# 第 8 章 神经网络语言模型&词向量

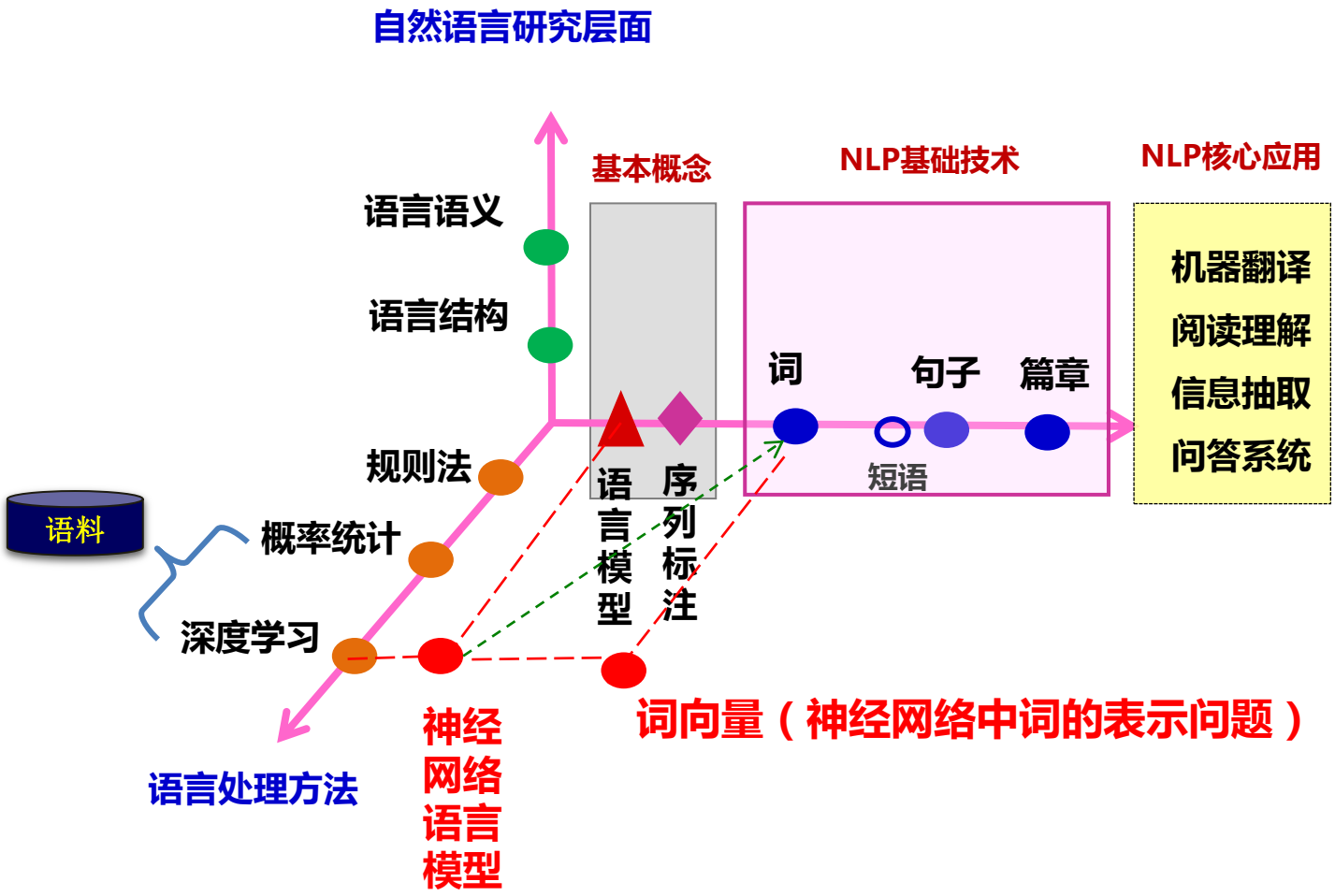
中科院信息工程研究所第二研究室

胡玥

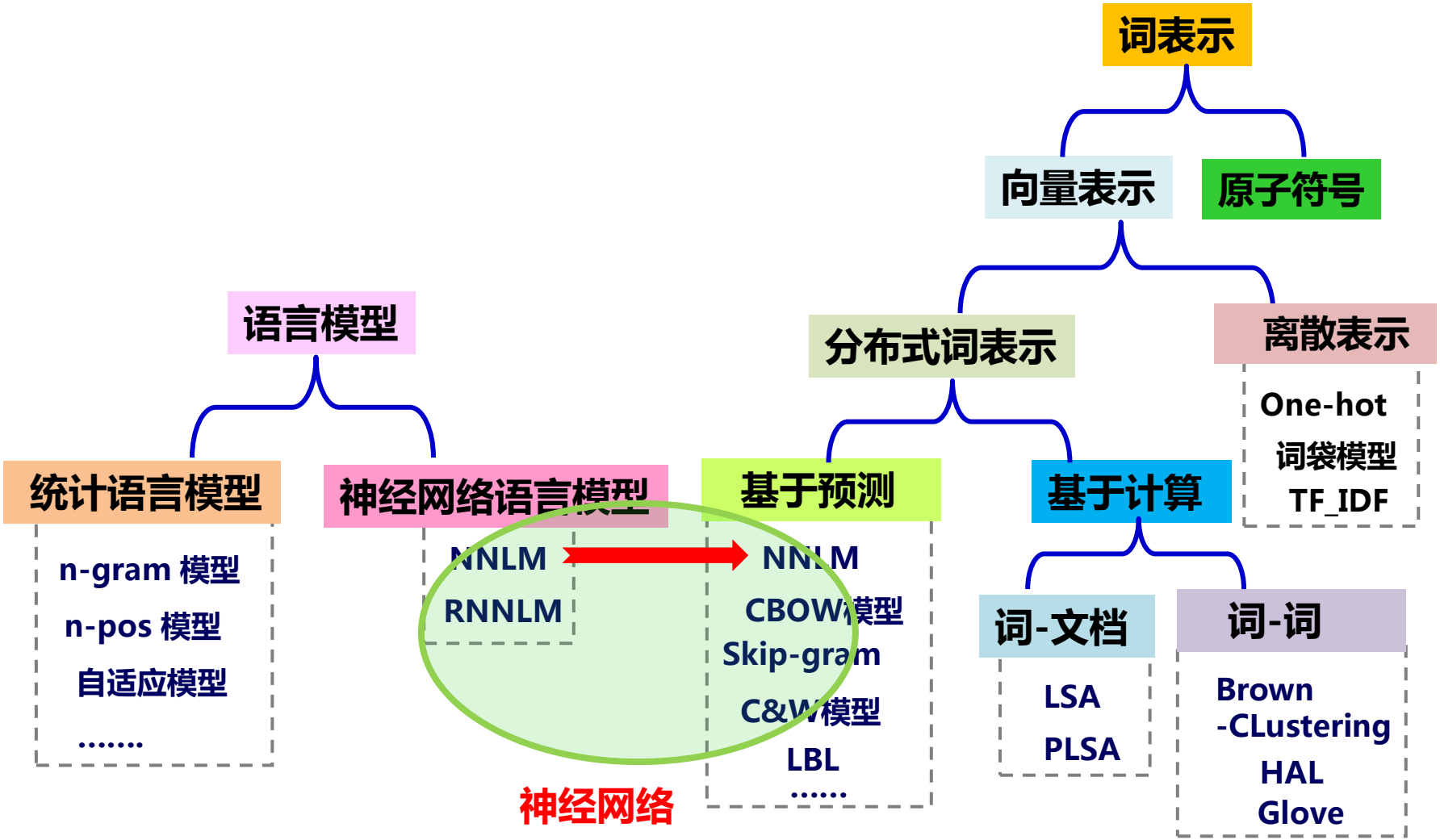
huyue@iie.ac.cn

# 自然语言处理课程内容及安排

✧ 课程内容：



# 本章主要内容


词表示
向量表示
原子符号
分布式词表示
离散表示
One-hot
词袋模型
TF_IDF
基于预测
基于计算
词-文档
LSA
PLSA
词-词
Brown
-CLustering
HAL
Glove
语言模型
统计语言模型
n-gram 模型
n-pos 模型
自适应模型
.....
神经网络语言模型
NNLM
RNNLM
NNLM
CBOW模型
Skip-gram
C&W模型
LBL
.....
神经网络

# 内 容 提 要

8.1　神经网络语言模型

8.2　神经网络词向量

# 语言模型回顾

## 语言模型

用句子S=$w_1,w_2,...,w_n$ 的 **概率** p(S) 来定量的刻画句子。

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

输入： 句子 S

输出： 句子概率p(S)

参数： $p(w_i|w_1,...,w_{i-1})$

统计语言模型：
用概率统计法学习参数

神经网络语言模型：
用神经网络学习参数

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

# 语言模型回顾

## 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

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

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

n-gram模型中，n 越大，能够保留的词序信息越多，语言模型越有效

但参数也越多，一般用 3-gram

# 语言模型回顾

## 统计语言模型 （用概率统计法学习语言模型参数）

- 对于n-gram 语言模型: $p(w_1,\ldots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-(n-1)}\ldots w_{i-1})$
- 模型参数 $p(w_i|w_{i-(n-1)}\ldots w_{i-1})$ 由最大似然估计（ MLE ）求得:

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

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。

存在数据稀疏问题，需要数据平滑

# 语言模型回顾

## 统计语言模型的缺点

- 平滑技术错综复杂而且需要回退到底阶，使得该模型无法面向更大的n元文法获取更多的词序信息

- 基于最大似然估计的语言模型缺乏对上下文的泛化，如 观察到 black car (黑汽车) 和 blue car(兰汽车) 不会影响估计出现 red car（红汽车）的概率

# 8.1 神经网络语言模型

**神经网络语言模型　（用神经网络学习语言模型参数）**

- 对于n-gram 语言模型: $p(w_1,\ldots,w_m)=\prod_{i=1}^{m} p(w_i|w_{i-(n-1)}\ldots w_{i-1})$
- 模型参数 $p(w_i|w_{i-(n-1)}\ldots w_{i-1})$ 由神经网络方法求得:

  根据所用神经网络不同，分为：

  - **NNLM 模型　（主要使用DNN）**
  - **RNNLM 模型　（主要使用RNN）**

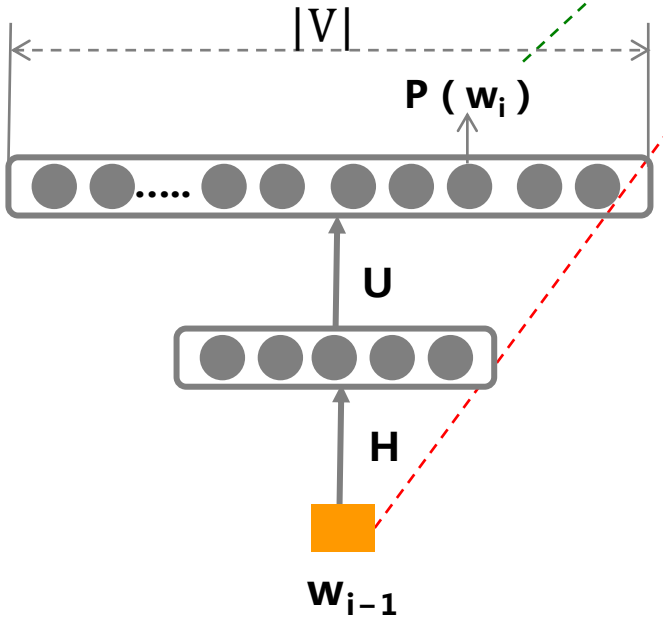
# 8.1 神经网络语言模型–NNLM

## NNLM模型（2-gram）

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

参数如何用神经网络学习

■ NNLM模型结构




softmax(y)

