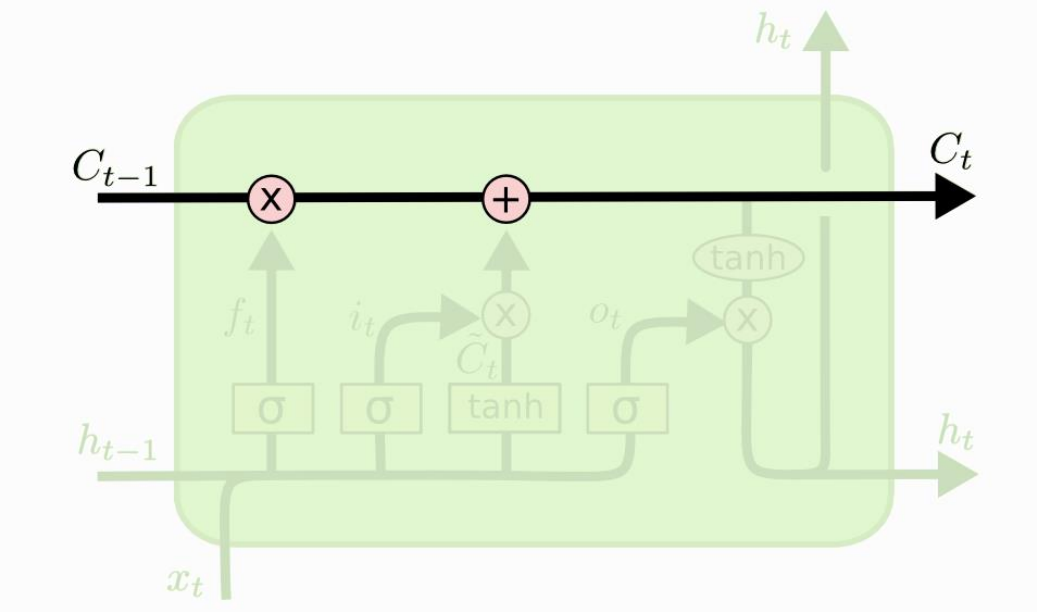
LSTM

- 不同的历史信息对当前信息的影响不一
- 长短期记忆网络（Long Short-Term Memory）



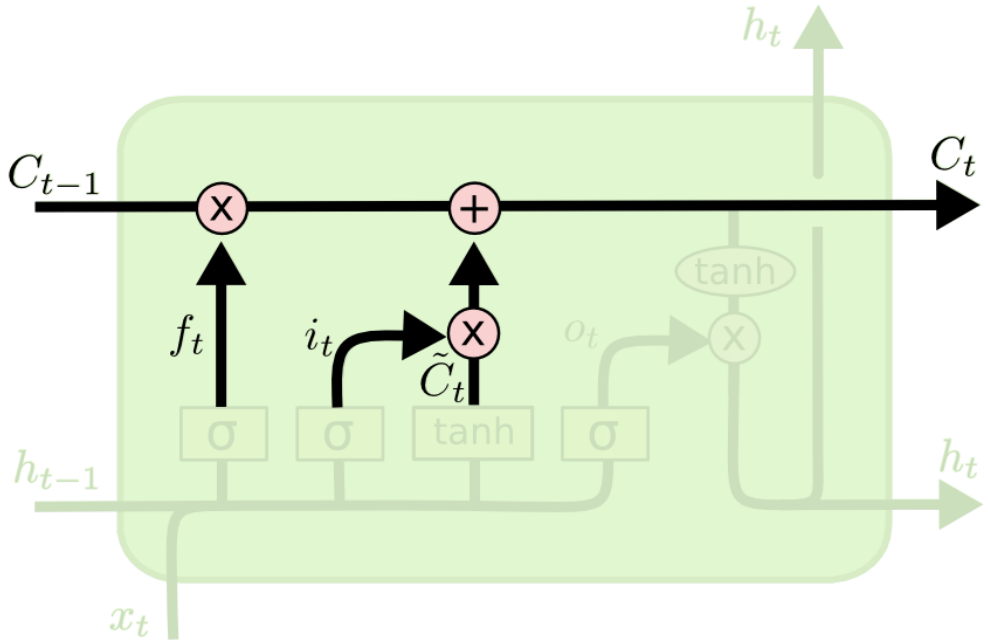
- 记忆单元 + 门控机制

LSTM：记忆单元



- 用向量C维持记忆，每一步通过加法更新，其内容由门控机制控制

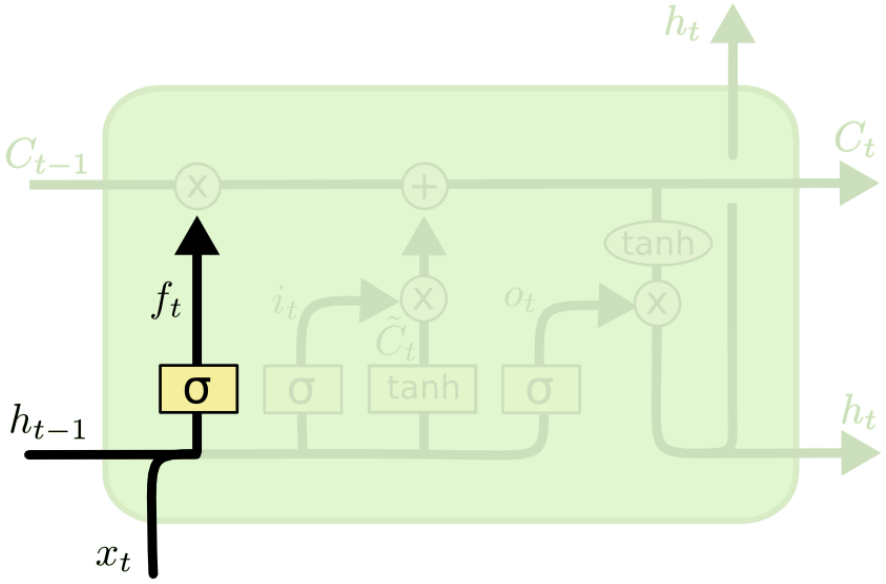
LSTM：记忆单元更新



- f_t：遗忘门数值越大，长期记忆保存越多
- i_t：输入门数值越大，短期记忆影响越大
- 更新后的记忆状态C_t是长期记忆C_{t-1}和短期记忆$\tilde{C}_t$的加权和

