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。

- 课程详情请咨询
  - 微信公众号：大数据分析挖掘
  - 新浪微博：ChinaHadoop

递归神经网络

主讲人： 李伟

纽约城市大学博士

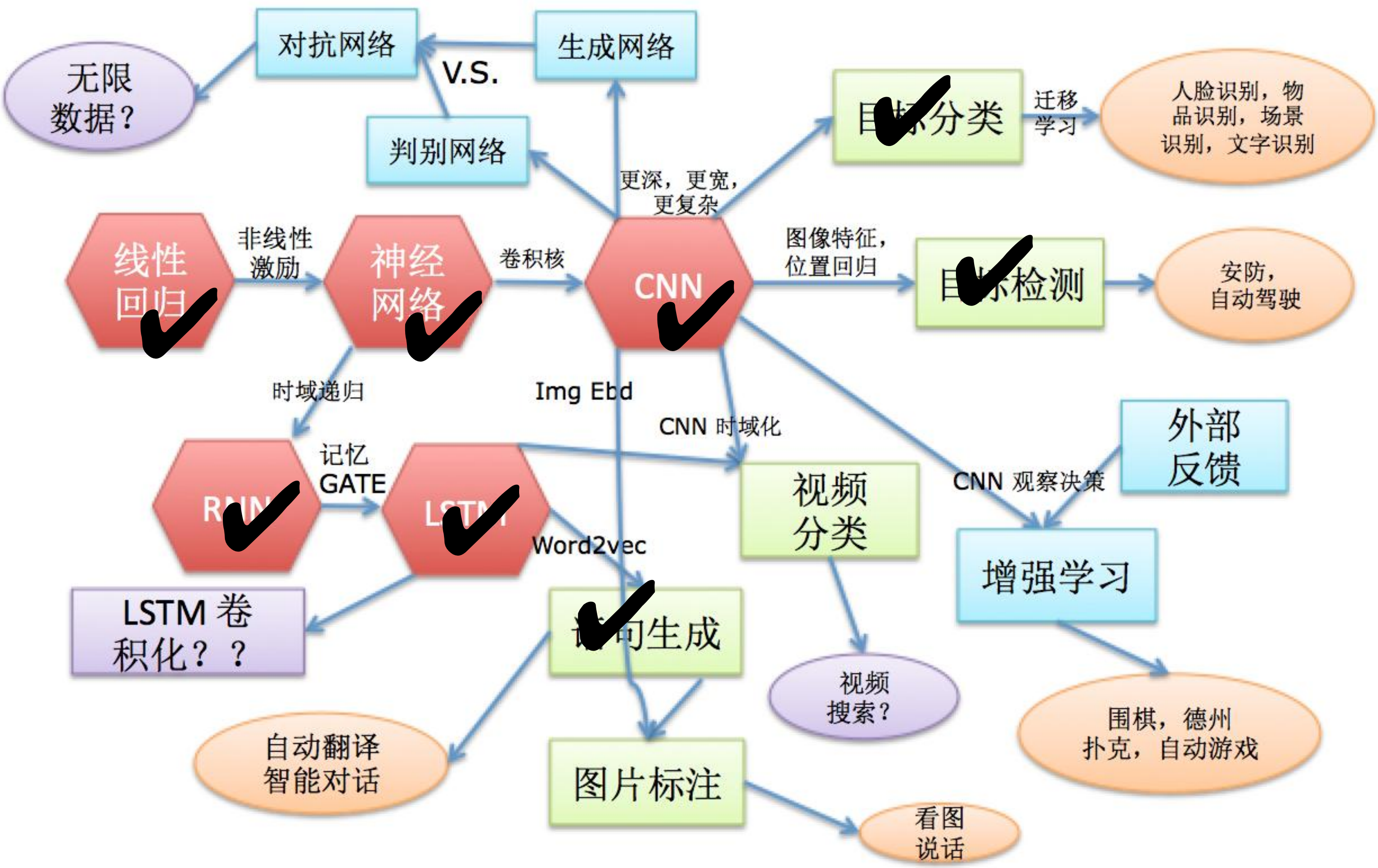
主要研究深度学习，计算机视觉，人脸计算

多篇重要研究文章作者，重要会议期刊审稿人

微博ID：weightlee03 （相关资料分享）

GitHub ID：wiibrew （课程代码发布）

结构



对抗网络
生成网络
V.S.
无限
数据？
判别网络
目标分类
迁移
学习
人脸识别，物
品识别，场景
识别，文字识别
更深，更宽，
更复杂
线性
回归
非线性
激励
神经
网络
卷积核
CNN
图像特征，
位置回归
目标检测
安防，
自动驾驶
时域递归
Img Ebd
CNN 时域化
外部
反馈
记忆
GATE
RNN
LSTM
视频
分类
CNN 观察决策
Word2vec
增强学习
LSTM 卷
积化？？
语句生成
视频
搜索？
围棋，德州
扑克，自动游戏
自动翻译
智能对话
图片标注
看图
说话

提纲

- 1. 递归神经网络RNN原理
- 2. 升级版RNN：LSTM
- 3. 语言处理特征提取：Word2Vec
- 4. 实例：LSTM用于语言处理

期待目标

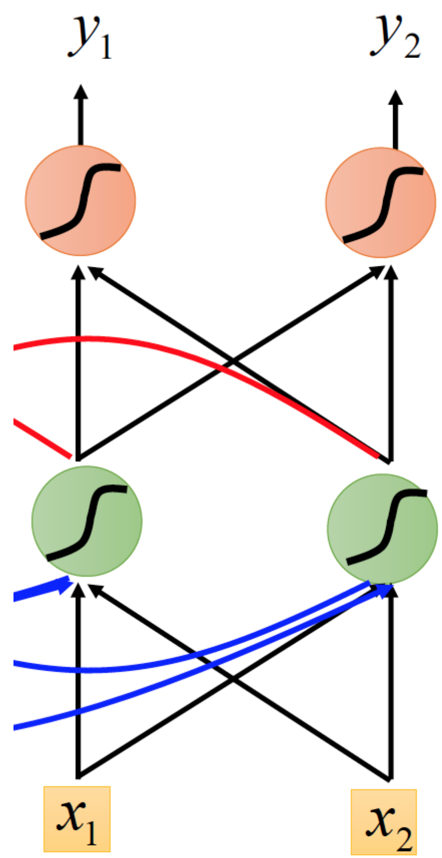
- 1. 理解从传统神经网络到递归神经网络RNN的转化
- 2. RNN特点，缺陷，LSTM的设计
- 3. 理解word2vec设计，特点，明白如何
- 4. 了解LSTM与word2vec结合用于语言相关应用

提纲

- **1. 递归神经网络RNN**
- 2. 升级版RNN：LSTM
- 3. 语言处理特征提取：Word2Vec
- 4.实例：LSTM用于语言处理

递归神经网络RNN

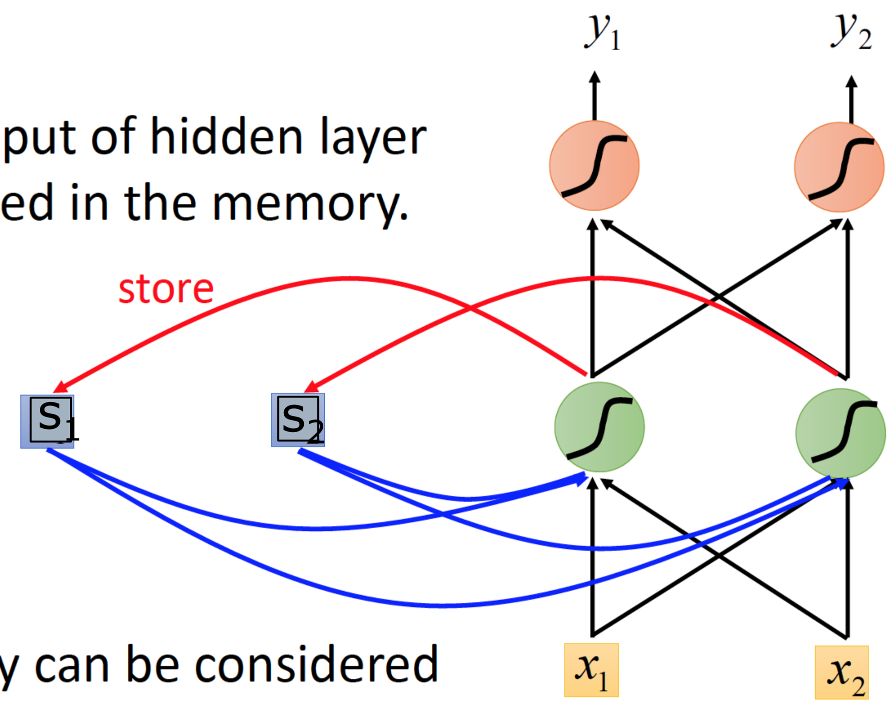
- 传统神经网络
- 输入，输出，隐含层
- 如果X为序列，输出影响？
- 是否有记忆能力？



递归神经网络RNN

- 递归神经网络
- 中间层激励保存
- 下一刻重新输入
- 记忆功能

The output of hidden layer are stored in the memory.

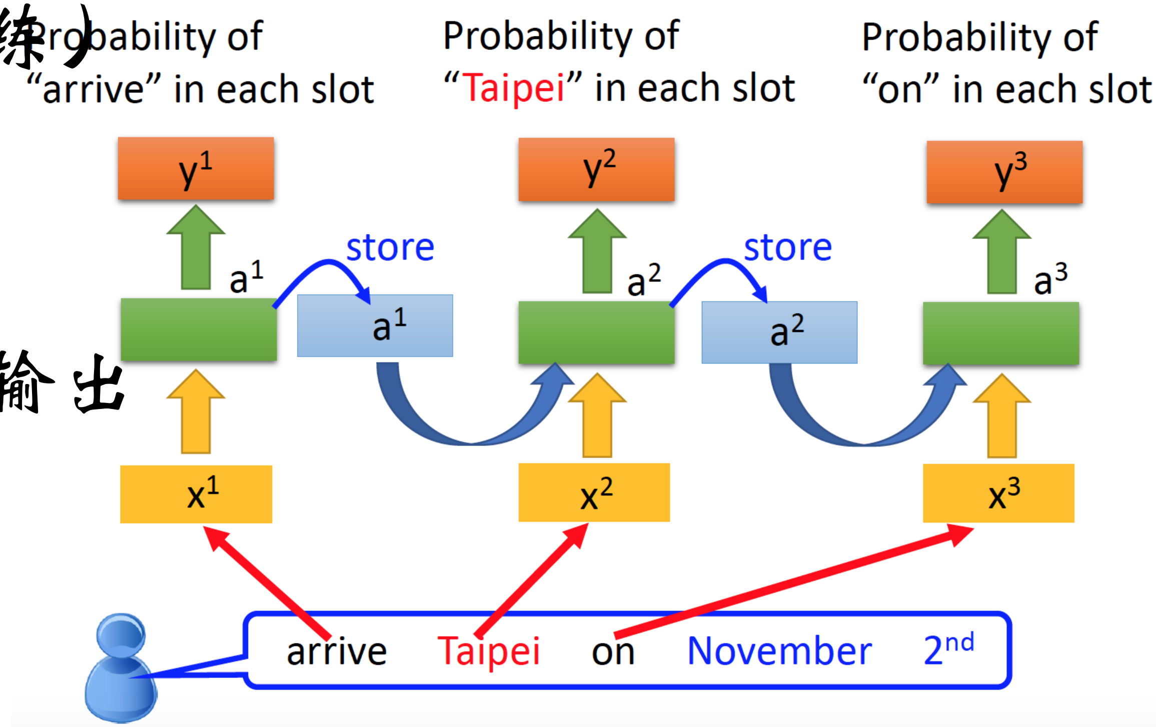


Memory can be considered as another input.

