2021-2022学年秋季学期

自然语言处理

Natural Language Processing

授课教师： 胡玥

助　教： 李运鹏

自然语言处理
Natural Language Processing

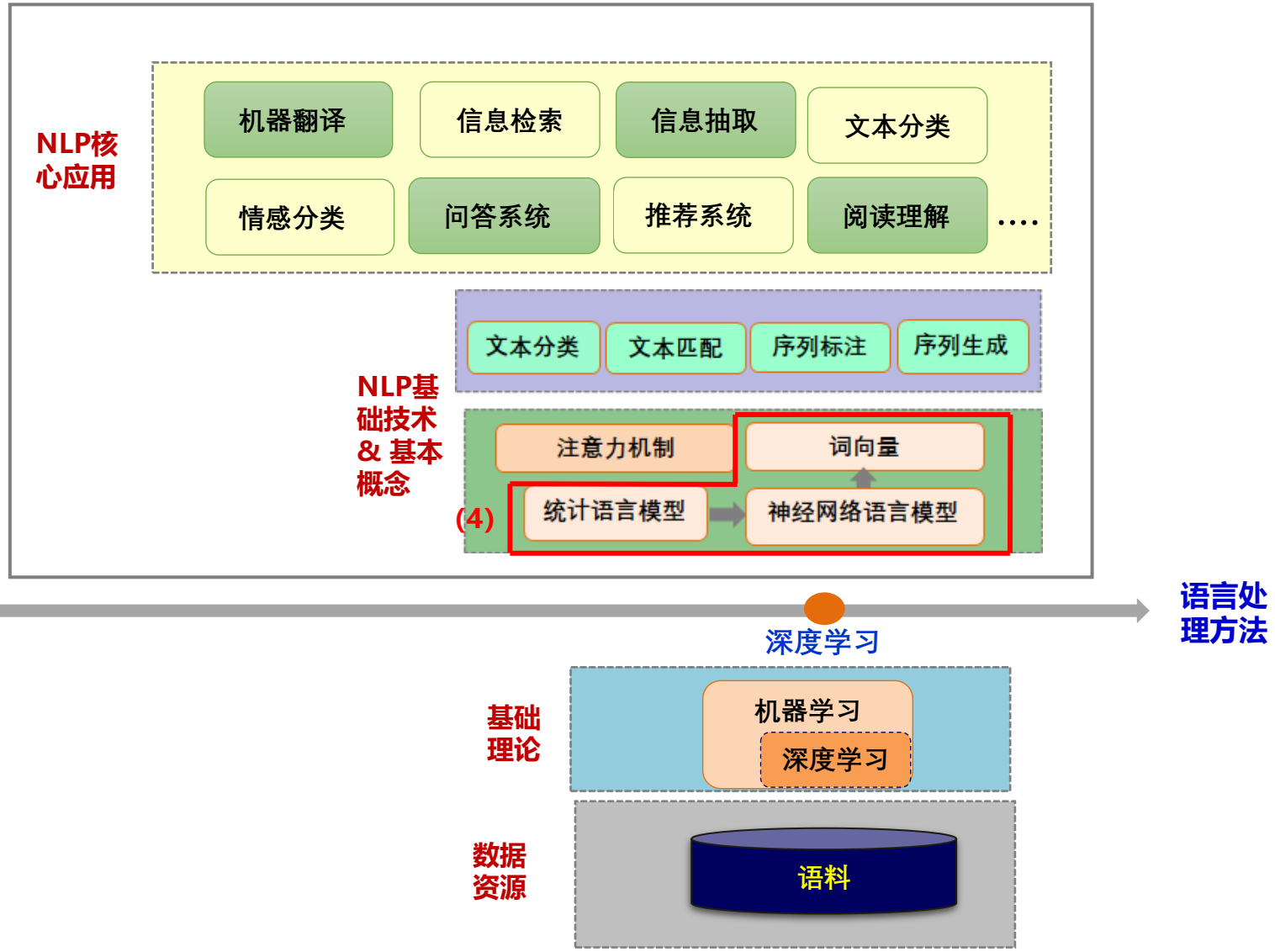
第 4 章　语言模型+词向量

授课教师：　胡玥

授课时间：2021.9

课程编码 201M4005H　课程名称 自然语言处理　授课团队名单 胡玥 、李运鹏

基于深度学习的自然语言处理课程内容



概 要

本章主要内容：介绍统计语言模型的基本概念，模型参数的学习方法，模型评价指标 和 语言模型的用途，并简要介绍几种语言模型的变形形式

本章教学目的：掌握语言模型的基本概念，理解其在语言处理中的作用，了解语言模型的变形形式：前向-后向语言模型，Skipping 语言模型

内 容 提 要

4.1　统计语言模型

4.2　神经语言模型

4.3　词向量（浅层）

4.1 统计语言模型

■ 统计语言模型

本节内容：

1. 语言模型基本概念
2. 语言模型参数估计
3. 参数的数据平滑
4. 语言模型性能评价
5. 语言模型应用

1. 语言模型基本概念

问题引入： （语音识别问题）

下表中，给定拼音串，如何确定对应的文字？

拼音串（无声调）	ni xian zai zai gan shen mo
候选字串	你 线 在 再 干 什 么
	你 现 在 在 干 什 么
	尼 先 在 在 感 什 么
	……
候选词串	你 现在 在 感什么
	你 现在 在 干什么
	你 先在 再 干什么
	……
正确文字串	你现在在干什么

如何判断语句合理性？

传统规则法：

句子是否合乎语法、语义（语言学分析）

问题：

判断过程复杂

其他方式？

1. 语言模型基本概念

弗莱德里克·贾里尼克

语言模型提出

弗莱德里克·贾里尼克（美国工程院院士）

提出了用数学的方法描述语言规律（语言模型）

语言模型基本思想：

用句子$S=w_1,w_2,\cdots,w_n$ 的概率 p(S) 刻画句子的合理性

（统计自然语言处理的基础模型）

对语句合理性判断：

规则法：判断是否合乎语法、语义（语言学定性分析）

统计法： 通过可能性（概率）的大小来判断（数学定量计算）

1. 语言模型基本概念

句子概率 p(S)定义：

语句 s = w_1 w_2 … w_n 的 概率p(S)定义为 ：

$$p(S)=p(w_1)p(w_2|w_1)...p(w_n|w_1,...,w_{n-1})$$

自然语言为上下文相关信息传递方式

$$=\prod_{i=1}^{n} p(w_i \mid w_1 \cdots\cdots w_{i-1})$$

其中：当 i=1 时，$p(w_1|w_0) = p(w_1)$

- **语言模型**

$$p(S) = \prod_{i=1}^{n} p(w_i \mid w_1 \cdots\cdots w_{i-1})$$

输入： 句子 S

输出： 句子概率p(S)

参数： $p(w_i|w_1,\cdots,w_{i-1})$

函数关系： $p(S) = \prod_{i=1}^{n} p(w_i \mid w_1 \cdots\cdots w_{i-1})$

1. 语言模型基本概念

说明：

(1) w_i 可以是字、词、短语或词类等等，称为统计基元。通常以“词”代之。

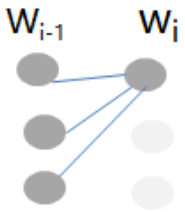
(2) w_i 的概率由 $w_1, \cdots, w_{i-1}$ 决定，由特定的一组$w_1, \cdots, w_{i-1}$ 构成的一个序列，称为 w_i 的历史（history）。

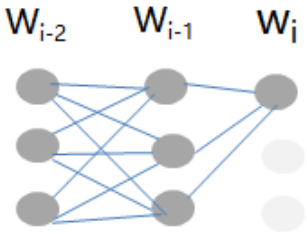
1. 语言模型基本概念

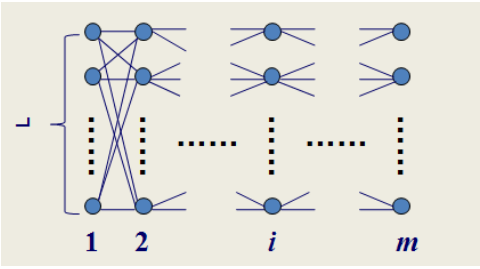
原始定义存在的问题：

对于$p(w_i|w_1,\cdots,w_{i-1})$：第 i（i >1）个统计基元，基元历史的个数为 i-1，如基元(如词汇表)有 L 个，i 基元就有 L^{i-1} 种不同的历史情况，模型有L^i个自由参数

如：$p(w_i|w_{i-1})$ 参数 3　　$p(w_i|w_{i-2}w_{i-1})$ 参数 9　　$p(w_m|w_1\cdots w_{m-1})$ 模型参数 L^m







如果 L=5000, m = 3, 自由参数的数目为 1250 亿！

一个汉语句子平均有22个词 — 参数太多

1. 语言模型基本概念

问题解决方法

减少历史基元的个数，马尔可夫方法：假设任意一个词 w_i出现的概率只与它前面的w_{i-1} 有关，问题得以简化

如：$p(S)=p(w_1)p(w_2|w_1)\ldots p(w_n|w_1,\ldots,w_{n-1})$

简化 $p(w_i|w_1,\ldots,w_{i-1}) \approx p(w_i|w_{i-1})$

$$p(s) = p(w_1)\times p(w_2|w_1)\times p(w_3|w_2)\times \ldots \times p(w_n|w_{n-1})$$

二元模型

1. 语言模型基本概念

n 元文法(n-gram)

n-gram 模型假设一个词的出现概率只与它前面的n-1个词相关，距离大于等于n的上文词会被忽略

即： $p(w_i|w_1,\ldots,w_{i-1}) \approx p(w_i|w_{i-(n-1)} \ldots,w_{i-1})$

- 1 元文法模型 (unigram) : $p(w_1,\cdots,w_m)=\prod_{i=1}^{m} p(w_i)$ w_i 独立于历史
- 2 元文法模型（bigram): $p(w_1,\cdots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-1})$ w_i 保留前1个词序
- 3 元文法模型（trigram ）:$p(w_1,\cdots,w_m)=\prod_{i=1}^{m} p(wi|w_{i-2}, w_{i-1})$ w_i 保留前2个词序
- ……
- n 元文法模型（n-gram）:$p(w_1,\cdots,w_m)=\prod_{i=1}^{m} p(wi|w_{i-(n-1)} \ldots w_{i-1})$ w_i 保留前n个词序

n-gram 就是对 $p(w_i|w_1,\cdots,w_{i-1})$的简化程度而定义

1. 语言模型基本概念

- 理论上N 越大越好，但再高阶模型也无法覆盖所有的语言现象
- N 越大，需要估计的参数越多
- **经验值：**

  tri-gram（3-gram）用的最多；

  four-gram（4-gram）以上需要太多的参数，少用。

1. 语言模型基本概念

例： 给定句子：John read a book 求 概率

解： 增加标记：<BOS> John read a book <EOS>

基于1元文法的概率为：

$$p(\text{John read a book}) = p(\text{John}) \times p(\text{read}) \times p(\text{a}) \times P(\text{book})$$

基于2元文法的概率为：

$$p(\text{John read a book}) = p(\text{John}|\text{<BOS>}) \times p(\text{read}|\text{John}) \times p(\text{a}|\text{read}) \times p(\text{book}|\text{a}) \times p(\text{<EOS>}|\text{book})$$

问题：如何获得 n 元语法模型中的各概率值(参数)?

2. 语言模型参数估计

任务：获得语言模型中所有词的条件概率（语言模型参数）

即：词表中词 W_i 的条件概论 $p(w_i|w_1,\ldots,w_{i-1})$

- 1 元文法模型 (unigram) : $p(w_i)$
- 2 元文法模型（bigram)：$p(w_i|w_{i-1})$
- 3 元文法模型（trigram : $p(wi|w_{i-2}, w_{i-1})$
- ……
- n 元文法模型（n-gram: $p(wi|w_{i-(n-1)} \ldots w_{i-1})$

2. 语言模型参数估计

参数估计

(1) 训练语料:

语言模型对于训练文本的类型、主题和风格等都十分敏感

语料选择:

- 训练语料应尽量和应用领域一致
- 语料尽量足够大
- 训练前应预处理

2. 语言模型参数估计

(2) 参数学习的方法

采用最大似然估计(maximum likelihood Evaluation, MLE)方法

对于 n-gram，参数 $p(w_i \mid w_{i-n+1}^{i-1})$

$$p(w_i \mid w_{i-n+1}^{i-1}) = f(w_i \mid w_{i-n+1}^{i-1}) = \frac{\sum_{w_i} c(w_{i-n+1}^{i})}{\sum_{w_i} c(w_{i-n+1}^{i-1})}$$

其中：

$\sum_{w_i} c(w_{i-n+1}^{i-1})$ 是历史串 w_{i-n+1}^{i-1} 在给定语料中出现的次数

$\sum_{w_i} c(w_{i-n+1}^{i})$，为 w_{i-n+1}^{i-1} 与 w_i 同现的次数

2. 语言模型参数估计

例1：给定训练语料：

“John read Moby Dick”

“Mary read a different book”

“She read a book by Cher”

如何 训练 2 元文法

解： 语料处理

<BOS>John read Moby Dick<EOS>

<BOS>Mary read a different book<EOS>

<BOS>She read a book by Cher<EOS>

2. 语言模型参数估计

如：以 p(John|<BOS>)，p(read|John)，p(a|read)， p(book|a)，p(<EOS>|book)

为例用最大似然估计求参数

$$p(John|<BOS>)=\frac{c(<BOS>\ John)}{\sum_w c(<BOS>\ w)}=\frac{1}{3} \qquad p(read\mid John)=\frac{c(John\quad read)}{\sum_w c(John\quad w)}=\frac{1}{1}$$

$$p(a\mid read)=\frac{c(read\quad a)}{\sum_w c(read\quad w)}=\frac{2}{3} \qquad p(book\mid a)=\frac{c(a\quad book)}{\sum_w c(a\quad w)}=\frac{1}{2}$$

$$p(<EOS>|book)=\frac{c(book\quad <EOS>)}{\sum_w c(book\quad w)}=\frac{1}{2}$$

语料：

<BOS>John read Moby Dick<EOS>

<BOS>Mary read a different book<EOS>

<BOS>She read a book by Cher<EOS>

2. 语言模型参数估计

例2： 求句子 *John read a book* 的概率 （2元文法）

解： p(John read a book) = p(John|<BOS>)×p(read|John)×p(a|read)×
p(book|a)×p(<EOS>|book)

$$P(\textit{John read a book}) = \frac{1}{3}\times 1\times\frac{2}{3}\times\frac{1}{2}\times\frac{1}{2}\approx 0.06$$

2. 语言模型参数估计

例3： 求： $p(Cher\ read\ a\ book) = ?$

解： $p(Cher\ read\ a\ book) = p(Cher|<BOS>) \times p(read|Cher) \times$

$p(a|read) \times p(book|a) \times p(<EOS>|book)$

$$p(Cher|<BOS>) = \frac{c(<BOS>\ Cher)}{\sum_w c(<BOS>\ w)} = \frac{0}{3}$$

$$p(read|Cher) = \frac{c(Cher\quad read)}{\sum_w c(Cher\quad w)} = \frac{0}{1}$$

于是： $p(Cher\ read\ a\ book) = 0$ ❓

问题：数据匮乏(稀疏)引起零概率问题

⇩

数据平滑

<BOS>John read Moby Dick<EOS>

<BOS>Mary read a different book<EOS>

<BOS>She read a book by Cher<EOS>

3. 参数的数据平滑

(1) 数据平滑的基本思想：

调整最大似然估计的概率值,使零概率增值，使非零概率下调，“劫富济贫”，消除零概率，改进模型的整体正确率。

基本目标：测试样本的语言模型困惑度越小越好。

基本约束： $\sum_{w_i} p(w_i \mid w_1, w_2, \cdots, w_{i-1}) = 1$

3. 参数的数据平滑

(2) 数据平滑方法：

- 加1法(Additive smoothing)
- 减值法/折扣法 (Discounting)

  1) Good-Turing　　2) Back-off (Katz)

