

期中复习

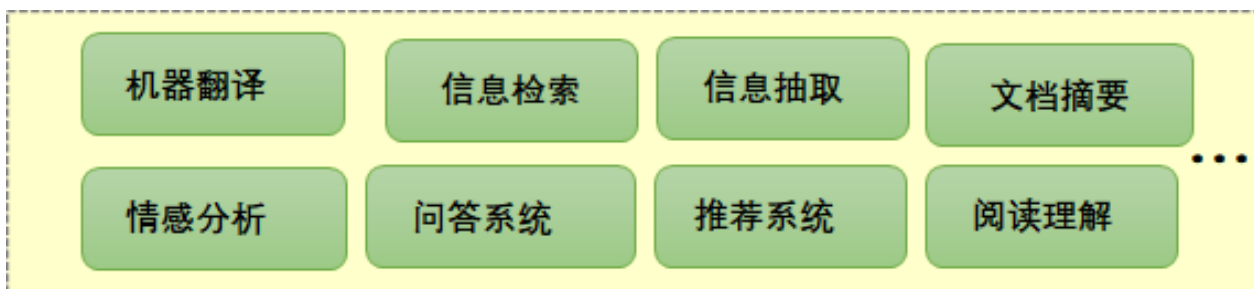
(上)

基于深度学习的自然语言处理课程内容

《自然语言处理》课程知识结构

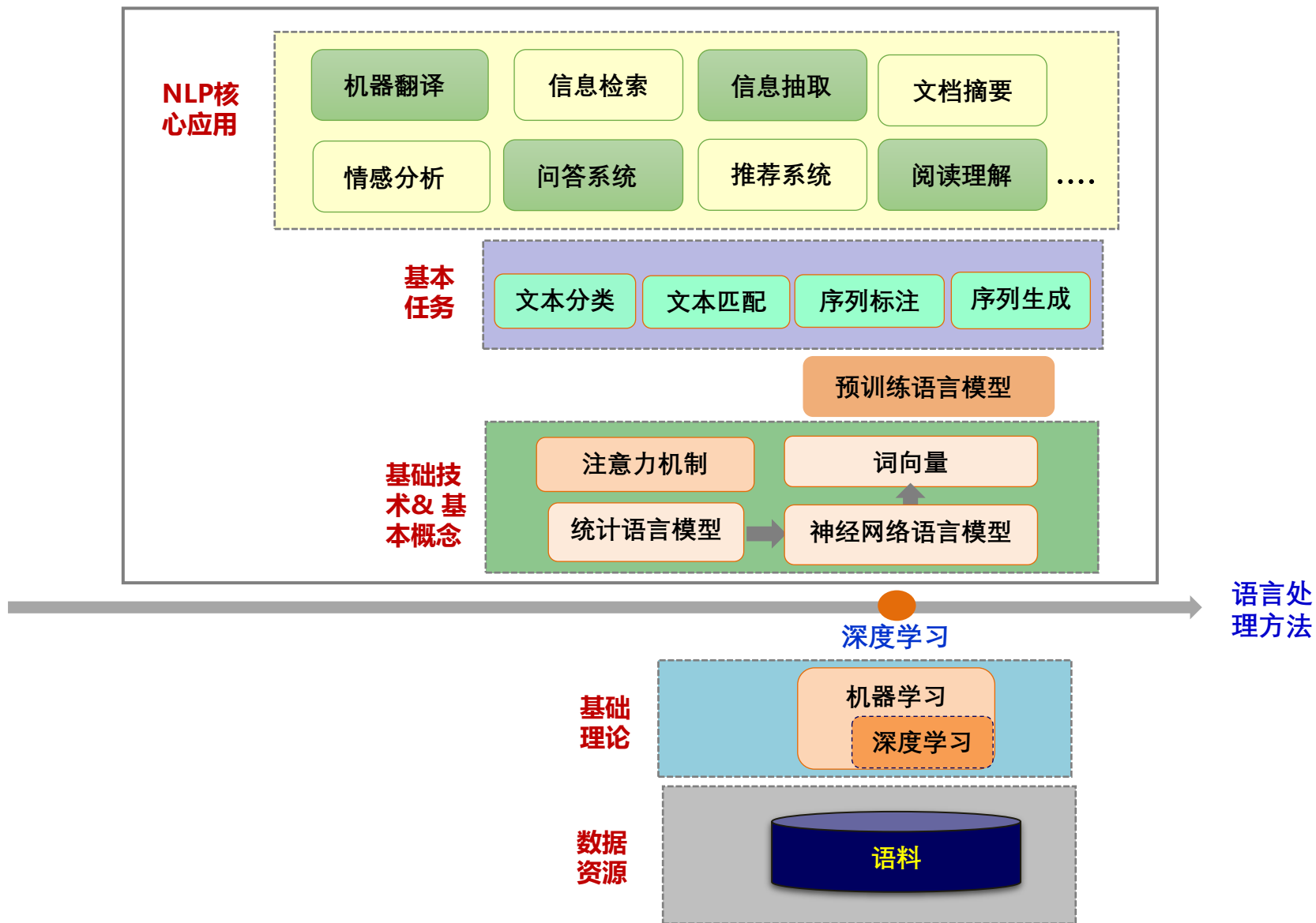
各类任务

每类任务是人类认知过程中对语言处理的真实需求



每类任务**相对独立**。根据处理方法不同可以分解为不同的子任务

基于深度学习的自然语言处理课程内容