递归神经网络RNN

- 递归神经网络
- 工作过程：

1. 文本处理（训练）
2. 一个神经元，
   不同时刻
3. 每个时刻都有输出



递归神经网络RNN

- 递归神经网络
- 表达式：

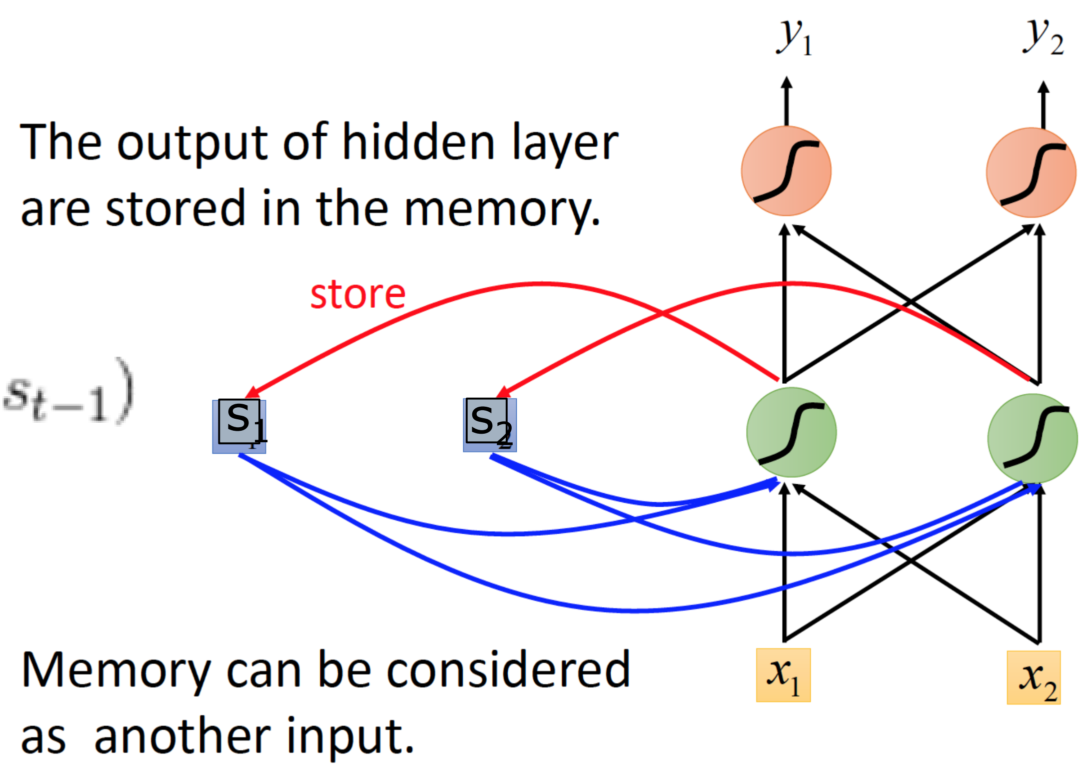
正向传播：

$$s_t = tanh(W \cdot x + U \cdot s_{t-1})$$

$$\hat{y}_t = \text{softmax}(V s_t)$$

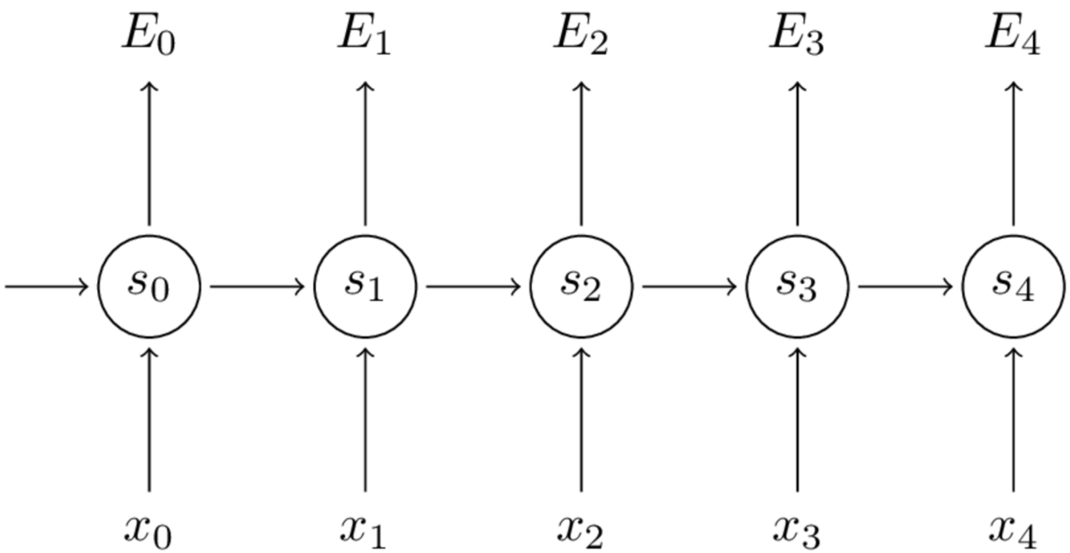
W：输入激励参数；

U：不同时刻状态转换参数



递归神经网络RNN

- 递归神经网络
- 损失函数



$$E_t(y_t, \hat{y}_t) = -y_t \log \hat{y}_t$$

$$E(y, \hat{y}) = \sum_t E_t(y_t, \hat{y}_t)$$

$$= -\sum_t y_t \log \hat{y}_t$$

递归神经网络RNN

- 递归神经网络
- 反向计算

1. 参数优化方法：同传统神经网络一样，梯度下降
2. 计算损失函数对参数的导数
3. 每个输出都对参数有影响

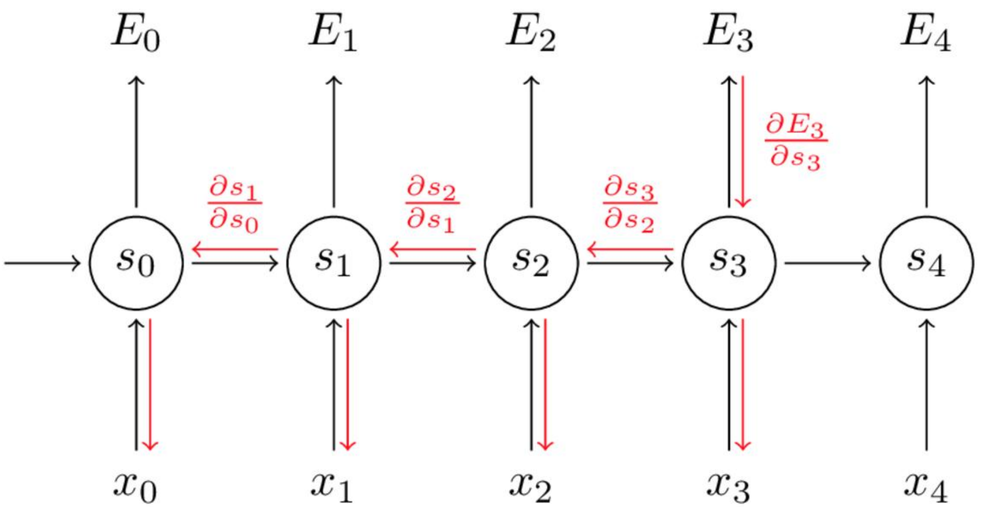
对参数的导数为各个输出对参数导数之和

$$\frac{\partial E}{\partial W} = \sum_t \frac{\partial E_t}{\partial W}.$$

递归神经网络RNN

- 递归神经网络
- 反向计算

1. 损失函数有多个，以E_3为分析对象
2. 链式法则，梯度前向传导
3. 多条路径t_1, t_2, t_3对参数W影响

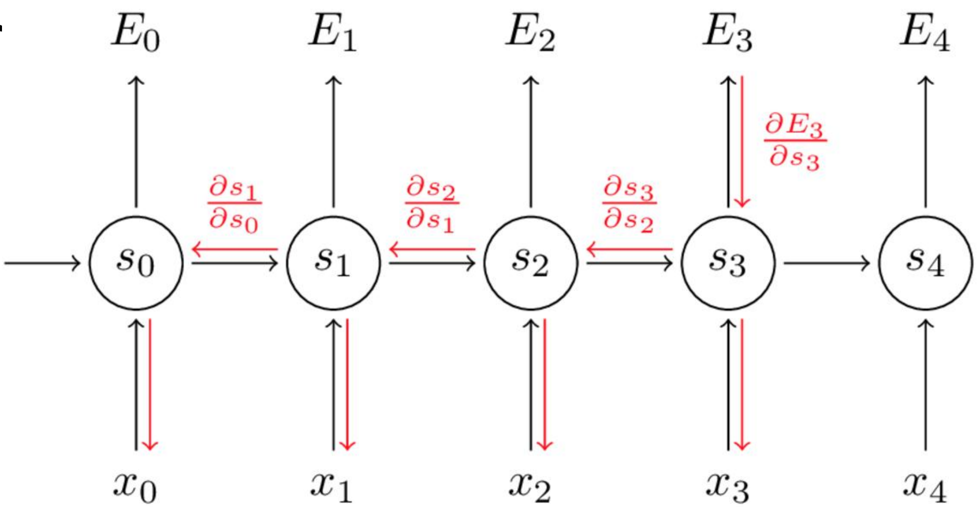


递归神经网络RNN

- 递归神经网络
- 反向计算：Backpropagation Through Time (BPTT)
- E_3由$t_0 - t_3$时刻x，W共同确定

ΔW的确定要考虑E_3在各个时刻对w导数

$$\Delta s_3 = \frac{\partial E_3}{\partial s_3}$$



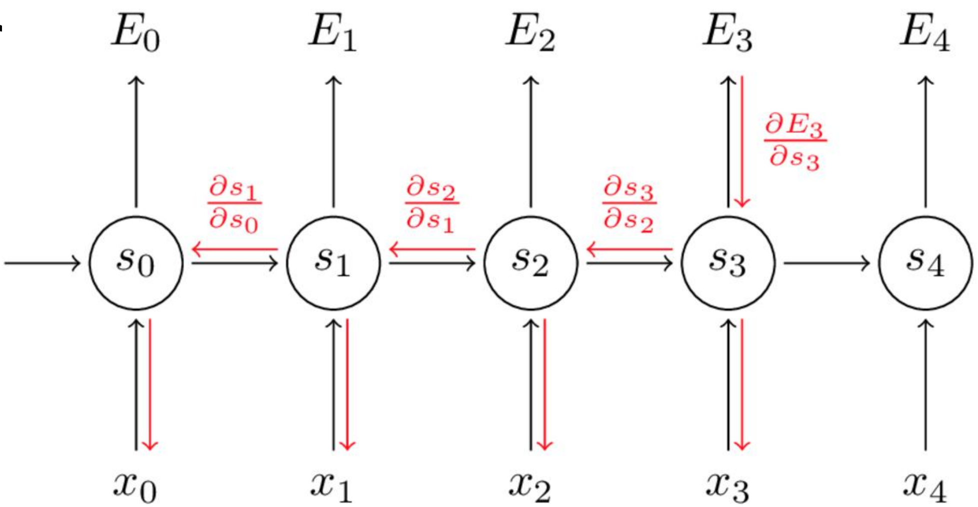