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

LSTM：遗忘门

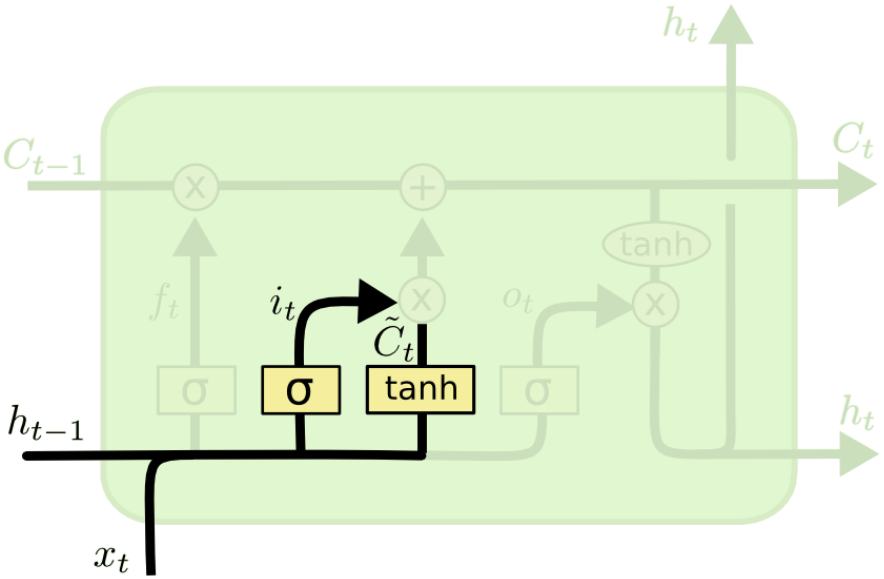


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

- 控制上一时刻记忆信息的保存

LSTM：输入门

- 生成当前时刻输入的记忆信息
- 控制输入信息对记忆单元的影响

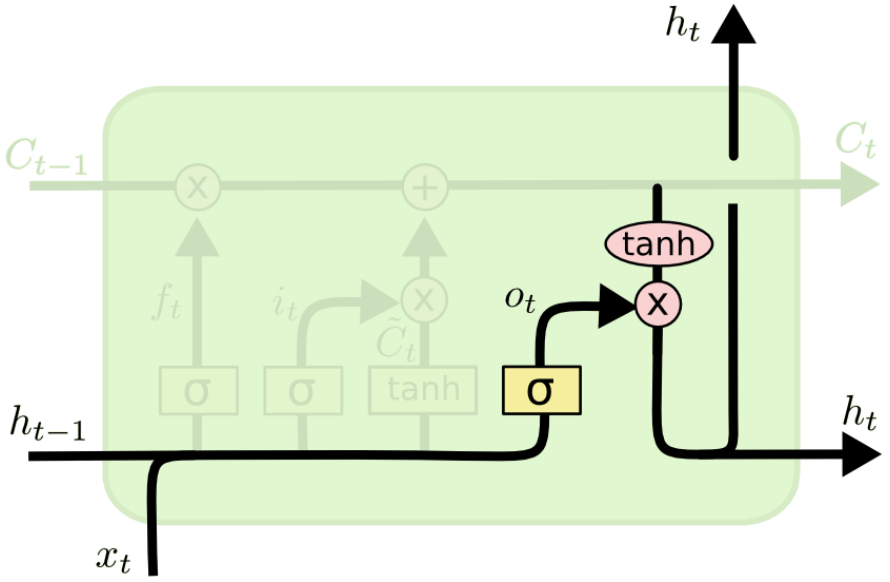


$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

LSTM：输出门

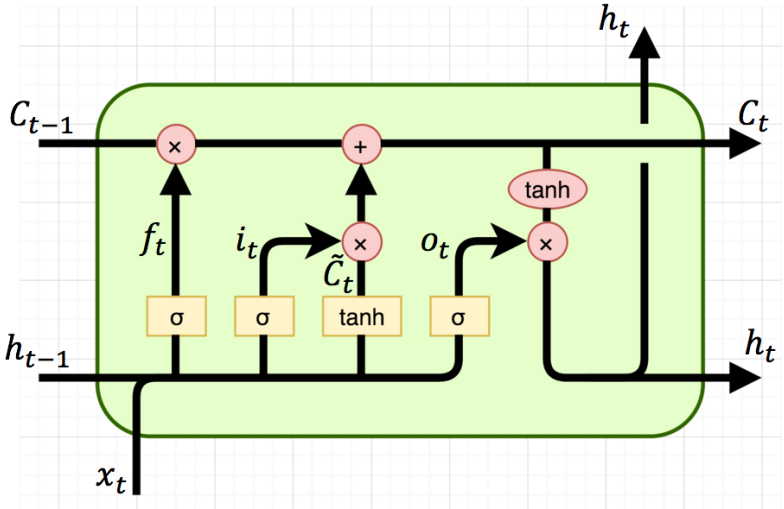
- 控制模型在某一时刻的最终输出
- 当o_t较大时，输出向量h_t也较大



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

$$h_t = o_t * \tanh\left(C_t\right)$$

LSTM：完整前向过程



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

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

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

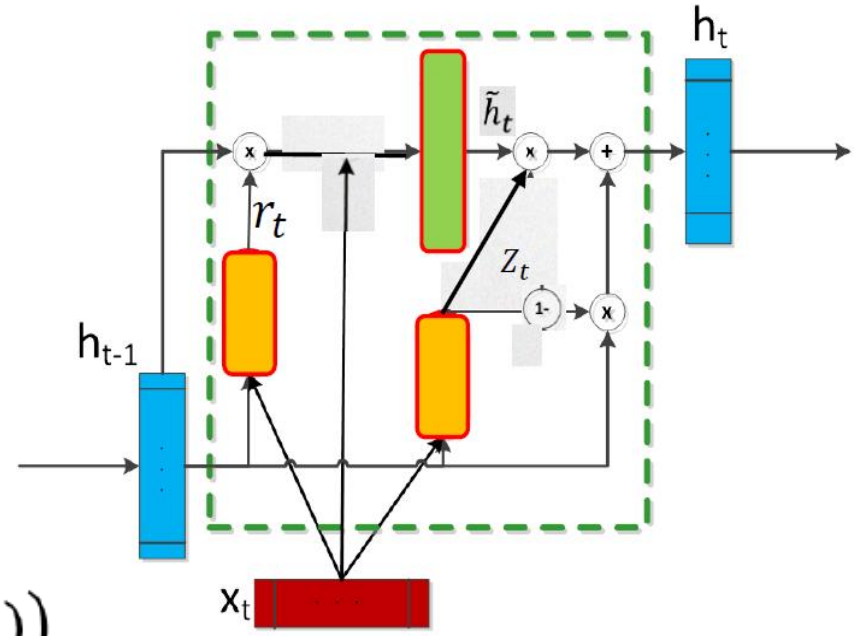
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

