法律声明

- 本课件包括：演示文稿，示例，代码，题库，视频和声音等，小象学院拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意，我们将保留一切通过法律手段追究违反者的权利。

关注 小象学院

第13讲 图像描述（图说）

计算机视觉

主讲人：叶梓

上海交通大学博士
主要研究方向：机器学习、深度学习、人工智能

本章内容

- 深度语言模型介绍
- RNN介绍
- LSTM原理解析
- 图说模型原理与结构
- 大数据集介绍
- 应用实例：
  - RNN的简单案例
  - Show and Tell

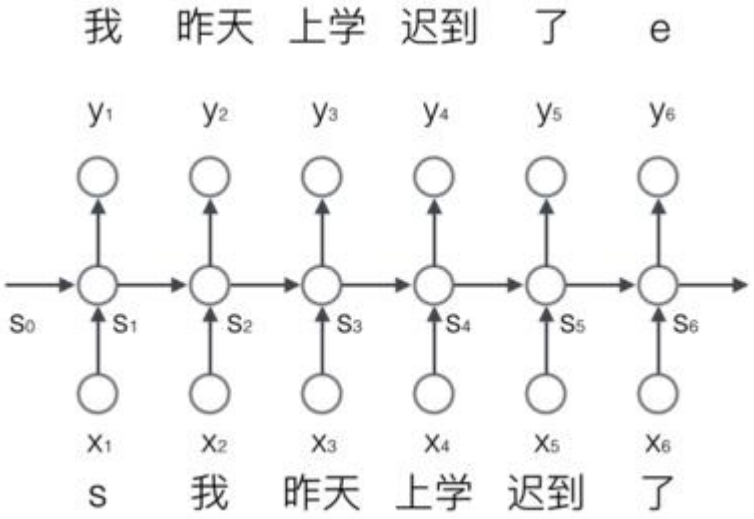
深度语言模型

递归神经网络RNN

- 有2类
  - 时间递归神经网络(Recurrent Neural Network)
    - 针对时间序列
  - 结构递归神经网络(Recursive Neural Network)
    - 针对树状结构
- 优化方法
  - 时序后向传播(Back propagation through time)
- 长时记忆/递归深度问题
  - 梯度爆炸(Gradient exploding) → 梯度剪切
  - 梯度消失(Gradient vanishing) → 特殊设计

RNN的应用

- 基于RNN语言模型。我们首先把词依次输入到循环神经网络中，每输入一个词，循环神经网络就输出截止到目前为止，下一个最可能的词。

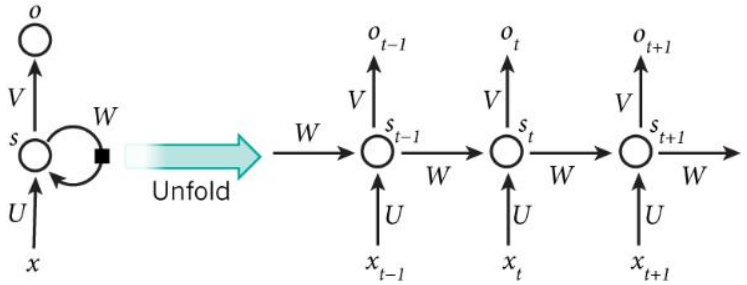


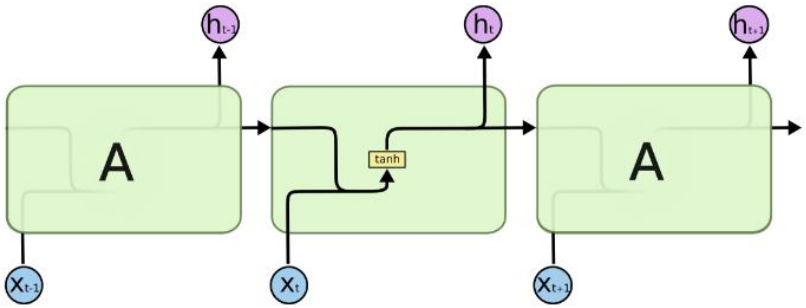
- **语言模型**是对下一个词出现的**概率**进行建模。怎样让神经网络输出概率呢？用softmax层作为神经网络的输出层。

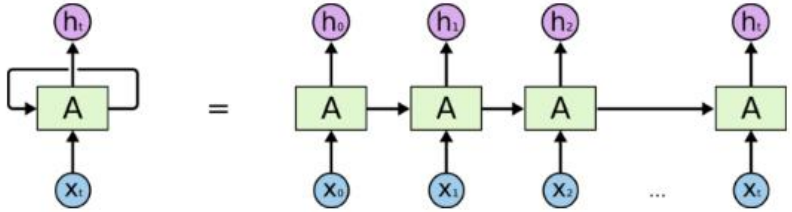
深度语言模型

朴素Vanilla-RNN

- 单层神经网络在时间上的扩展
- t-1时刻的隐层状态(Hidden state)会参与t时刻输出的计算
- 严重的梯度消失问题

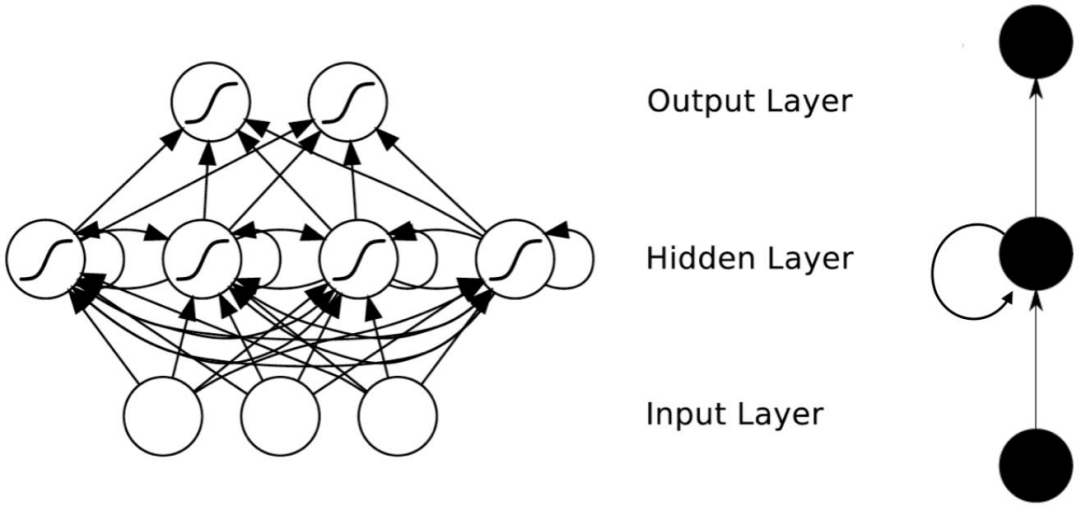






深度语言模型

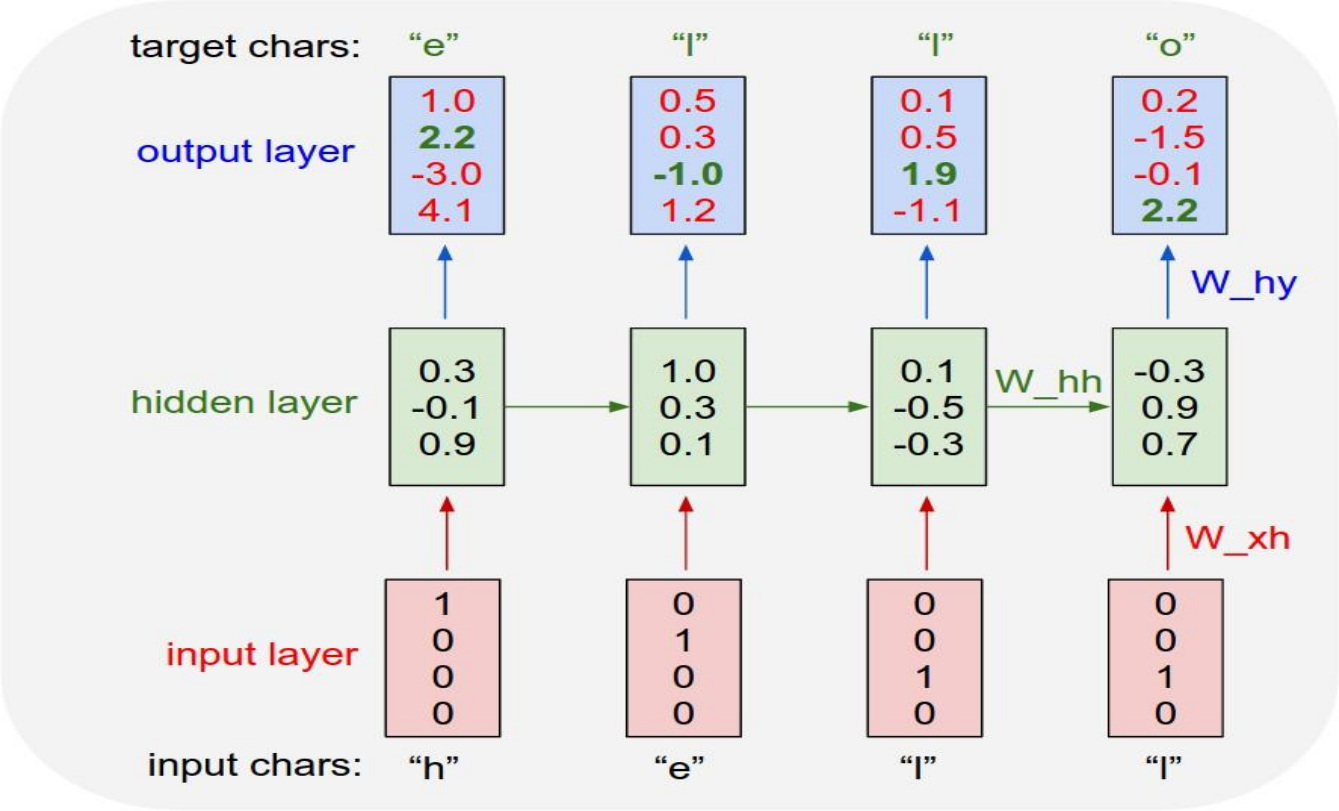
- 关键：Hidden Layer会有连向下一时间Hidden Layer的边，是全连接！