递归神经网络RNN

- 递归神经网络
- 反向计算：Backpropagation Through Time (BPTT)
- E_3由$t_0 - t_3$时刻x，W共同确定

ΔW的确定要考虑E_3在各个时刻对w导数

t3:

$$\frac{\partial E_3}{\partial w} = \frac{\partial s_3}{\partial w}\Delta s_3$$



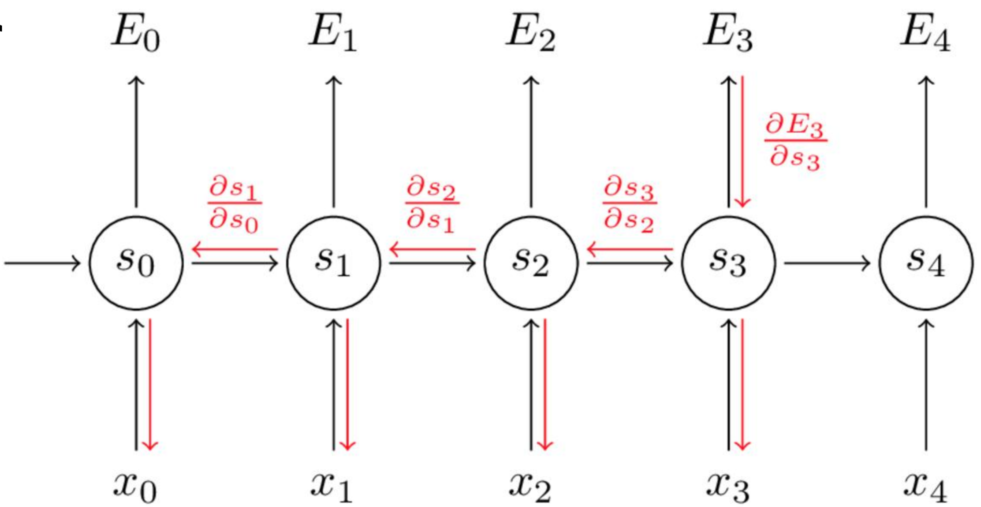
递归神经网络RNN

- 递归神经网络
- 反向计算：Backpropagation Through Time (BPTT)
- E_3由t_0-t_3时刻x，W共同确定

Δ W的确定要考虑E_3在各个时刻对w导数

t2:

$$\frac{\partial E_3}{\partial w} = \frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial w}\Delta s_3$$



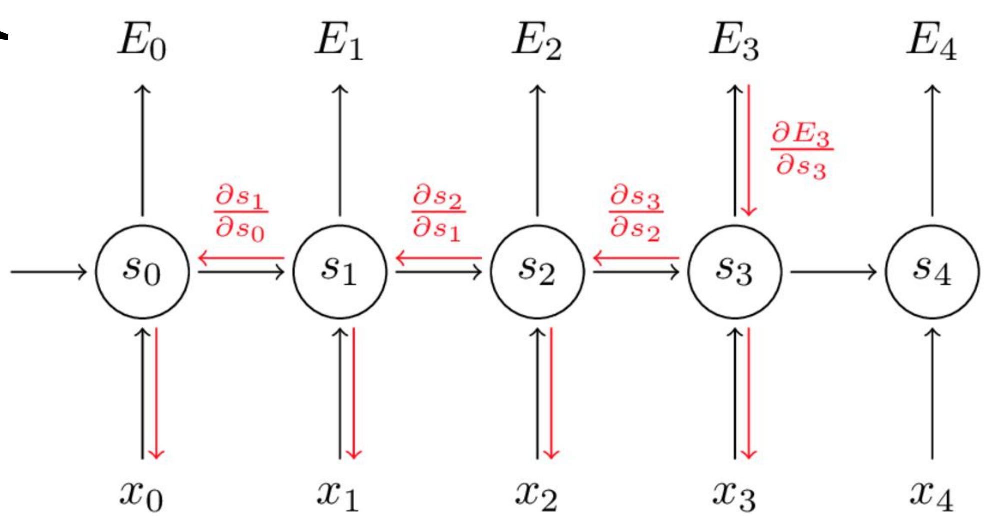
递归神经网络RNN

- 递归神经网络
- 反向计算：Backpropagation Through Time (BPTT)
- E_3由t_0-t_3时刻x，W共同确定

ΔW的确定要考虑E_3在各个时刻对w导数

t1:

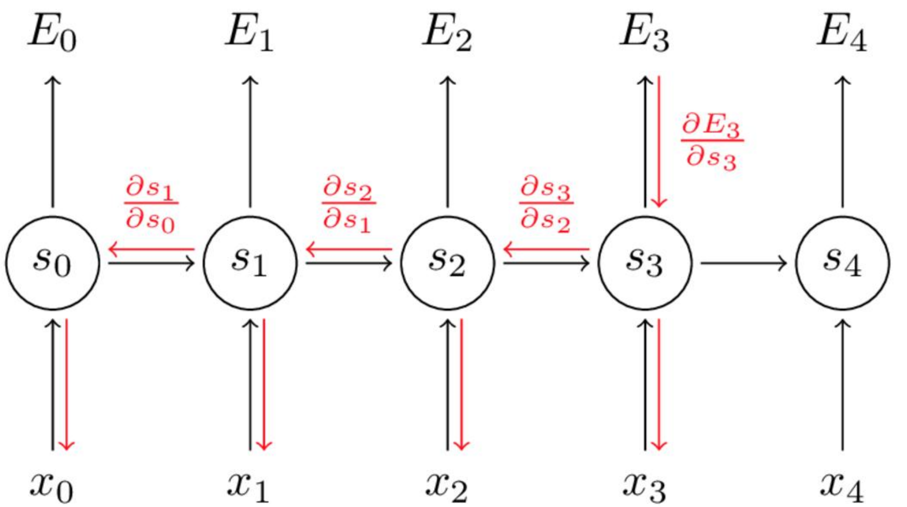
$$\frac{\partial E_3}{\partial w}=\frac{\partial s_3}{\partial s_2}\frac{\partial s_2}{\partial s_1}\frac{\partial s_1}{\partial w}\Delta s_3$$



递归神经网络RNN

- 递归神经网络
- 反向计算：Backpropagation Through Time (BPTT)
- E_3由t_0-t_3时刻x，W共同确定

$$\frac{\partial E_3}{\partial w}=\sum_{k=0}^{3}\frac{\partial s_3}{\partial s_k}\frac{\partial s_k}{\partial w}\Delta s_3$$