  3) 绝对减值(H. Ney)　4) 线性减值

- 删除减值法：低阶代替高阶

……

3. 参数的数据平滑

◆ **加1法 (Additive smoothing)**

基本思想: 每一种情况出现的次数加1。

如，对于2-gram 有：

$$p(w_i \mid w_{i-1}) = \frac{1 + c(w_{i-1}w_i)}{\sum_{w_i}[1 + c(w_{i-1}w_i)]} = \frac{1 + c(w_{i-1}w_i)}{|V| + \sum_{w_i} c(w_{i-1}w_i)}$$

其中，V 为被考虑语料的词汇量（全部可能的基元数）

3. 参数的数据平滑

- **减值法/折扣法 (Discounting)**

基本思想：修改训练样本中事件的实际计数，使样本中(实际出现的)不同事件的概率之和小于1，剩余的概率量分配给未见概率。

1) Good-Turing　　2) Back-off (Katz)

3) 绝对减值(H. Ney)　　4) 线性减值

- **删除插值法(Deleted interpolation)**

基本思想：用低阶语法估计高阶语法，即当 3-gram 的值不能从训练数据中准确估计时，用 2-gram 来替代，同样，当 2-gram 的值不能从训练语料中准确估计时，可以用 1-gram 的值来代替。插值公式：

$$p(w_3 \mid w_1 w_2) = \lambda_3 p'(w_3 \mid w_1 w_2) + \lambda_2 p'(w_3 \mid w_2) + \lambda_1 p'(w_3)$$

其中，$\lambda_1 + \lambda_2 + \lambda_3 = 1$

（详情略）

3. 参数的数据平滑

例3： 求： $p(Cher\ read\ a\ book) = ?$

$$p(Cher\ read\ a\ book) = p(Cher|<BOS>)\times p(read|Cher)\times$$
$$p(a|read)\times p(book|a)\times p(<EOS>|book)$$

解： **采用加1法进行平滑处理**

原来：

$p(Cher|<BOS>) = 0/3$
$p(read|Cher) = 0/1$
$p(a|read) = 2/3$
$p(book|a) = 1/2$
$p(<EOS>|book)= 1/2$

平滑以后：

$p(Cher|<BOS>) = (0+1)/(11+3) = 1/14$
$p(read|Cher) = (0+1)/(11+1) = 1/12$
$p(a|read) = (1+2)/(11+3) = 3/14$
$p(book|a) = (1+1)/(11+2) = 2/13$
$p(<EOS>|book)= (1+1)/(11+2) = 2/13$

词汇量：
|V| = 11

$$p(Cher\ read\ a\ book) = p(Cher|<BOS>)\times p(read|Cher)\times$$
$$p(a|read)\times p(book|a)\times p(<EOS>|book)$$
$$= \frac{1}{14}\times\frac{1}{12}\times\frac{3}{14}\times\frac{2}{13}\times\frac{2}{13}\approx 0.00003$$

3. 参数的数据平滑

对于句子 John read a book 也需数据数据平滑处理：

原来：

$p(John|<BOS>) = 1/3,$

$p(read|John) = 1/1,$

$p(a|read) = 2/3,$

$p(book|a) = 1/2,$

$p(<EOS>|book) = 1/2$

平滑后：

$p(John|<BOS>) = 2/14,$

$p(read|John) = 2/12,$

$p(a|read) = 3/14,$

$p(book|a) = 2/13,$

$p(<EOS>|book) = 2/13$

$$p(John\ read\ a\ book)$$

$$= p(John|<BOS>) \times p(read|John) \times p(a|read) \times p(book|a) \times p(<EOS>|book)$$

$$= \frac{2}{14} \times \frac{2}{12} \times \frac{3}{14} \times \frac{2}{13} \times \frac{2}{13} \approx 0.0001$$

经网络语言模型不需要数据平滑 **Why ?**

4. 语言模型性能评价

主要有两种评价方法：

1. 实用方法：

通过查看该模型在实际应用（如拼写检查、机器翻译）中的表现来评价，优点是直观、实用，缺点是缺乏针对性、不够客观。

2. 理论方法：

用模型的 迷惑度/困惑度/混乱度（preplexity）衡量。其基本思想是能给测试集赋予较高概率值（低困惑度）的语言模型较好

4. 语言模型性能评价

困惑度定义

平滑的 n-gram 模型句子的概率: $p(s)=\prod_{i=1}^{m+1} p(w_i \mid w_{i-n+1}^{i-1})$

假定测试语料 T 由 l_T 个句子构成 $(t_1, \cdots, t_{l_T})$

则整个测试集的概率为: $p(T)=\prod_{i=1}^{l_T} p(t_i)$

模型 $p(w_i \mid w_{i-n+1}^{i-1})$ 对于测试语料的交叉熵:

$$H_p(T)=-\frac{1}{W_T}\log_2 p(T)$$

其中，W_T 是测试文本 T 的词数。

模型 p 的 困惑度 $PP_p(T)$ 定义为:

$$PP_P(T)=2^{H_p(T)}$$

5. 语言模型应用

语言模型的用途

语言模型是自然语言处理中的基础模型，在自然语言处理各项任务中有着广泛的应用场景

早期统计语言模型主要用途

- 决定哪一个词序列的可能性更大
- 已知若干个词，预测下一个词
- ….

5. 语言模型应用

例1： 给定拼音串对应的汉字串？

给定拼音串：ta shi yan jiu sheng wu de

解： 列出所有可能的汉字串：

拼音串（无声调）	ta shi yan jiu sheng wu de
候选字串	踏 实 研 究 生 物 的
	他 实 验 救 生 物 的
	他 是 研 究 生 物 的
	他 使 烟 酒 生 雾 的
	……

5. 语言模型应用

使用 n-gram（2-gram）计算所有可能字串概率：

$p(CString_1) = p(踏实|<BOS>) \times p(研究|踏实) \times p(生物|研究) \times p(的|生物) \times p(<EOS>|的)$

$p(CString_2) = p(他|<BOS>) \times p(实验|他) \times p(救|实验) \times p(生物|救) \times p(的|生物) \times p(<EOS>|的)$

……

然后选择概率最大的字串作为结果

5. 语言模型应用

例2： 已知若干个词，预测下一个词

如，联想输入法：基于 n-gram 的智能狂拼、微软拼音输入法等

语言模型变种：

◆ 前向-后向语言模型

$$\text{sentence}(t_1, t_2, \ldots, t_N)$$

前向语言模型：

$$p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_1, t_2, \ldots, t_{k-1}).$$

后向语言模型：

$$p(t_1, t_2, \ldots, t_N) = \prod_{k=1}^{N} p(t_k \mid t_{k+1}, t_{k+2}, \ldots, t_N).$$

◆ K-Skipping N-gram Model

一个词的出现概率只与它前(后)面的距离为K的 n-1个词相关。核心思想是刻画远距离约束关系

◆ Class-based N-gram Mode

该方法基于词类建立语言模型，以缓解数据稀疏问题，且可以方便融合部分语法信息 如，n-pos 模型

语言模型变种：

◆ 指数语言模型

传统的n-gram语言模型，只是考虑了词形方面的特征，而没有词性以及语义层面上的知识最大熵模型MaxEnt、最大熵马尔科夫模型MEMM、条件随机域模型CRF可以更好的融入多种知识源，刻画语言序列特点，较好的用于解决序列标注问题。

◆ Topic-based N-gram Mode

该方法将训练集按主题划分成多个子集，并对每个子集分别建立N-gram语言模型，以解决语言模型的主题自适应问题。

◆ Cache-based N-gram Model

该方法利用cache缓存前一时刻的信息，以用于计算当前时刻概率，以解决语言模型动态自适应问题。

……

详情请参考相关资料

内 容 提 要

4.1　统计语言模型

4.2　神经语言模型

4.3　词向量（浅层）

4.2 神经语言模型

■ 神经语言模型

本节内容：

1. 神经语言模型概述
2. DNN语言模型（NNLM）
3. RNN语言模型（RNNLM）
4. RNN语言模型变形

1. 神经语言模型概述

■ 神经语言模型概念

语言模型

$$p(S) = \prod_{i=1}^{n} p(w_i \mid w_1 \cdots\cdots w_{i-1})$$

输入： 句子 S

输出： 句子概率p(S)

参数： $p(w_i|w_1,\cdots,w_{i-1})$

函数关系： $p(S) = \prod_{i=1}^{n} p(w_i \mid w_1 \cdots\cdots w_{i-1})$

对于语言模型参数 $p(w_i|w_1,\cdots,w_{i-1})$

统计语言模型： 用概率统计法学习参数

神经语言模型： 用神经网络学习参数

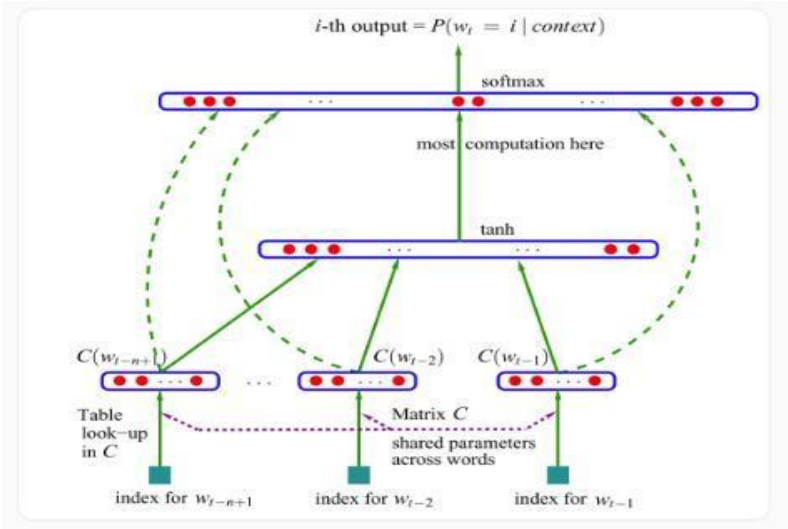
- 使用DNN 学习模型参数 ： NNLM 模型
- 使用RNN 学习模型参数 ： RNNLM 模型

1. 神经语言模型概述

神经网络语言模型提出

Bengio 等人2001年提出了第一个神经网络语言模型 (Neural language models)

它是一种前馈神经网络，该模型在学习语言模型的同时，也得到了词向量。



神经语言模型（Neural language models）— NLP 里程碑成果

Yoshua Bengio, et.al. A neural probabilistic language model（2001, 2003）

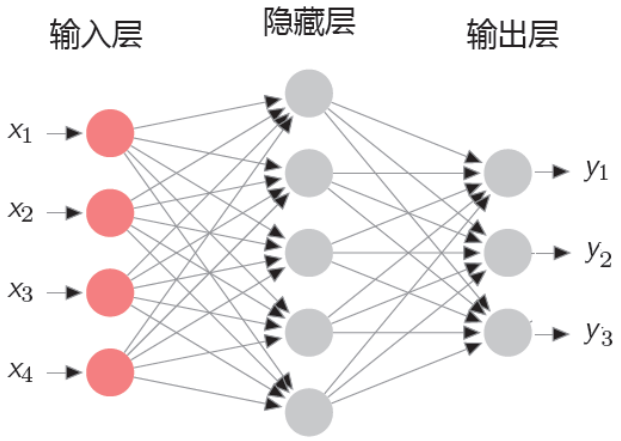
2. DNN语言模型（NNLM）

■ DNN语言模型（NNLM）

目标： **用神经网络DNN 学习语言模型** $p(S) = \prod_{i=1}^{n} p(w_i \mid w_1 \cdots\cdots w_{i-1})$

模型参数 $p(w_i | w_{i-(n-1)} \cdots w_{i-1})$

问题：如何用DNN学习模型参数?



2. DNN语言模型（NNLM/2-gram）

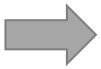
■ 2 元文法模型（bigram ）：　$p(w_1,\ldots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-1})$

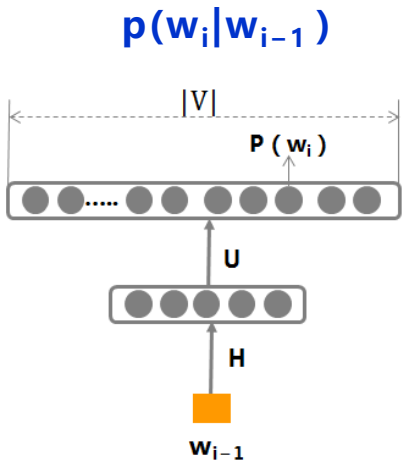
参数：　$p(w_i|w_{i-1})$

任务：用DNN学习　$p(w_i|w_{i-1})$

模型设计：

- 神经网络：DNN
- 输入：　w_{i-1}
- 输出：　$p(w_i)$



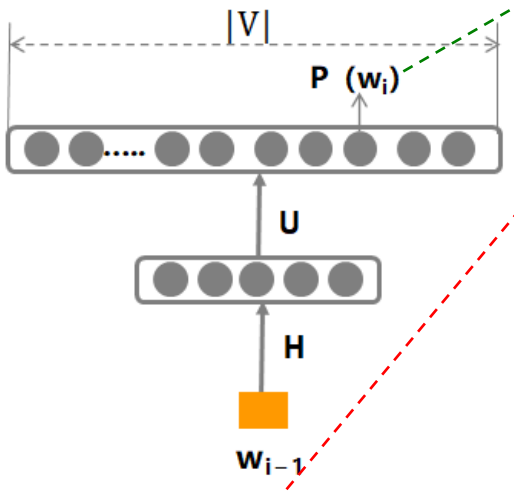


2. DNN语言模型（NNLM/2-gram）

（1）NNLM模型结构

语言模型参数

2-gram: $p(w_1,\ldots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-1})$



输出层：$p(w_i|w_{i-1})$
$=\text{softmax}\left(b^2+U\left(\tanh\left(XH+b^1\right)\right)\right)$

隐藏层：$h=\tanh(XH+b^1)$

输入层：X: 词 w_{i-1}

参数：$\theta=\{H, U, b^1, b^2\}$

神经网络参数

输出层有|V|个元素，V是有限词表包括未登录词标识UNK和句子开始和结束补齐符号，一般在10000≈1000000左右，常见规模 70000左右

2. DNN语言模型（NNLM/2-gram）

（2）NNLM模型学习 (2-gram)

◆ 语料：(“无监督”)

文本： $S=w_1, w_2, \cdots, w_n, \cdots$

实例：X： w_{i-1}

$\hat{Y}$ ： w_i

◆ 目标函数：

采用log损失函数 $L(Y, P(Y|X))=-logP(Y|X)$

对于整个语料而言，语言模型需要最大化：

$$\sum_{w_{i-1}} \sum_{i \in D} \log P(w_i \mid w_{i-1})$$

2. DNN语言模型（NNLM/2-gram）

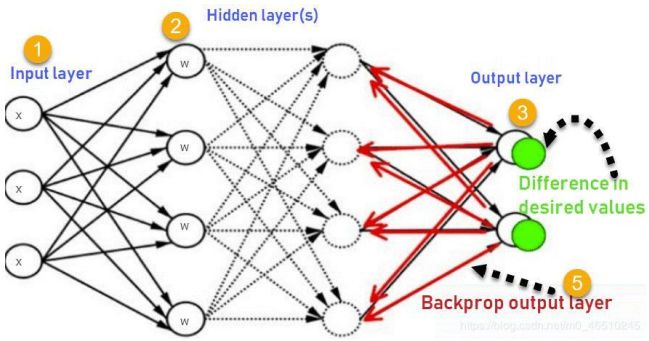
(2) NNLM模型学习 (2-gram)

◆ 参数训练：

(BP) 随机梯度下降法优化训练目标:
每次迭代，随机从语料D 中选取一段文本
$w_{i-(n-1)}, \cdots, w_i$作为训练样本进行一次梯度迭代

$$\theta \leftarrow \theta + \alpha \frac{\partial \log P(w_i \mid w_{i-1})}{\partial \theta}$$

其中，α学习率，$\theta = \{ H, U, b^1, b^2 \}$



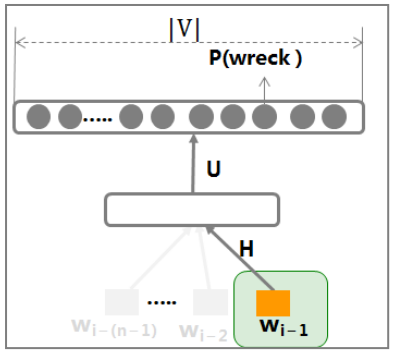
2. DNN语言模型（NNLM/2-gram）

（3）NNLM模型预测

例： P（“wreck a nice beach”）

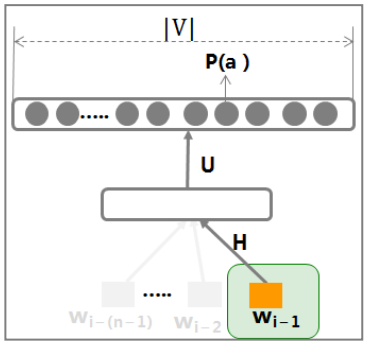
=P(wreck | START)P(a | wreck)P(nice | a)P(beach | nice)