输出层：$p(w_i|w_{i-1}) = \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|}\exp(y(v_k))}$

$y(w_i)=b^2+U(\tanh(XH+b^1))$

隐藏层：$h=\tanh(XH+b^1)$

输入层：X: 词 $w_{i-1}$

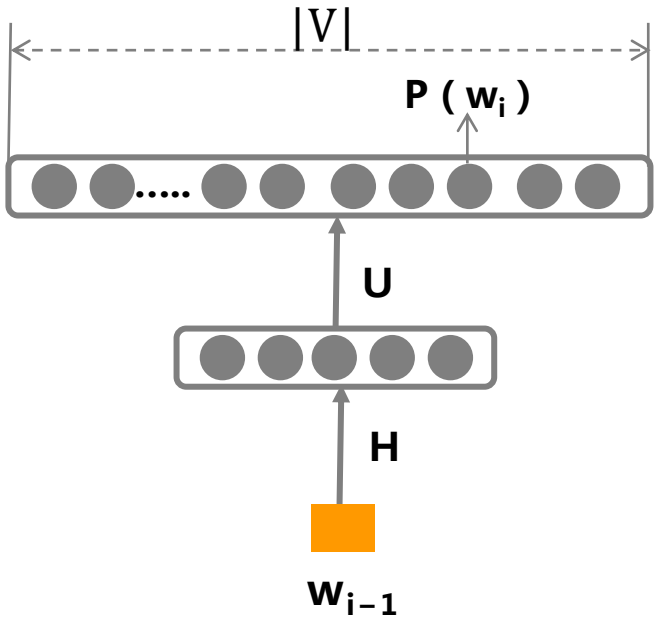
输出层有|V|个元素，V是有限词表包括未登录词标识UNK和句子开始和结束补齐符号，一般在10000≈1000000左右，常见规模 70000左右

# 8.1 神经网络语言模型–NNLM

## NNLM模型（2-gram）

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

■ NNLM模型结构




输入: $X: w_{i-1}$

输出: $p(w_i|w_{i-1})$

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

运算关系:

$$p(w_i|w_{i-1}) = \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|}\exp(y(v_k))}$$

$$y(w_i)=b^2+U(\tanh(XH+b^1))$$

# 8.1 神经网络语言模型–NNLM

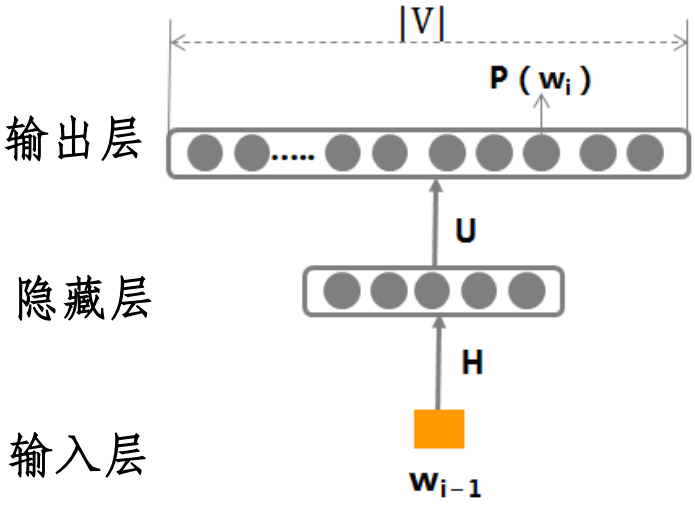
## ■ NNLM模型学习( 2-gram )

**参数：** $\theta = \{ H, U, b^1, b^2 \}$

**输出：** $p(w_i | w_{i-1})$

$$= \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|} \exp(y(v_k))}$$

$y(w_i) = b^2 + U(\tanh(XH + b^1))$




● **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 $L(Y, P(Y|X)) = -\log P(Y|X)$

对于整个语料而言，语言模型需要最大化：

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

● **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

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

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

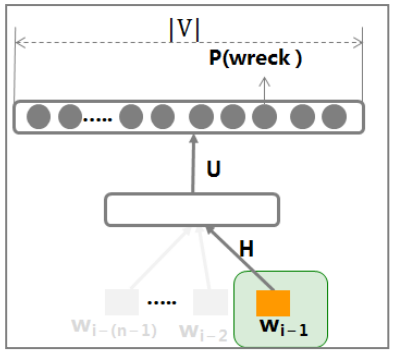
# 8.1 神经网络语言模型–NNLM

## ■ NNLM模型预测 （2-gram）

**例：** P（“wreck a nice beach”）

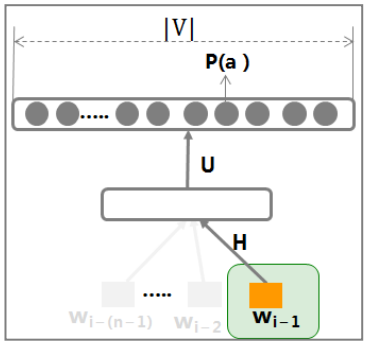
=P(wreck | START)P(a | wreck)P(nice | a)P(beach | nice)

P(wreck | START)



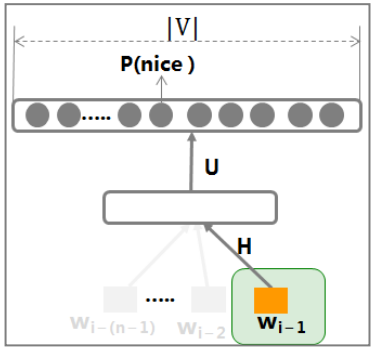

“START”

P(a | wreck)



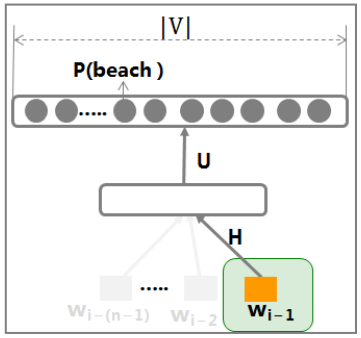

“wreck”

P(nice | a)




“a”

P(beach | nice)




“nice”

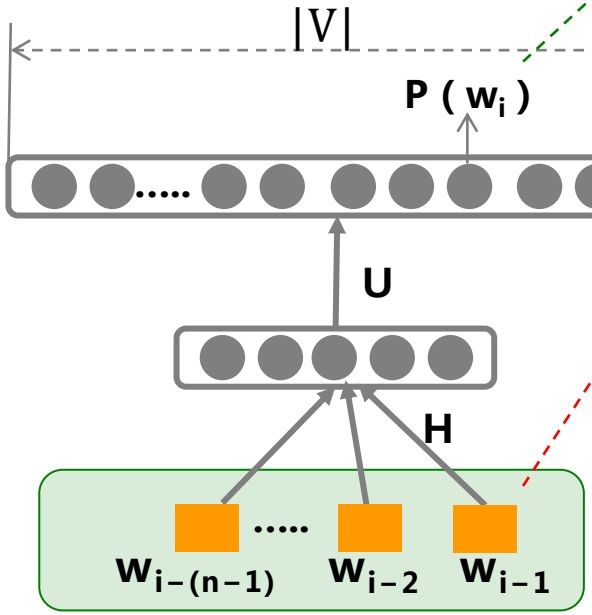
# 8.1 神经网络语言模型–NNLM

## NNLM模型（n-gram）

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

参数如何用神经网络学习

■ NNLM模型结构




softmax(y)

输出层：$p(w_i|w_{i-(n-1)}\ \ldots\ w_{i-1})=\frac{\exp(y(w_i))}{\sum_{k=1}^{|V|}\exp(y(v_k))}$

$y(w_i)=b^2+U(\tanh(XH+b^1))$

隐藏层：$h=\tanh(XH+b^1)$

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

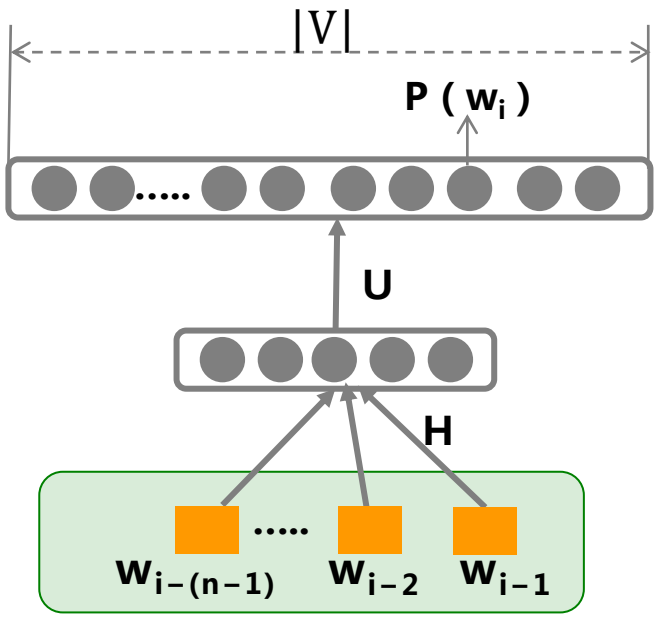
词以什么形式输入网络→ 词向量问题

# 8.1 神经网络语言模型–NNLM

## NNLM模型（n-gram）

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

■ NNLM模型结构




**输入:** $X: w_{i-(n-1)}\ldots w_{i-2}, w_{i-1}$

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

**参数：** $\theta=\{H, U, b^1, b^2\}$

**运算关系：**

$$p(w_i|w_{i-(n-1)}\ldots w_{i-1}) = \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|}\exp(y(v_k))}$$

$$y(w_i)=b^2+U(\tanh(XH+b^1))$$

# 8.1 神经网络语言模型–NNLM

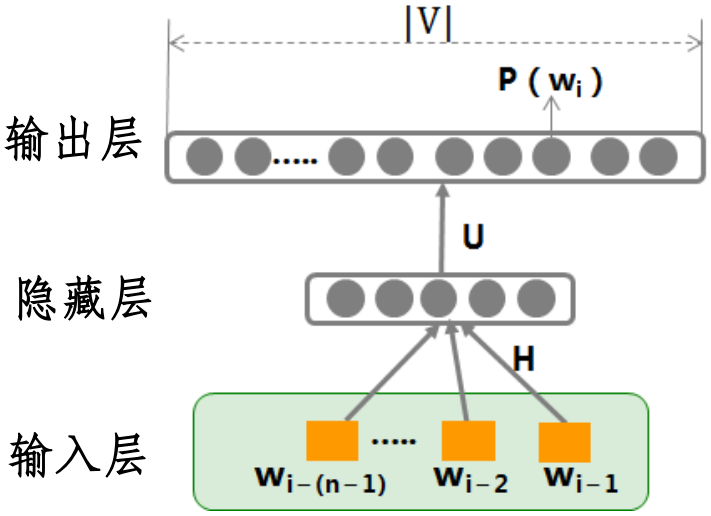
## ■ NNLM模型学习（n-gram）

**参数：** $\theta = \{ H, U, b^1, b^2 \}$

**输出：** $p(w_i | w_{i-(n-1)} \cdots w_{i-1})$

$$= \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|} \exp(y(v_k))}$$

$y(w_i) = b^2 + U(\tanh(XH + b^1))$




## ● 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 $L(Y, P(Y|X)) = -\log P(Y|X)$

对于整个语料而言，语言模型需要最大化：

$$\sum_{w_{i-(n-1)} \cdots i \in D} \log P(w_i | w_{i-(n-1)}, \ldots, w_{i-1})$$

## ● 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

$$\theta \leftarrow \theta + \alpha \frac{\partial \log P(w_i | w_{i-(n-1)}, \ldots, w_{i-1})}{\partial \theta}$$