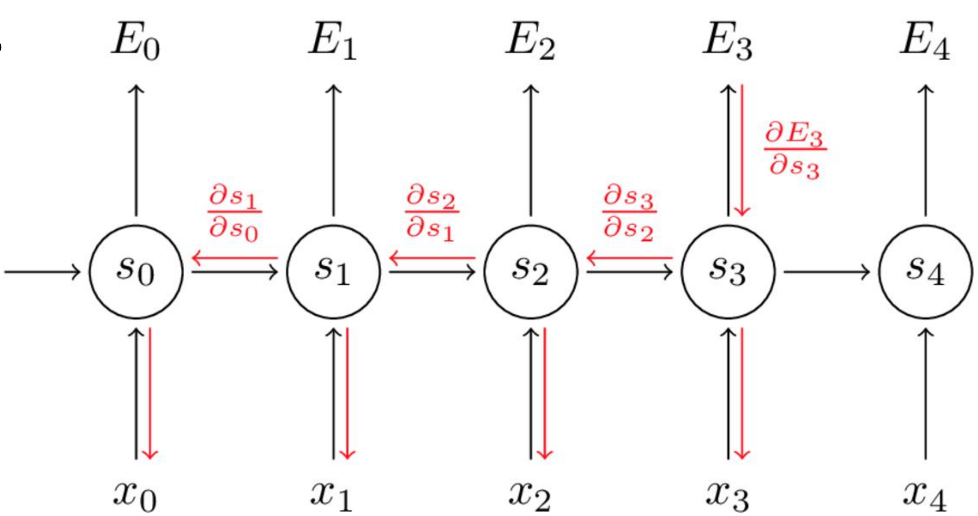
递归神经网络RNN

- 递归神经网络
- 反向计算：Backpropagation Through Time (BPTT)
- 如何对U求导？

1. U是不同时刻中间状态之间变化关系

2. E_3 生成过程中U参与3次

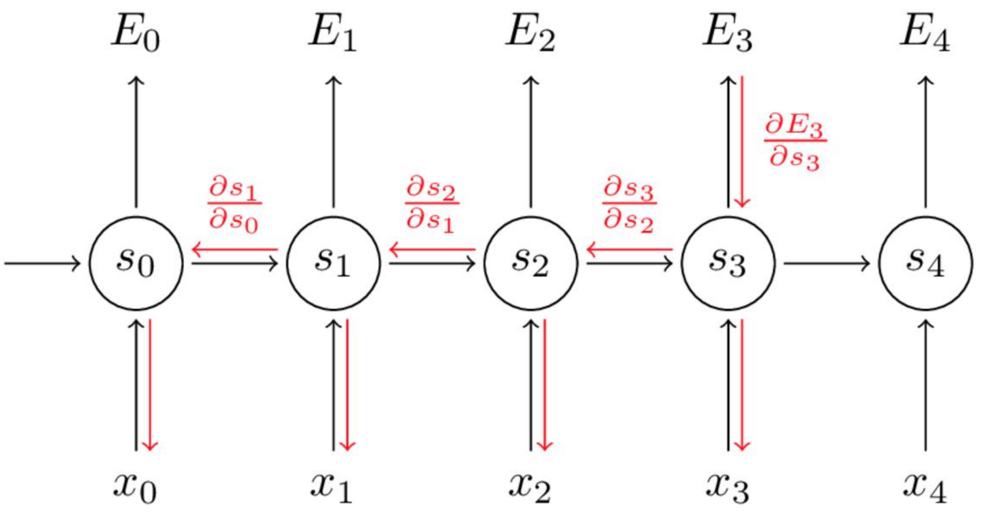
3. 对U求导为三次之和



递归神经网络RNN

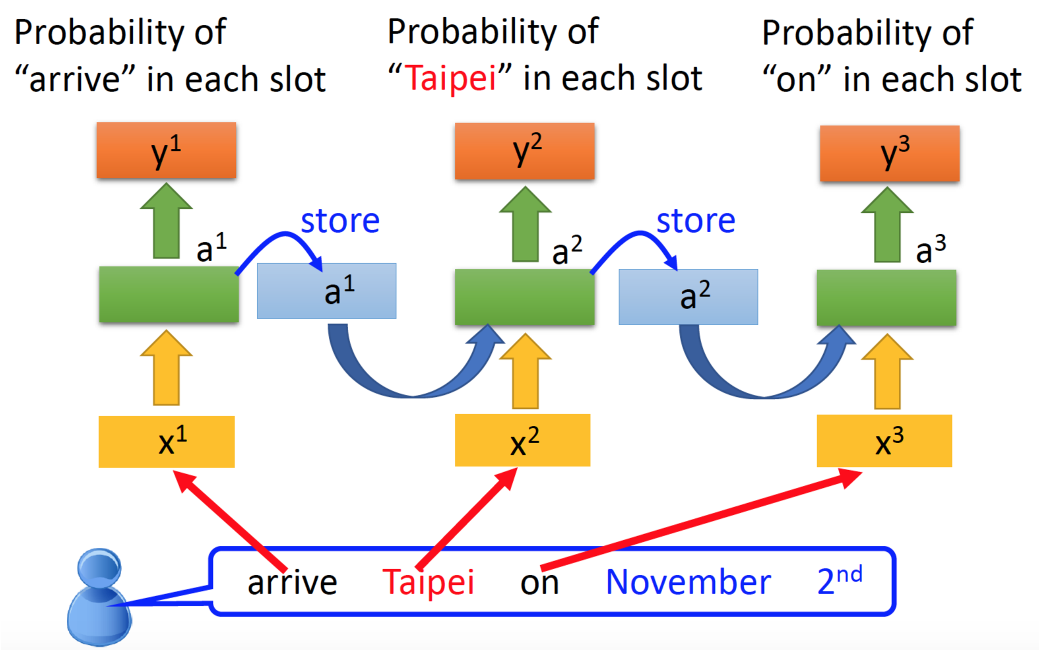
- 递归神经网络
- 反向计算：Backpropagation Through Time (BPTT)
- 如何对U求导？

$$\frac{\partial E_3}{\partial U} = \sum_{k=0}^{3} \frac{\partial s_3}{\partial s_k} \frac{\partial s_k}{\partial U} \Delta s_3$$



递归神经网络RNN

- 递归神经网络
- 作用：语言，文本信息处理

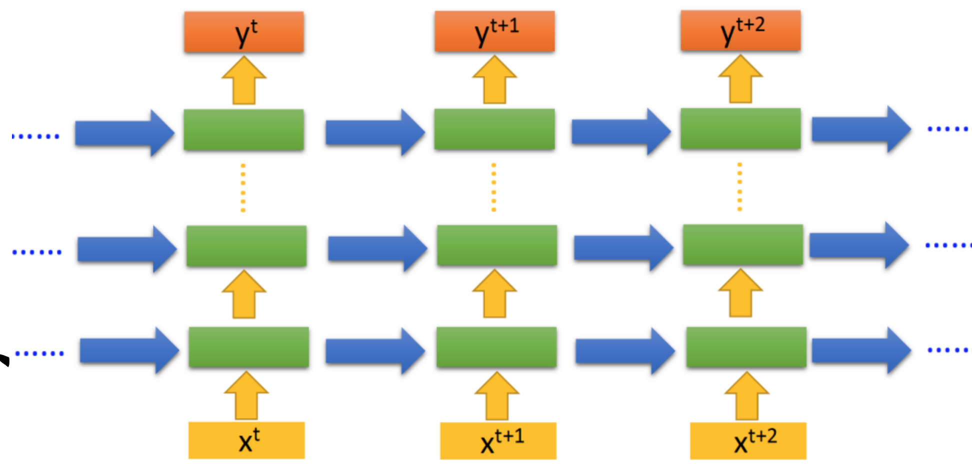


递归神经网络RNN

- 递归神经网络
- 结构：多层网络，双向网络

如何理解多层结构？

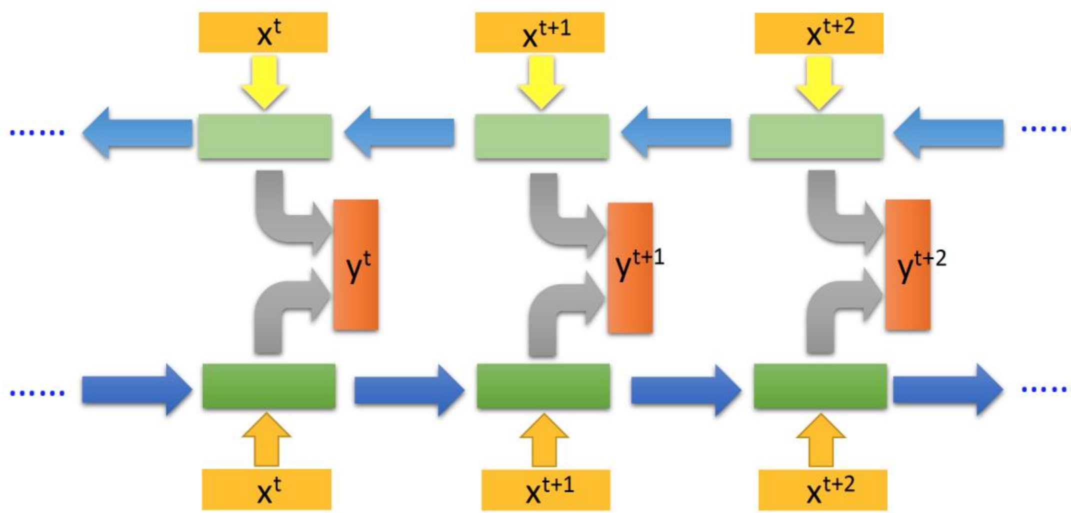
类比传统神经网络单层到多层的结构变化，额外添加上层前一状态



递归神经网络RNN

- 递归神经网络
- 结构：多层网络，双向网络

输入信息正向，反向输入RNN，原因：信息的依赖关系顺序不定的。



递归神经网络RNN

- 递归神经网络
- Vanishing Gradient 问题

$$\frac{\partial E_3}{\partial w} = \sum_{k=0}^{3} \frac{\partial s_3}{\partial s_k} \frac{\partial s_k}{\partial w} \Delta s_3$$

Jacobian matrix

$$\frac{\partial E_3}{\partial W} = \sum_{k=0}^{3} \left(\boxed{\prod_{i=k+1}^{3} \frac{\partial s_i}{\partial s_{i-1}}} \right) \frac{\partial s_k}{\partial W} \Delta s_3$$