- 还有一种结构是Bidirectional Networks，也就是说会有来自下一时间的Hidden Layer传回来的边。

RNN的示例

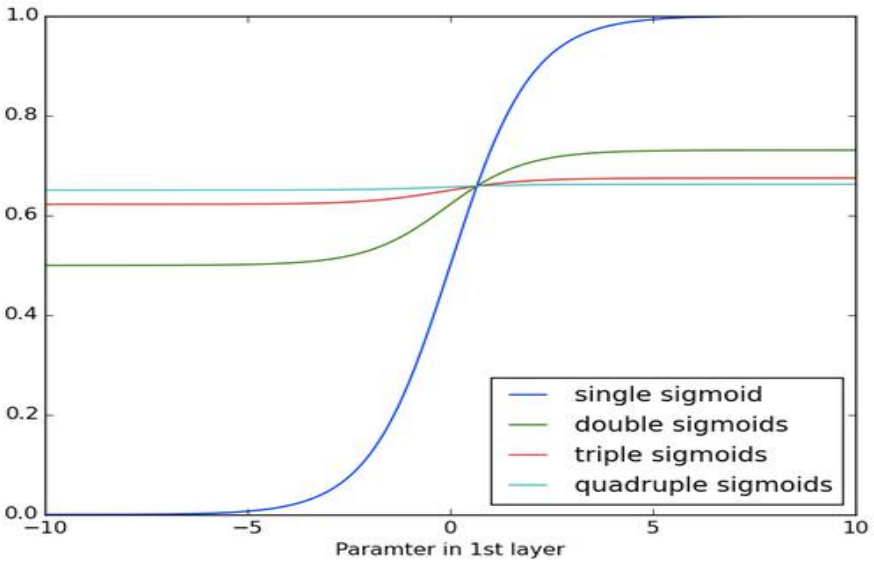
- 希望output层中绿色的数字足够大，而红色的数字足够小；
- 这样经过softmax后，才能输出正确的内容。



深度语言模型

时序后向传播（BPTT）

- 传统后向传播(BP)在时间序列上的扩展
- t时刻的梯度是前t-1时刻所有梯度的累积
- 时间越长，梯度消失越严重



BPTT

- BPTT算法是针对**循环层**的训练算法，它的基本原理和BP算法是一样的，也包含同样的三个步骤：
  - 前向计算每个神经元的输出值；
  - 反向计算每个神经元的**误差项**值δ_j，它是误差函数E对神经元j的**加权输入**的偏导数net_j；
  - 计算每个权重的梯度。
- 最后再用**随机梯度下降**算法更新权重。

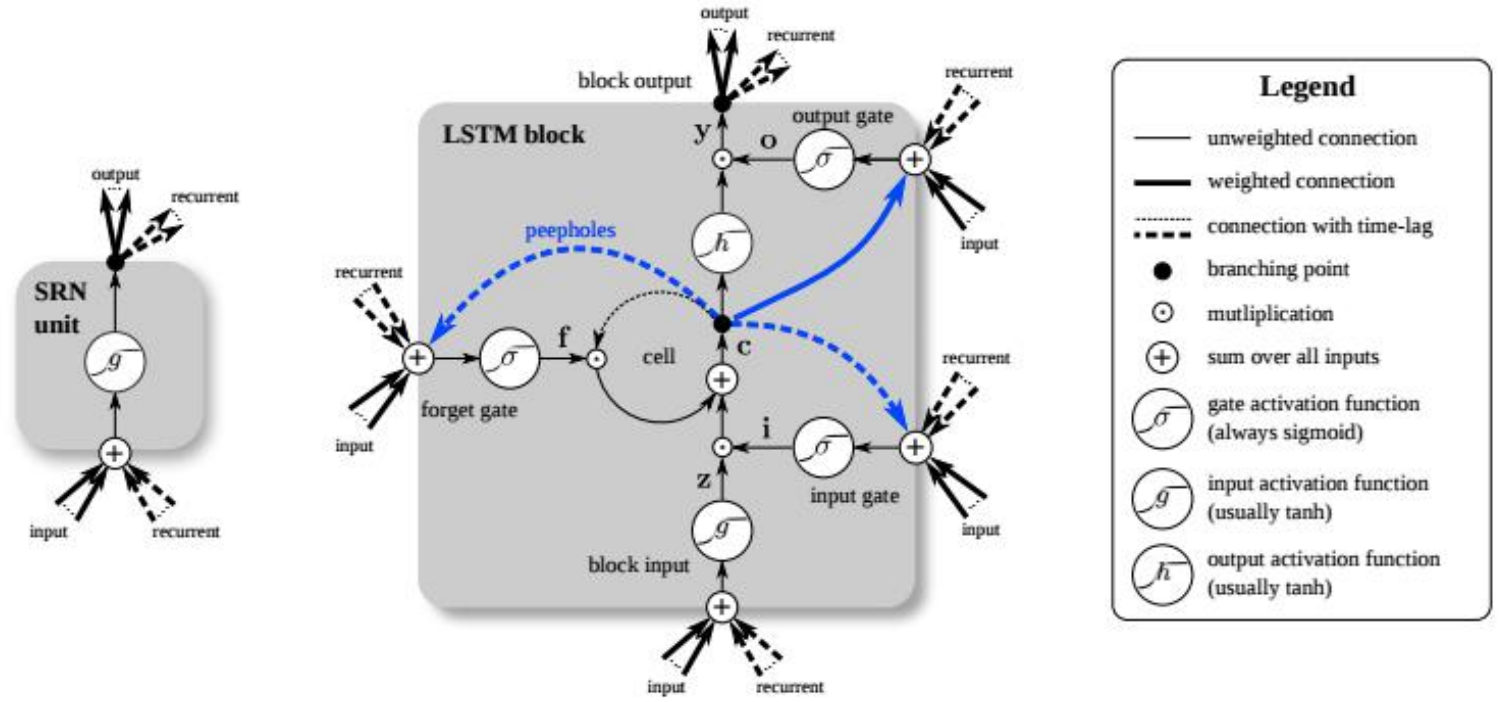
深度语言模型

LSTM长短时记忆模型(Long Short-Term Memory)

- Hochreiter & Schmidhuber 于1997提出
- 有效捕捉长时记忆(Long dependency)
- 包含4个神经元组
  - 1个记忆神经元(Memory cell)
  - 3个控制门神经元
    - 输入门(Input gate)
    - 忘记门(Forget gate)
    - 输出门(Output gate)