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

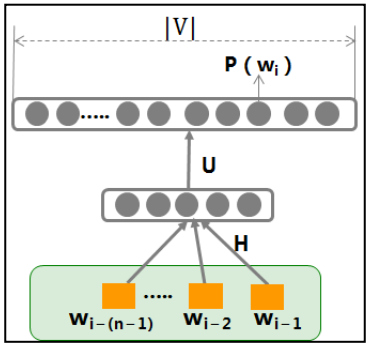
# 8.1 神经网络语言模型–NNLM

■ **NNLM模型预测** **（n-gram）** 如 n=4

**例：** P（“wreck a very nice beach”）

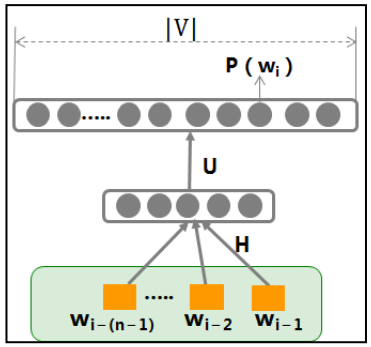
=P(very | START wreck a )P(nice | wreck a very ) P(beach | a very nice )

P(very | START wreck a )



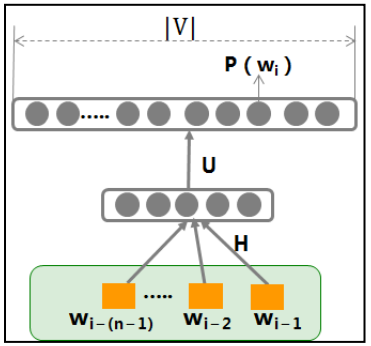

“START wreck a ”

P(nice | wreck a very )




“wreck a very ”

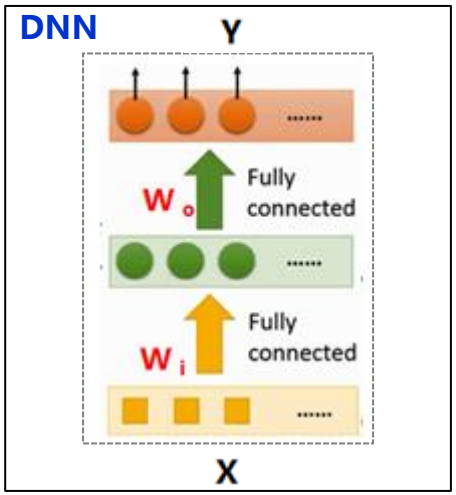
P(beach | a very nice )

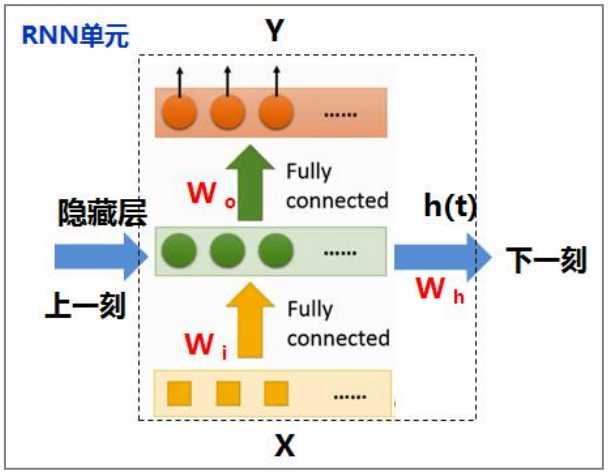


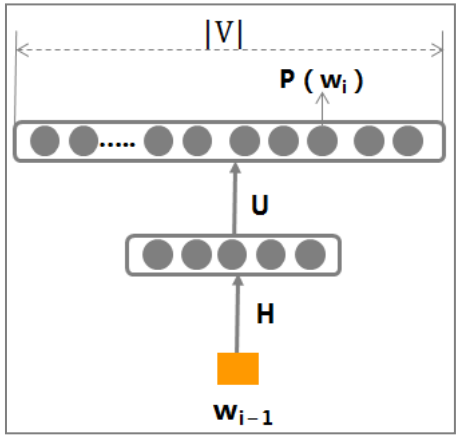

“a very nice ”

# 8.1 神经网络语言模型–RNNLM

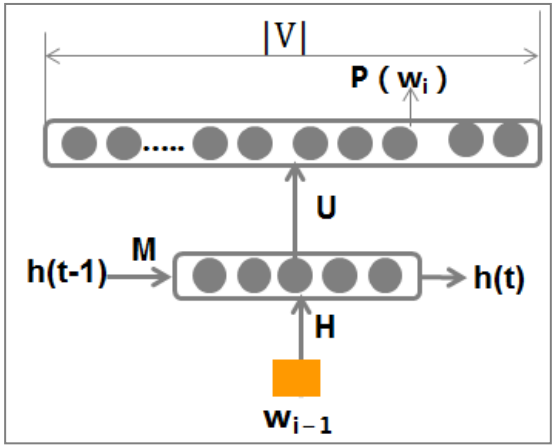
## RNNLM模型




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

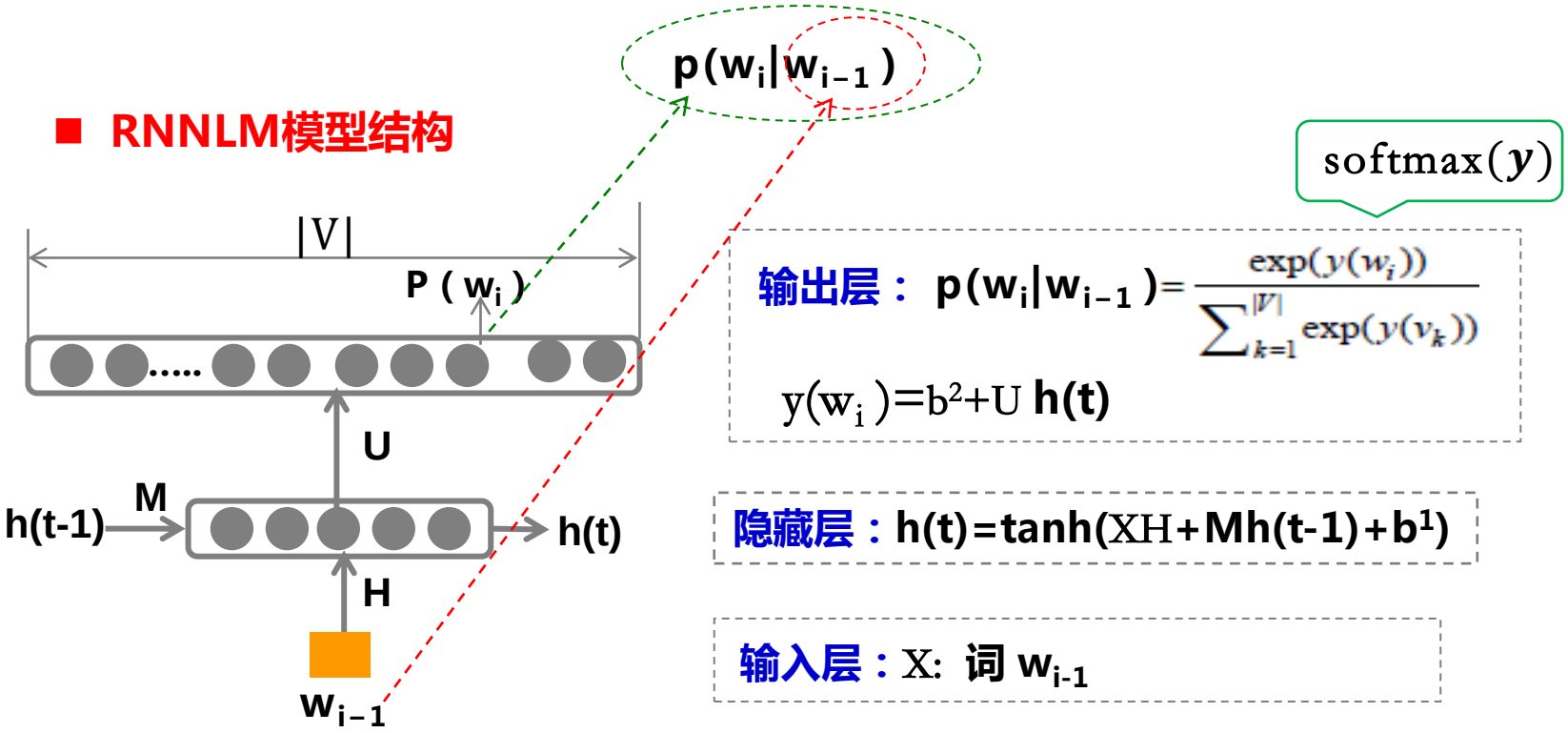



$p(w_i|w_{i-1})$

# 8.1 神经网络语言模型–RNNLM

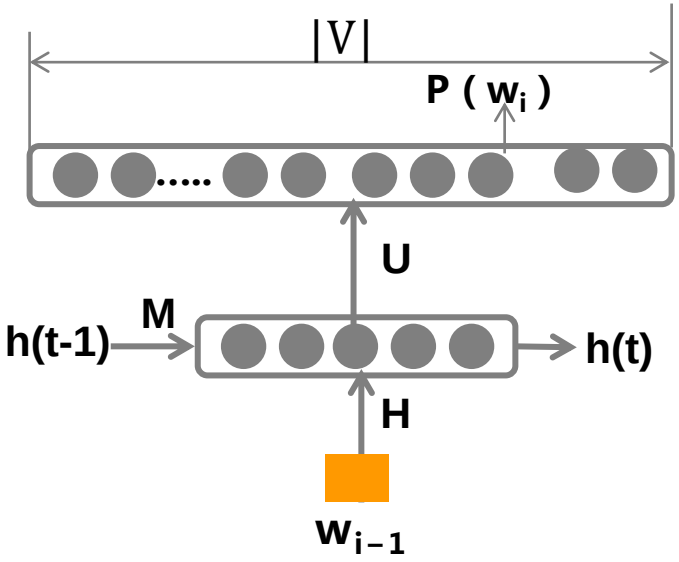
## RNNLM模型

■ RNNLM模型结构



# 8.1 神经网络语言模型–RNNLM

■ **RNNLM模型结构**




**输入:** X: $w_{i-1}$ 和 $h(t-1)$

**输出：** $p(w_i|w_{i-1})$ 和 $h(t)$ :(内部隐层)

**参数：** $\theta=\{H, U, M, b^1, b^2\}$

**运算关系：**

$$p(w_i|w_{i-1}) = \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|}\exp(y(v_k))}$$

$$y(w_i)=b^2+U(\tanh(XH+Mh(t-1)+b^1))$$

$$h(t)=\tanh(XH+Mh(t-1)+b^1)$$

# 8.1 神经网络语言模型–RNNLM

## ■ RNNLM模型学习

**参数：** $\theta = \{ H, U, M, b^1, b^2 \}$

**输出：** $p(w_i | w_{i-1}) = \dfrac{\exp(y(w_i))}{\sum_{k=1}^{|V|} \exp(y(v_k))}$

$y(w_i) = b^2 + U(\tanh(XH + Mh(t-1) + b^1))$

### ● 语料：（“无监督”）

文本： $S = w_1, w_2, \ldots, w_n$ ,

……

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

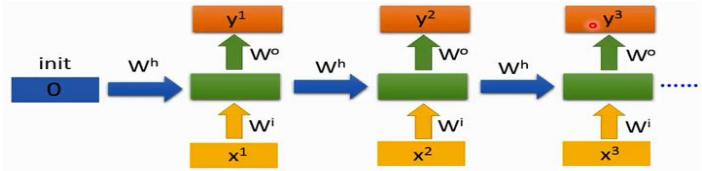
$\hat{Y}$ ： $w_1, w_2, \ldots, w_{n-1}\ w_n$