$U_{nxn} S_{i-1} = S_i$

U最大特征值大于1 爆炸

小于1 消失

>1 nan 容易察觉,<1 难以发现

小象学院 ChinaHadoop.cn

递归神经网络RNN

- 递归神经网络
- Vanishing Gradient 问题

影响，较长的记忆无法产生作用

- 如何解决？

1. 非线性激励更换

2. LSTM 长短记忆单元

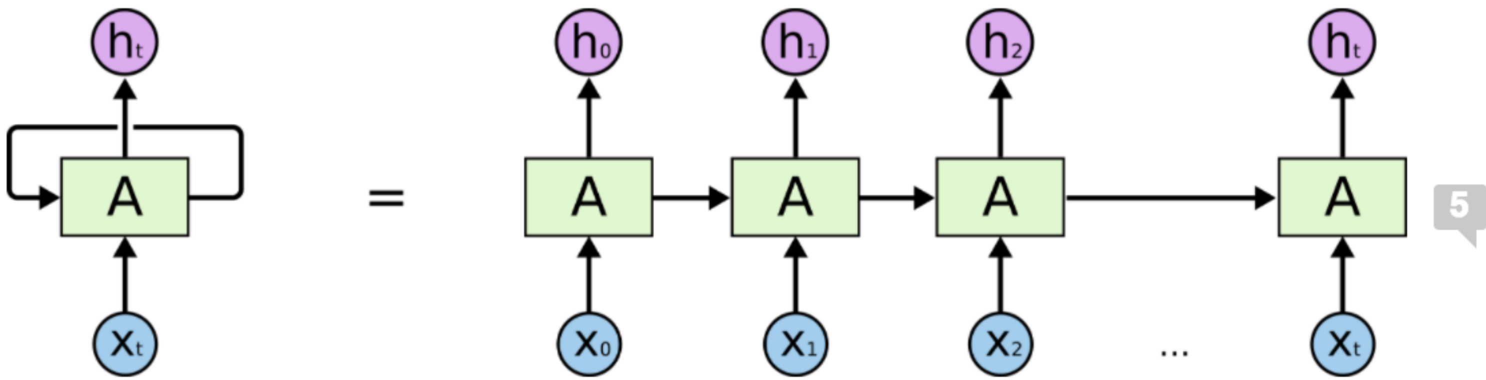
提纲

- 1. 递归神经网络RNN
- **2. 升级版RNN：LSTM**
- 3. 语言处理特征提取：Word2Vec
- 4.实例：LSTM用于语言处理

升级版RNN：LSTM

- RNN局限

RNN展开

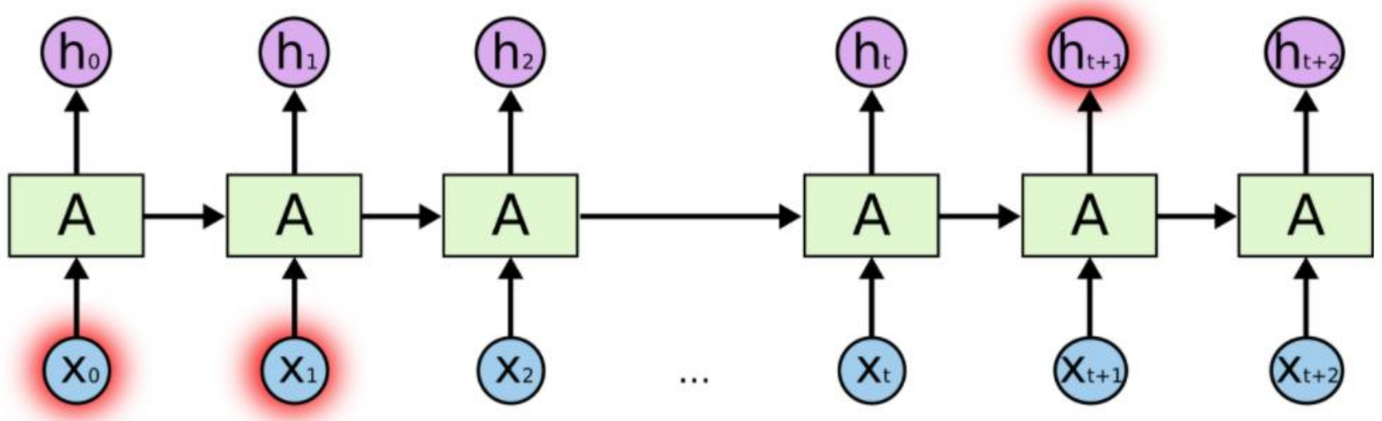


升级版RNN：LSTM

- RNN局限

前后依赖

I am from China, I speak Chinese.



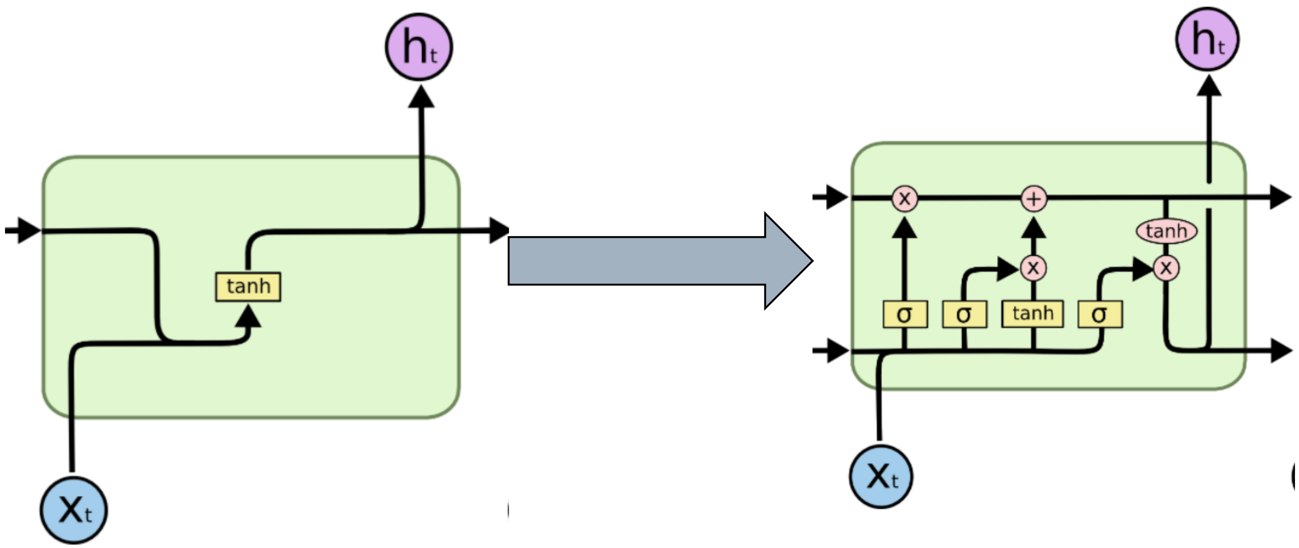
China->Chinese, 决定作用，距离太远难以产生关联

升级版RNN：LSTM

- RNN局限

解决方案－设计Gate，保存重要记忆

RNN　　　　　　　　　　　　　LSTM



升级版RNN：LSTM

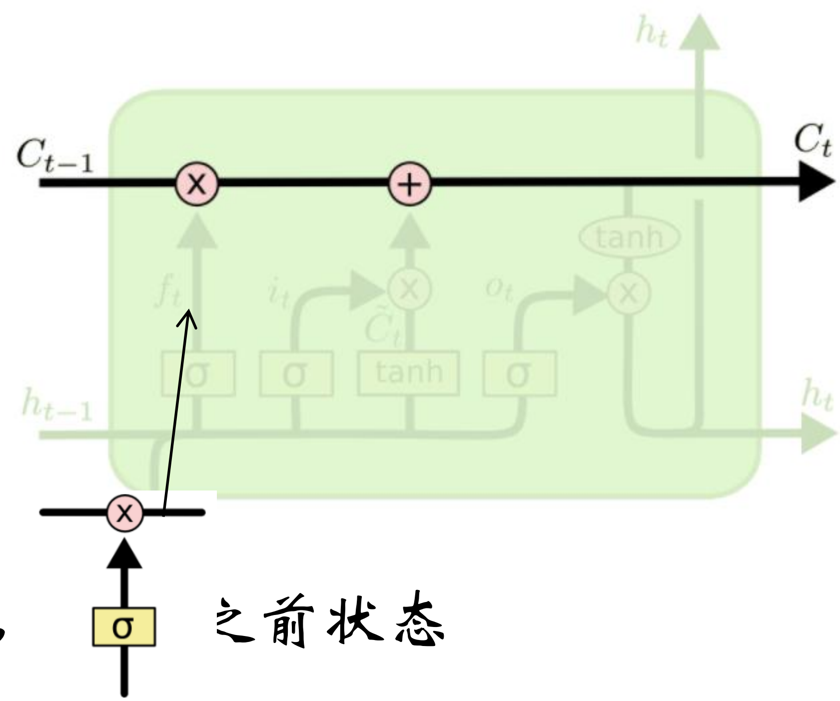
- LSTM形成
- 核心内容C_t

信息流控制的关键，参数决定了h_t传递过程中，那些被保存或舍弃。参数被Gate影响

怎样实现Gate对C影响？

Sigmoid函数系数决定C_t参数的变化

而Sigmoid函数决定于——输入，



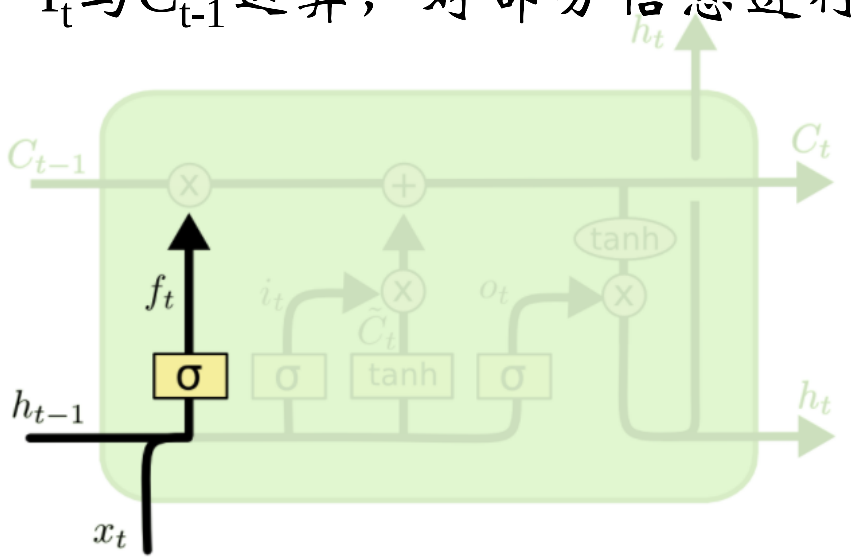
之前状态

升级版RNN：LSTM

- LSTM
- 分步分析LSTM原理

第一步：新输入x_t前状态h_{t-1}决定C哪些信息可以舍弃

f_t与C_{t-1}运算，对部分信息进行去除



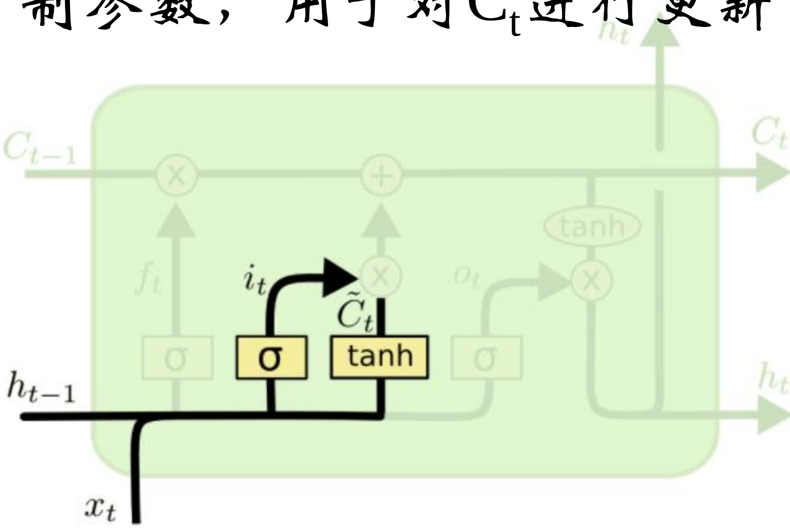
$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