P(wreck | START)



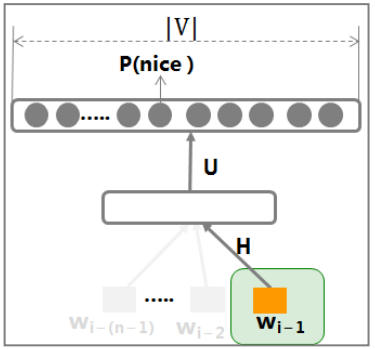
“START”

P(a | wreck)



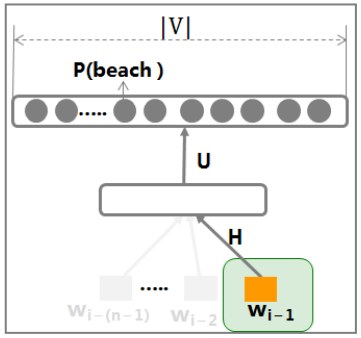
“wreck”

P(nice | a)



“a”

P(beach | nice)



“nice”

每求一个参数用一遍神经网络

2. DNN语言模型（NNLM/n-gram）

■ **n 元文法模型（n-gram ）：** $p(w_1,\cdots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-(n-1)} \ldots w_{i-1})$

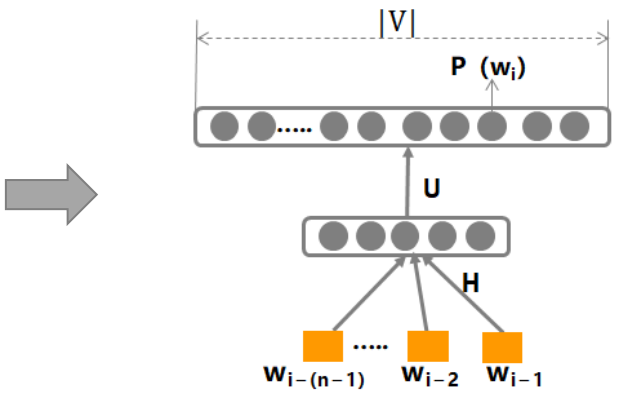
参数： $p(w_i|w_{i-(n-1)} \ldots w_{i-1})$

任务： 用DNN学习 $p(w_i|w_{i-(n-1)} \ldots w_{i-1})$

模型设计：

$p(w_i|w_{i-(n-1)} \ldots w_{i-1})$

- 神经网络：DNN
- 输入： $w_{i-(n-1)} \ldots w_{i-1}$
- 输出： $p(w_i)$

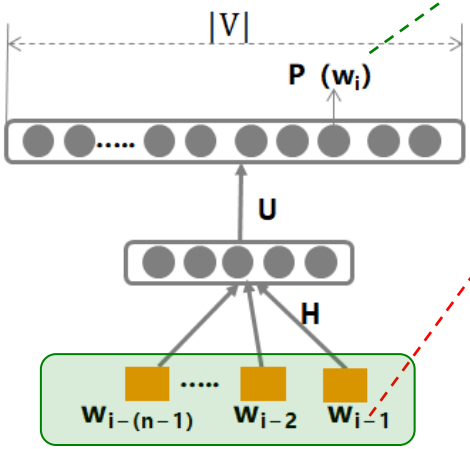


2. DNN语言模型（NNLM/n-gram）

（1）NNLM模型结构

语言模型参数

n-gram: $p(w_1,\ldots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-(n-1)}\ \ldots\ w_{i-1})$



词以什么形式输入网络
→ 词向量问题

输出层：$p(w_i|w_{i-(n-1)}\ \ldots\ w_{i-1})$
$=\mathrm{softmax}\left(b^2+U\left(\tanh\left(XH+b^1\right)\right)\right)$

隐藏层：$h=\tanh(XH+b^1)$

输入层：X: n-1个词 $w_{i-(n-1)},\cdots,w_{i-1}$

参数：$\theta=\{H, U, b^1, b^2\}$

神经网络参数

2. DNN语言模型（NNLM/n-gram）

（2）NNLM模型学习 (n-gram)

◆ 语料：(“无监督”)

文本： $S=w_1, w_2, \cdots, w_n$, …

实例： X： $w_1, w_2, \cdots, w_{i-1}$

$\hat{Y}$ ： w_i

◆ 目标函数：

采用log损失函数 $L(Y, P(Y|X))=-logP(Y|X)$

对于整个语料而言，语言模型需要最大化：

$$\sum_{w_{i-1}} \sum_{i \in D} \log P(w_i \mid w_{i-1})$$

2. DNN语言模型（NNLM/n-gram）

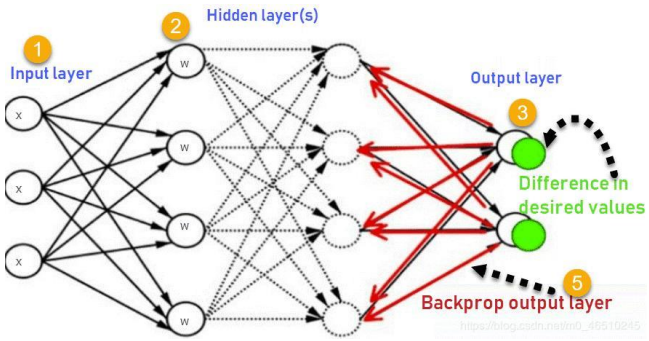
(2) NNLM模型学习 (n-gram)

◆ 参数训练：

(BP) 随机梯度下降法优化训练目标:
每次迭代，随机从语料D 中选取一段文本
$w_{i-(n-1)}, \cdots, w_i$作为训练样本进行一次梯度迭代

$$\theta \leftarrow \theta + \alpha \frac{\partial \log P(w_i \mid w_{i-1})}{\partial \theta}$$

其中，α学习率，$\theta = \{ H, U, b^1, b^2 \}$



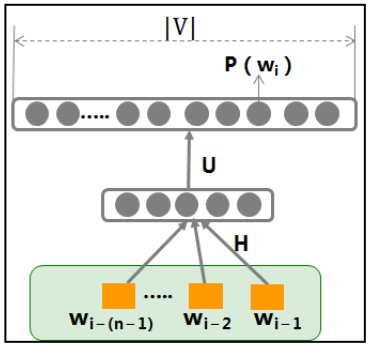
2. DNN语言模型（NNLM/n-gram）

(3) NNLM模型预测 （n-gram） 如 n=4

例： P（“wreck a very nice beach”）

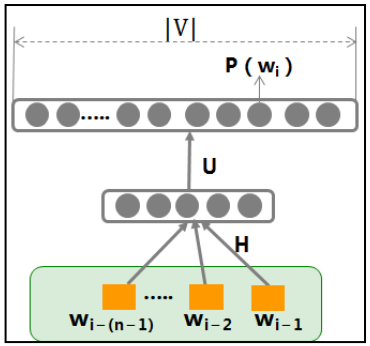
=P(very | START wreck a)P(nice | wreck a very) P(beach | a very nice)

P(very | START wreck a)



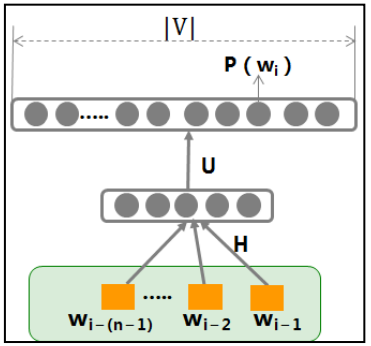
“START wreck a ”

P(nice | wreck a very)



“wreck a very ”

P(beach | a very nice)



“a very nice ”

每求一个参数用一遍神经网络

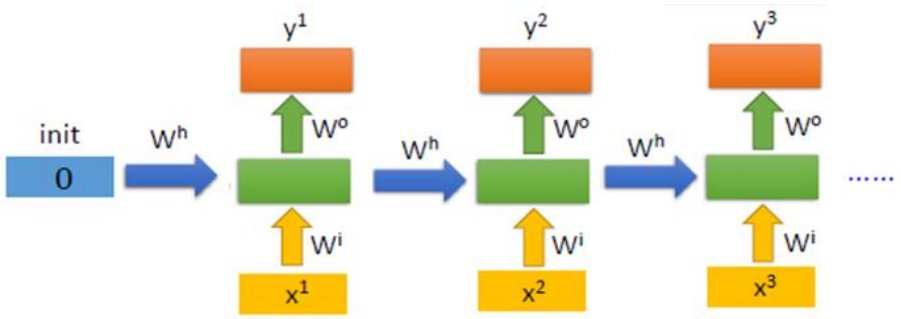
3. RNN语言模型（RNNLM）

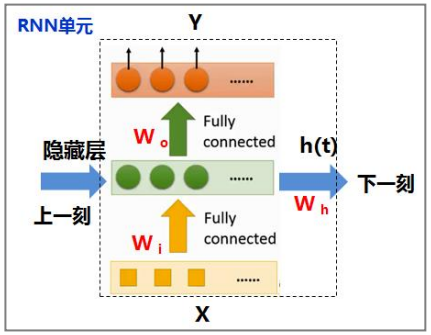
■ RNN语言模型（RNNLM）

目标： **用神经网络RNN 学习语言模型** $p(S) = \prod_{i=1}^{n} p(w_i \mid w_1 \cdots\cdots w_{i-1})$

模型参数 $p(w_i|w_{i-(n-1)} \ldots w_{i-1})$

问题：如何用RNN学习模型参数？





3. RNN语言模型（RNNLM）

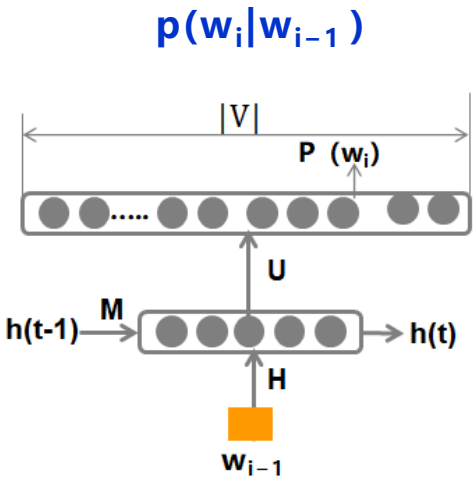
■ 2 元文法模型（bigram）：$p(w_1,\ldots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-1})$

参数： $p(w_i|w_{i-1})$

任务： 用RNN学习 $p(w_i|w_{i-1})$

模型设计：

- 神经网络：RNN
- 输入： w_{i-1}
- 输出： $p(w_i)$

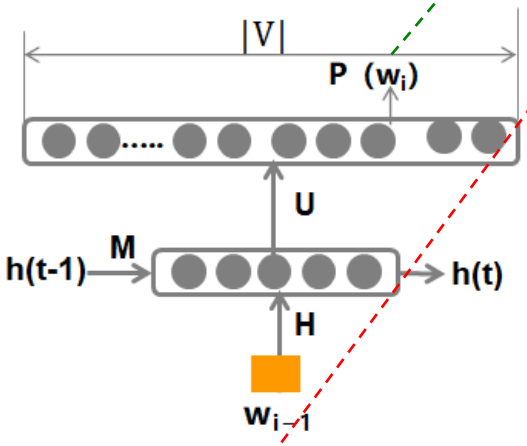


3. RNN语言模型（RNNLM）

(1) RNNLM模型结构

语言模型参数

2-gram: $p(w_1,\ldots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-1})$



输出层：$p(w_i|w_{i-1})$
$=\text{softmax}\ (b^2+U\ h(t))$

隐藏层：$h(t)=\tanh(XH+Mh(t-1)+b^1)$

输入层：X: 词 w_{i-1}

参数：$\theta=\{H, U, M, b^1, b^2\}$ 神经网络参数

3. RNN语言模型（RNNLM）

(2) RNNLM模型学习 (2-gram)

◆ 语料：(“无监督”)

文本： $S= w_1, w_2,..., w_n, \ldots\ldots$

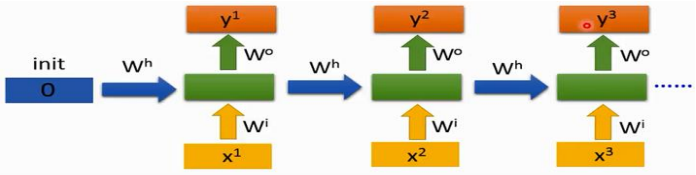
实例：X： START, $w_1, w_2,..., w_{n-1}$

$\hat{Y}$： $w_1, w_2,..., w_{n-1}\ w_n$

◆ 目标函数：

对于整个语料而言，语言模型需要最大化：

$$\sum_{w_{i-1}} \sum_{i \in D} \log P(w_i \mid w_{i-1})$$



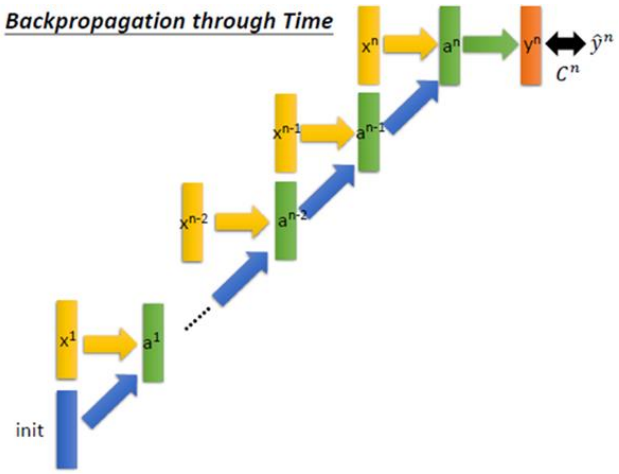
3. RNN语言模型（RNNLM）

(2) RNNLM模型学习 (2-gram)

◆ 参数训练：

(BPTT) 随机梯度下降法优化训练目标:

每次迭代，随机从语料D 中选取一段文本 $w_{i-(n-1)},\cdots,w_i$作为训练样本进行一次梯度迭代



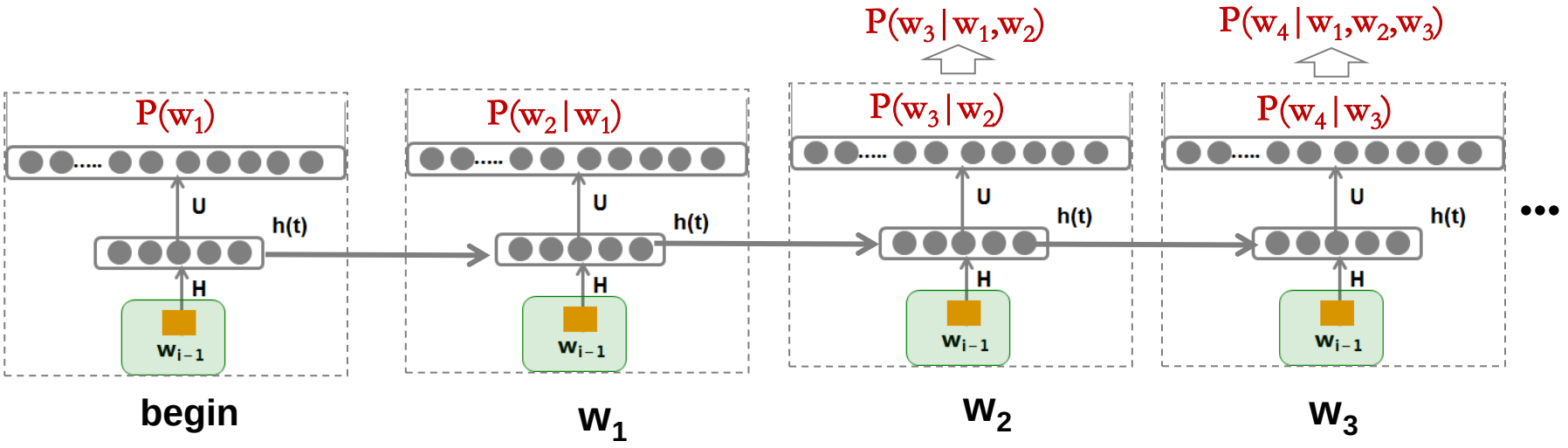
3. RNN语言模型（RNNLM）

(3) RNNLM模型预测

例：　$P(w_1, w_2, w_3, \cdots, w_n)$

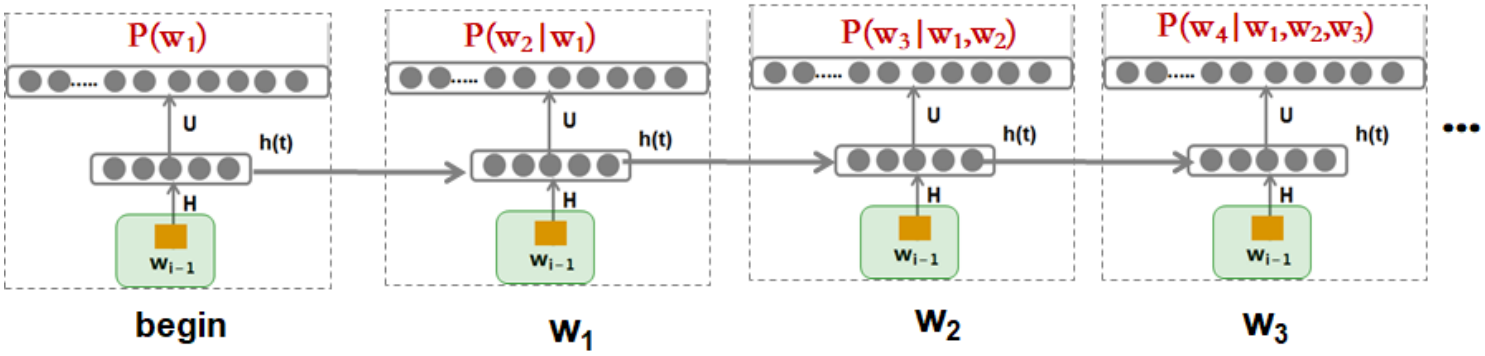
$$\text{2-gram}: P(w_1)P(w_1|w_2)P(w_3|w_2) \cdots\cdots P(w_n|\ w_{n-1})$$

$$= P(w_1)P(w_1|w_2)P(w_3|w_1,w_2) \cdots\cdots P(w_n|w_1,w_2 \cdots w_{n-1})$$



3. RNN语言模型（RNNLM）

(3) RNNLM模型预测



随着模型逐个读入语料中的词w_1;w_2 ….隐藏层不断地更新为h(1),h(2)…. ,通过这种迭代推进方式，每个隐藏层实际上包含了此前所有上文的信息，相比NNLM 只能采用上文*n* 元短语作为近似，RNNLM 包含了更丰富的上文信息，也有潜力达到更好的效果。

3. RNN语言模型（RNNLM）

RNNLM 优点：

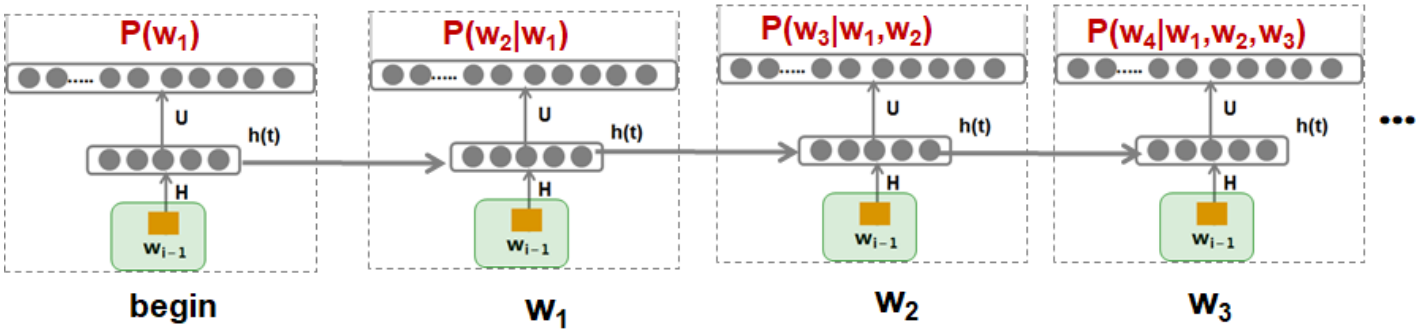
- RNNLM 模型可以保留每个词的全部历史信息，不需简化为n-gram
- 引入词向量作为输入后不需要数据平滑

在神经网络一般用RNN语言模型

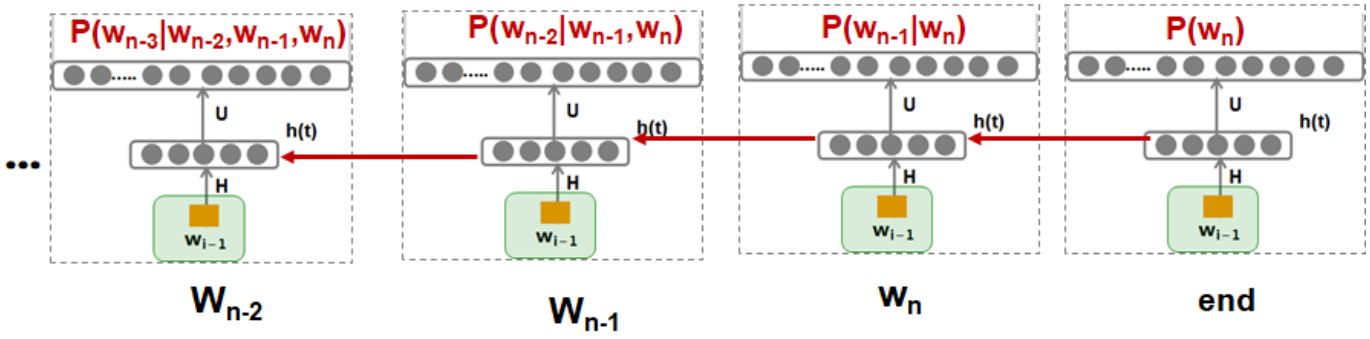
4. RNN语言模型变形

$P(w_1, w_2, w_3, \cdots, w_n)$

◆ 正向语言模型

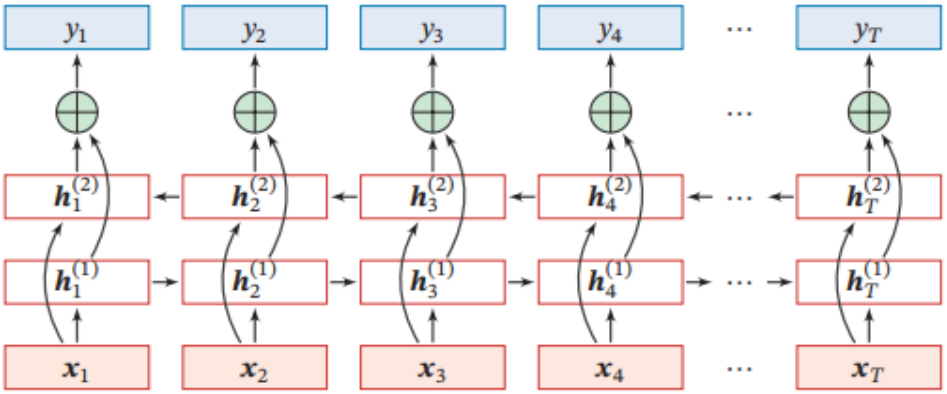


◆ 反向语言模型



4. RNN语言模型变形

◆ 双向语言模型



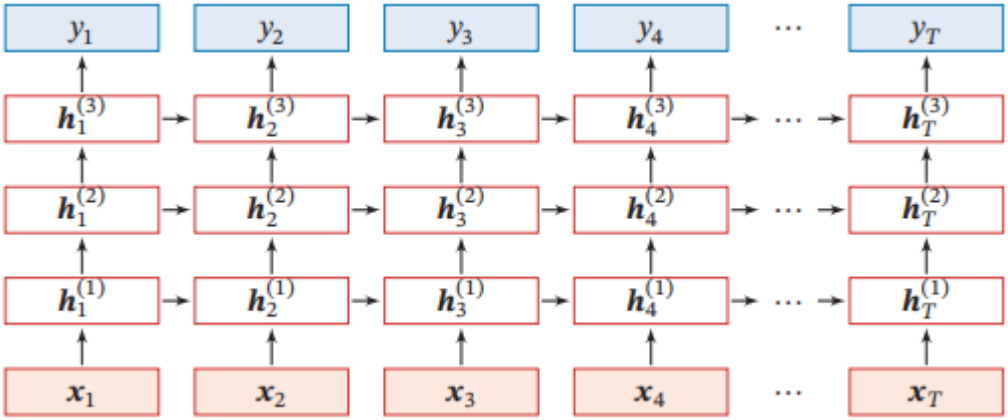
$$\vec{h}_t = f(\vec{W}x_t + \vec{V}\vec{h}_{t-1} + \vec{b})$$
$$\overleftarrow{h}_t = f(\overleftarrow{W}x_t + \overleftarrow{V}\overleftarrow{h}_{t+1} + \overleftarrow{b})$$
$$y_t = g(U[\vec{h}_t; \overleftarrow{h}_t] + c)$$

每个时刻都有一个正向输入的隐层 $\overrightarrow{ht}$ 和·一个反向输入隐层 $\overleftarrow{h_t}$

两个隐层分别可以表示一个词的上文信息和下文信息

4. RNN语言模型变形

◆ 单向多层RNN语言模型



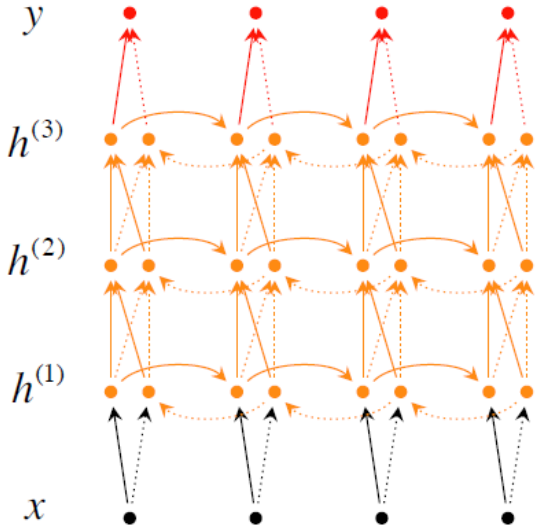
$$h^i(t) = \sigma(W^i{}_i\, h^{i-1}(t) + W^i{}_h\, h^i(t-1) + b^i)$$

$$Y = \mathrm{softmax}(W_o h^L(t))$$

采用多个隐层，每个隐层向后一层传递序列信息

4. RNN语言模型变形

◆ 双向多层RNN语言模型



$$\overrightarrow{h}_t^{(i)} = f(\overrightarrow{W}^{(i)} h_t^{(i-1)} + \overrightarrow{V}^{(i)} \overrightarrow{h}_{t-1}^{(i)} + \overrightarrow{b}^{(i)})$$

$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

$$y_t = g(U[\overrightarrow{h}_t^{(L)}; \overleftarrow{h}_t^{(L)}] + c)$$

各层每个时刻都有一个正向输入 $\overrightarrow{h^i_t}$ 和·一个反向输入 $\overleftarrow{h^i_t}$

每个隐层向后一层传递序列信息

内 容 提 要

4.1　统计语言模型

4.2　神经语言模型

4.3　词向量（浅层）

4.3 词向量（浅层）

■ 词向量（浅层）

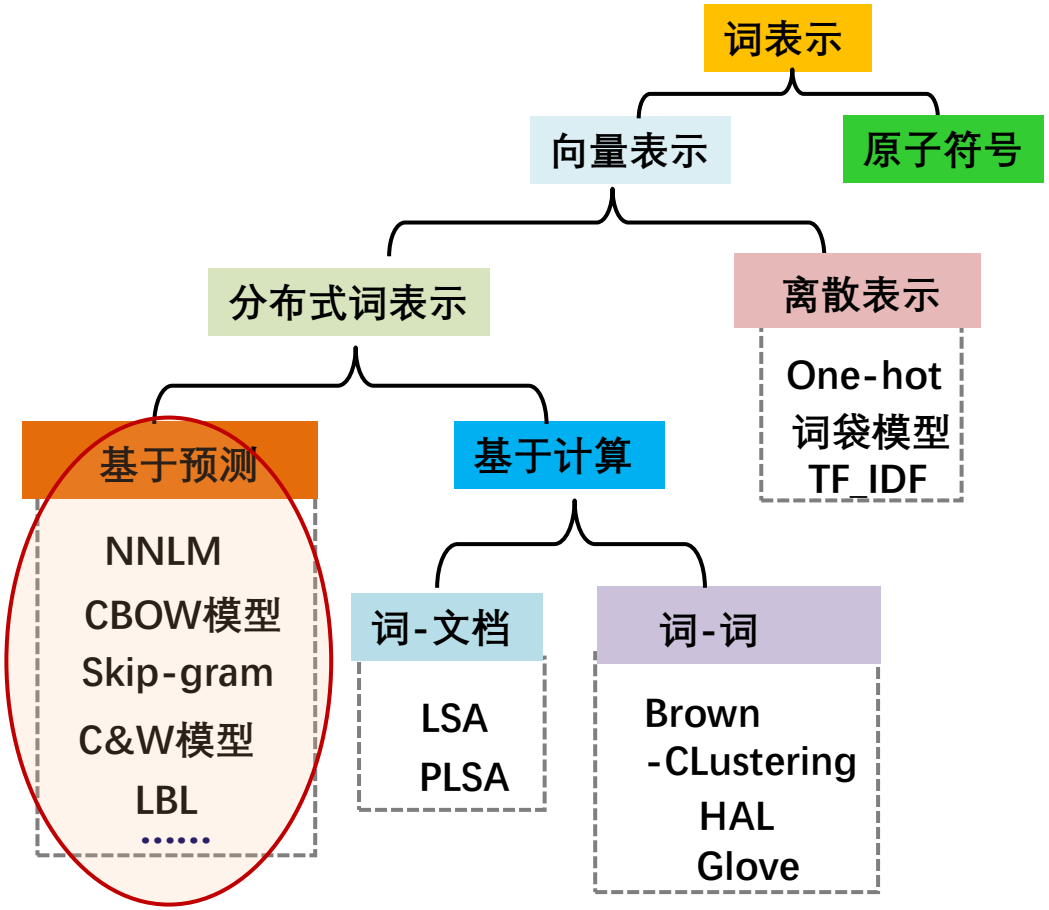
本节内容：

1. 词表示概述
2. 经典词向量(浅层)表示模型
3. 词向量特性及应用

1. 词表示概述

词的表示

自然语言问题要用计算机处理时，第一步要找一种方法把这些符号数字化，成为计算机方便处理的形式化表示。



1. 词表示概述

■ 符号表示

大部分基于规则和基于统计的自然语言处理任务把词看作原子符号

如，减肥　瘦身

■ 离散表示

1、One-hot 表示

NLP 中最直观，也是最常用的词表示方法

如：
- 减肥 [0 0 0 1 0 0 0 0 0 0 0 0 0 0 0]
- 瘦身 [1 0 0 0 0 0 0 0 0 0 0 0 0 0 0]

优势：稀疏方式存储非常的简洁

不足：词汇鸿沟，维数灾难

1. 词表示概述

2、词袋模型

每个数表示该词在文档出现的次数（One-hot的加和）

3、TF_IDF

每个数代表该词在整个文档中的占比

■ 词的分布式表示

核心思想：用一个词附近的其他词来表示该词

分布式假设：在相同上下文中出现的词倾向于具有相同的含义 [Harris ,1954]

如，Marco saw a hairy little wampinuk Crouching behind a tree .