### ● 目标函数：

对于整个语料，语言模型需要最大化

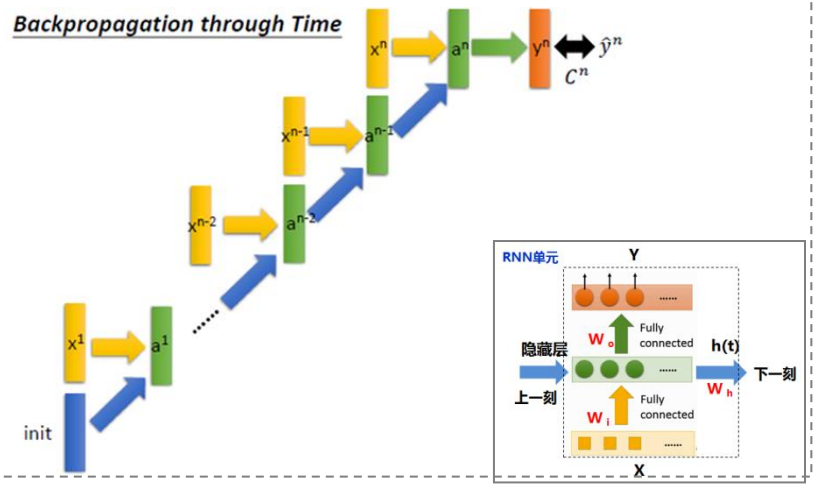
$$\sum_{w_{i-1}} \sum_{i \in D} \log P(w_i \mid w_{i-1})$$

**RNN网络**



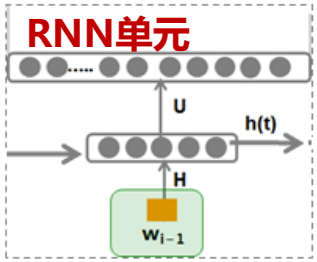

### ● 参数训练：

(BPTT) 随机梯度下降法优化训练目标:



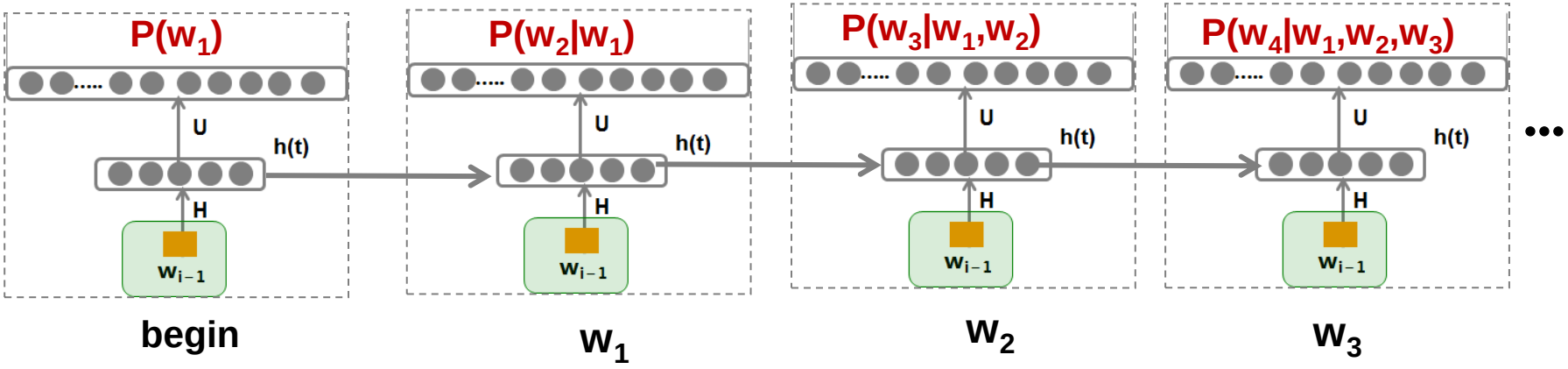
# 8.1 神经网络语言模型–RNNLM

## RNNLM模型预测




例： $P(w_1, w_2, w_3, \cdots, w_n)$

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




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。

# 内 容 提 要

8.1　神经网络语言模型

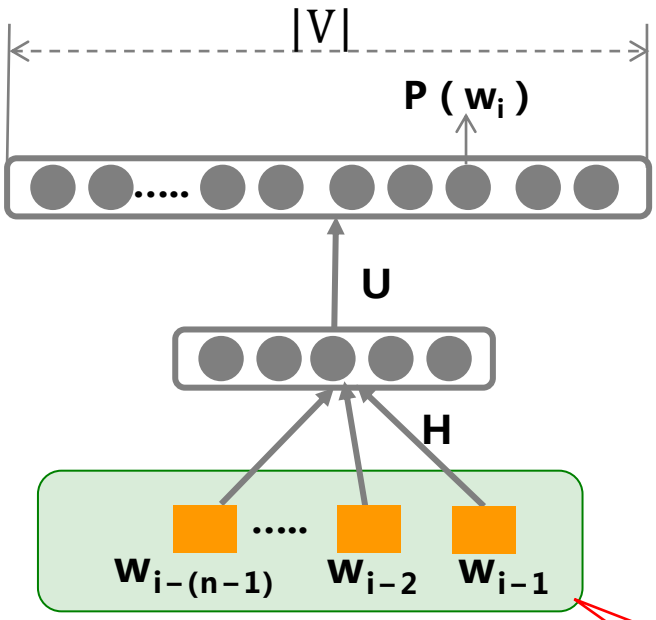
8.2　神经网络词向量

# 8.2 神经网络词向量

**问题引入：**

**在使用深度学习技术解决自然语言处理任务时，最基础的问题是词的表示。**

**NNLM模型结构**




**输出层：** $p(w_i|w_{i-(n-1)} \cdots w_{i-1}) = \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|}\exp(y(v_k))}$

$y(w_i) = b^2 + U(\tanh(XH + b^1))$

**隐藏层：** $h = \tanh(XH + b^1)$

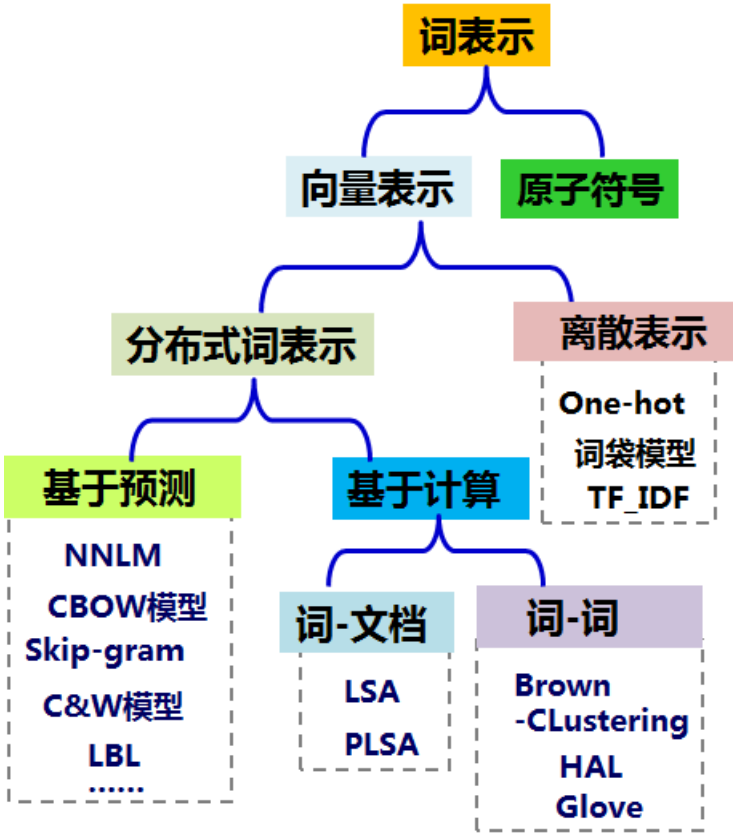
**输入层：** X: n-1个词 $w_{i-(n-1)}, \ldots, w_{i1}$

**神经网络输入需是向量，用什么样的向量表示一个“词”？→ 词的表示问题**

# 8.2 神经网络词向量

## 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。




■ **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

# 8.2 神经网络词向量

**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

**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。

**核心思想：用一个词附近的其他词来表示该词**

# 8.2 神经网络词向量

## 经典分布表示模型

| 名称 | 上下文 | 上下文与目标词之间的建模（技术手段） |
| --- | --- | --- |
| LSA/LSI<br>HAL<br>GloVe<br>Jones & Mewhort | 文档<br>词<br>词<br>n-gram | 矩阵 |
| Brown Clustering | 词 | 聚类 |
| Skip-gram<br>CBOW<br>Order<br>LBL<br>NNLM<br>C&W | 词<br>n-gram（加权）<br>n-gram（线性组合）<br>n-gram（线性组合）<br>n-gram（非线性组合）<br>n-gram（非线性组合） | 神经网络 |

- **基于计算的分布式**

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。

- **基于预测的分布式表示（神经网络词向量）**

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 。

# 8.2 神经网络词向量

## 词向量与神经网络

词向量模型可以从大规模无标注语料中自动学习到句法和语义信息，基于词向量的神经网络模型也为多项自然语言处理任务带来了性能的提升，在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。

# 8.2 神经网络词向量

## NNLM模型-词向量

- Xu 等人在2000 年首次使用神经网络求解二元语言模型【1】
- 2001年，Bengio 等人正式提出神经网络语言模型（Neural Network Language Model ，NNLM ）【2,3】。