升级版RNN：LSTM

- LSTM
- 分步分析LSTM原理

第二步：新输入xt前状态ht-1告诉C哪些新信息想要保存

i_t：新信息添加时的系数（对比f_t）$\hat{C}_t$ 单独新数据形成的控制参数，用于对C_t进行更新



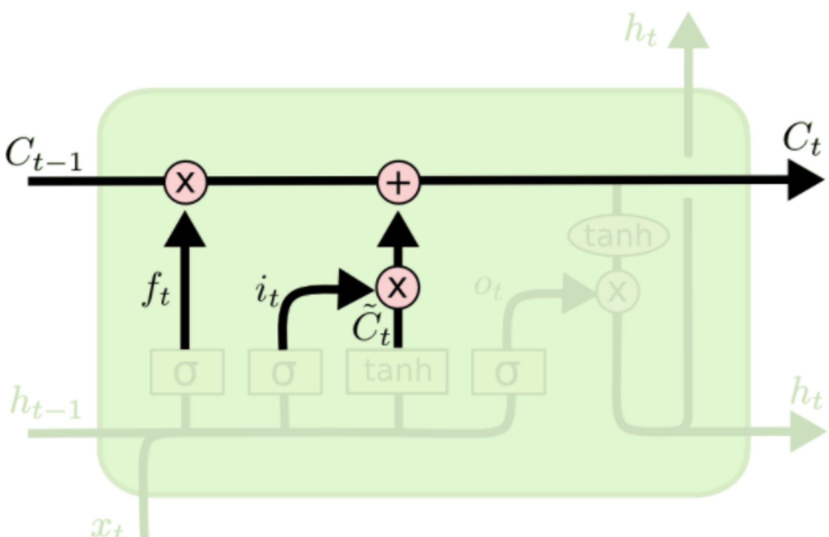
$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \ + \ b_i\right)$$

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

升级版RNN：LSTM

- LSTM
- 分步分析LSTM原理

第三步：根据旧的控制参数Ct-1，新生成的更新控制参数$\hat{C}_t$组合生成最终生成该时刻最终控制参数：

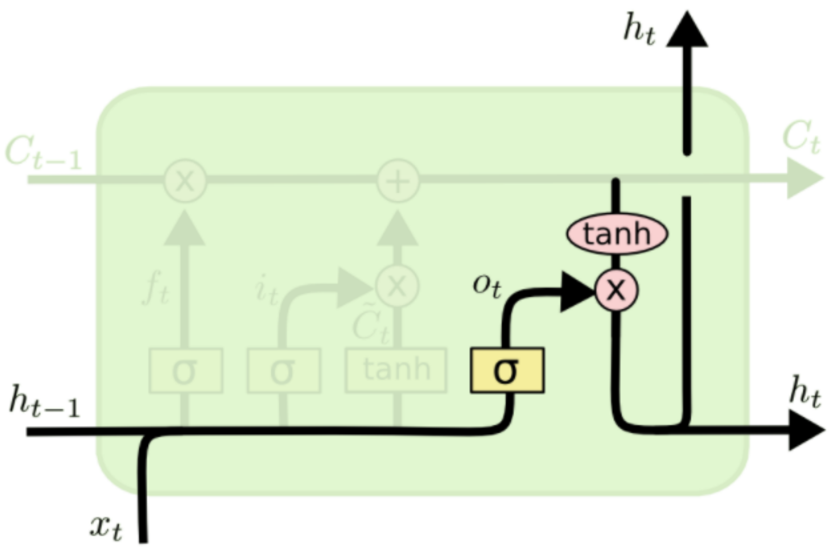


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

升级版RNN：LSTM

- LSTM
- 分步分析LSTM原理

第四步：根据控制参数Ct产生此刻的新的LSTM输出：



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

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

升级版RNN：LSTM

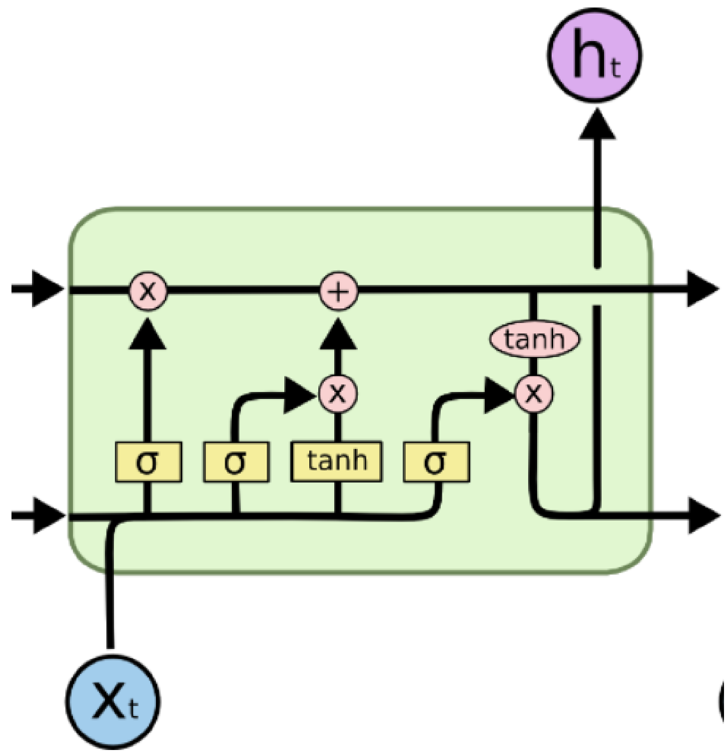
- LSTM
- 分步分析LSTM原理

主要内容：

1. C_t信息舍弃
2. C_t局部生成
3. C_t更新
4. C_t运算

Gate作用在哪里？

有用的就信息如何保存？

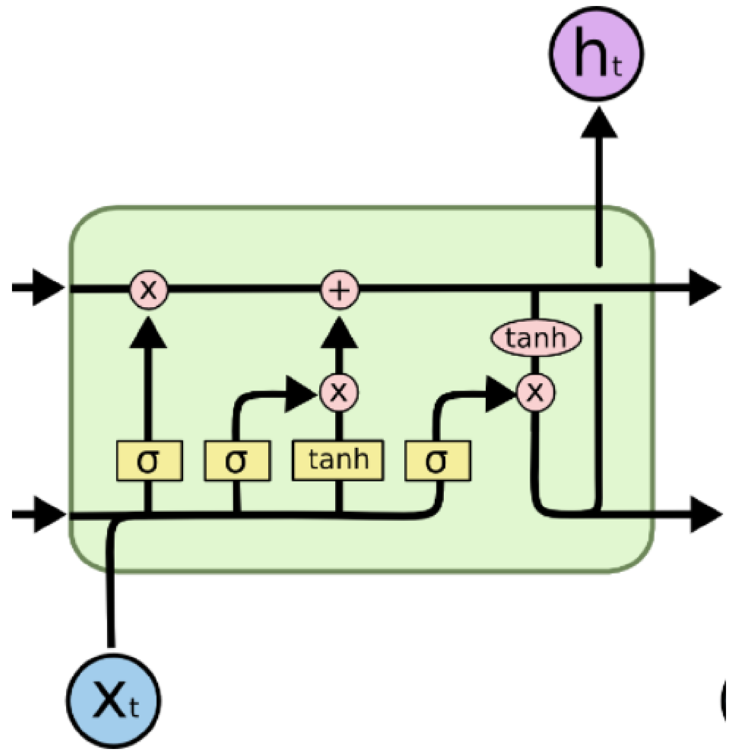


升级版RNN：LSTM

- LSTM
- 分步分析LSTM原理

主要内容：

1. C_t信息舍弃
2. C_t局部生成
3. C_t更新
4. C_t运算



Gate作用在哪里？ Gate输出 i_t，f_t，o_t 引导Ct生成

有用的就信息如何保存？训练后，C_t 相关参数为1

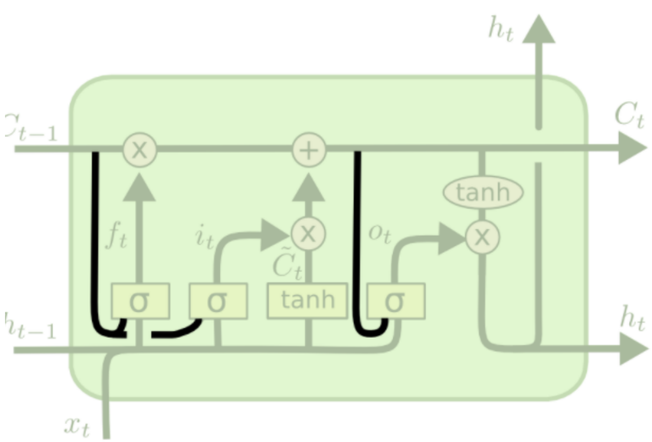
小象学院 ChinaHadoop.cn

升级版RNN：LSTM

- LSTM
- LSTM变种

1.Peephole connection

正常：Ct受到Gate参数影响→二者相互影响



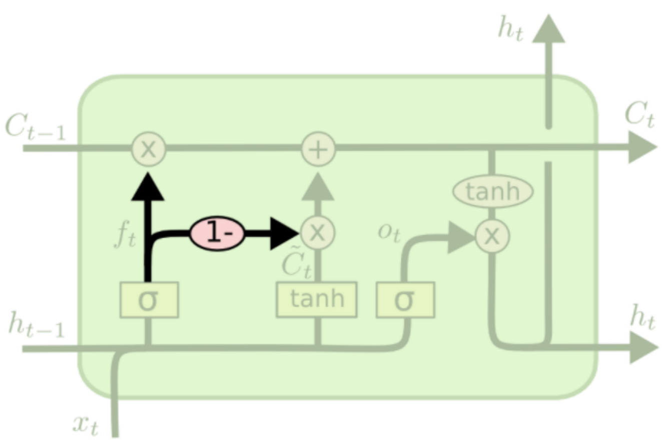
$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] + b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] + b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] + b_o\right)$$

升级版RNN：LSTM

- LSTM
- LSTM变种

1. Gate忘记／更新不再独立

正常：第一二步互不影响→遗忘／更新部分互为补充



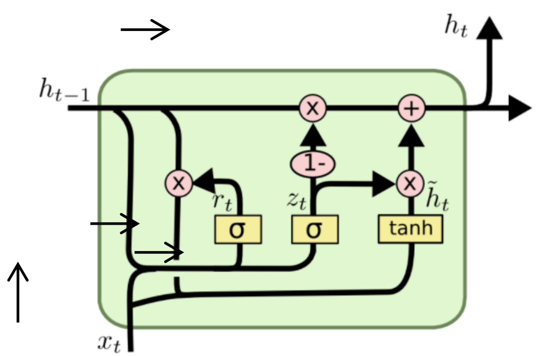
$$C_t = f_t * C_{t-1} + (1 - f_t) * \tilde{C}_t$$