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

$$h_t = o_t * \tanh\left(C_t\right)$$

门控循环单元（GRU）

- LSTM的简化版
  - 去掉显式的记忆单元
- 设计了
  - 重置门(reset gate)
  - 更新门(update gate)



重置门：$r_t = \sigma(W_r x_t + U_r h_{t-1})$

暂存器：$\tilde{h}_t = \tanh\left(W x_t + U(r_t \circ h_{t-1})\right)$

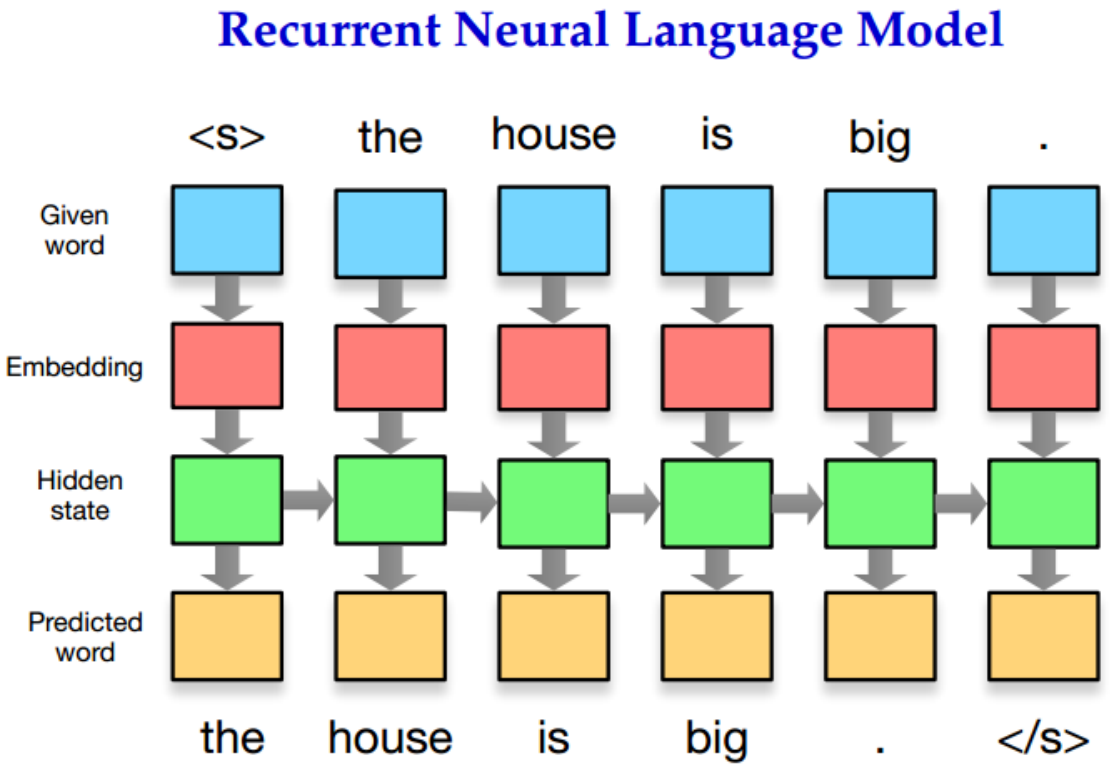
更新门：$z_t = \sigma(W_z x_t + U_z h_{t-1})$

隐状态：$h_t = (1 - z_t) \circ h_{t-1} + z_t \circ \tilde{h}_t$

Cho, Kyunghyun, Bart van Merrienboer, Dzmitry Bahdanau and Yoshua Bengio. “On the Properties of Neural Machine Translation: Encoder-Decoder Approaches.” *SSST@EMNLP* (2014).