  该模型**在学习语言模型的同时，也得到了词向量。**

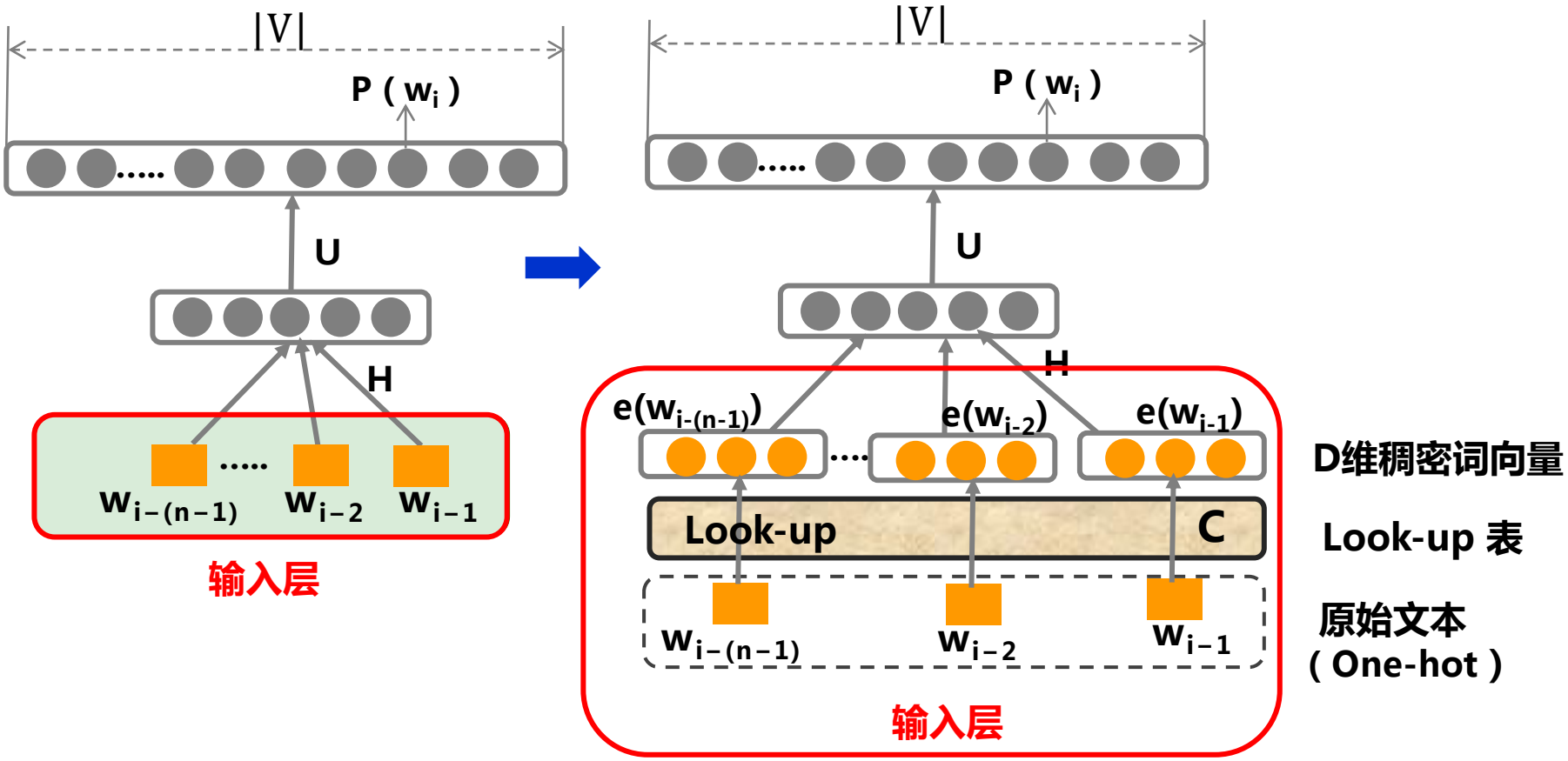

【1】Wei Xu and Alex Rudnicky. Can artificial neural networks learn language models In Sixth International Conference on Spoken Language Processing, 2000.

【2】Yoshua Bengio, et.al.. A neural probabilisticlanguage model. In Advances in Neural Information Processing Systems,2001

【3】Yoshua Bengio, et.al. ANeural Probabilistic Language Model. The Journal of Machine Learning Research,2003

# 8.2 神经网络词向量

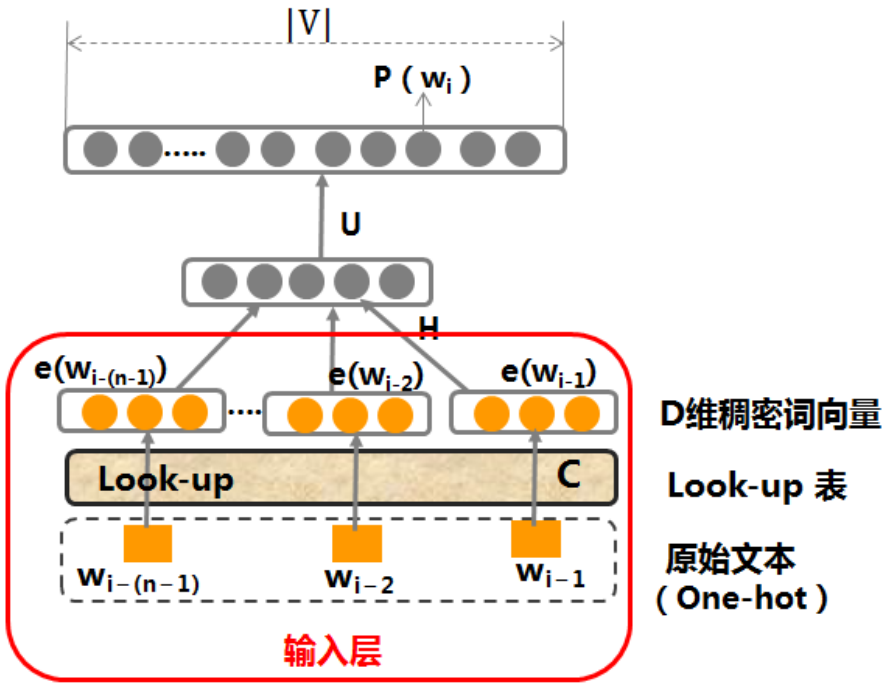
## NNLM模型输入表示问题




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

# 8.2 神经网络词向量

## NNLM模型输入表示问题




例，look-up 表 C 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 ]

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

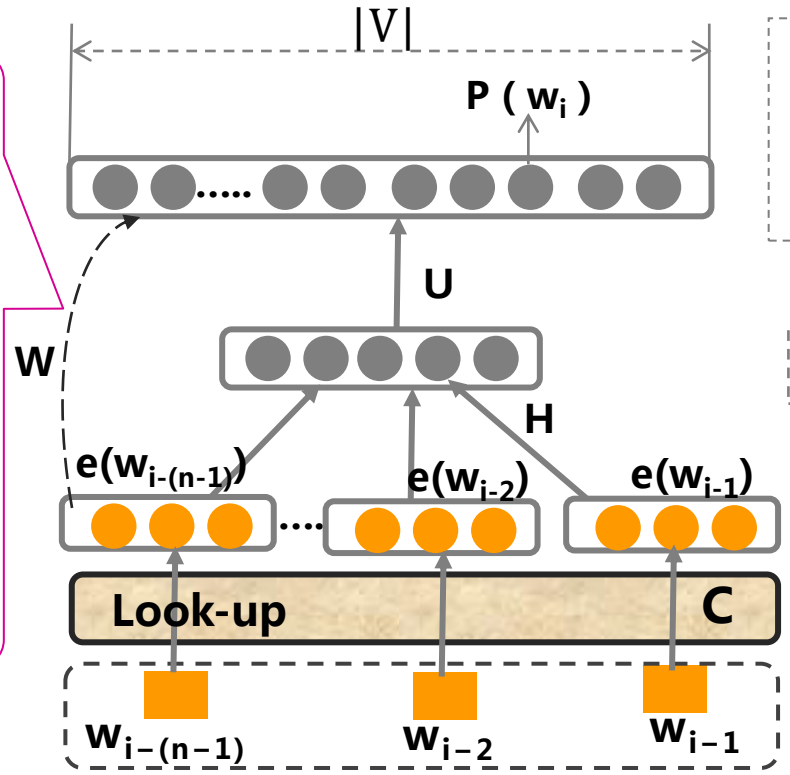
稠密向量表示**Look-up**表 **C** 是|D| x | V|维实数投影矩阵，| V|表示词表的大小，|D| 表示词向量e的维度（一般50维以上）；各词的词向量存于C中。词w 到其词向量e(w) 的转化是从该矩阵中取出相应的列。

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

# 8.2 神经网络词向量– NNLM

## NNLM模型-词向量

■ NNLM模型结构




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

softmax(y)

输出层：$p(w_i|w_{i-(n-1)} \dots w_{i-1}) = \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|}\exp(y(v_k))}$

$y(w_i) = b^2 + Wx + U h$

隐藏层：$h=\tanh(XH+b^1)$

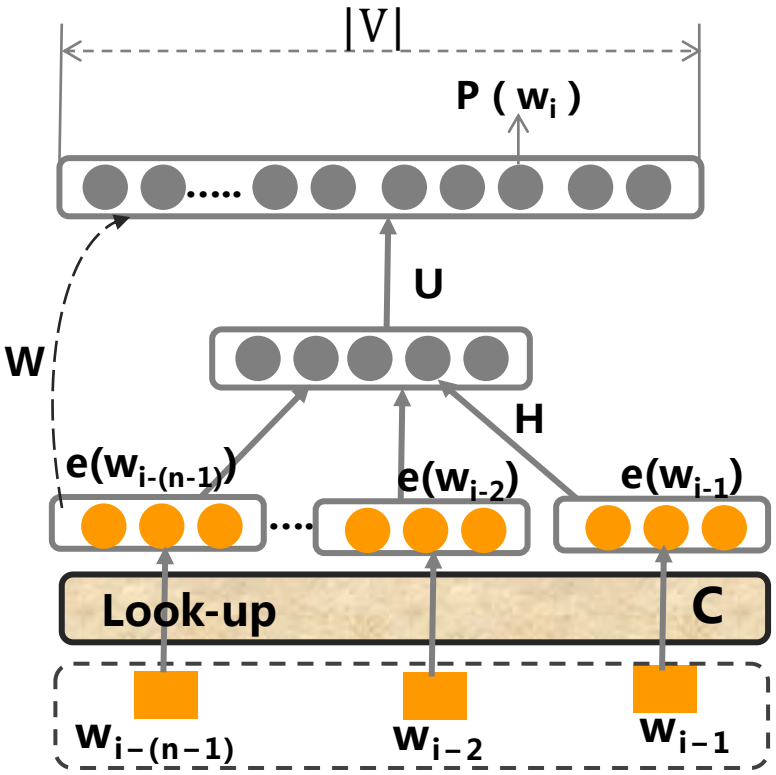
输入层：X: n-1个词 $w_{i-(n-1)},\dots,w_{i-1}$ 的词向量拼接 $X=[e(w_{i-(n-1)}) \dots e(w_{i-1})]$

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

# 8.2 神经网络词向量– NNLM

## NNLM模型-词向量

### ■ NNLM模型结构




**输入：** $X=[e(w_{i-(n-1)}).... e(w_{i-1})]$ (词向量初值)

**输出：** $p(w_i | w_{i-(n-1)} \dots w_{i-1})$

训练结束→训练好的词向量

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

**运算关系：**

$$p(w_i | w_{i-(n-1)} \dots w_{i-1}) = \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|} \exp(y(v_k))}$$

$$y(w_i) = b^2 + Wx + U(\tanh(XH + b^1))$$

$$X=[e(w_{i-(n-1)}).... e(w_{i-1})]$$

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

# 8.2 神经网络词向量– NNLM

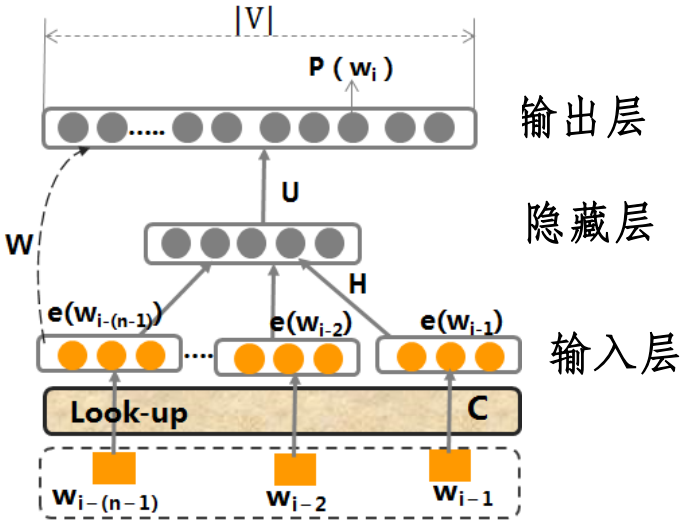
## ■ NNLM模型-词向量学习

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

**输出：** $p(w_i \mid w_{i-(n-1)} \cdots w_{i-1}) = \dfrac{\exp(y(w_i))}{\sum_{k=1}^{|V|} \exp(y(v_k))}$

$y(w_i) = b^2 + Wx + U(\tanh(XH + b^1))$

$X = [e(w_{i-(n-1)}) \ldots e(w_{i-1})]$




● **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 $L(Y, P(Y|X)) = -\log P(Y|X)$