深度语言模型

Vanilla-RNN vs LSTM

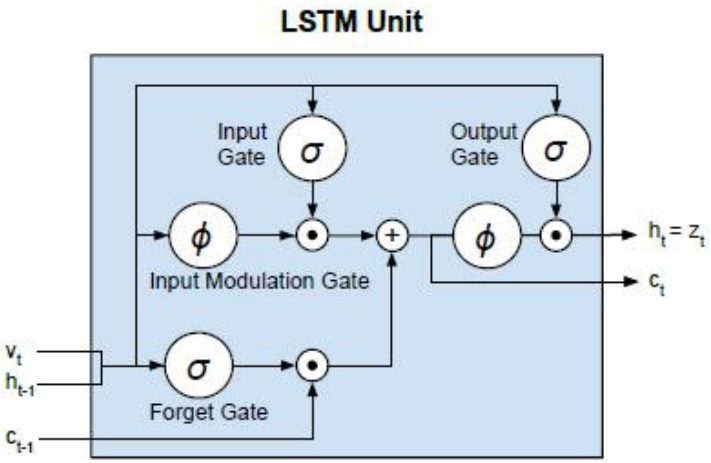


Vanilla-RNN LSTM

深度语言模型

LSTM数学模型

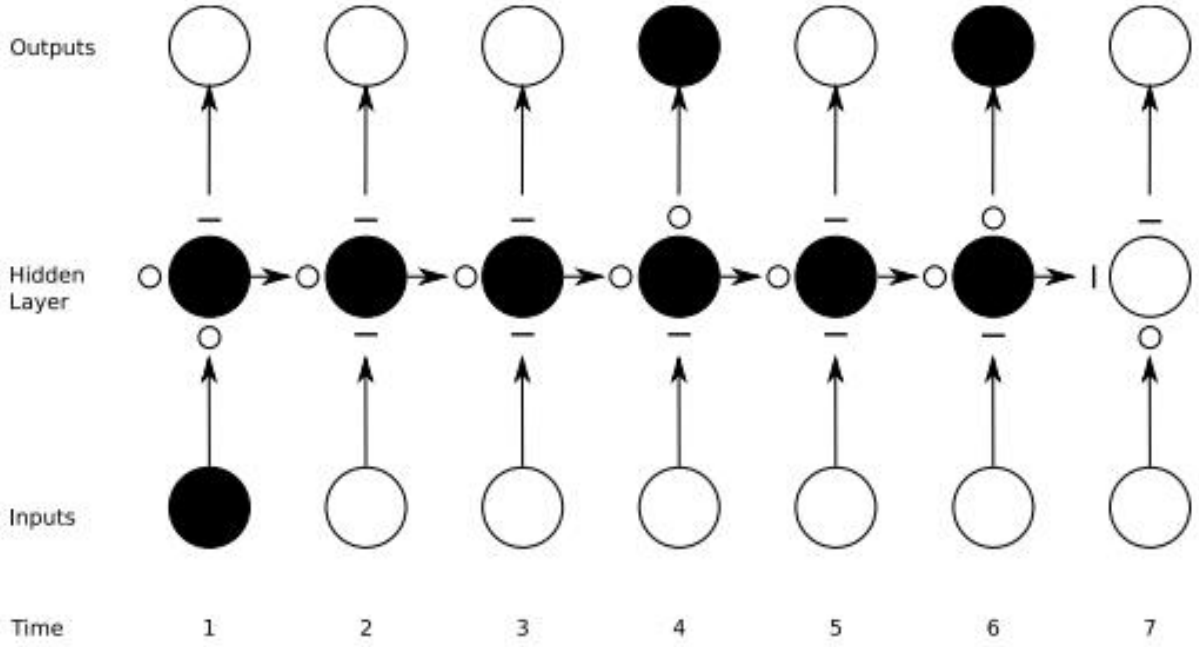
- 3个输入
  - 前一时刻的隐含状态h_{t-1}
  - 前一时刻的记忆状态c_{t-1}
  - 当前时刻的输入v_t
- 2个输出
  - 当前时刻的隐含状态h_t
  - 当前时刻的记忆状态c_t



$$
\begin{aligned}
\boldsymbol{i}_t &= \sigma(W_{vi}\boldsymbol{v}_t + W_{hi}\boldsymbol{h}_{t-1} + \boldsymbol{b}_i) \\
\boldsymbol{f}_t &= \sigma(W_{vf}\boldsymbol{v}_t + W_{hf}\boldsymbol{h}_{t-1} + \boldsymbol{b}_f) \\
\boldsymbol{o}_t &= \sigma(W_{vo}\boldsymbol{v}_t + W_{ho}\boldsymbol{h}_{t-1} + \boldsymbol{b}_o) \\
\boldsymbol{g}_t &= \phi(W_{vg}\boldsymbol{v}_t + W_{hg}\boldsymbol{h}_{t-1} + \boldsymbol{b}_g) \\
\boldsymbol{c}_t &= \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \boldsymbol{g}_t \\
\boldsymbol{h}_t &= \boldsymbol{o}_t \odot \phi(\boldsymbol{c}_t)
\end{aligned}
$$

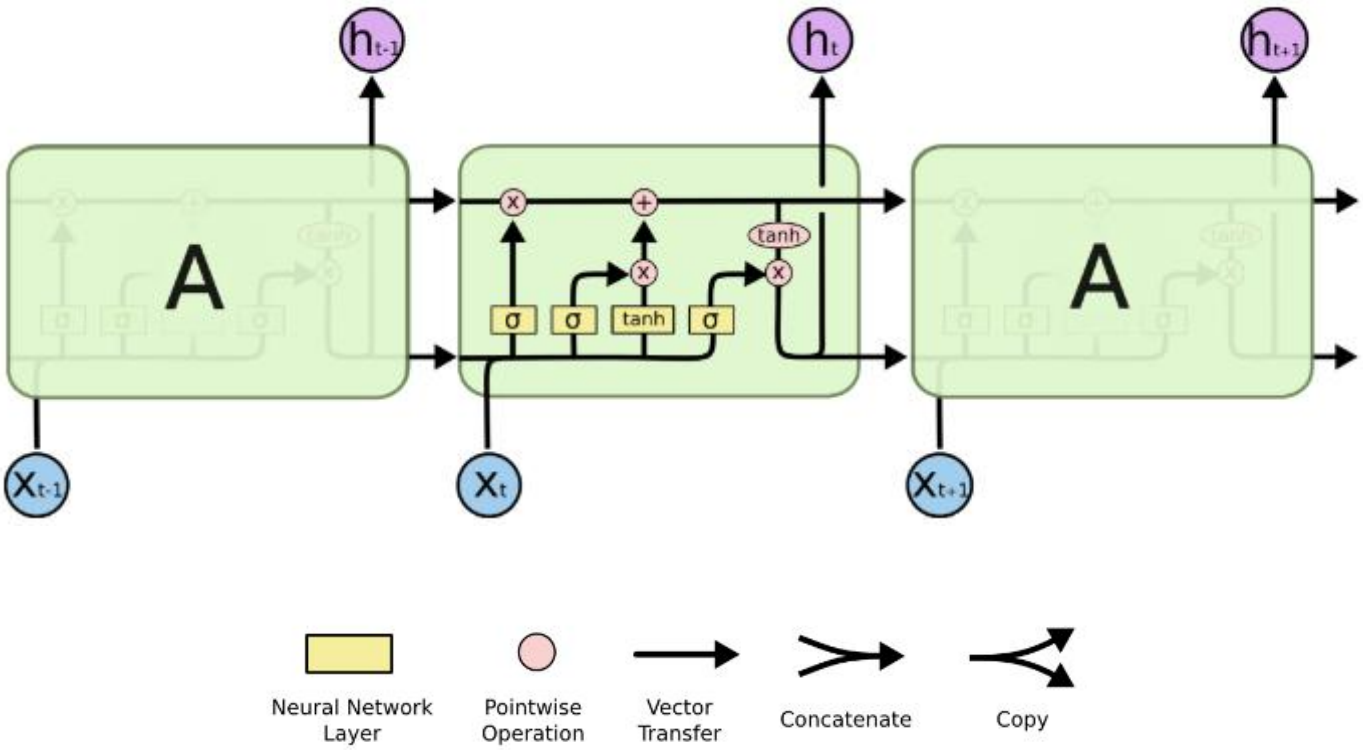
深度语言模型

LSTM控制门作用



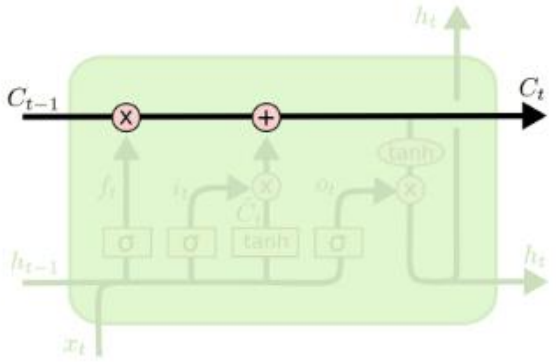
深度语言模型

LSTM结构图

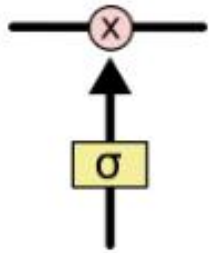


深度语言模型

LSTM结构图



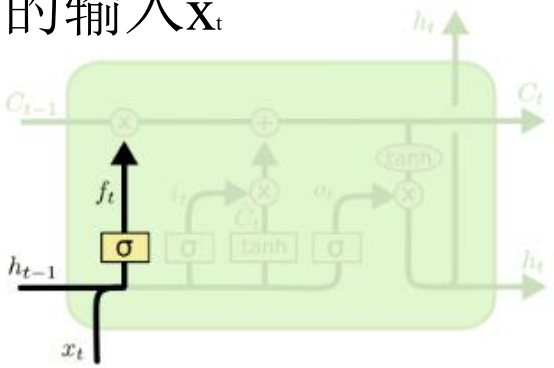
- 记忆状态(cell state) → 信息
  - 存储之前时刻的信息
  - 避免长时记忆问题的核心
- 控制门(gate) → 选择性控制信息流入
  - 由元素乘操作实现
  - 配有sigmoid激活函数的神经层
  - 值域[0,1]
    - 0 不通过任何信息
    - 1 通过所有信息



深度语言模型

LSTM结构图

- 忘记门
  - 决定前一时刻中多少记忆状态被移除
  - Sigmoid激活
  - 2个输入
    - 前一时刻的隐含状态h_{t-1}
    - 当前时刻的输入x_t

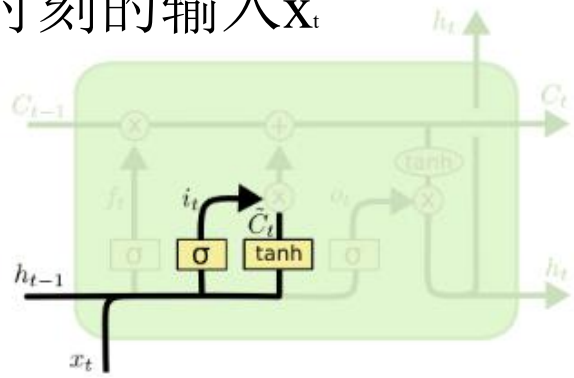


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

深度语言模型

LSTM结构图

- 输入门
  - 决定当前时刻有多少新输入信息需要存入记忆状态
  - Sigmoid激活
  - 2个输入
    - 前一时刻的隐含状态h_{t-1}
    - 当前时刻的输入x_t