RNN的典型应用：神经语言模型

- 语言模型：给定一个单词序列之后，预测下一个词产生的概率

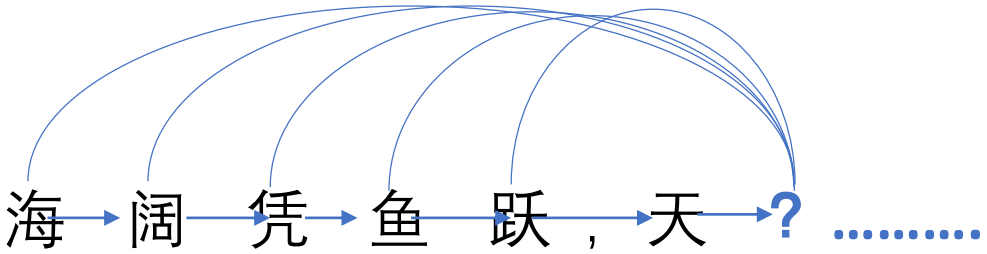


主要内容

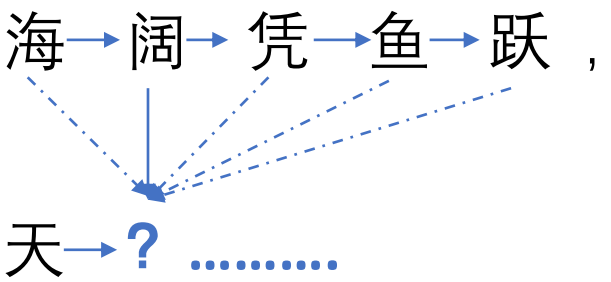
- 卷积神经网络（Convolution Neural Network，CNN）
- 循环神经网络（Recurrent Neural Network，RNN）
  - 长短期记忆网络（Long-short Term Memory，LSTM）
  - 门控循环单元（Gate Recurrent Unit，GRU）
- **注意力机制（Attention）**
- Transformer