对于整个语料而言，语言模型需要最大化：

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

● **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

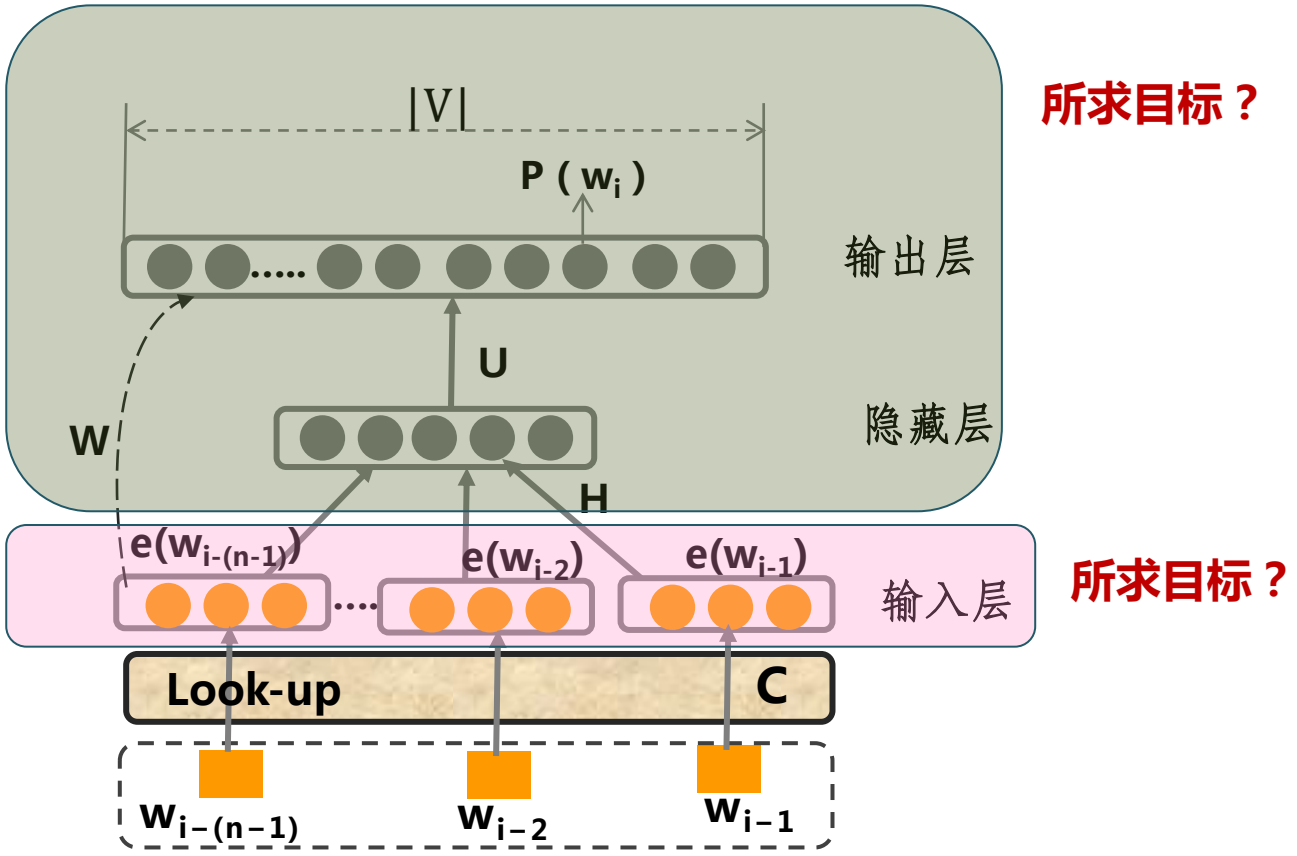
$$\theta \leftarrow \theta + \alpha \frac{\partial \log P(w_i \mid w_{i-(n-1)}, \ldots, w_{i-1})}{\partial \theta}$$

其中，$\alpha$学习率，$\theta = \{H, U, W, b^1, b^2, \text{词向量}\}$

# 8.2 神经网络词向量– NNLM

## ■ NNLM模型应用

- 语言模型




所求目标？

- 词向量

所求目标？

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

# 8.2 神经网络词向量– NNLM

**学得的词向量具有如下语言学特性**

分布假说：

- **语义相似的词，其词向量空间距离更相近**

***1-of-N Encoding***

apple = [ 1 0 0 0 0]

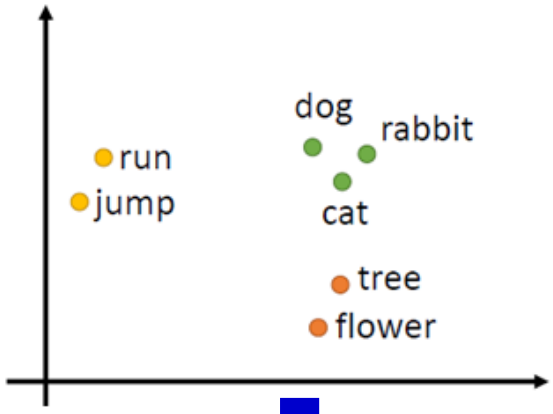
bag = [ 0 1 0 0 0]

cat = [ 0 0 1 0 0]

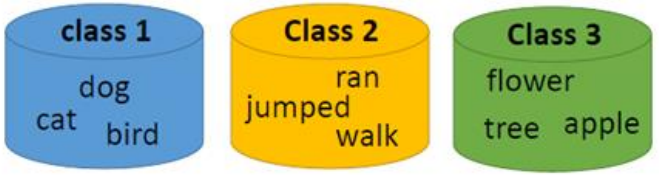
dog = [ 0 0 0 1 0]

elephant = [ 0 0 0 0 1]

***Word Embedding***




***Word Class***




**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**

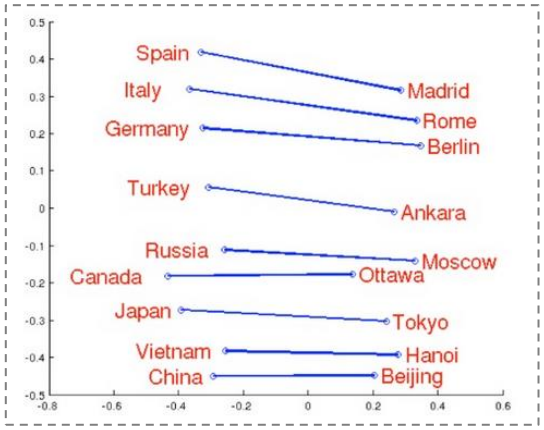
# 8.2 神经网络词向量– NNLM

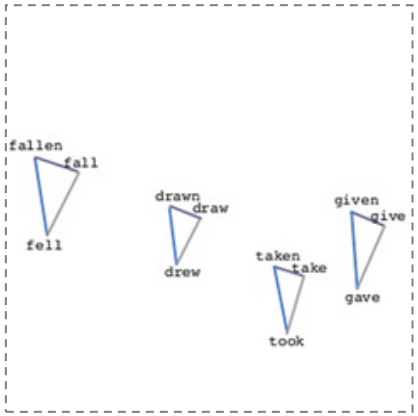
**学得的词向量具有如下语言学特性**

- **相似关系词对的词向量之差也相似**

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

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



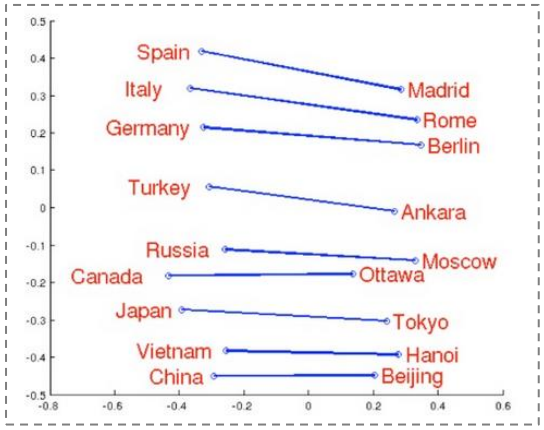




Source: http://www.slideshare.net/hustwj/cikm-keynotenov2014

# 8.2 神经网络词向量– NNLM

**词向量的语言学特性可以用来完成语义相关任务**

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

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

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

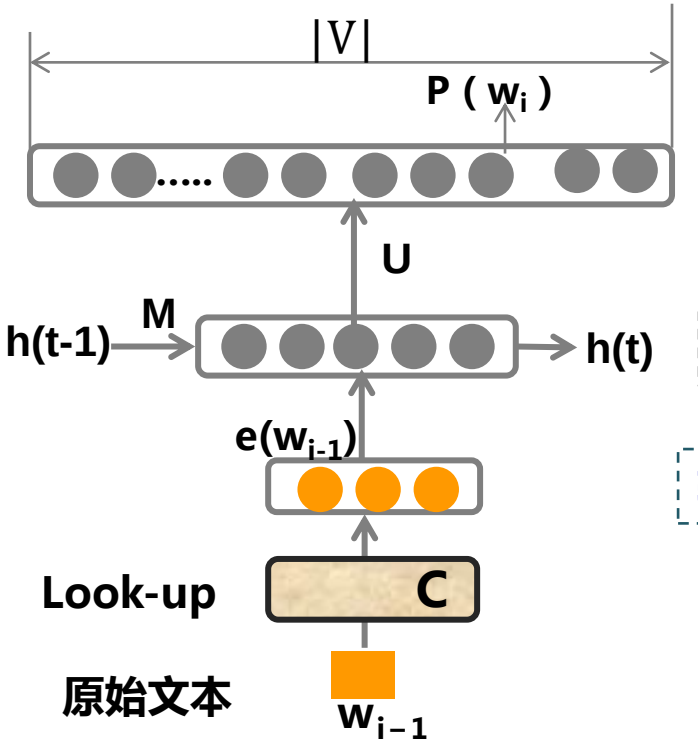
$$V(Germany) \approx \text{V(Berlin)} - \text{V(Rome)} + \text{V(Italy)}$$

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

# 8.2 神经网络词向量– RNNLM

## RNNLM模型-词向量

■ NNLM模型结构




softmax(y)

输出层：$p(w_i|w_{i-1}) = \frac{\exp(y(w_i))}{\sum_{k=1}^{|V|}\exp(y(v_k))}$

$y(w_i)=b^2+U\,h(t)$

隐藏层：$h(t)=\tanh(X+Mh(t-1)+b^1)$

输入层：X: 词 $w_{i-1}$的词向量 $X= e(w_{i-1})$

输入: $X= e(w_{i-1})$ (词向量初值)，$h(t-1)$

输出： $p(w_i|w_{i-1})$ ， $h(t)$ :(内部隐层)