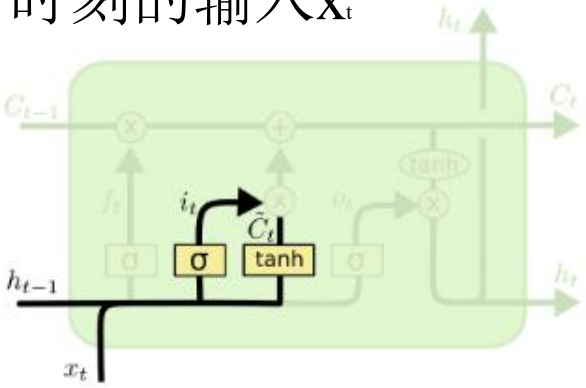
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right)$$

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

深度语言模型

LSTM结构图

- 输入调制
  - 决定当前时刻有多少新输入信息需要存入记忆状态
  - Tanh激活
  - 2个输入
    - 前一时刻的隐含状态h_{t-1}
    - 当前时刻的输入x_t



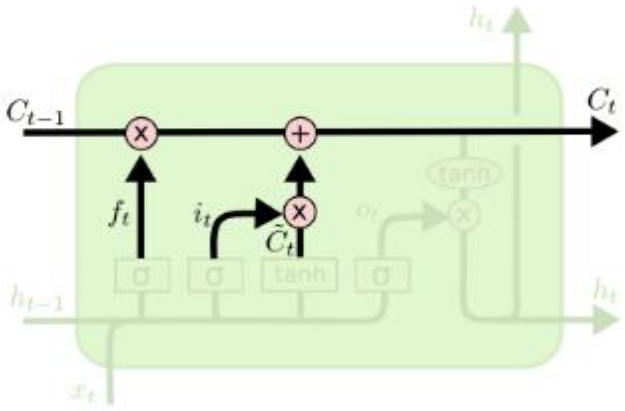
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

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

深度语言模型

LSTM结构图

- 记忆状态更新
  - 选择性移除前一时刻态的旧信息（记忆状态）
  - 选择性添加当前时刻的新信息（调制输入）



选择性移除

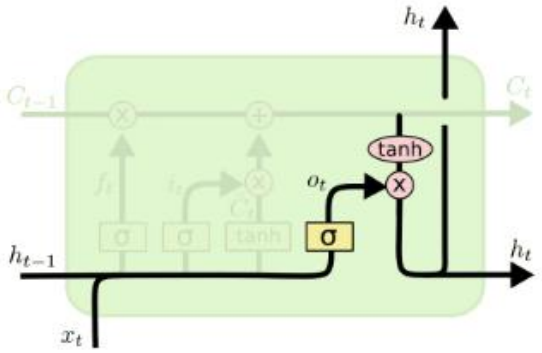
$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

选择性添加

深度语言模型

LSTM结构图

- 输出门
  - 决定当前时刻多少记忆状态用于输出
  - 2个输入
    - 前一时刻的隐含状态h_{t-1}
    - 当前时刻的输入x_t
  - 2个激活
    - Tanh激活
      - 压缩记忆状态[-1,1]
    - Sigmoid激活



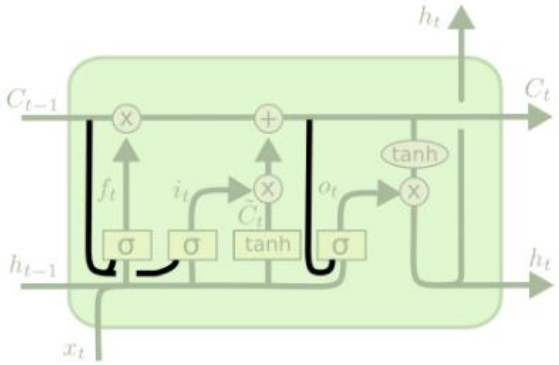
$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$

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

深度语言模型

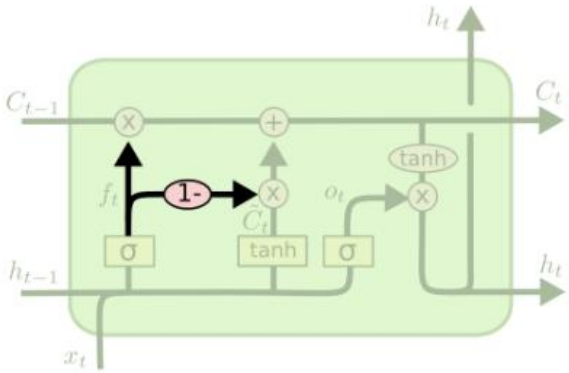
LSTM变种

- Peephole



$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] + b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] + b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] + b_o\right)$$

- Coupled 忘记-输入门

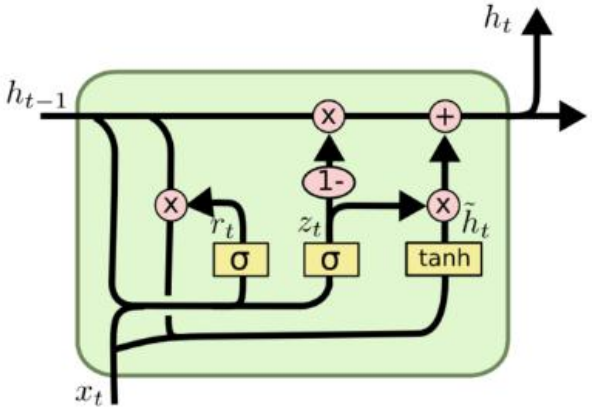


$$C_t = f_t * C_{t-1} + (1 - \boldsymbol{f_t}) * \tilde{C}_t$$

深度语言模型

GRU门限递归单元（Gated Recurrent Unit）

- 2个改动（Cho, et al., 2014）
  - 合并输入门和忘记门
  - 合并记忆状态和隐藏状态
- 2个控制门
  - 重置门(Reset gate)
  - 更新门(Update gate)
- 2个输入（与LSTM一致）
- 1个输出



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$

$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$

$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$

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

深度语言模型

LSTM vs GRU

- LSTM
  - 模型复杂，参数多，拟合能力强
  - 数据要求：大规模、复杂度高
- GRU
  - 模型精简，参数少，拟合能力相对弱
  - 适用于偏小规模、不是很复杂的数据

图说模型Image Captioning

为图片生成描述语言

- 输入：图片
- 输出：客观描述图片内容的句子



A person is holding a **gecko** in their hand.



A large **blimp** in a blue sky.



A black and white **skunk** is eating grass.



A plate of food with a fork and a **hollandaise**.

图说模型Image Captioning

一个视觉-语言研究问题

- 理解为一种特殊的机器翻译：视觉→语言
- 模型需要有复杂的场景理解能力
  - 图片理解→计算机视觉
  - 语言理解→自然语言处理
  - 复合、多模态理解→多媒体(Multi-Media)

图说模型Image Captioning

研究难点与挑战

- 多模态理解与推理
  - 图片：捕捉真实世界的原始刻画
  - 自然语言：代表更高一级的抽象
- 复合理解与推理
  - 多个元素：物体、动作、场景、事件等
  - 多步骤、迭代过程

图说模型Image Captioning

理解模式

1. 完整理解图片所有内容
2. 用语言描述出自己的理解



模型策略

1. 传统的分段处理策略
2. 新的端对端对点策略(End-to-end trainable way)

图说模型Image Captioning

传统的分段处理策略

- 流程
  1. 图片内容→文本标签
  2. 文本标签→描述语句
- 优势
  - 滤除干扰信息
  - 模块化结构
  - 直接使用CV和NLP的研究成果
- 不足
  - 第一步中错误判断会单向影响第二步的语言推理

图说模型Image Captioning

传统的分段处理策略

- 流程
  1. 将图片跟文本映射到同一共享空间下
  2. 翻译：图片特征→语言描述
- 优势
  - 同时训练，最优协作
  - 模块化结构
- 劣势
  - 黑箱严重

图说模型Image Captioning

$$P(S|I;\theta) = \prod_{t=0}^{N} P(S_t|S_0, S_1, \ldots, S_{t-1}, I; \theta)$$

$$\log P(S|I;\theta) = \sum_{t=0}^{N} \log P(S_t|S_0, S_1, \ldots, S_{t-1}, I; \theta)$$

$$\theta^* = \arg\max_{\theta} \sum_{(I,S)} \log P(S|I;\theta)$$

State-of-the-art模型组成

- DNN框架
  - CNN： 图片理解VGG，ResNet，GoogLeNet
  - RNN：语言理解及生成Multimodal-RNN, LSTM, GRU
  - 特殊功能模块：Attention