注意力机制： Attention

RNN处理自然语言的一个问题



实际上的影响是



所能获得的山下文信息对当前的影响作用不一

引入注意力这一概念

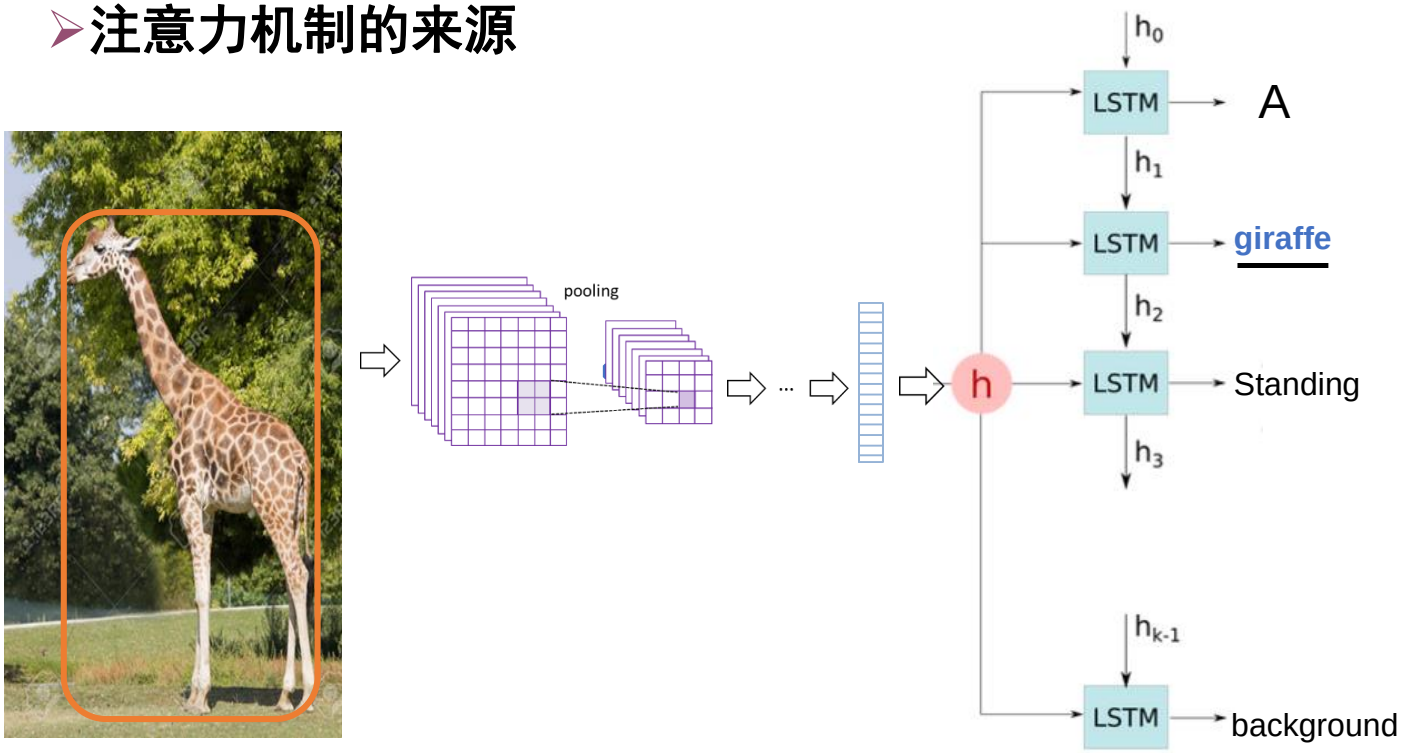
注意力机制：Attention

- 注意力机制：**对所摄入的场景的局部具有强烈的注意，而对其周围的场景弱化的机制。**

- 这一概念源于神经科学（Neuroscience ）和计算神经科学（ Computational Neuroscience ）领域
- 研究大多集中于视觉注意力（ Visual Attention ）：**动物或者人对所看到的场景做出反应时，都会集中于所看到场景的局部区域。**

注意力机制： Attention

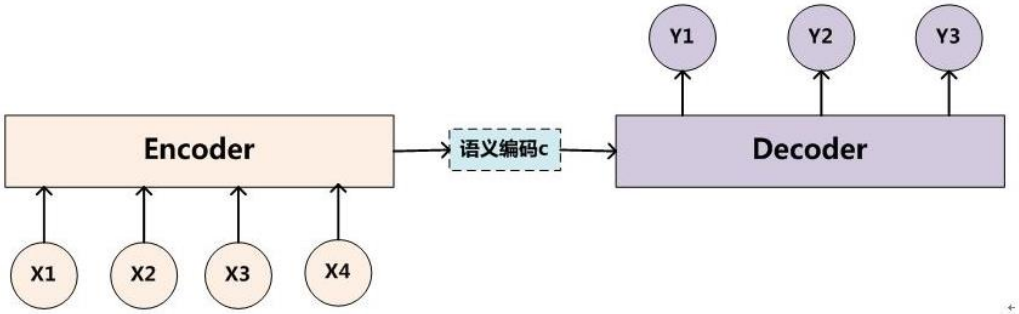