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

训练结束→训练好的词向量

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

# 8.2 神经网络词向量– RNNLM

## ■ RNNLM模型-词向量学习

**参数：** $\theta = \{U, M, b^1, b^2, \text{词向量}\}$

**输出：** $p(w_i \mid w_{i-1}) = \dfrac{\exp(y(w_i))}{\sum_{k=1}^{|V|} \exp(y(v_k))}$

$y(w_i) = b^2 + U(\tanh(XH + Mh(t-1) + b^1))$

● **语料：（“无监督”）**

文本： $S = w_1, w_2, \ldots, w_n$ ,

……

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

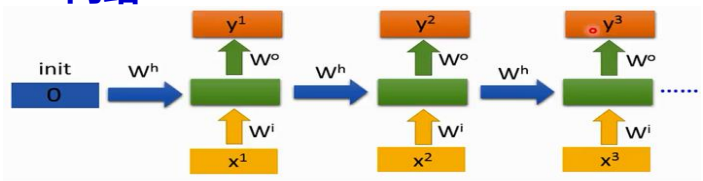
$\hat{Y}$ ： $w_1, w_2, \ldots, w_{n-1}\ w_n$

● **目标函数：**

对于整个语料，语言模型需要最大化

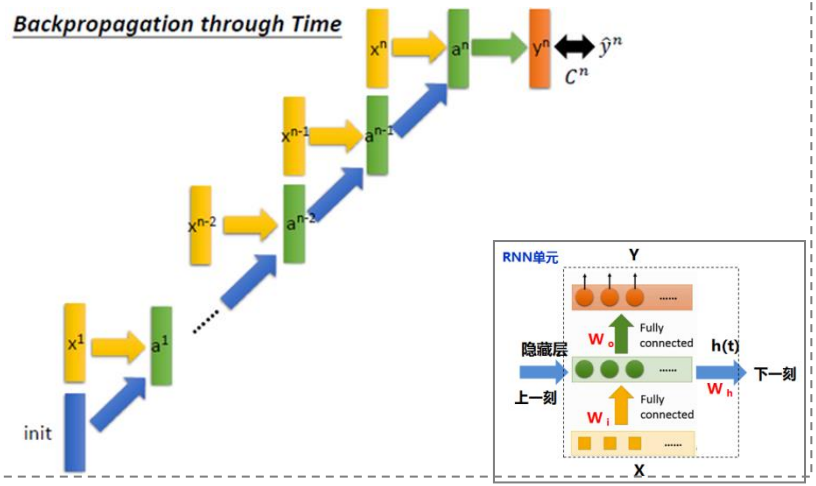
$$\sum_{w_{i-1}} \sum_{i \in D} \log P(w_i \mid w_{i-1})$$

**RNN网络**




● **参数训练：**

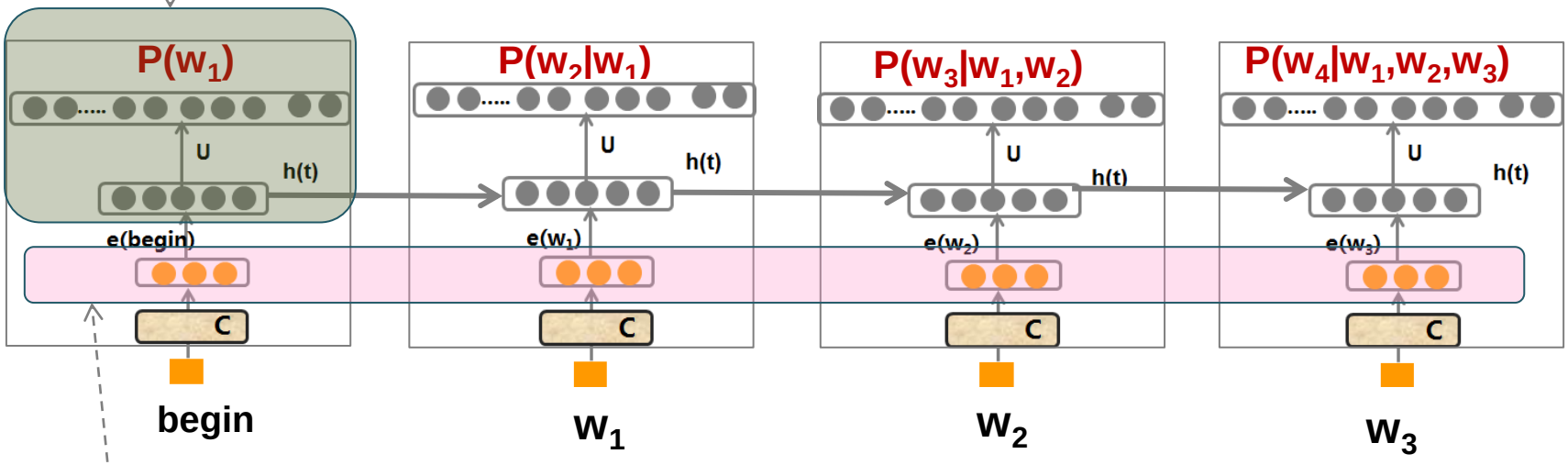
(BPTT) 随机梯度下降法优化训练目标:



# 8.2 神经网络词向量– RNNLM

## RNNLM模型应用

- 语言模型




- 词向量

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

# 8.2 神经网络词向量– C&W

**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。

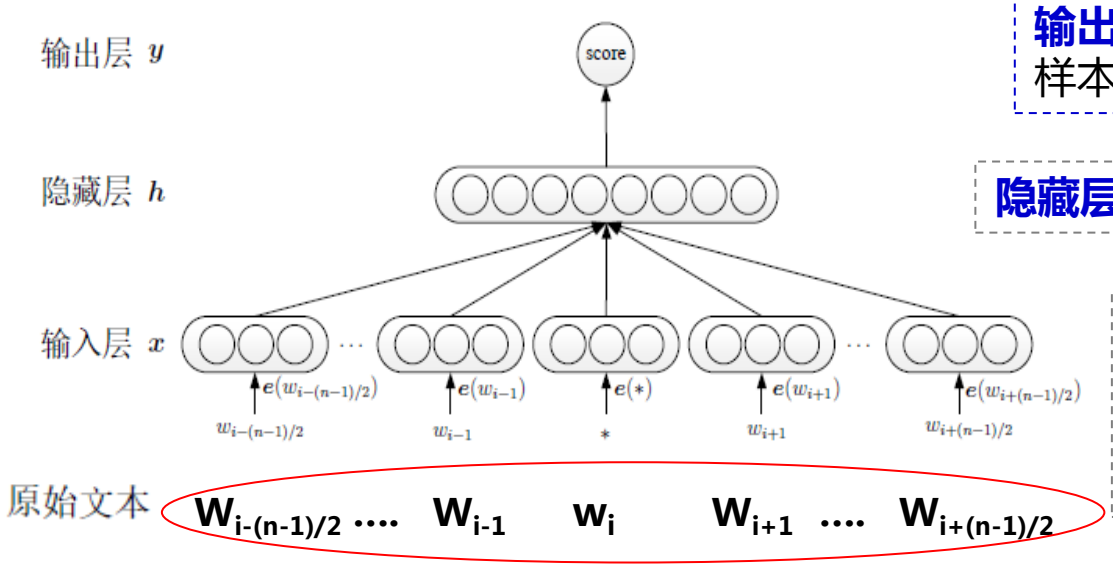
**特点**：C&W 模型的目标函数是求目标词$w$与其上下文$c$的联合打分，而其他模型均为根据上下文$c$，预测目标词$w$。