NIC模型

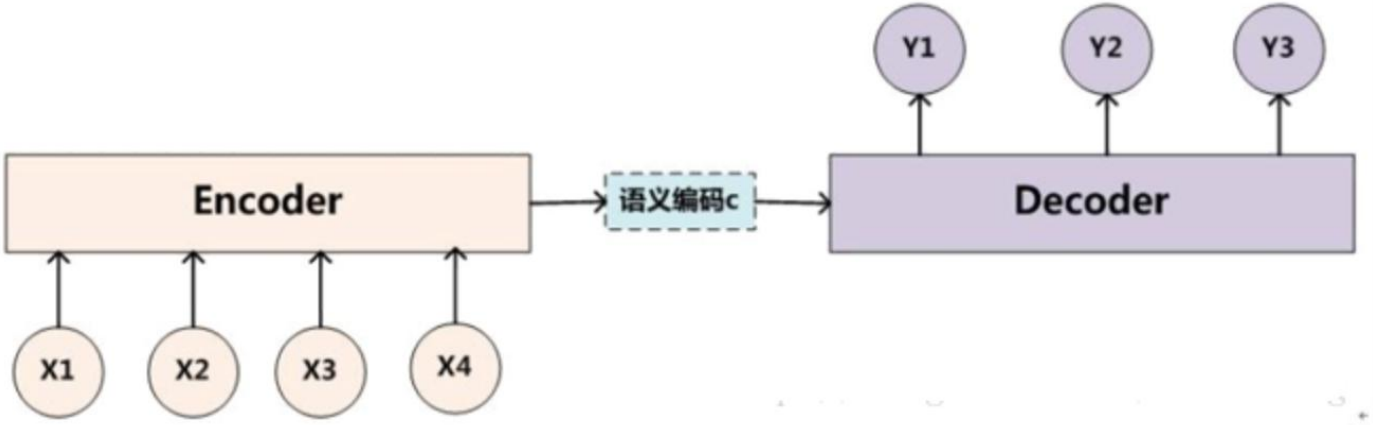
- NIC模型的结构非常"简单"：就是利用encoder-decoder框架，
- 首先利用CNN（这里是GoogLeNet）作为encoder，将 Softmax 之前的那一层固定维数的向量作为图像特征；
- 再使用LSTM作为decoder，其中图像特征输入decoder（图像特征仅在开始时刻输入，后面就没用了）。
- 模型的训练就是和任务描述那里介绍的一样，使用最大化对数似然来训练，然后在测试阶段采用beam search来减小搜索空间。

encoder-decoder简介

- 2014年，Sutskever提出的简单Encoder-Decoder模型，解决sequence to sequence转换的问题。
- 应用场景：
  - 机器翻译：待翻译的文本序列-->翻译文本序列
  - 语音识别：声学特征序列-->识别文本序列
  - 问答系统：问题描述单词序列-->生成答案单词序列
  - 文本摘要：文本序列-->摘要序列

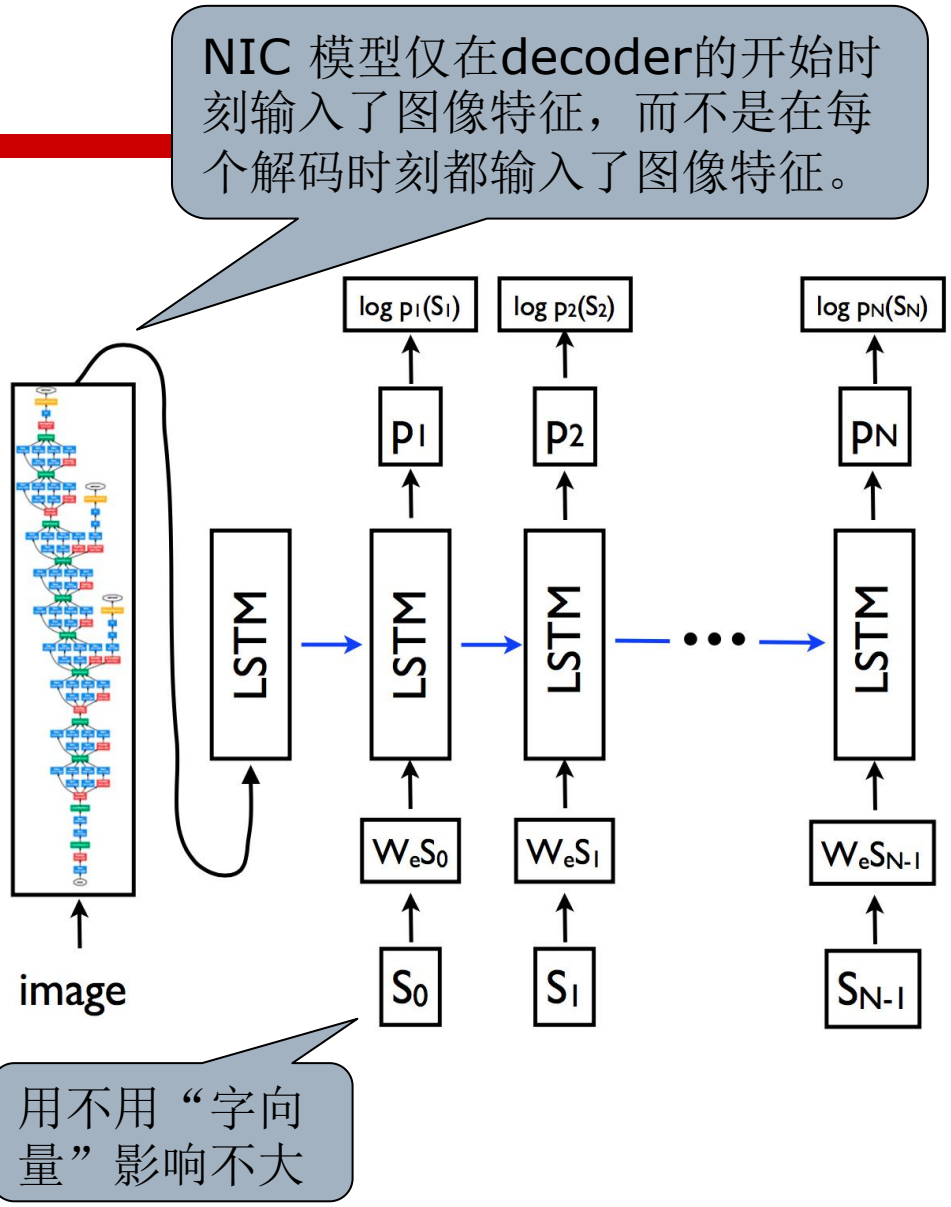
encoder-decoder简介

- 编码是将source sequence（x1,x2,x3,x4,...）转化为一个固定长度的context vector(c)，
- 解码是将该context vector转化为target sequence(y1,y2,y3,...)
- Encoder最后一个时间步的状态作为整个句子的中间语义context vector
- context vector直接作为Decoder的初始状态。



NIC模型

- CNN + LSTM
  - Inception v3生成图片特征（最后全连接层）
  - 特征映射矩阵We: 将文本映射到图片特征空间
  - 文本编码：one-hot
- LSTM
  - CNN特征作为第一个词
  - 句子中的词作为后续序列



NIC模型

- LSTM语言生成器

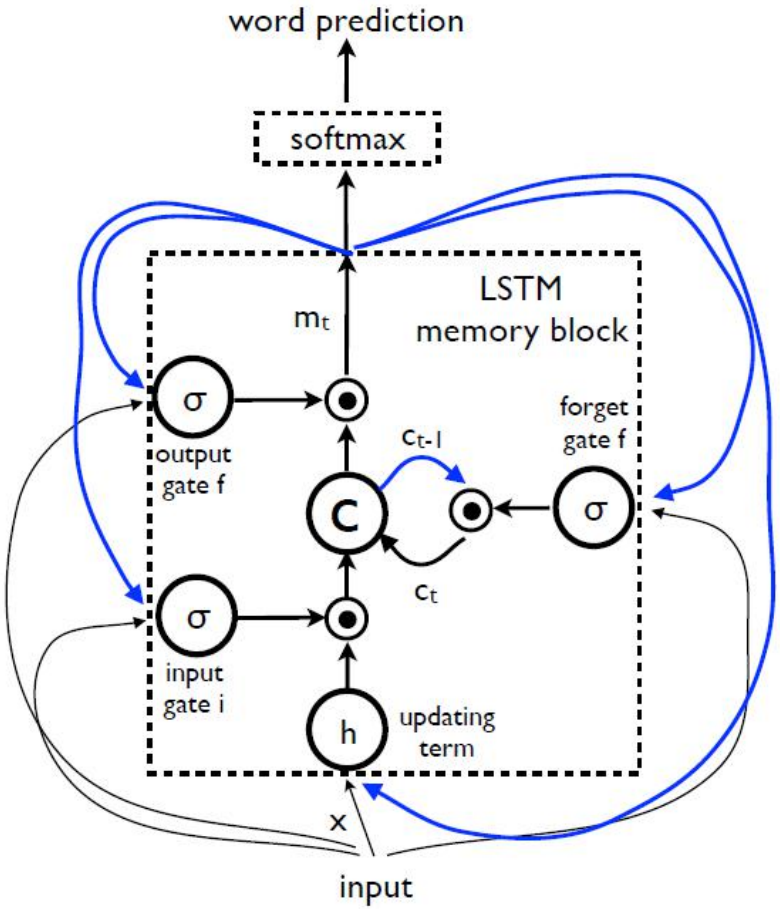
$$
\begin{aligned}
i_t &= \sigma(W_{ix}x_t + W_{im}m_{t-1}) \\
f_t &= \sigma(W_{fx}x_t + W_{fm}m_{t-1}) \\
o_t &= \sigma(W_{ox}x_t + W_{om}m_{t-1}) \\
c_t &= f_t \odot c_{t-1} + i_t \odot h(W_{cx}x_t + W_{cm}m_{t-1}) \\
m_t &= o_t \odot c_t \\
p_{t+1} &= \text{Softmax}(m_t)
\end{aligned}
$$