wampinuk 的含义可由其上下文推断。

分布式语义学：根据词语在大型文本语料中的分布特性量化词语及词语语义相似性。

1. 词表示概述

■ 经典分布表示模型

基于计算的分布式

名称	上下文	上下文与目标词之间的建模（技术手段）
LSA/LSI HAL GloVe Jones & Mewhort	文档 词 词 n-gram	矩阵
Brown Clustering	词	聚类

利用全部上下文或利用一定窗口内的上下文词捕获语法和语义信息

局限性：耗空间过大、稀疏等问题，需用降维方法（如，SVD分解的方法）构造低维稠密向量作为词的分布式表示 (25~1000维)。与深度学习模型框架差异大。

1. 词表示概述

■ 经典分布表示模型

基于预测的分布式表示 (神经网络词向量)

名称	上下文	上下文与目标词之间的建模（技术手段）
Skip-gram	词	神经网络
CBOW	n-gram（加权）	
Order	n-gram（线性组合）	
LBL	n-gram（线性组合）	
NNLM	n-gram（非线性组合）	
C&W	n-gram（非线性组合）	

不计算词之间的共现频度，直接用“基于词的上下文词来预测当前词”或“基于当前词预测上下文词”的方法构造构造低维稠密向量作为词的分布式表示 。

深度学习自然语言处理技术中，词向量是神经网络技术中重要的组成部分

1. 词表示概述

■ 词向量(词嵌入)

词嵌入在2001年首次出现。而 Mikolov 等人在2013年作出的主要创新是通过删除隐藏层和近似目标来使这些单词嵌入的训练更有效。虽然这些变化本质上很简单，但它们与高效的 word2vec（word to vector，用来产生词向量的相关模型）组合在一起，使得大规模的词嵌入模型训练成为可能。

词嵌入（Word Embeddings）— NLP 里程碑成果

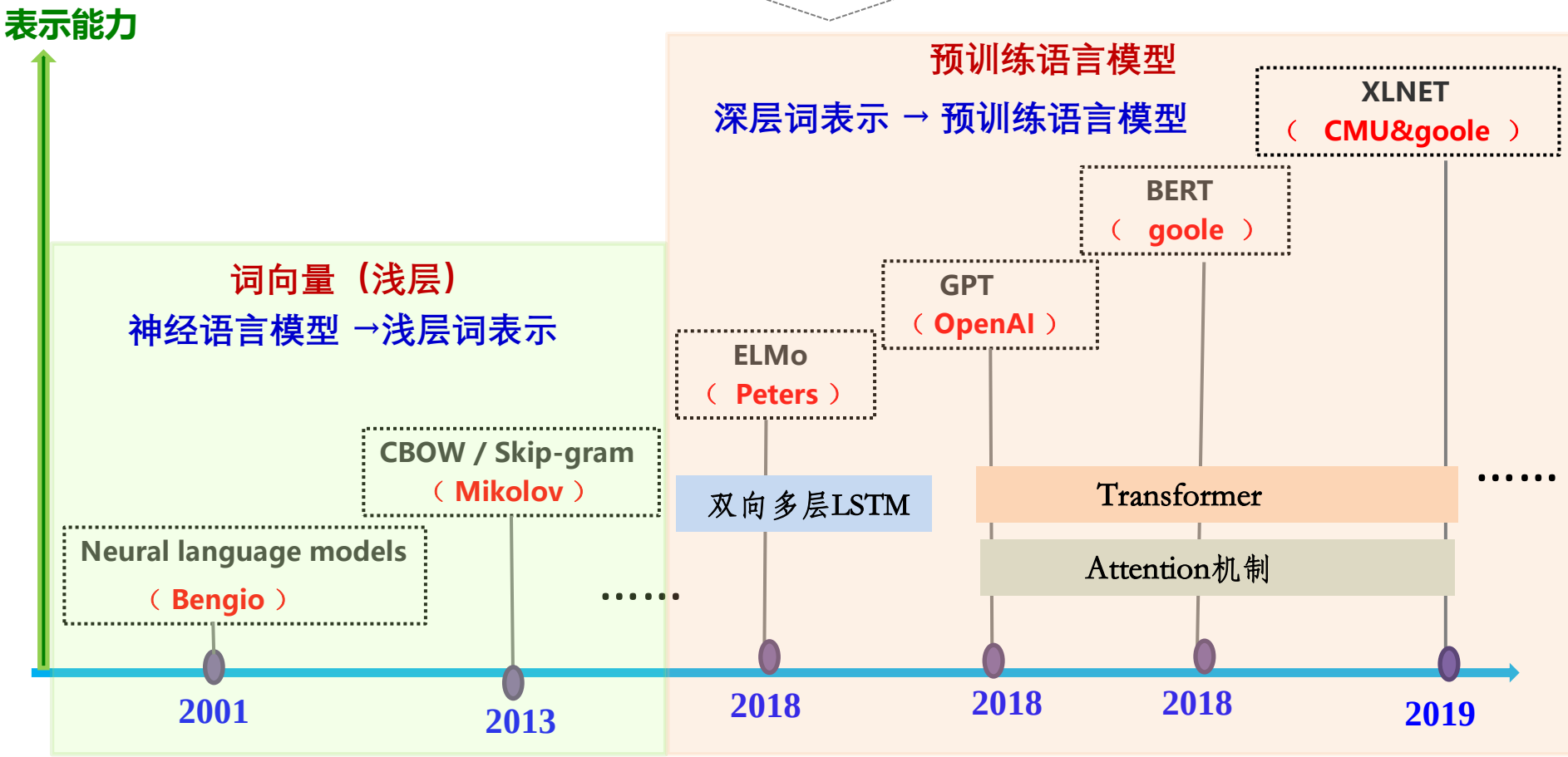
Yoshua Bengio, et.al. A neural probabilistic language model（2001, 2003）

Tomas Mikolov:Efficient estimation of word representations in vector space，2013

1. 词表示概述

■ 词向量发展历程

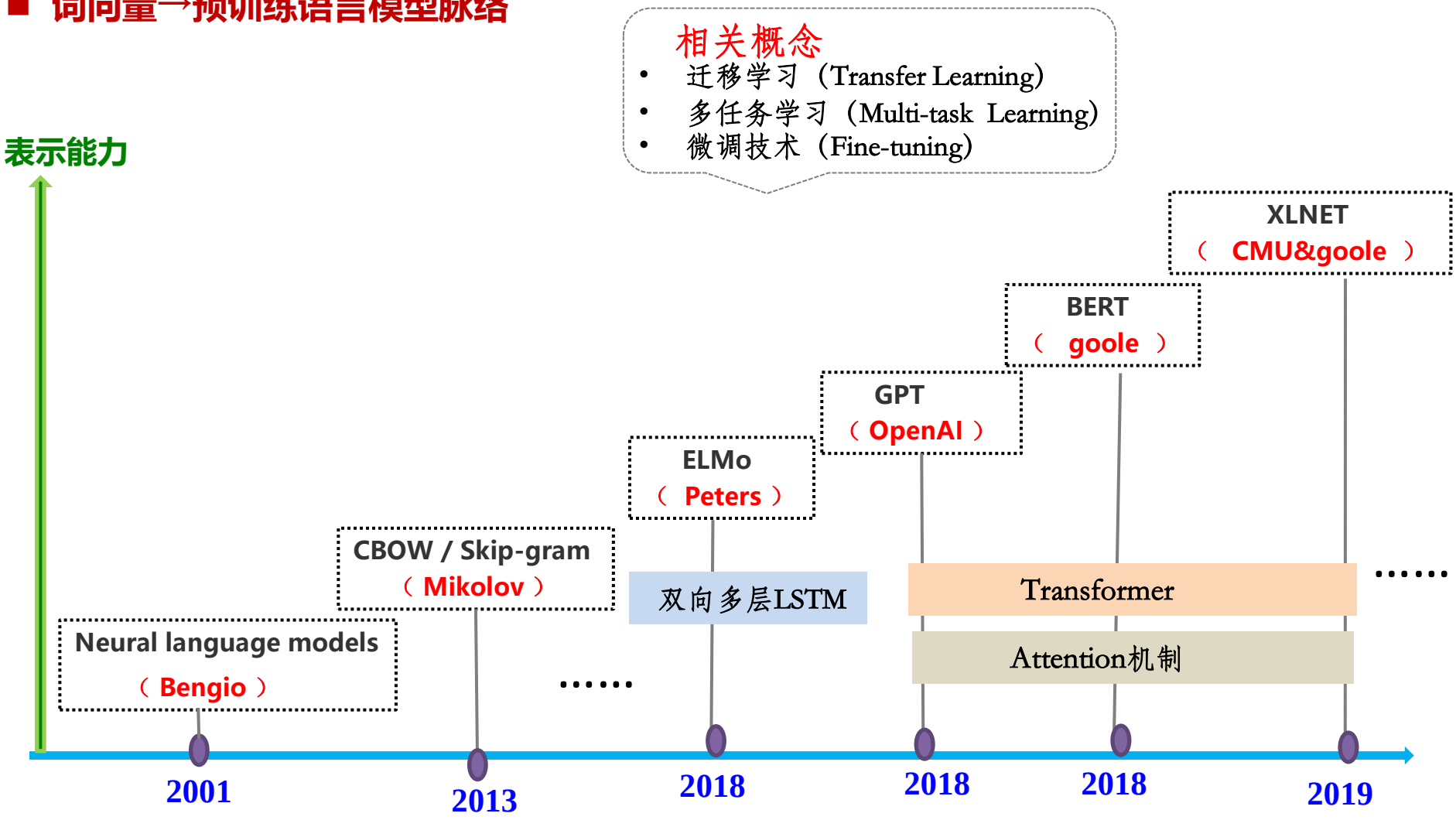
相关概念
- 迁移学习（Transfer Learning）
- 多任务学习（Multi-task Learning）
- 微调技术（Fine-tuning）



神经语言模型 → 浅层词向量→ 深层词向量→预训练语言模型

1. 词表示概述

■ 词向量→预训练语言模型脉络



神经语言模型 → 浅层词向量→ 深层词向量→预训练语言模型

2. 经典词向量(浅层)表示模型

词向量（浅层）表示有多种，本节介绍五种典型的词向量

（1）NNLM模型词向量

（2）RNNLM模型词向量

（3） C&W 模型词向量

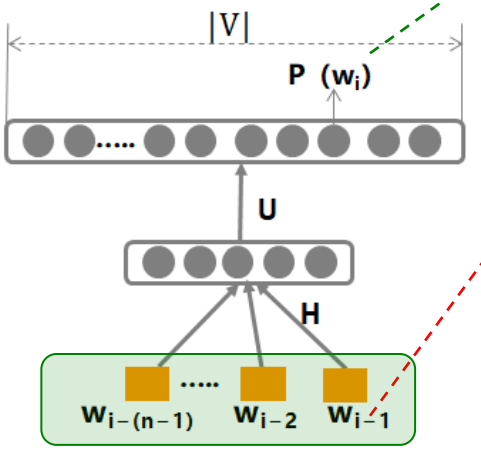
（4） CBOW 模型词向量

（5）Skip-gram模型词向量

（1） NNLM模型词向量

问题引入： NNLM语言模型（n-gram）回顾

$$p(w_1,\ldots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-(n-1)}\ \ldots\ w_{i-1})$$



词以什么形式输入？

输出层：$p(w_i|w_{i-(n-1)}\ \ldots\ w_{i-1})$
$=\text{softmax}\ (b^2+U\ (\tanh\ (XH+b^1)))$

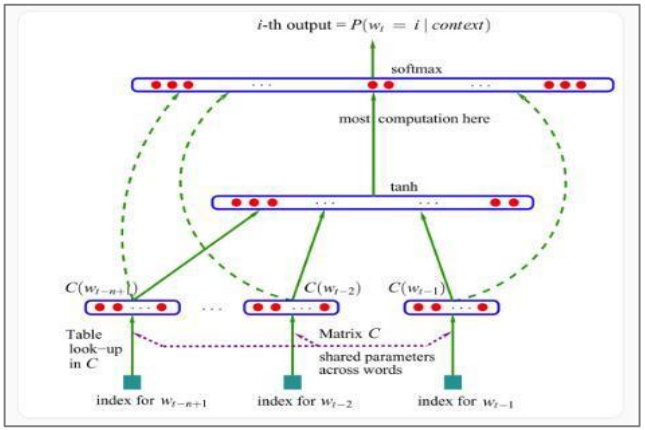
隐藏层：$h=\tanh(XH+b^1)$

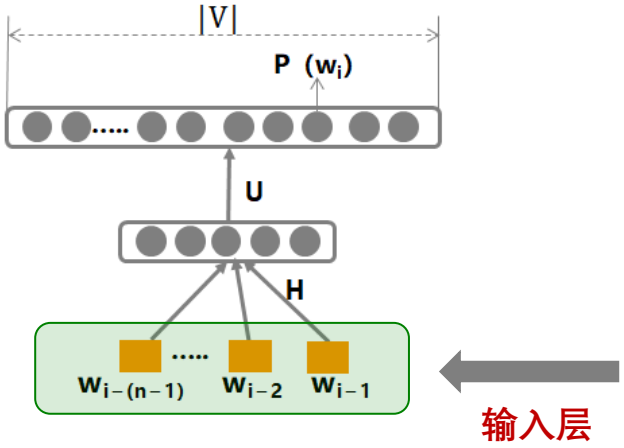
输入层：X: n-1个词 $w_{i-(n-1)},\cdots,w_{i-1}$

参数：$\theta=\{H, U, b^1, b^2\}$

(1) NNLM模型词向量

NNLM模型-输入表示





词的 one-hot 表示

张： 0000100…..00

三：0010000……00

李：00000010…..00

四：0100000…….00

……..