- **注意力机制的来源**



A **giraffe** standing in a forest with trees in the background.

注意力机制： Attention

- **对联生成为例（Encoder-Decoder框架）**



例如“天高任鸟飞”的这句下联的生成：上联“海阔凭鱼跃”里每个字对 “高”可能都有影响，而根据对联成文的习惯显然“阔”对于 “高”中起了做大的作用。

引入Attention机制，应该在生成 “高”的时候，体现出上联中的字对于“高”的不同的影响程度，比如给出一个下列概率分布值：

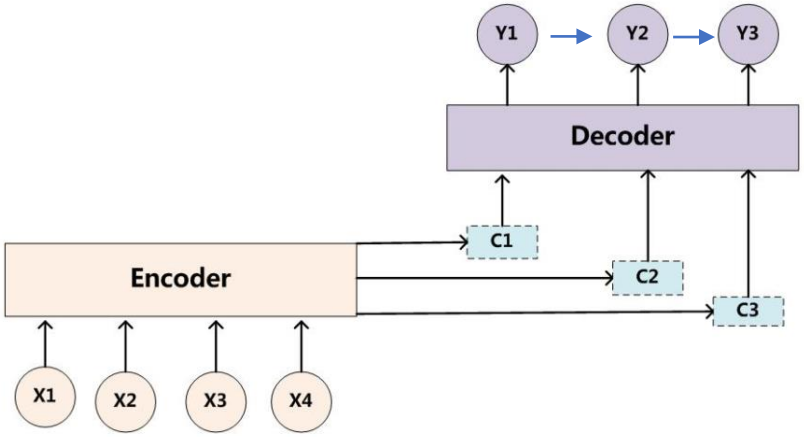
(海,0.2) (阔,0.5) (凭,0.1) (鱼,0.1) (跃,0.1)