- CNN+LSTM图说

$$
\log p(S|I) = \sum_{t=0}^{N} \log p(S_t|I, S_0, \ldots, S_{t-1})
$$

$$
\begin{aligned}
x_{-1} &= \text{CNN}(I) \\
x_t &= W_e S_t, \quad t \in \{0 \ldots N-1\} \\
p_{t+1} &= \text{LSTM}(x_t), \quad t \in \{0 \ldots N-1\}
\end{aligned}
$$

$$
\theta^\star = \arg\max_\theta \sum_{(I,S)} \log p(S|I;\theta)
$$

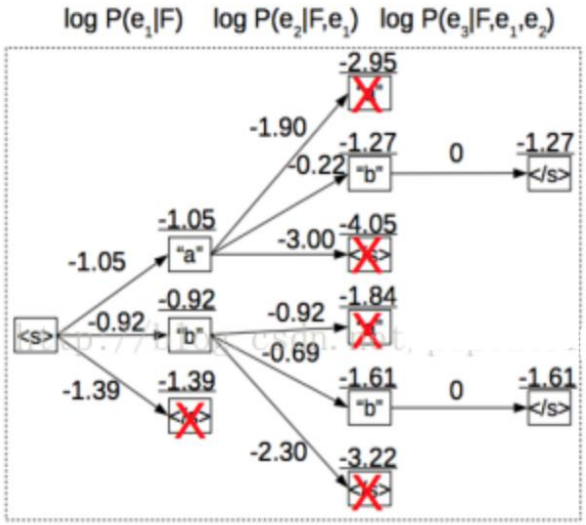


NIC模型

- 训练细节
  - 第一步：固定CNN学习率参数，无动量，训练LSTM语言模型500K
    - 词条化后去掉了词频小于5的词；
    - CNN参数：在ImageNet数据集预训练好的参数
    - 训练拆分：一句话n个词→n-1组训练序列
  - 第二步：微调CNN参数，CNN&LSTM一起训练100K
- 推理策略
  - Beam Search (尺寸=3)
  - 每一步获取Top3概率的词作为备选

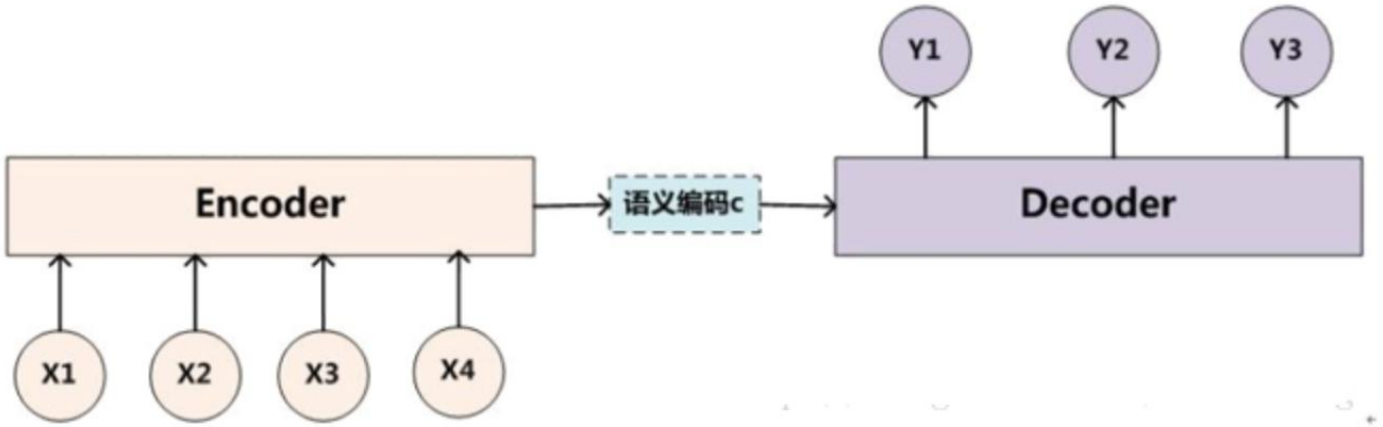
Beam Search

- 假设beam size为2；词典大小为3，内容为（a,b,c）。
- 生成第1个词的时候，选择概率最大的2个词，假设为a,c，则当前序列就是a,c。
- 生成第2个词的时候，将当前序列a和c，分别与词表中的所有词进行组合，得到新的6个序列aa ab ac ca cb cc,然后从其中选择2个得分最高的，作为当前序列。
- 后面会不断重复这个过程，直到遇到结束符为止。最终输出2个得分最高的序列。



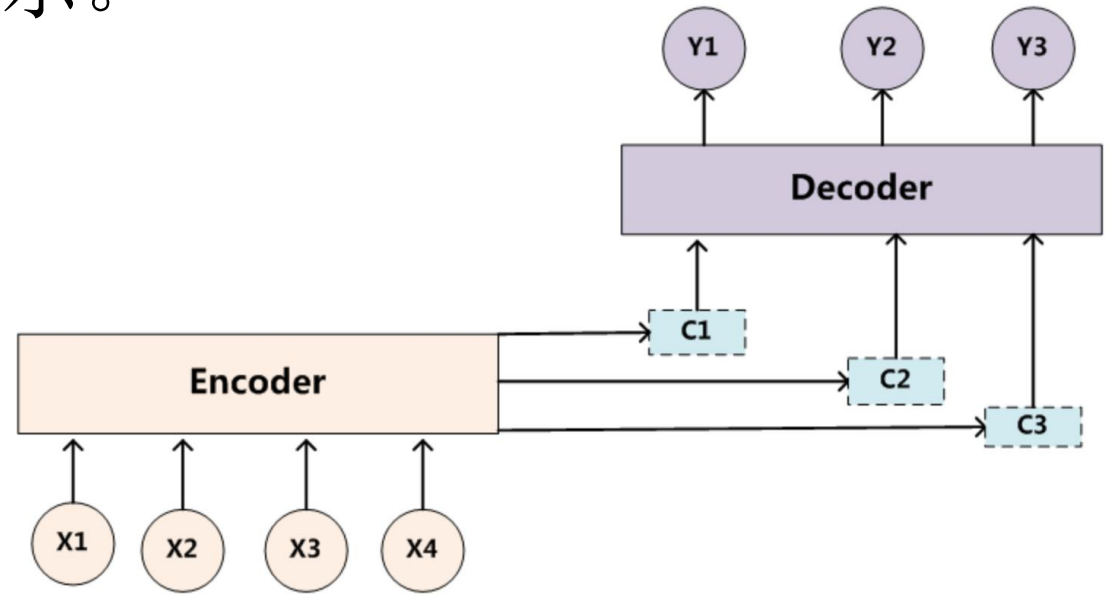
注意力机制

- 语义编码C是由句子Source的每个单词经过Encoder 编码产生的，
- 这意味着不论是生成哪个单词，句子Source中任意单词对生成某个目标单词yi来说影响力都是相同的。



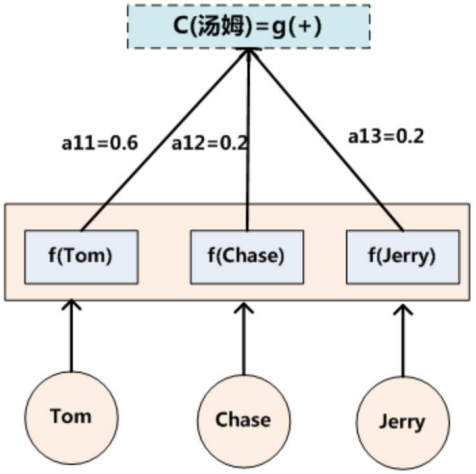
注意力机制

- 固定的中间语义表示C换成了根据当前输出单词来调整成加入注意力模型的变化的C_i。
- 增加了注意力模型的Encoder-Decoder框架如图所示。



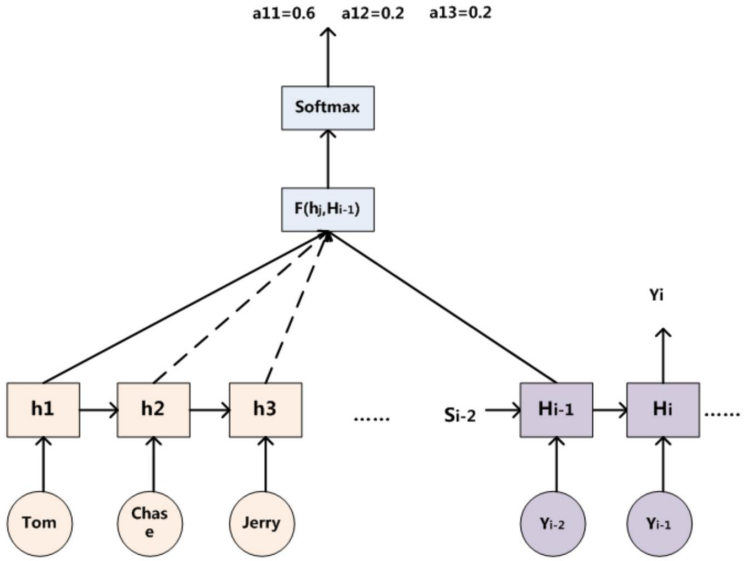
注意力机制

- Attention-based Encoder-Decoder，Encoder部分和传统的一致，Decoder部分引入Attention机制，使得c更灵活。
- 这时c就是由$h_1,...,h_T$加权求和获得，且每个h_i前面的权重系数W_i不一样，增加模型的灵活性。