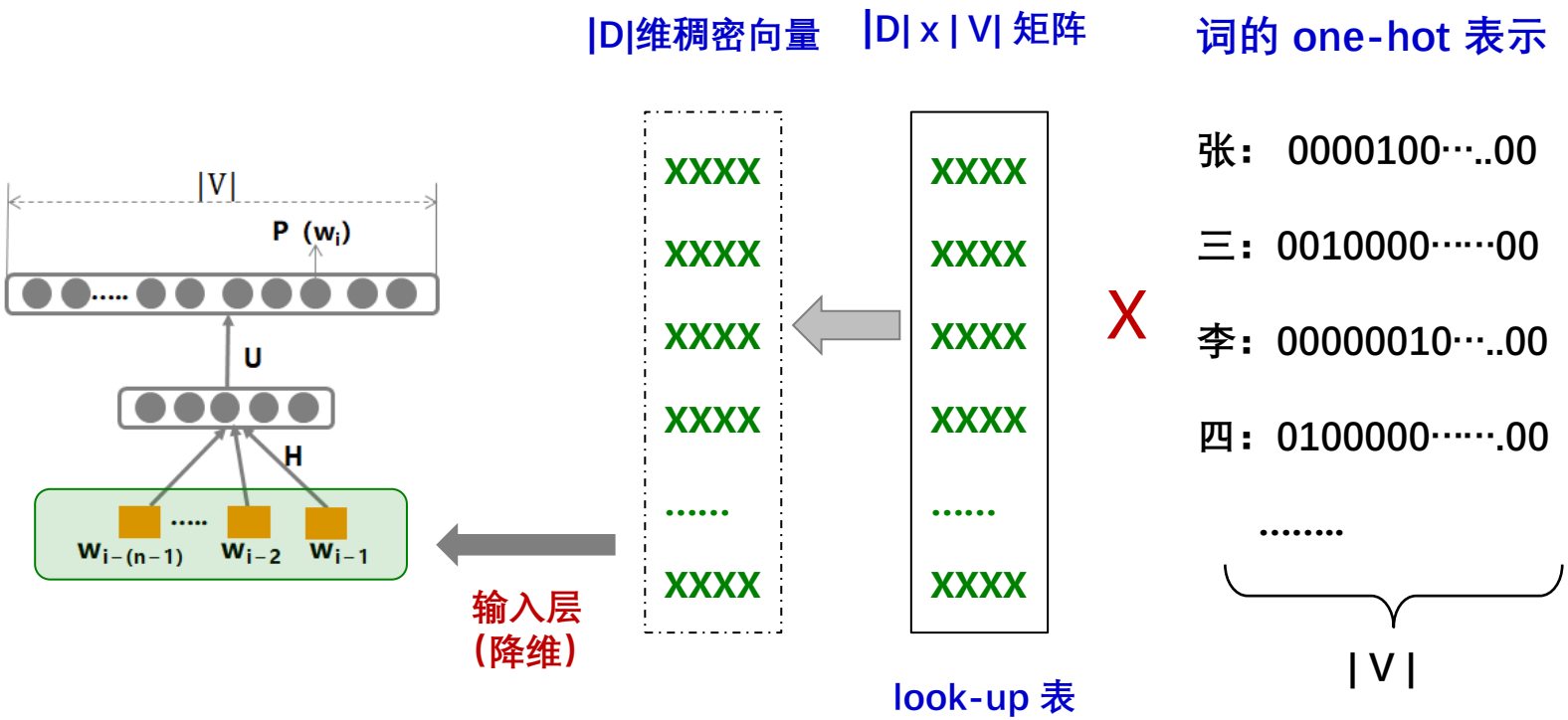
|V|

问题：one-hot 表示维度太高

Yoshua Bengio, et.al. A neural probabilistic language model（2001, 2003）

(1) NNLM模型词向量

NNLM模型-输入表示

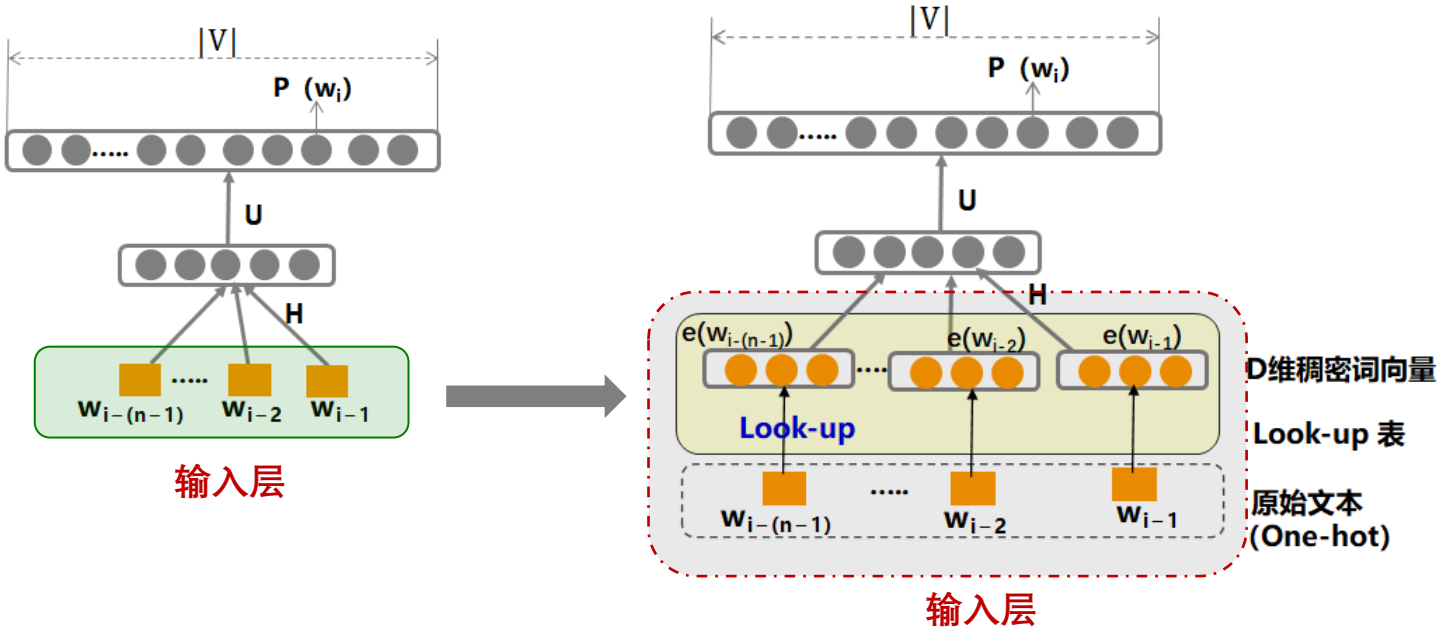


方法：采用低维稠密向量降维

Yoshua Bengio, et.al. A neural probabilistic language model（2001, 2003）

(1) NNLM模型词向量

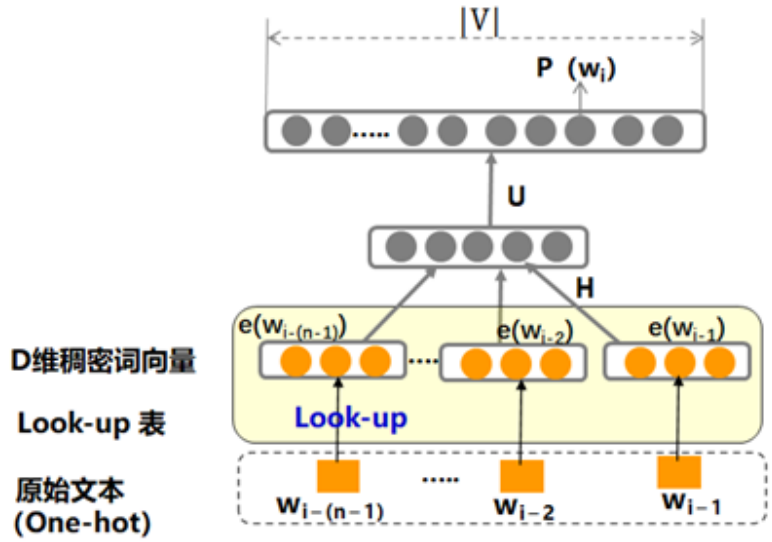
NNLM模型-输入表示



Yoshua Bengio, et.al. A neural probabilistic language model（2001, 2003）

(1) NNLM模型词向量

NNLM模型-输入表示



例，look-up 表 如下：

$$C = \begin{pmatrix} (w_1)_1 & (w_2)_1 & \cdots & (w_V)_1 \\ (w_1)_2 & (w_2)_2 & \cdots & (w_V)_2 \\ \vdots & \vdots & \ddots & \vdots \\ (w_1)_D & (w_2)_D & \cdots & (w_V)_D \end{pmatrix}$$

w_2 = [0 1 0⋯ 0]

$e(w_2) = (w_2)_1\ (w_2)_2 \cdots (w_2)_D$

稠密向量表示Look-up表是|D| x | V|维实数投影矩阵，| V|表示词表的大小，|D|表示词向量e的维度（一般50维以上）；各词的词向量存于表中。词w 到其词向量e(w) 的转化是从该矩阵中取出相应的列。

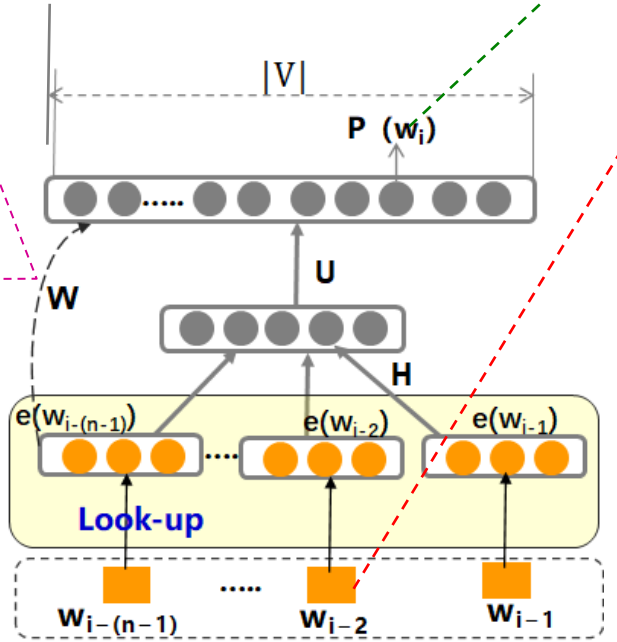
(1) NNLM模型词向量

NNLM模型结构(词向量)

语言模型参数

n-gram: $p(w_1,\dots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-(n-1)}\dots w_{i-1})$

用该直连边边，迭代次数可减半；不用生成语言模型性能更好。后续工作，很少有使用该边



输出层：$p(w_i|w_{i-(n-1)}\dots w_{i-1})$
$=\mathrm{softmax}\ (b^2+Wx+U\,h)$

隐藏层：$h=\tanh(XH+b^1)$

输入层：X: n-1个词 $w_{i-(n-1)},\cdots,w_{i-1}$
的词向量拼接 $X=[e(w_{i-(n-1)})\cdots e(w_{i-1})]$
(词向量初值)

参数：$\theta=\{H, U, W, b^1, b^2,$ 词向量 $\}$

神经网络参数

训练结束后look-up表中为训练好的词向量

(1) NNLM模型词向量

NNLM模型(词向量)学习

◆ 学习参数： $\theta = \{ H, U, W, b^1, b^2, \text{词向量} \}$

◆ 语料：(“无监督”)

文本： $S=w_1, w_2, \cdots, w_n, \cdots$

实例： X： $w_1, w_2, \cdots, w_{i-1}$

$\hat{Y}$ ： w_i

◆ 目标函数：

采用log损失函数 $L(Y, P(Y|X))=-logP(Y|X)$

对于整个语料而言，语言模型需要最大化：

$$\sum_{w_{i-1}} \sum_{i \in D} \log P(w_i \mid w_{i-1})$$

(1) NNLM模型词向量

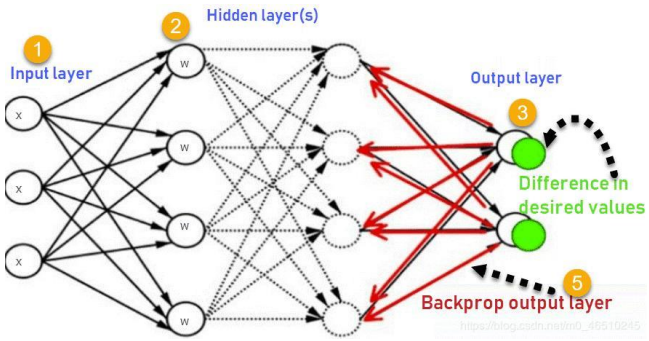
NNLM模型(词向量)学习

◆ 参数训练：

(BP) 随机梯度下降法优化训练目标:
每次迭代，随机从语料D 中选取一段文本
$w_{i-(n-1)}, \cdots, w_i$作为训练样本进行一次梯度迭代

$$\theta \leftarrow \theta + \alpha \frac{\partial \log P(w_i \mid w_{i-1})}{\partial \theta}$$

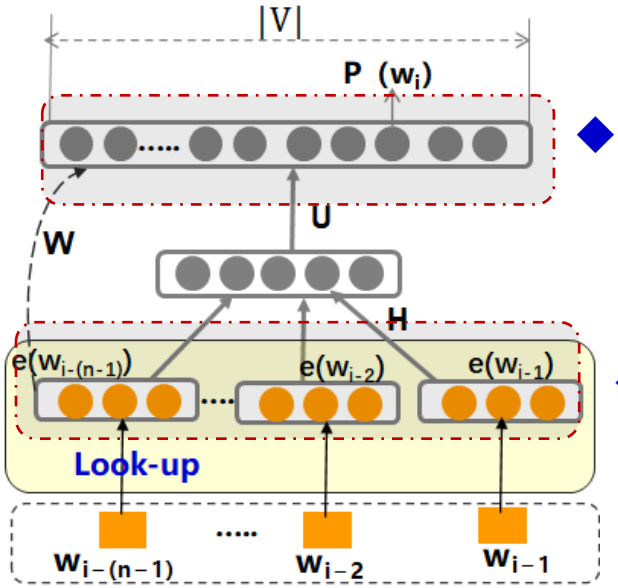
其中，α学习率，θ＝{ H, U, W， b^1, b^2 ，词向量 }



(1) NNLM模型词向量

NNLM模型作用

NNLM 语言模型在训练语言模型同时也训练了词向量



◆ 语言模型：$p(w_i | w_{i-(n-1)} \ldots w_{i-1})$

◆ 词向量：$w_{i-(n-1)} \ldots w_{i-1}$ 的词向量

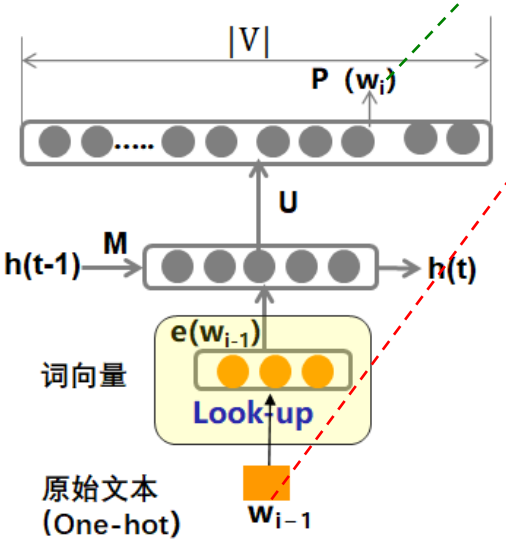
Yoshua Bengio, et.al. A neural probabilistic language model（2001, 2003）

(2) RNNLM模型词向量

■ RNNLM模型结构(词向量)

语言模型参数

2-gram: $p(w_1,\ldots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-1})$



输出层：$p(w_i|w_{i-1})$
$=\mathrm{softmax}\left(b^2+U\,h(t)\right)$

隐藏层：$h(t)=\tanh(X+Mh(t-1)+b^1)$

输入层：X: 词 w_{i-1}的词向量 $X= e(w_{i-1})$初值
$h\,(t-1)$

参数：θ ={ U,M, b^1, b^2 ,词向量}

神经网络参数

- Tomas Mikolov, et.al.Statistical language models based on neural networks.2012
- Tomas Mikolov, et.al.Recurrent neural network based language model.2010

(2) RNNLM模型词向量

■ RNNLM模型学习(词向量)　(2-gram)

◆ 语料：(“无监督”)

文本：　S= $w_1, w_2, \ldots, w_n$ ，　……

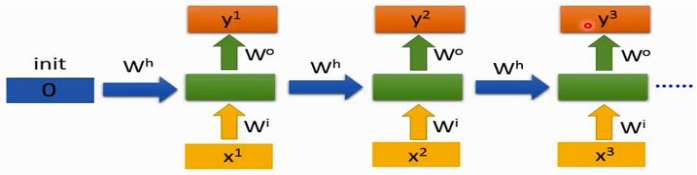
实例：X：　START，$w_1, w_2, \ldots, w_{n-1}$

$\hat{Y}$ ：　$w_1, w_2, \ldots, w_{n-1}$　w_n

◆ 目标函数：

对于整个语料而言，语言模型需要最大化：

$$\sum_{w_{i-1}} \sum_{i \in D} \log P(w_i \mid w_{i-1})$$



(2) RNNLM模型词向量

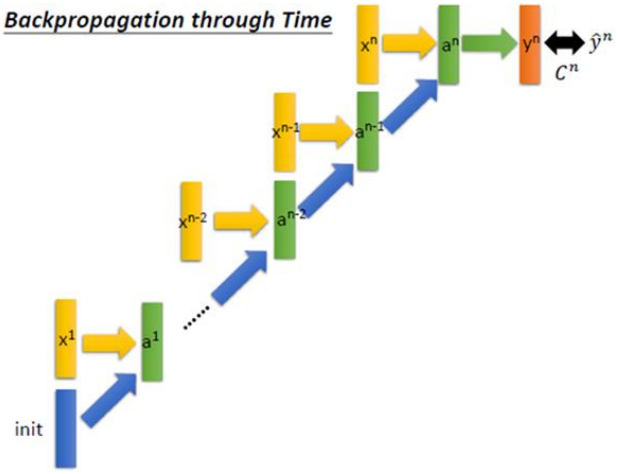
■ RNNLM模型学习(词向量)　(2-gram)

◆ 参数训练：

(BPTT) 随机梯度下降法优化训练目标:

每次迭代，随机从语料D 中选取一段文本 $w_{i-(n-1)},\cdots,w_i$作为训练样本进行一次梯度迭代

学习参数： $\theta=\{U, W, b^1, b^2$ ，词向量 $\}$

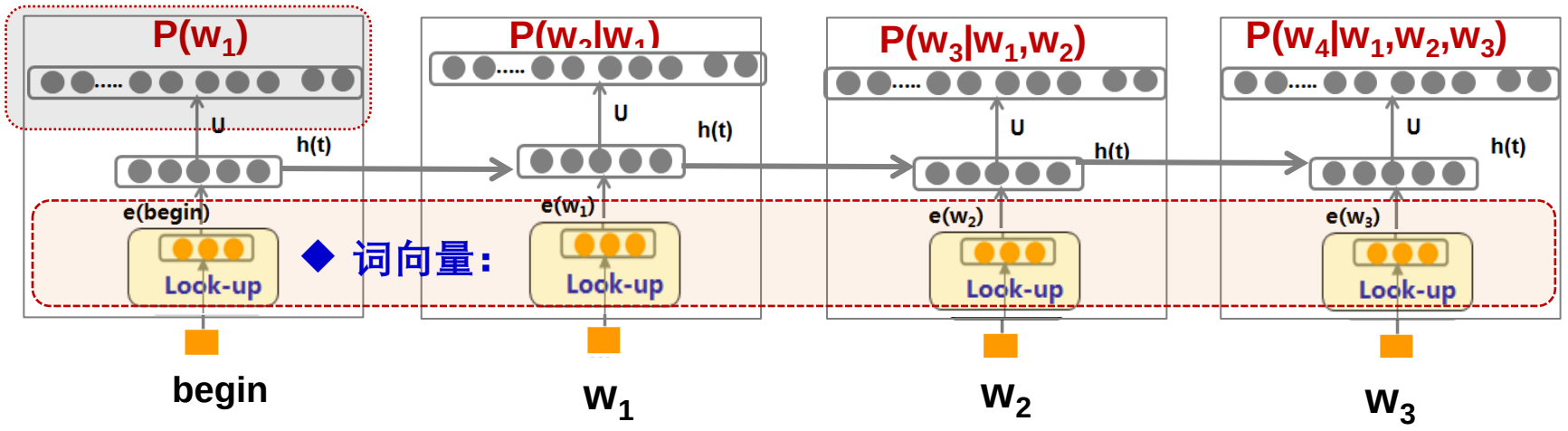


(2) RNNLM模型(词向量)

RNNLM模型应用

RNNLM 语言模型在训练语言模型同时也训练了词向量

◆ 语言模型：



- Tomas Mikolov, et.al.Statistical language models based on neural networks.2012
- Tomas Mikolov, et.al.Recurrent neural network based language model.2010

(3) C&W 模型词向量

■ **C&W 模型** （Collobert ,Weston ,2008）

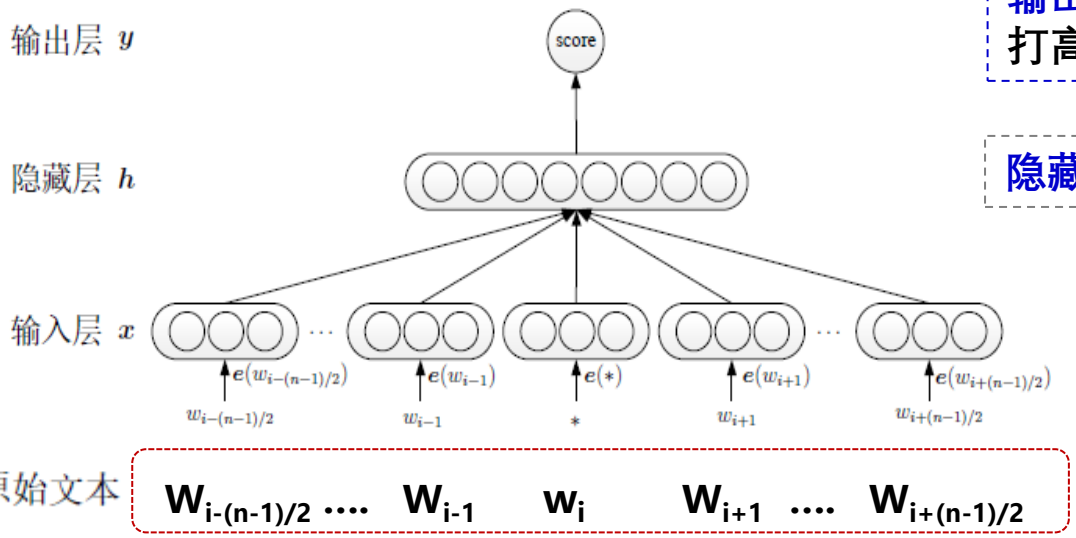
Collobe等人2008提出的第一个以直接快速生成词向量为目标的模型，采用直接对n元短语打分的方式替代语言模型中求解条件概率的方法：对于语料中出现过的n元短语，对其打高分；对于语料中没有出现的随机短语，对其打低分。通过这种方式，C&W模型可以更直接地学习得到符合分布假说的词向量。

特点：C&W 模型的目标函数是求目标词w与其上下文c的联合打分，而其他模型均为根据上下文c，预测目标词w。

Ronan Collobert : A unified architecture for natural language processing : Deep neural networks with multitask learning，2008

(3) C&W 模型词向量

■ C&W模型结构



输出层：对输入序列打分（正样本打高分、负样本打低分）

隐藏层:上下文和目标词的联合表示

输入：目标词W_i及其上下文

$x = [e(w_{i-(n-1)}), \cdots, e(w_i), \cdots, e(w_{i+(n-1)})]$

为从语料中选出的一个n元短语$w_{i-(n-1)/2}$，…, w_i, …, $w_{i+(n-1)/2}$一般n为奇数，以保证上文和下文的词数一致； w_i为目标词(序列中间的词) $x = [e(w_{i-(n-1)}), \cdots, e(w_i), \cdots, e(w_{i+(n-1)})]$

(3) C&W 模型词向量

■ C&W模型学习

优化目标：：

对于整个语料最小化:

$$\sum_{(w,c)\in D}\sum_{w'\in V}\max(0,1-\text{score}(w,c)+\text{score}(w',c))$$

其中，(w,c) :c 表示目标词 w 的上下文

- 正样本(w,c)来自语料
- 而负样本(w'c)则是将正样本序列中的中间词替换成其它词

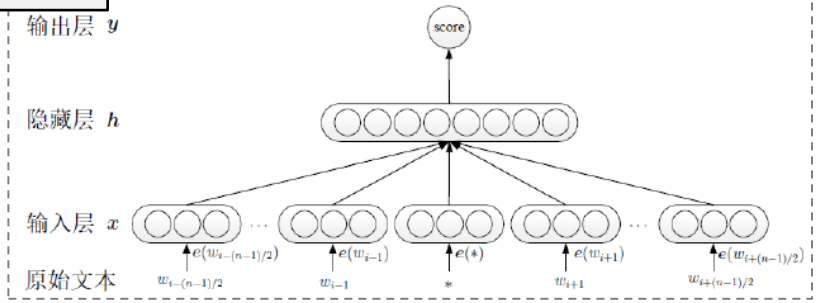
参数训练：

采用pairwise的方式对文本片段进行优化，即可得词向量

(3) C&W 模型词向量

C&W模型与NNLN模型对比

C&W模型

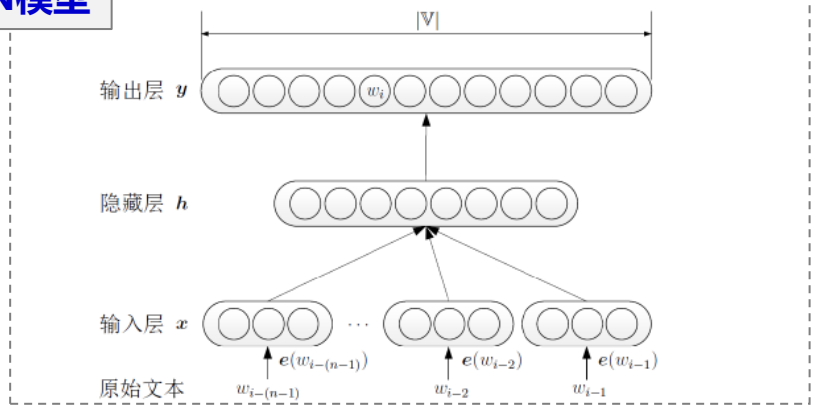


目标词在输入层

输出层只有1个节点

最后一层只需|h|次运算

NNLN模型



目标词在输出层

输出层有|V|个节点

最后一层需|Vl x lh|次运算，

且需要进行softmax运算

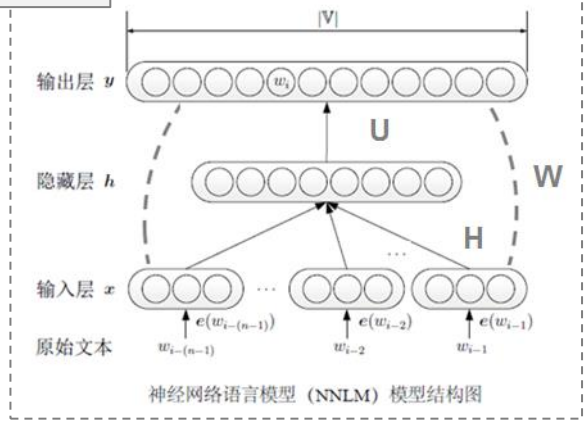
C&W 模型在运算速度上优于NNLM模型，但在许多语言学任务上，效果不如其它模型

(4) CBOW / Skip-gram模型

CBOW / Skip-gram模型

Mikolov等人在2013年，同时提出CBOW（Continuous Bag-of-Words）和Skip-gram模型。主要针对NNLM训练复杂度过高的问题进行改进，简化模型，保留核心部分，以更高效的方法获取词向量。

NNLN模型



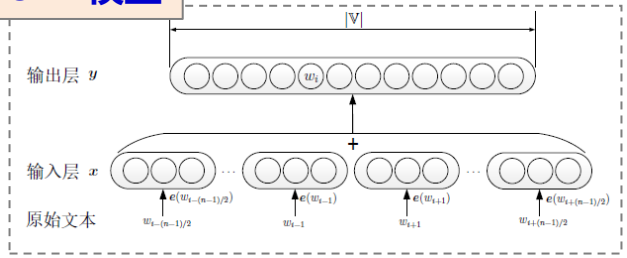
神经网络语言模型（NNLM）模型结构图

研表究明，汉字序顺并不定一影阅响读！事证实明了也许当你看这完句话之后才发字现都乱是的。

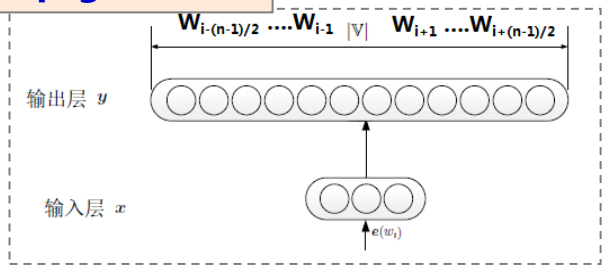
简化：

1. 除去隐藏层
2. 除去词序

CBOW 模型

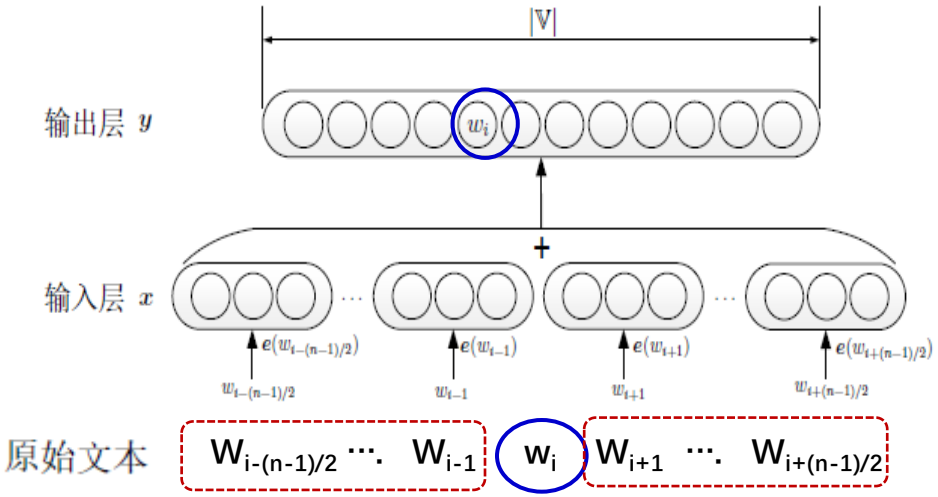


Skip-gram模型



(4) CBOW 模型

■ CBOW 模型结构



输出层： $p(w_i|C) = \frac{\exp\left(e'(w_i)^{\mathrm{T}}x\right)}{\sum_{w'\in V}\exp\left(e'(w')^{\mathrm{T}}x\right)}$

输入层： X: 词 w_i的上下文词向量平均值 $x = \frac{1}{n-1}\sum_{w_j\in c} e(w_j)$

为从语料中选出的一个n元短语$w_{i-(n-1)/2}$，…，w_i，…，$w_{i+(n-1)/2}$一般n为奇数，以保证上文和下文的词数一致；w_i为目标词，w_i上下文C为不包括w_i的n-1元短语

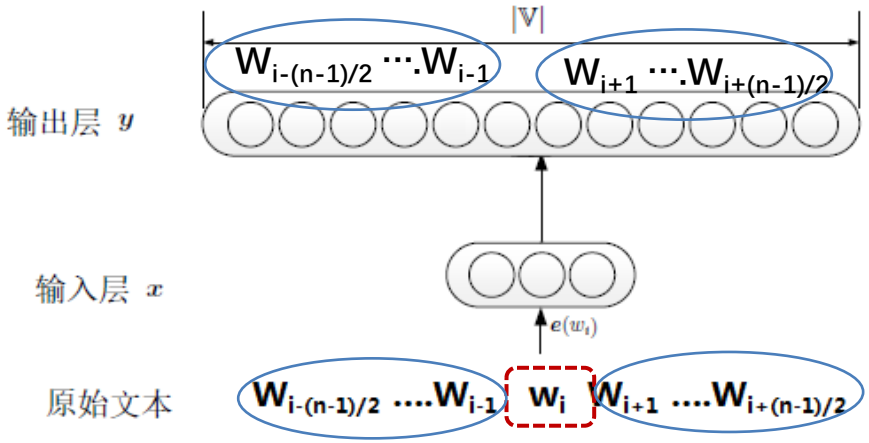
■ 模型学习

- **优化目标：** 最大化 $\sum_{(w,c)\in D}\log P(w|c)$
- **参数训练：** 梯度下降法

Tomas Mikolov:Efficient estimation of word representations in vector space，2013

(5) Skip-gram模型

■ Skip-gram模型



输出层： $P(w_j \mid w_i) = \frac{\exp\left(e'(w_j)^{\mathrm{T}} x\right)}{\sum_{w' \in \mathrm{V}} \exp\left(e'(w')^{\mathrm{T}} x\right)}$

输入层： X: 词 w_i 的词向量

将目标词w_i的词向量作为输入，每次从w_i的上下文C 中选一个词作为预测词进行预测。目标词w_i及上下文C 定义同CBOW 模型

■ 模型学习

- **优化目标：** 最大化 $\sum_{(w,c)\in D} \sum_{w_j \in c} \log P(w_j \mid w)$
- **参数训练：** 梯度下降法

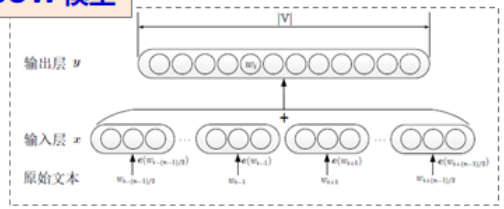
Tomas Mikolov:Efficient estimation of word representations in vector space，2013

(5) CBOW / Skip-gram模型

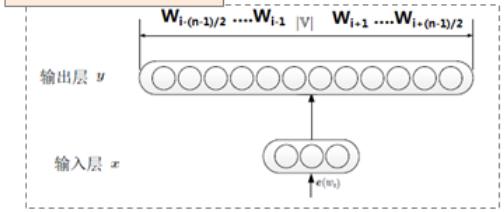
CBOW / Skip-gram模型中softmax优化问题:

计算 y 中的每个分量的值非常耗时，实际运算中采用层级softmax 函数优化。

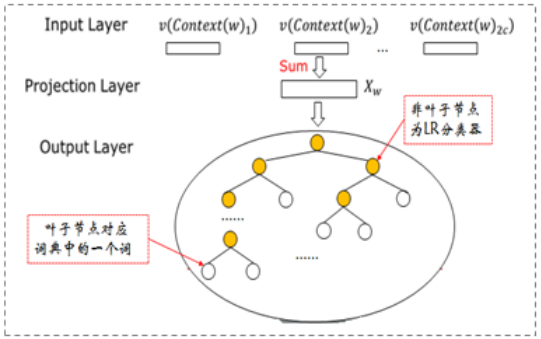
CBOW 模型



Skip-gram模型

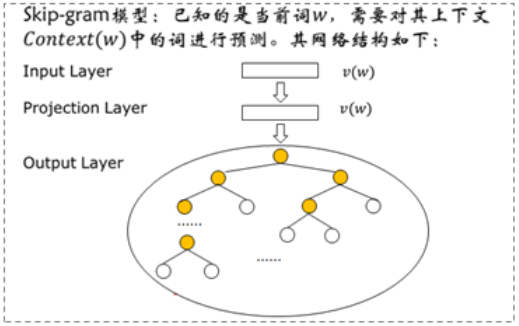


CBOW Hierarchical Softmax



求： $p(w_i \mid Context_i)$

Skip-gram Hierarchical Softmax



求： $p(Context_i \mid w_i)$

详情参阅：

Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In Proceedings of the international workshop on artificial intelligence and statistics, , 2005.