每个括号里的概率代表了生成前“高”时，注意力分配模型分配给不同上联单字的的注意力大小。

注意力机制： Attention

- **对联生成为例（Encoder-Decoder框架）**

引入上述概率分布值对正确生成下联中的每个字显然是有帮助的。
相应的，下联中的每个字都需要其对应的上联中每个字的注意力分配概率信息。
这意味着在生成每个下联字y_i的时候，原先都是相同的语义中间表示C将会被替换成根据当下生成单词而不断变化的c_i。这个c_i就是C经过了Attention机制计算后的结果。



生成目标句子单词的过程变成了如下形式：

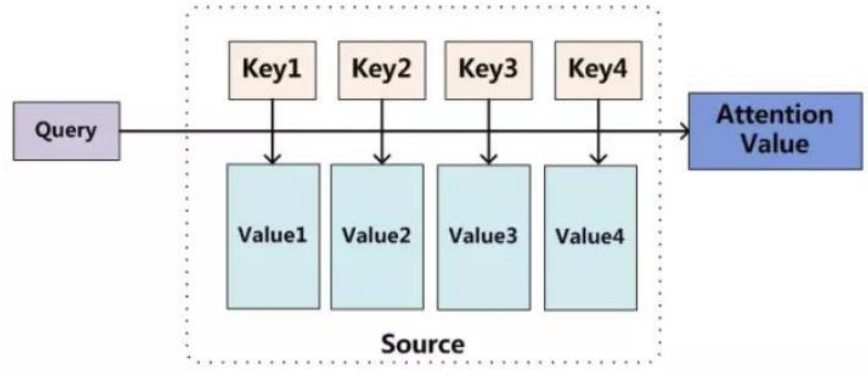
$$y_1 = f_1(c_1)$$
$$y_2 = f_2(c_2, y_1)$$
$$y_3 = f_3(c_3, y_1, y_2)$$

如：C(高)=g(0.2*f(海).0.5*(阔)0.1*(凭). 0.1*(鱼). 0.1*(跃))

注意力机制：Attention

➢ **Attention计算的另一种理解**

可以用上述图来抽象表示：将上联想象成由一系列数据对Key构成，此时给定下联中的某个元素Query，通过计算Query和各个Key间的相似性或者相关性，得到每个Key对应Value的权重系数，然后对Value进行加权求和，即得到了最终的Attention Value。



根据Query和某个Key值，计算两者的相似性或相关性，最常见的方法包括：

1. 求两者的向量点积$Similarity(Query, Key_i) = Query \cdot Key_i$
2. 求两者的向量cosine相似性
   $Similarity(Query, Key_i)=\frac{Query \cdot Key_i}{\|Query\| \cdot \|Key_i\|}$
3. 引入额外的神经网络求值
   $Similarity(Query, Key_i)=MLP(Query, Key_i)$

注意力机制： Attention

➢attention常见的分类方式

1.在attention score(匹配度或权值)的计算方式上变种：

① 点积attention score(Basic dot-product attention)
② 乘积attention score(Multiplicative attention)
③ 加法attention score(Addictive attention)

2.在attention向量加权求和计算方式上的变种：

① Soft attention与Hard attention
② Local attention与Global attention
③ 静态attention与动态attention

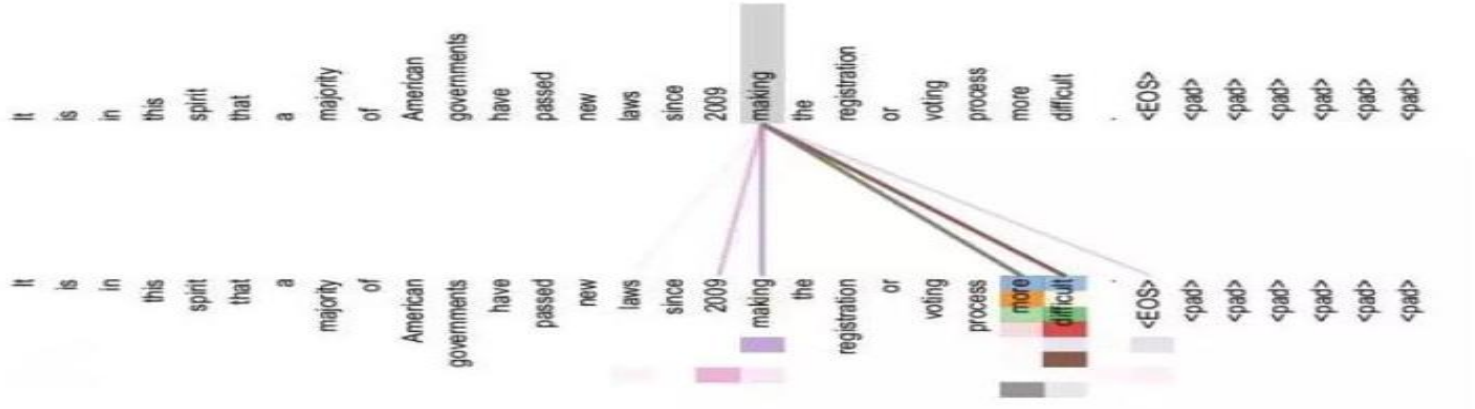
3.更加特殊的attention(常见于Transformer)