升级版RNN：LSTM

- LSTM – GRU (Gated Recurrent Unit)
- 重要变种 GRU

1. 遗忘，更新Gate结合（不是独立，不是互补）
2. 控制参数C_t与输出h_t结合，直接产生带有长短记忆能力的输出 link: http://wiseodd.github.io/techblog/2016/08/12/lstm-backprop/ (code)

（反向推导）http://r2rt.com/written-memories-understanding-deriving-and-extending-the-lstm.html



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

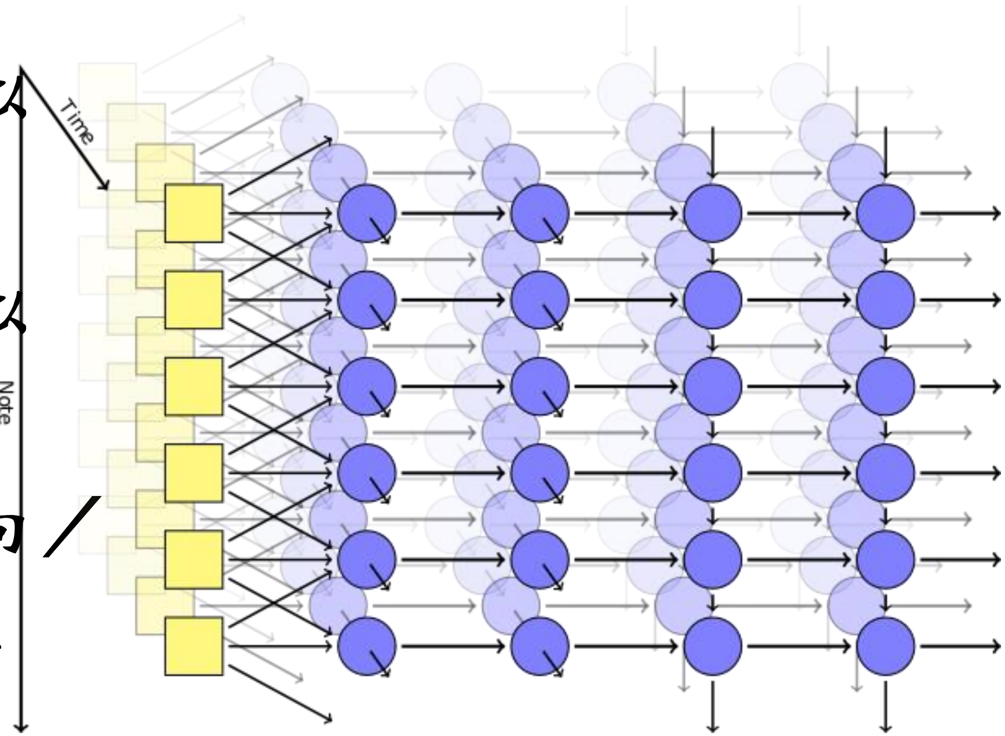
更新

遗忘

临时控制参数变形

升级版RNN：LSTM

- LSTM工作方式



- 结构设计上：可以认为是NN进行设计
- 中间层的特征可以最终输出或所有输出
- 额外参数：单双向／梯度上限／梯度计算范围

提纲

- 1. 递归神经网络RNN
- 2. 升级版RNN：LSTM
- **3. 语言处理特征提取：Word2Vec**
- 4.实例：LSTM用于语言处理

语言处理特征提取：Word2Vec

- 语言文本信息的表达形式

字符串形式难以直接理解

- 机器学习的输入输出数据形式

向量，多维数组

语言处理特征提取：Word2Vec

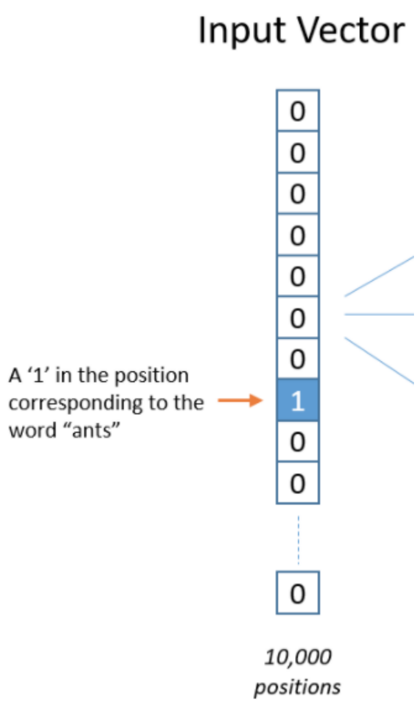
- Word2Vec
- 1. 建立字典，每个词生成one – hot向量

Word个数为n，产生n维向量

第i个word的向量为

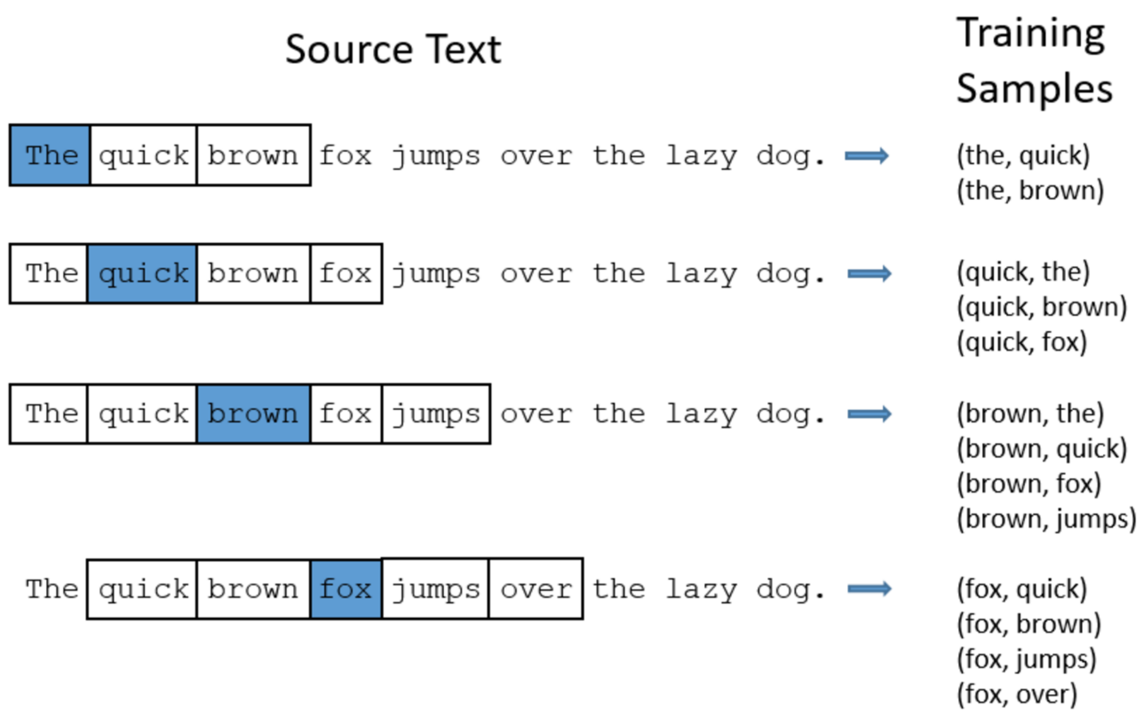
（0, 0,.... 1, 0, 0, 0, 0）

第i个位置



语言处理特征提取：Word2Vec

- Word2Vec
- 2. 训练数据集构建



语言处理特征提取：Word2Vec

- Word2Vec
- 3. 简单神经网络

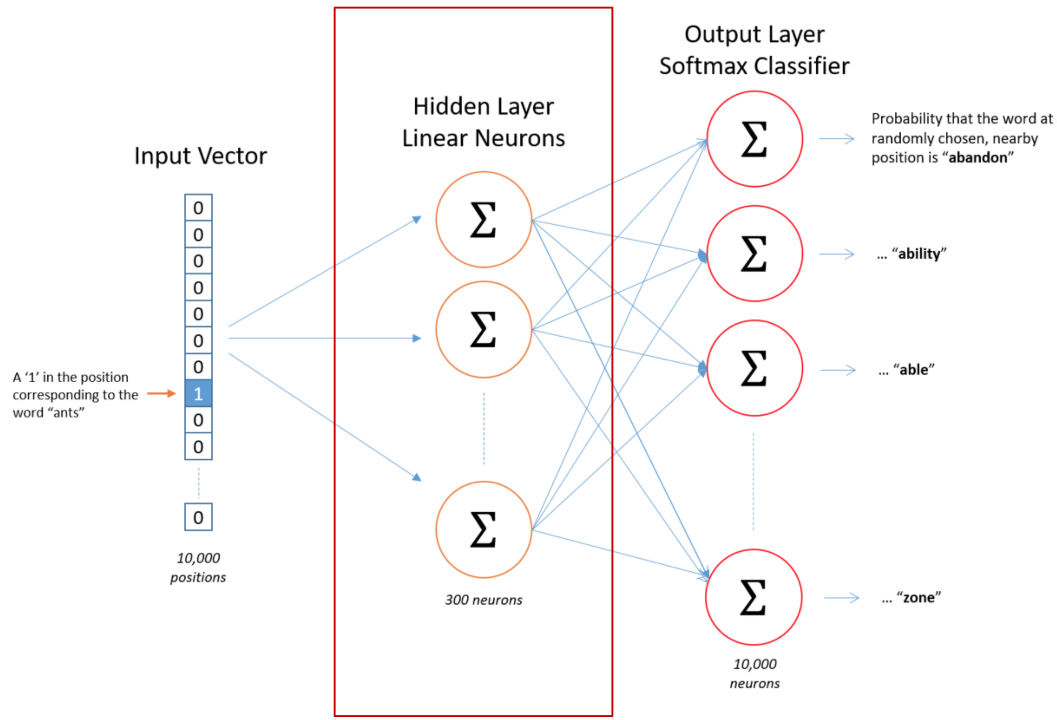
简易三层神经网络

各层神经元个数：

N-m-N

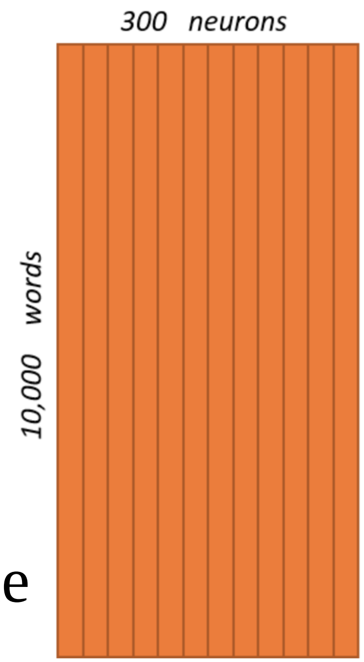
学的是词语映射到

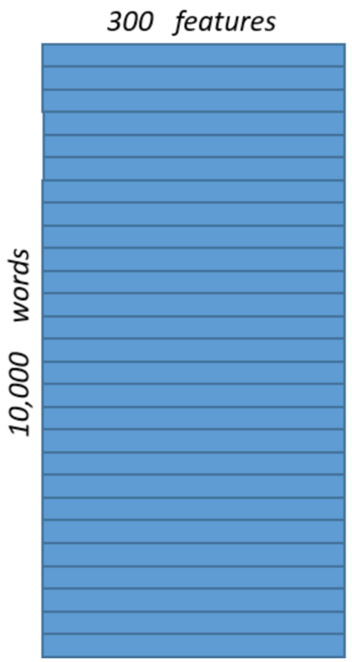
临近词的映射(无意义)