(5) CBOW / Skip-gram模型

基于CBOW/Skip-gram模型的词向量工具

Word2vec 是Google开源的将词表征为实数值向量的高效工具，其利用深度学习的思想，可以通过训练，把对词的处理简化为K维向量空间中的向量运算

Word2vec训练得到的词向量可以用于机器翻译，相似词查找，关系挖掘，中文聚类等任务中。

Word2vec总共有两种类型，每种类型有两个策略，总共4种

模型	CBOW		Skip-Gram	
方法	Hierarchical Softmax	Negative Sampling	Hierarchical Softmax	Negative Sampling

经典模型小结

模型特点

模型	目标词与上下文位置(n-gram)	模型输入	模型输出	目标词与上下文词之间的关系
NNLM	(上文)(目标词)	上文词向量拼接	目标词概率	上文在输入层、目标词在输出层，优化预测关系
C&W	(上文)(目标词)(下文)	上下文及目标词词向量拼接	上下文及目标词联合打分	上下文和目标词都在输入层，优化组合关系
CBOW	(上文)(目标词)(下文)	上下文各词词向量平均值	目标词概率	上下文在输入层、目标词在输出层，优化预测关系
Skip-gram	(上文)(目标词)(下文)	目标词词向量	上下文词概率	目标词在输入层、上下文在输出层，优化预测关系

经典模型小结

不同模型对比

指标	对比情况
模型复杂度	NNLM>C&W>CBOW>Skip-g ram
参数个数	NNLM>(cBoW=Skip-gram)>C&W
时间复杂度	NNLM>(cBoW=Skip-g ram)>C&W

3. 词向量特性及应用

词向量具有如下语言学特性

- 语义相似的词，其词向量空间距离更相近 （分布假说 ）

1-of-N Encoding

apple = [1 0 0 0 0]

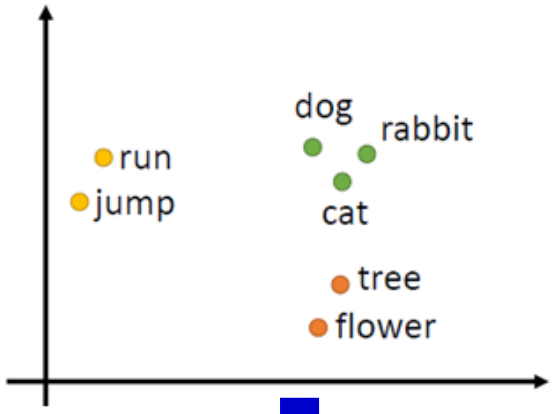
bag = [0 1 0 0 0]

cat = [0 0 1 0 0]

dog = [0 0 0 1 0]

elephant = [0 0 0 0 1]

Word Embedding

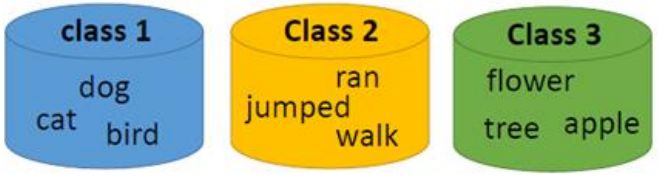


优点：降维，消除词汇鸿沟

其语言模型自带平滑功能

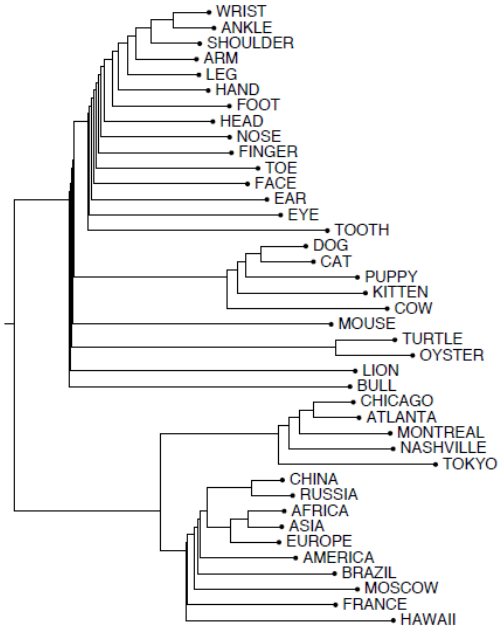
应用：同义词检测、单词类比等

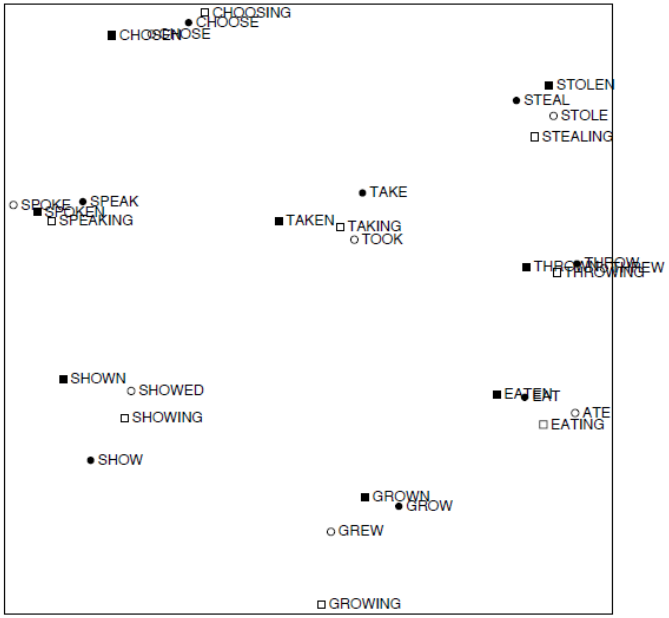
Word Class



3. 词向量特性及应用

应用：语义相似度度量





Rohde et al. An Improved Model of Semantic Similarity Based on Lexical Co-Occurrence, 2005

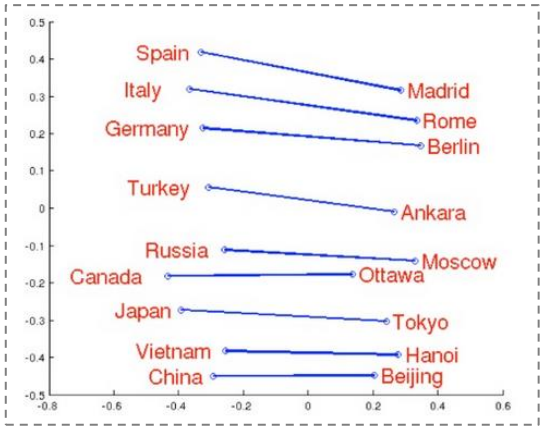
3. 词向量特性及应用

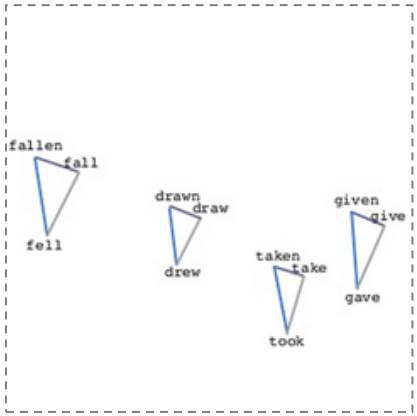
词向量具有如下语言学特性

- **相似关系词对的词向量之差也相似**

$$V(king) - V(queen) \approx V(uncle) - V(aunt)$$

$$V(hotter) - V(hot) \approx V(bigger) - V(big)$$



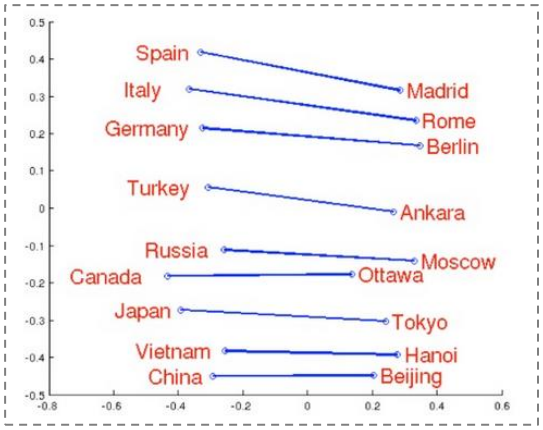


Source: http://www.slideshare.net/hustwj/cikm-keynotenov2014

3. 词向量特性及应用

应用：直接使用词向量的加减法进行推理

如： Rome : Italy = Berlin : ?



用相似关系词对的词向量之差也相似，

直接使用词向量的加减法

$$\text{Compute } V(Berlin) - V(Rome) + V(Italy)$$

$$\text{Find the word w with the closest V(w)}$$

$$V(Germany) \approx \text{V(Berlin)} - \text{V(Rome)} + \text{V(Italy)}$$

Source: http://www.slideshare.net/hustwj/cikm-keynotenov2014

3. 词向量特性及应用

词向量应用范围

- **利用词向量的语言学特性完成任务**

  利用词向量语义相似的词，其词向量空间距离相近的特征可完成语义相关性任务。如，同义词检测、单词类比等

- **将词向量作为静态特征（输入）**

使用词向量作为模型输入，在模型训练过程中，只调整模型参数，不调整输入词向量。如，基于平均词向量的文本分类、命名实体识别等

采用不同的词向量会影响系统性能。

- **将词向量作为动态初值（输入）（浅层表示模型）**

使用词向量作为神经网络的初始值，模型训练过程中会调整词向量的初值，从而提升神经网络模型的优化效果。

如，基于卷积神经网络的文本分类、词性标注等

3. 词向量特性及应用

神经语言模型对比概率语言模型的优势

概率语言模型存在问题

1. 由于参数数量问题需要对词 i 的历史简化 n-gram
2. 需要数据平滑

神经网络语言模型

利用RNN 语言模型可以解决以上概率语言模型问题

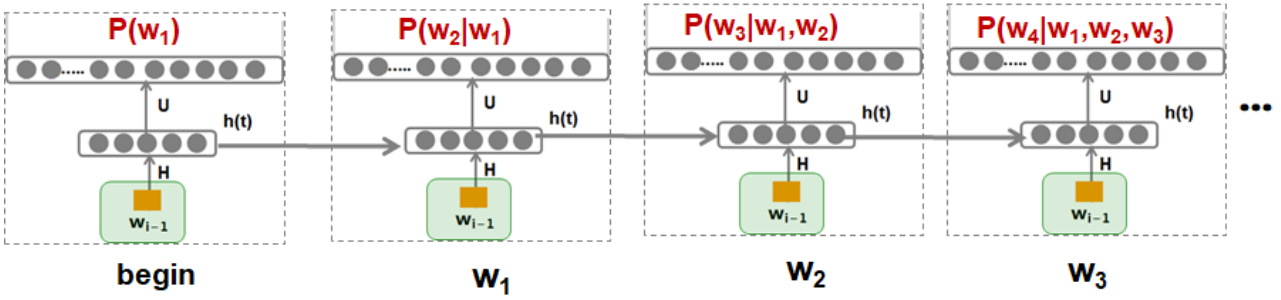
3. 词向量特性及应用

问题1. 由于参数数量问题需要对词 i 的历史简化 n-gram

解决： RNNLM模型

$$P(w_1, w_2, w_3, \cdots, w_n)$$

$$= P(w_1)P(w_1 \mid w_2)P(w_3 \mid w_1, w_2) \cdots\cdots P(w_n \mid w_1, w_2 \cdots w_{n-1})$$



RNNLM模型可以保留每个词的所有历史信息

3. 词向量特性及应用

问题2：需要数据平滑

平滑问题： $p(Cher\ read\ a\ book) = ?$

$$= p(Cher|<BOS>) \times p(read|Cher) \times p(a|read) \times p(book|a) \times p(<EOS>|book)$$

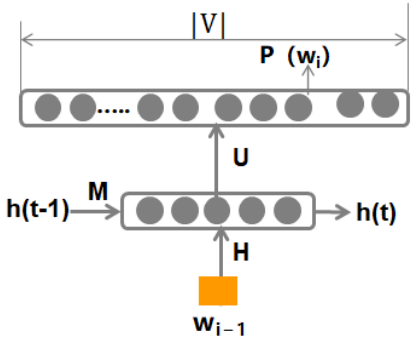
统计语言模型：

$$p(Cher|<BOS>) = \frac{c(<BOS>\ Cher)}{\sum_w c(<BOS>\ w)} = \frac{0}{3}$$

$$p(read|Cher) = \frac{c(Cher\ \ read)}{\sum_w c(Cher\ \ w)} = \frac{0}{1}$$

$p(Cher\ read\ a\ book) = 0$ ➡ **需要数据平滑**

解决：神经网络语言模型：

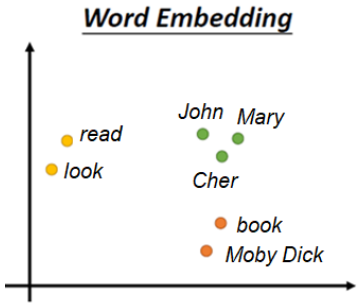


当语言模型训练好后，模型网络参数固定，这时给任意的W_{i-1} $P(W_i)$ 不会为 0 ➡ **无需数据平滑**

3. 词向量特性及应用

RNN 语言模型 + 词向量

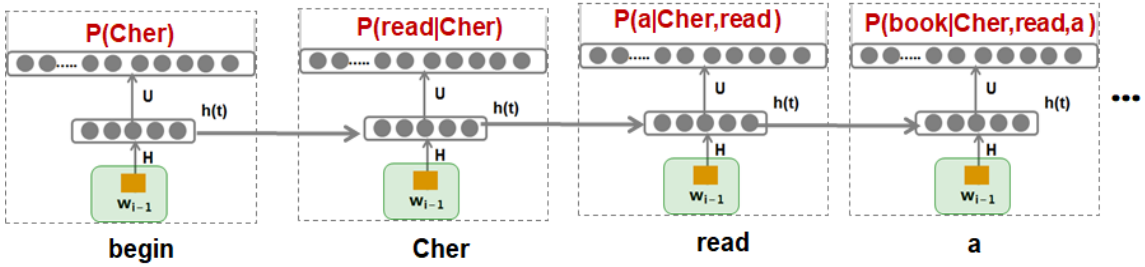
词向量特征:



优点：降维，消除词汇鸿沟

词意相近的词其词向量空间距离近

如，输入 $p(Cher\ read\ a\ book)$



Cher 和 Join 的词向量比较接近模型具有更好泛化能力

深度学习中语言模型常用“RNN语言模型+词向量” 模式

参考文献：

李宏毅课程

http://speech.ee.ntu.edu.tw/~tlkagk/courses_ML16.html

来斯惟，基于神经网络的词和文档语义向量表示方法研究，博士学位论文

licstar的博客： http://licstar.net/archives/328

深度学习word2vec学习笔记
http://download.csdn.net/detail/mytestmy/8565959

宗成庆，统计自然语言处理（第2版）课件

在 此 表 示 感 谢！

谢谢各位！

Q&A