注意力机制

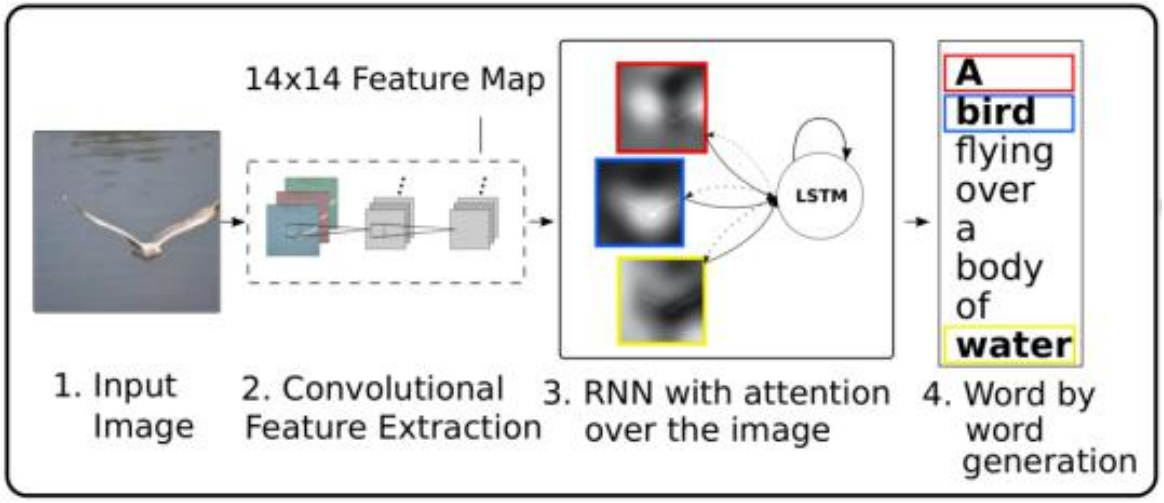
- 可以用Target输出句子i-1时刻的隐层节点状态H_{i-1}去一一和输入句子Source中每个单词对应的RNN隐层节点状态h_j进行对比，
- 即通过函数$F(h_j,H_{i-1})$来获得目标单词y_i和每个输入单词对应的对齐可能性，然后函数F的输出经过Softr



Show, attend and tell(SAT)模型

- Encoder，模型使用CNN来提取 L 个 D 维的特征vector作为注释向量，每一个都对应图像的一个区域，如式： $a = \{\mathbf{a}_1, \ldots, \mathbf{a}_L\}, \mathbf{a}_i \in \mathbb{R}^D$
- Decoder，解码阶段用LSTM网络生成caption，其中C是句子长度，K是词表大小，y是各个词的one-hot编码所构成的集合。如式：

$$y = \{\mathbf{y}_1, \ldots, \mathbf{y}_C\}, \mathbf{y}_i \in \mathbb{R}^K$$



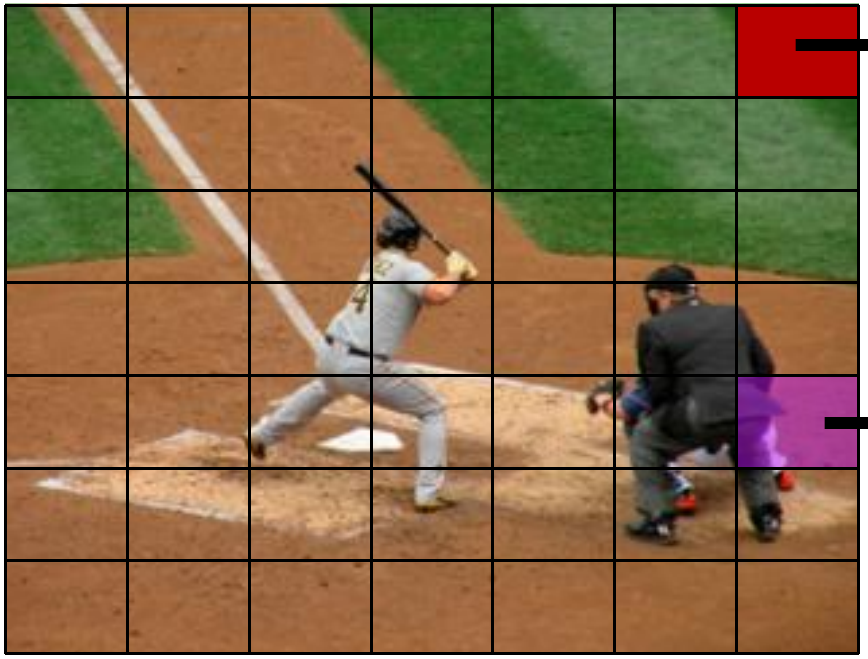
Show, attend and tell(SAT)模型

- CNN + LSTM + Attention module
- CNN
  - VGG最后卷基层生成图片特征
- 特征映射矩阵: 将文本映射到图片特征空间
- 文本编码
  - one-hot
- LSTM
  - 添加第3个输入：基于attention的图片特征