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

# 8.2 神经网络词向量– C&W

## C&W 模型

### ■ 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

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

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

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

# 8.2 神经网络词向量– C&W

## ■ 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

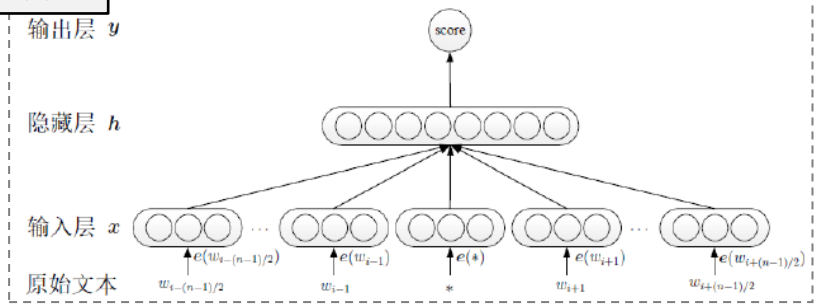
### ● 参数训练：

采用pairwise的方式对文本片段进行优化，即可得词向量

# 8.2 神经网络词向量– C&W

## C&W模型与NNLN模型对比

**C&W模型**

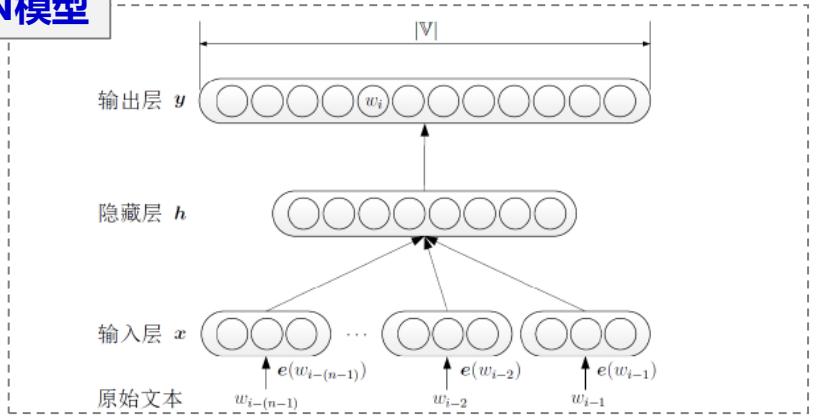



目标词在输入层

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

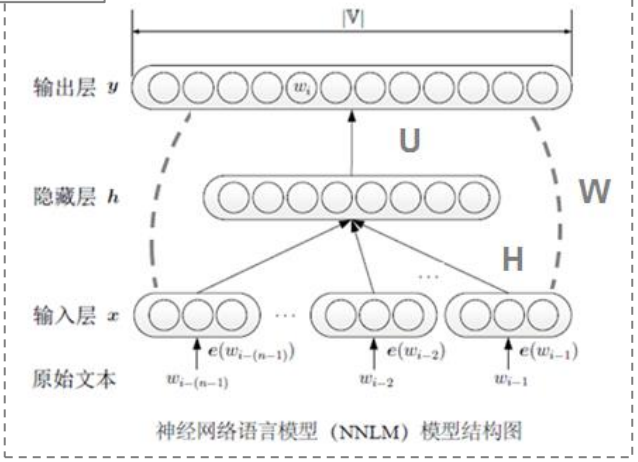
# 8.2 神经网络词向量– CBOW/Skip-gram

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。

## 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，以更高效的方法获取词向量。

**NNLN模型**

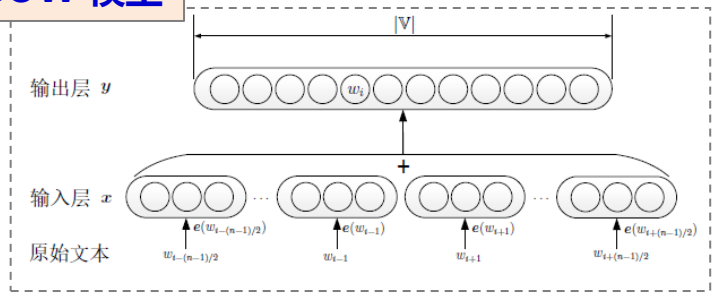



神经网络语言模型（NNLM）模型结构图

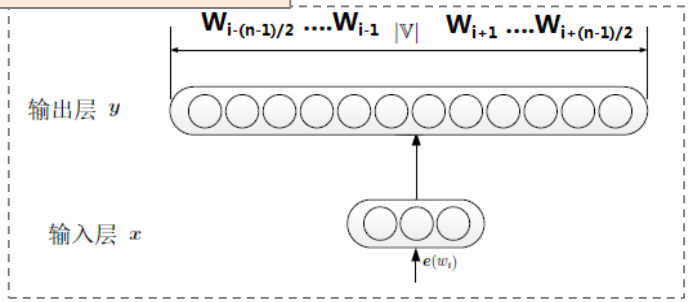
**简化：**

1. 除去隐藏层
2. 除去词序

**CBOW 模型**



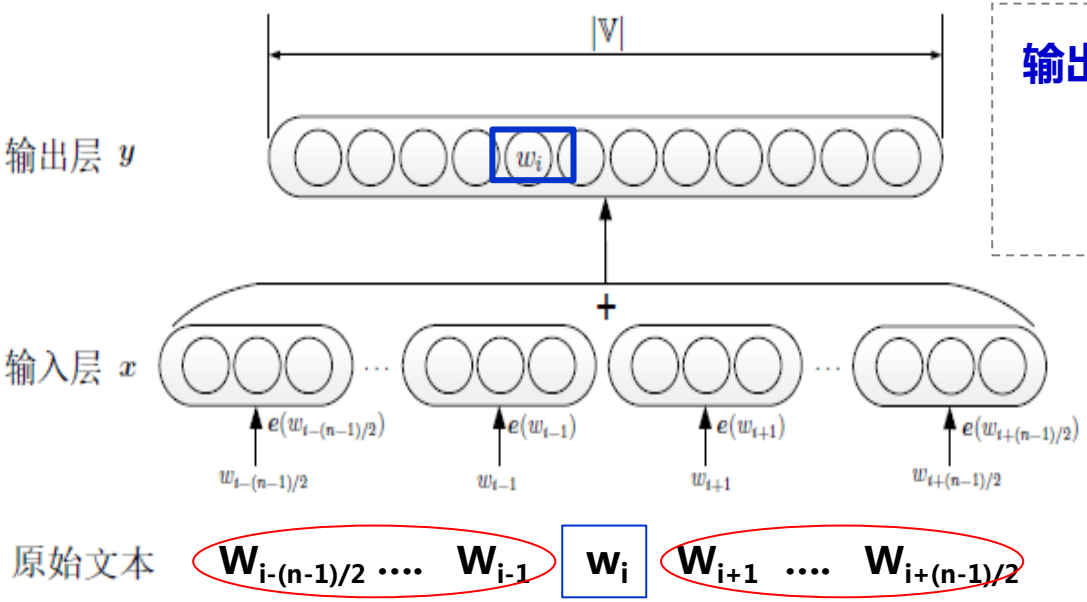

**Skip-gram模型**




在NNLM、RNNLM 和C&W 模型的基础上，简化模型，保留核心部分。

# 8.2 神经网络词向量– CBOW

## CBOW 模型




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

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

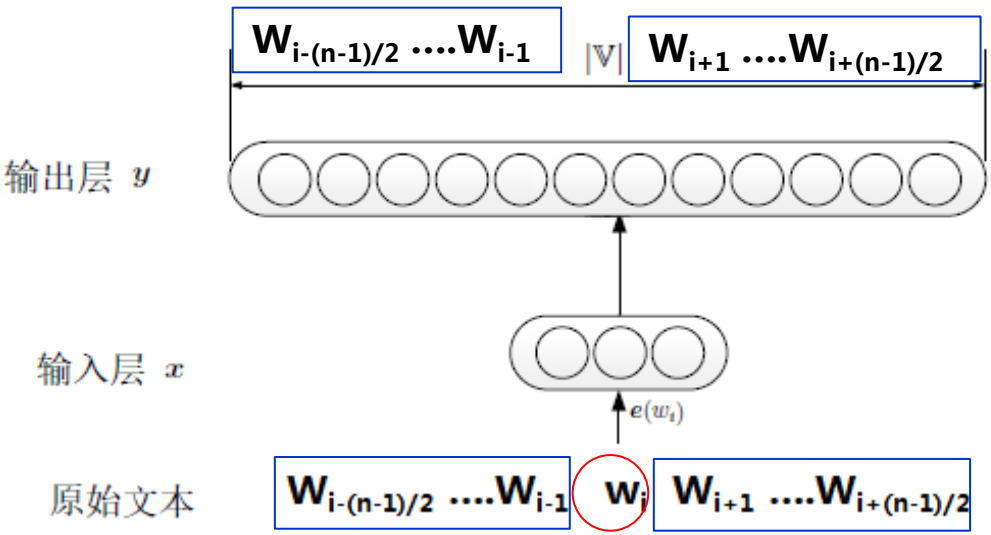
- **优化目标：** **最大化** $\sum_{(w,c)\in D}\log P(w|c)$
- **参数训练：** **梯度下降法**

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

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

# 8.2 神经网络词向量– Skip-gram

## Skip-gram模型




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

**输入层：** X: 词 $w_i$ 的词向量

- **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

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

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

# 8.2 神经网络词向量– CBOW/Skip-gram

**softmax优化问题：**

由于需要得到 y 中的每个分量的值，计算非常耗时，实际运算中采用**层级softmax** 函数优化。

**Hierachical softmax 基本思想：**

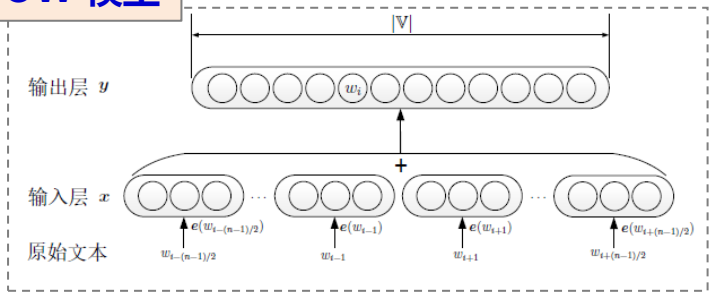
对于词典中的任意词w，哈夫曼树中必然存在一条从根结点到该词所在的叶子结点的路径。将路径上存在的所有分支视为一次二分类，将每次分类的概率乘起来，就得到了最终p(w| Context(w))

**详情参阅：**

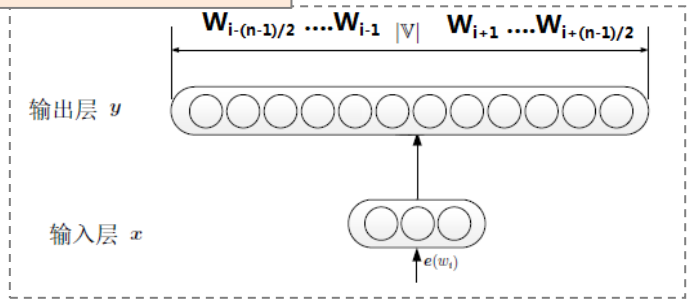
Frederic Morin and Yoshua Bengio. Hierarchical probabilistic neural network language model. In Proceedings of the international workshop on artificial intelligence and statistics, , 2005.

# 8.2 神经网络词向量– CBOW/Skip-gram

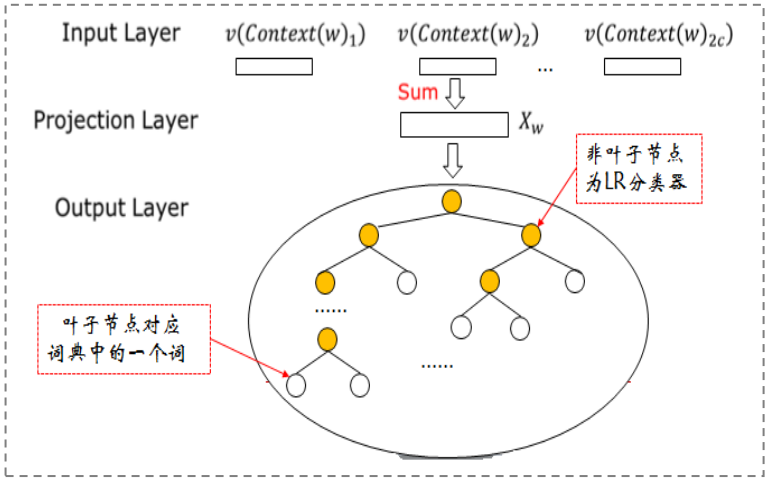
**CBOW 模型**




**Skip-gram模型**

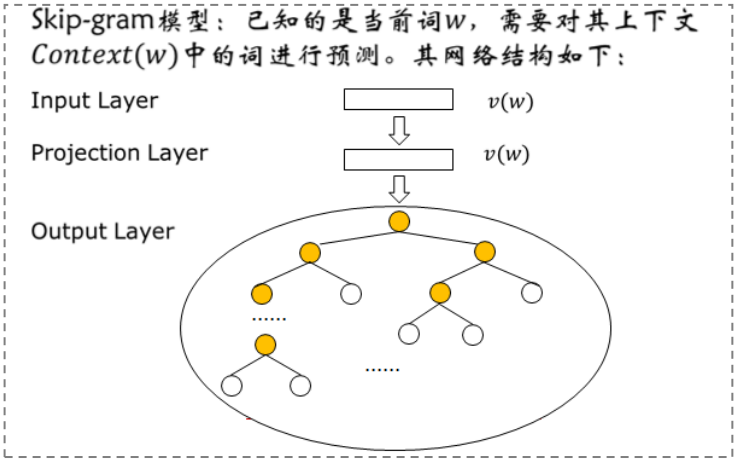



## CBOW Hierarchical Softmax




求： $p(w_i \mid Context_i)$

## Skip-gram Hierarchical Softmax




求： $p(Context_i \mid w_i)$

# 8.2 神经网络词向量– CBOW/Skip-gram

## 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 |

# 8.2 神经网络词向量– 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 |

# 8.2 神经网络词向量– 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 |

# 8.2 神经网络词向量

**构建词向量参考建议：**

**1.语料**：选择领域内的语料，对同领域的任务会有显著的提升；对于确定类型的语料，语料规模越大，词向量的性能越好。

**2.模型**：当语料较小时，应当选用模型结构最简单的Skip-gram 模型，而当语料较大时，选用CBOW 模型会有更好的效果。

简单的模型在绝大多数情况下已经足够好。预测目标词的模型比目标词与上下文呈组合关系的模型（C&W 模型）在多个任务中有更好的性能。

**3. 训练时**：迭代优化的终止条件最好根据具体任务的验证集来判断，或者近似地选取其它类似的任务作为指标.

**4. 词向量维度选择：**一般需要选择50 维及以上，特别当衡量词向量的语言学特性时，词向量的维度越大，效果越好。

# 8.2 神经网络词向量

## 词向量用途

- **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）**

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

# 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



**在 此 表 示 感 谢 ！**

信息内容安全技术国家工程实验室
National Engineering Laboratory for Information Security Technologies

# 谢谢各位！