① Self attention
② Multi-head attention
③ Key-value attention

注意力机制： Attention

- self-attention

对于英-中机器翻译来说，Source是英文句子，Target是对应的翻译出的中文句子，Attention机制发生在Target的元素和source中的所有元素之间。
Self-Attention也叫intra-attention，指的不是发生在Target和source之间的Attention机制，而是指单语内部元素之间的Attention机制，也可以理解为target=source这种特殊情况下的注意力机制。



注意力机制： Attention

- self-attention**优点**

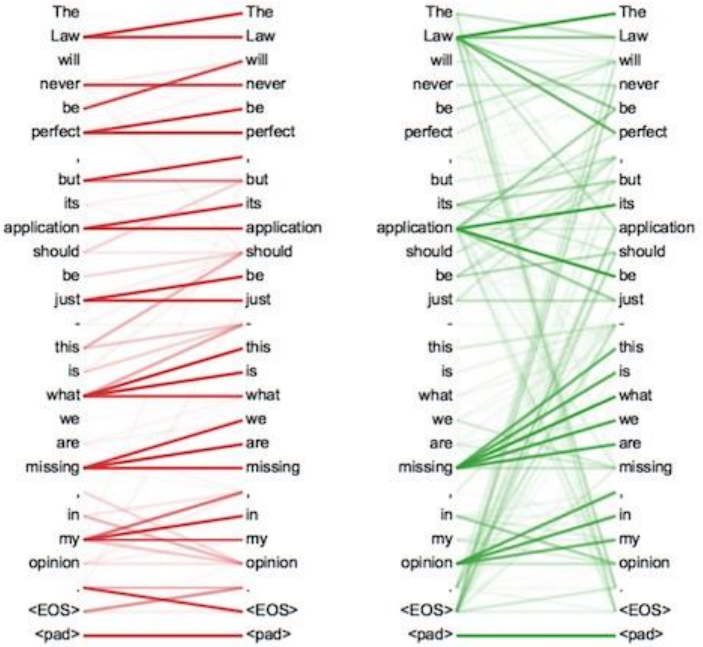
显然，引入self-attention之后会更容易捕捉句子中长距离的相互依赖的特征，因为如果是RNN或者LSTM，需要依次序序列计算，对于远距离的相互依赖的特征，要经过若干时间步步骤的信息累积才能将两者联系起来，而且距离越远，有效捕捉的可能性越小。
但self-attention在计算过程中会直接将句子中任意两个单词的联系通过一个计算步骤直接联系起来，所以远距离依赖特征之间的距离被极大缩短，有利于有效的利用这些特征。
除此之外， self-attention对于增加计算的并行性也有直接帮助作用，这也是为何self-attention逐渐被广泛使用的原因。

注意力机制：Attention

- **Multi-head attention**

Multi head attention(多头注意力机制)可视作多个self-attention的结合，每个head学习到在不同表示空间中的特征，如图所示，两个head学习到的attention侧重点可能不同，这给了模型更大的容量。

Multi head attention计算过程与self-attention基本相同，只在开始和结尾多出了切分和合成部分。



主要内容

- 卷积神经网络（Convolution Neural Network，CNN）
- 循环神经网络（Recurrent Neural Network，RNN）
  - 长短期记忆网络（Long-short Term Memory，LSTM）
  - 门控循环单元（Gate Recurrent Unit，GRU）
- 注意力机制（Attention）
- **Transformer**