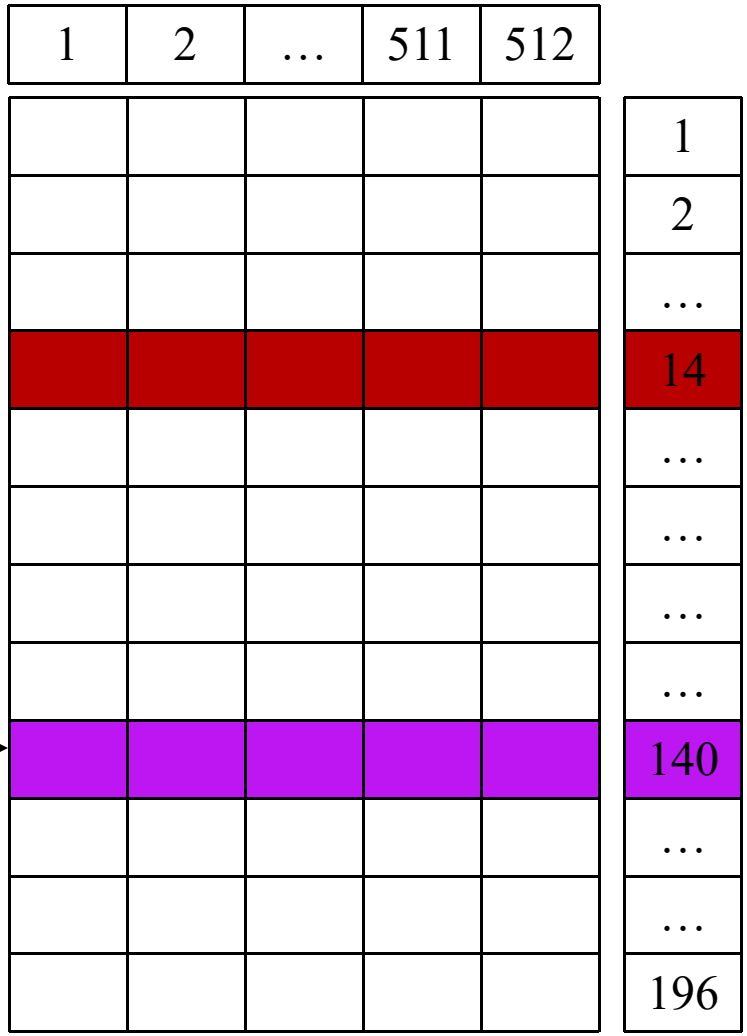
Show, attend and tell(SAT)模型

注意力机制的CNN特征

- VGG最后卷积层输出14x14x512



14 X 14 网格拆分



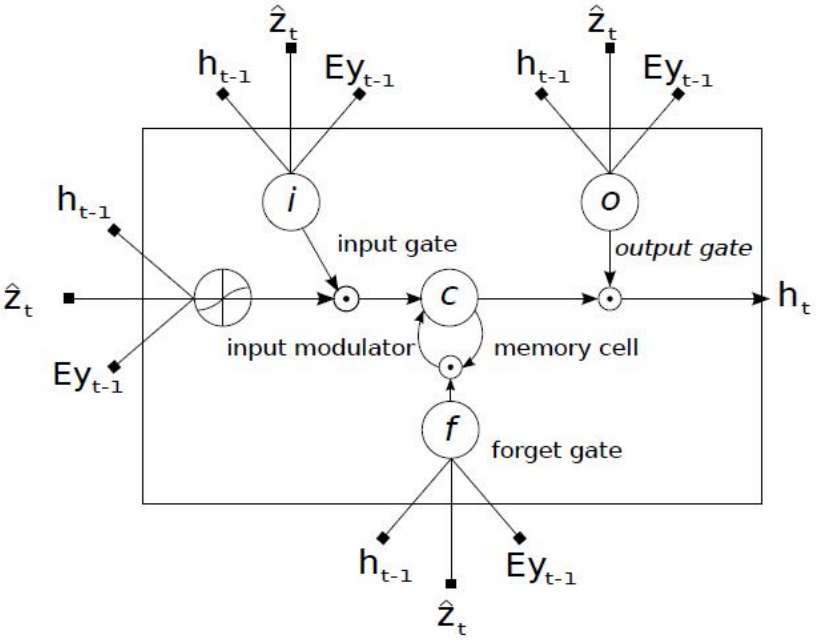
Show, attend and tell(SAT)模型

- LSTM改进

视觉信息

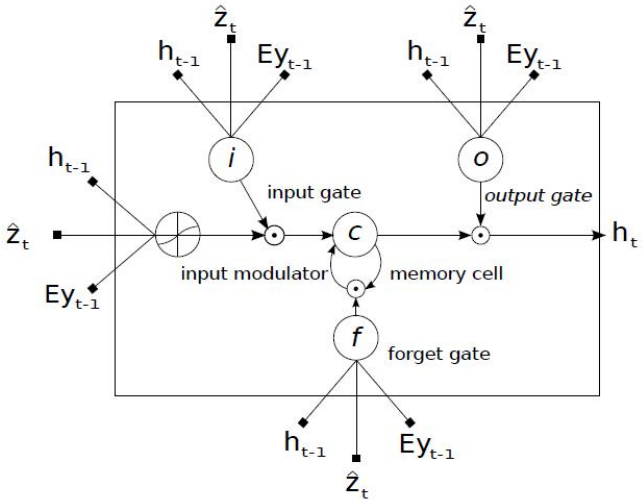
$$
\begin{aligned}
\mathbf{i}_t &= \sigma(W_i E\mathbf{y}_{t-1} + U_i \mathbf{h}_{t-1} + Z_i \hat{\mathbf{z}}_t + \mathbf{b}_i),\\
\mathbf{f}_t &= \sigma(W_f E\mathbf{y}_{t-1} + U_f \mathbf{h}_{t-1} + Z_f \hat{\mathbf{z}}_t + \mathbf{b}_f),\\
\mathbf{c}_t &= \mathbf{f}_t \mathbf{c}_{t-1} + \mathbf{i}_t \tanh(W_c E\mathbf{y}_{t-1} + U_c \mathbf{h}_{t-1} + Z_c \hat{\mathbf{z}}_t + \mathbf{b}_c),\\
\mathbf{o}_t &= \sigma(W_o E\mathbf{y}_{t-1} + U_o \mathbf{h}_{t-1} + Z_o \hat{\mathbf{z}}_t + \mathbf{b}_o\\
\mathbf{h}_t &= \mathbf{o}_t \tanh(\mathbf{c}_t).
\end{aligned}
$$

- 输入、遗忘和输出门均由sigmoid激活，所以得到的值在0~1之间，可以直接作为概率值；
- 候选向量c_t 和h_t由tanh激活，值在-1~1之间。



Show, attend and tell(SAT)模型

- Ey_{t-1}是look-up得到词 y_{t-1}的 m 维词向量；
- h_{t-1}是LSTM上一时刻的隐状态；
- $\hat{z}_t$是LSTM真正的"输入"，代表的是捕捉了特定区域视觉信息的上下文向量。
- 隐状态和细胞状态的初始值的计算方式：两个独立的多层感知机，感知机的输入是各个图像区域特征的平均。
- 根据以上，就可以求得当前时刻最大概率的输出词，并作为下一时刻的输入，从而获得整个描述（caption）结果。



$$\mathbf{c}_0 = f_{\text{init,c}}\left(\frac{1}{L}\sum_i^L \mathbf{a}_i\right)$$

$$\mathbf{h}_0 = f_{\text{init,h}}\left(\frac{1}{L}\sum_i^L \mathbf{a}_i\right)$$

$$p(\mathbf{y}_t|\mathbf{a}, \mathbf{y}_1^{t-1}) \propto \exp(\mathbf{L}_o(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{L}_h\mathbf{h}_t + \mathbf{L}_z\hat{\mathbf{z}}_t))$$

Show, attend and tell(SAT)模型

SAT模型Attention module

实现attention的方式就是：z_t由时间和"注意力"位置区域共同决定，对于每个时间的每个区域都定义一个权重值α_{ti}。

- 注意机制模块
  - 注意权重推断

概率表达

$$e_t = f_{att}(a, h_{t-1}) = W_{att}\tanh(W_a a + W_h h_{t-1} + b)$$

$$\alpha_t = \mathrm{softmax}(e_t)$$

  - 14x14特征中单个特征

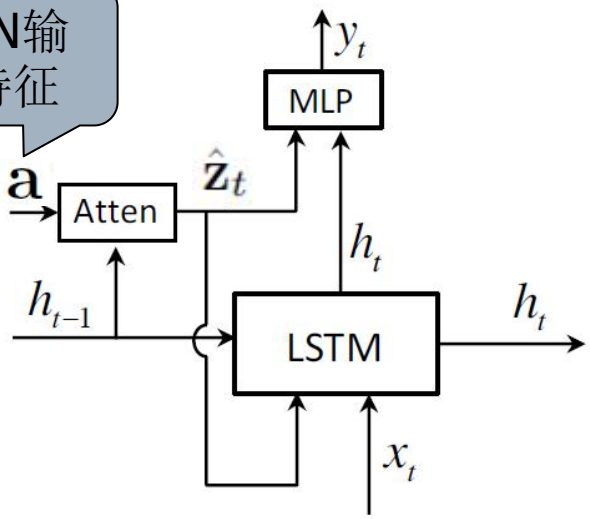
$$e_{ti} = f_{\mathrm{att}}(\mathbf{a}_i, \mathbf{h}_{t-1})$$

$$\alpha_{ti} = \frac{\exp(e_{ti})}{\sum_{k=1}^{L}\exp(e_{tk})}$$

  - 特征融合（权重相加）

$$\hat{\mathbf{z}}_t = \phi(\{\mathbf{a}_i\}, \{\alpha_i\}) = \sum_i^L \alpha_i \mathbf{a}_i = \alpha_t \cdot \mathrm{a}$$

CNN输出特征



Show, attend and tell(SAT)模型

- Hard attention，这里权重α_{ti}所起的作用的是否被选中，只有0、1两个选项，所以引入了变量$s_{t,i}$，当区域i被选中时为1，否则为0。

$$\hat{\mathbf{z}}_t = \sum_i s_{t,i}\mathbf{a}_i$$

- Soft attention，对每个区域都关注，只是关注的重要程度不一样，所以此处的权重α_{ti}就对应着此区域所占比重，那么z_t就可以直接通过比重加权求和得到。

$$\mathbb{E}_{p(s_t|a)}[\hat{\mathbf{z}}_t] = \sum_{i=1}^{L} \alpha_{t,i}\mathbf{a}_i$$

Show, attend and tell(SAT)模型

SAT模型Attention module（注意机制）



A woman is throwing a frisbee in a park.



A dog is standing on a hardwood floor.



A little girl sitting on a bed with a teddy bear.



A group of people sitting on a boat in the water.

图说数据集

数据集（目前主要集中在英文）

- MSCOCO标注集&竞赛
  - 描述(Captions)
  - 80个类别的物体(Object)语义分割
  - 100,000人的肢体关键点(Keypoints)
  - 其他附属标注集

	训练集	评估集	测试集	图片标注
MSCOCO	82783	40504	40775	5句描述
Flickr30k	28000	1000	1000	5句描述
Flickr8k	6000	1000	1000	5句描述

图说的性能指标

- Meteor
  - 与人类评判结果最接近
- CIDEr
  - 与人类评判结果次相关
- BLEU@N（N代表n-gram，分别是1,2,3,4）
- ROUGE-L
- Perplexity

图说的性能指标

- Meteor是用来评测机器翻译的，对模型给出的译文与参考译文进行词对齐，计算词汇完全匹配、词干匹配和同义词匹配等各种情况的准确率、召回率和F值。
- 先计算 unigram 情况下的准确率 P 和召回率 R，得到调和均值（F值）然后进行调整。
- Meteor的特别之处在于，它不希望生成很"碎"的译文：比如参考译文是"A B C D"，模型给出的译文是"B A D C"，虽然每个 unigram 都对应上了，但是会受到很大的惩罚。

图说的性能指标

- BLEU@n也是机器翻译评测标准，n 常取1到4，代表N-gram的长度。
- 基于思路：计算模型生成的译文中出现的 n-gram，有多少出现在参考译文中，其百分比作为 precision 。
- 针对基本思路的两点改进：
  - 1、解决"停用词"：与参考译文的交集来评分
  - 2、解决小短句：对短句子进行惩罚
- BLEU的优点是：粒度为 n-gram，考虑了更长的匹配信息；
- BLEU的缺点是：不考虑 n-gram 的词性。

图说的性能指标

- CIDEr：这个指标将每个句子都看作"文档"，将其表示成 tf-idf 向量的形式，然后计算参考caption与模型生成的caption的余弦相似度，作为打分。实际上就是向量空间模型。
- 通过全部 n-gram 构成的VSM的词表。其中 idf 的分母部分代表的是出现于参考caption的图片个数。
- 这个指标的发明动机之一是BLEU的一个缺点，就是对所有匹配上的词都同等对待，而实际上有些词应该更加重要。

参考资料

- https://www.cnblogs.com/Determined22/p/6910277.html
- https://blog.csdn.net/chinabing/article/details/78763454
- https://www.jianshu.com/p/7582df96b081

问答互动

在所报课的课程页面，

1、点击“全部问题”显示本课程所有学员提问的问题。

2、点击“提问”即可向该课程的老师和助教提问问题。



联系我们

小象学院：互联网新技术在线教育领航者

– 微信公众号：**小象学院**