语言处理特征提取：Word2Vec

- Word2Vec
- 4. 生成最终Vect
- 训练model特征提取
- 每个one-hot对应一个300-d向量
- 生成最终look up word table





语言处理特征提取：Word2Vec

- Word2Vec
- 1. 建立字典，每个词生成one－hot向量
- 2. 根据文本训练数据构建映射关系用以训练
- 3. 构建简单神经网络，神经网络要做的是word到word的映射
- 4. 中间层特征提取，word2vec

语言处理特征提取：Word2Vec

- Word2Vec 特点

1. 利用上下文(context)进行学习

两个词上下文类似，生成的vector会接近

2. 具有类比特性

king-queen+female=male

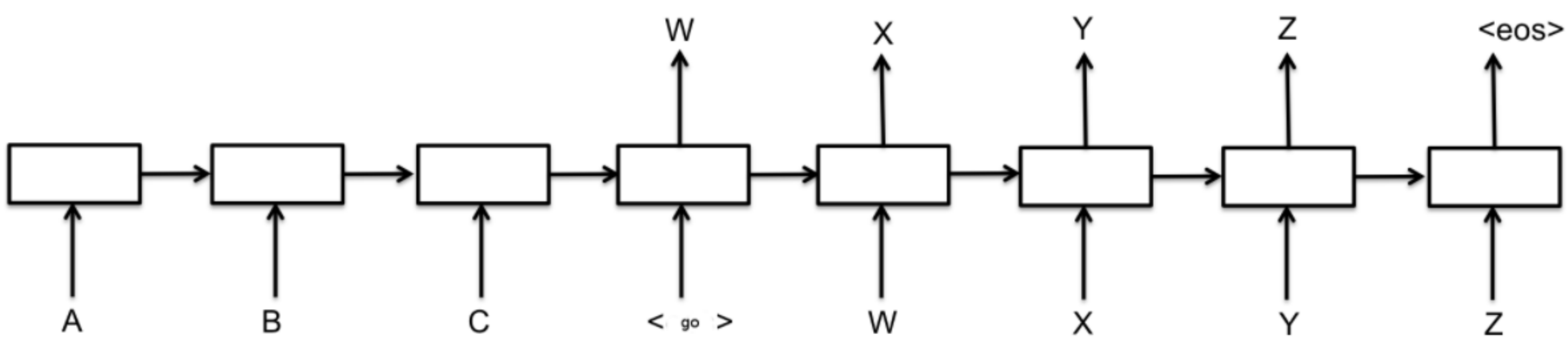
3. 字符→数据，方便机器学习处理

提纲

- 1. 递归神经网络RNN
- 2. 升级版RNN：LSTM
- 3. 语言处理特征提取：Word2Vec
- **4. 实例：LSTM用于语言处理**

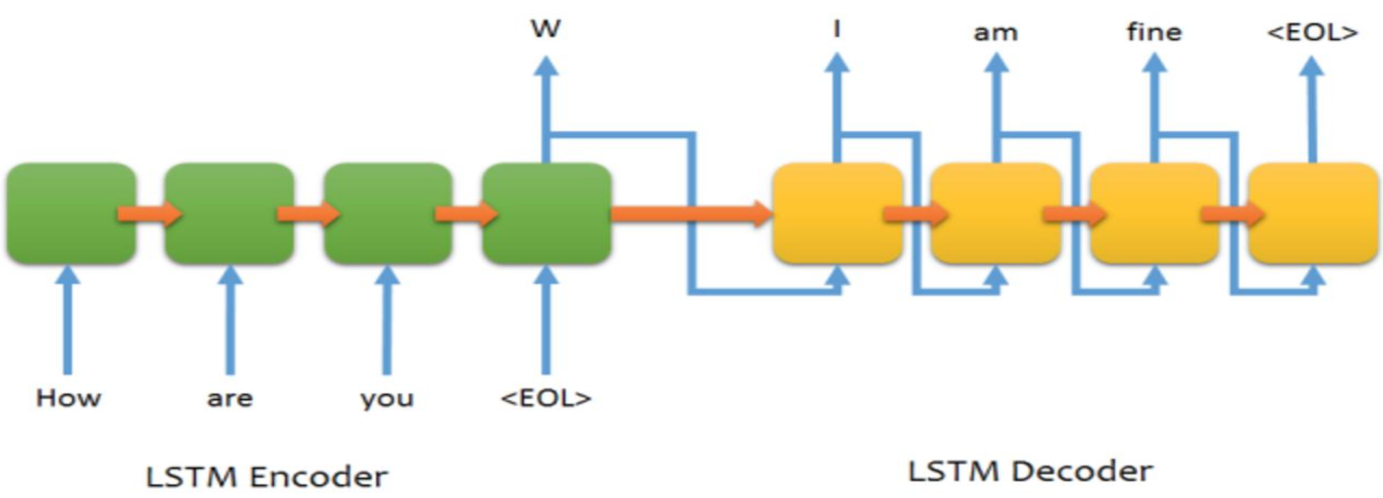
实例：LSTM用于语言处理

- LSTM语言生成
- 1. word形式：Word2Vec
- 2. 训练过程：words→word
- 3. LSTM网络只有最后输出有用



实例：LSTM用于语言处理

- LSTM语言生成
- 1. word形式：Word2Vec
- 2. 训练过程：words→word
- 3. LSTM网络只有最后输出有用



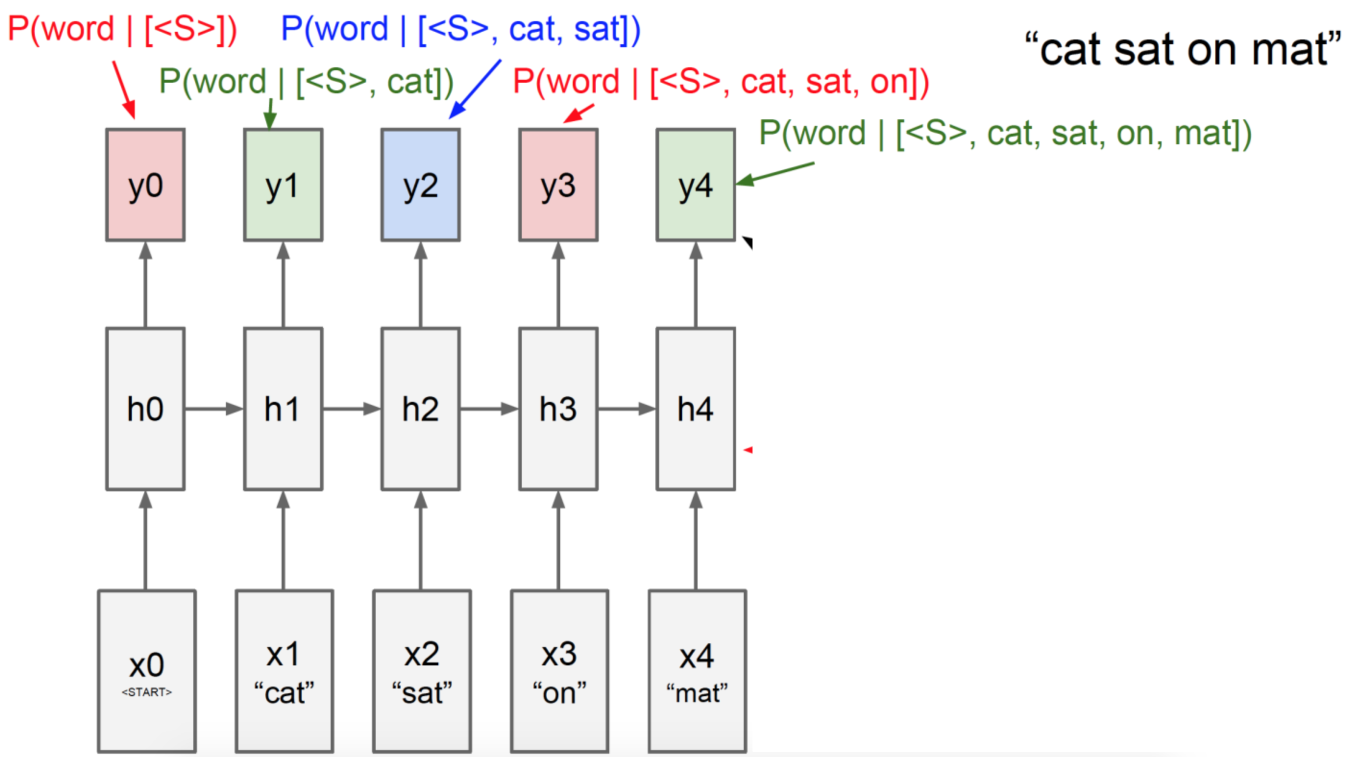
实例：LSTM用于语言处理

- LSTM语言生成
- 训练目标：生成单词间的条件概率
- 句子：cat sat on mat.

P(cat | [<S>])
P(sat | [<S>, cat])
P(on | [<S>, cat, sat])　　→ 1
P(mat | [<S>, cat, sat, on])

实例：LSTM用于语言处理

- LSTM语言生成
- 训练过程



总结

- 1. 递归神经网络RNN原理
- 2. 升级版RNN：LSTM
- 3. 语言处理特征提取：Word2Vec
- 4. 实例：LSTM用于语言处理

总结

- 有问题请到课后交流区
  - 问题答疑： http://www.xxwenda.com/
    - 可邀请老师或者其他人回答问题
- 课堂QQ群438285995，微信群
- 讲师微博：weightlee03，每周不定期分享DL资料
- GitHub ID：wiibrew （课程代码发布）

  https://github.com/wiibrew/DeepLearningCourseCodes