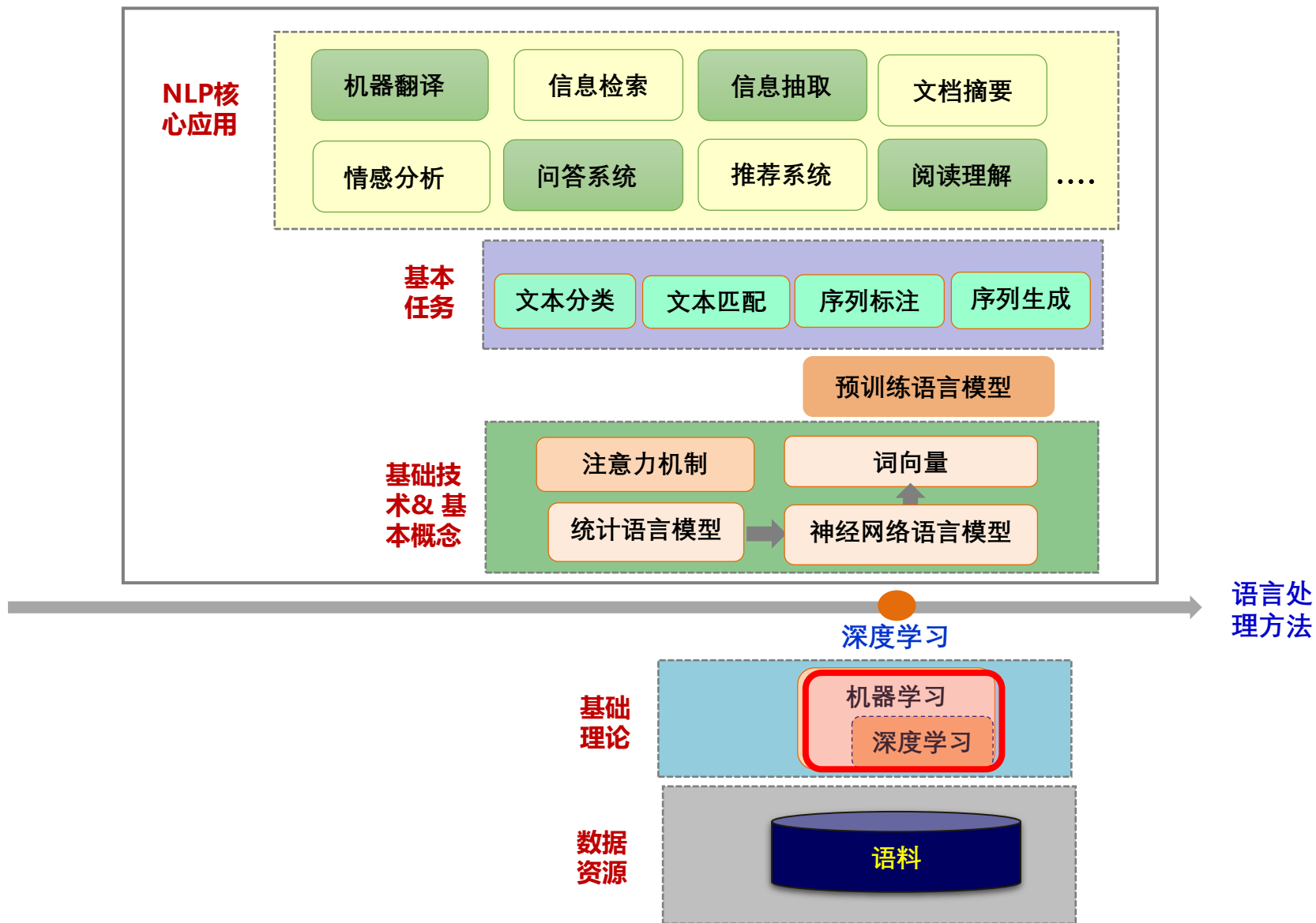
自然语言处理
Natural Language Processing

第 4-7 章 深度学习基础

授课教师：胡玥

授课时间：2020.9

基于深度学习的自然语言处理课程内容



第 4-7 章 深度学习基础

要求：

基本概念

- 神经网络调参方法？
- 梯度下降法有几种？各自特点？
- DNN 学习算法？
- RNN 学习算法？
- CNN 各层的作用？
- 图卷积的步骤？

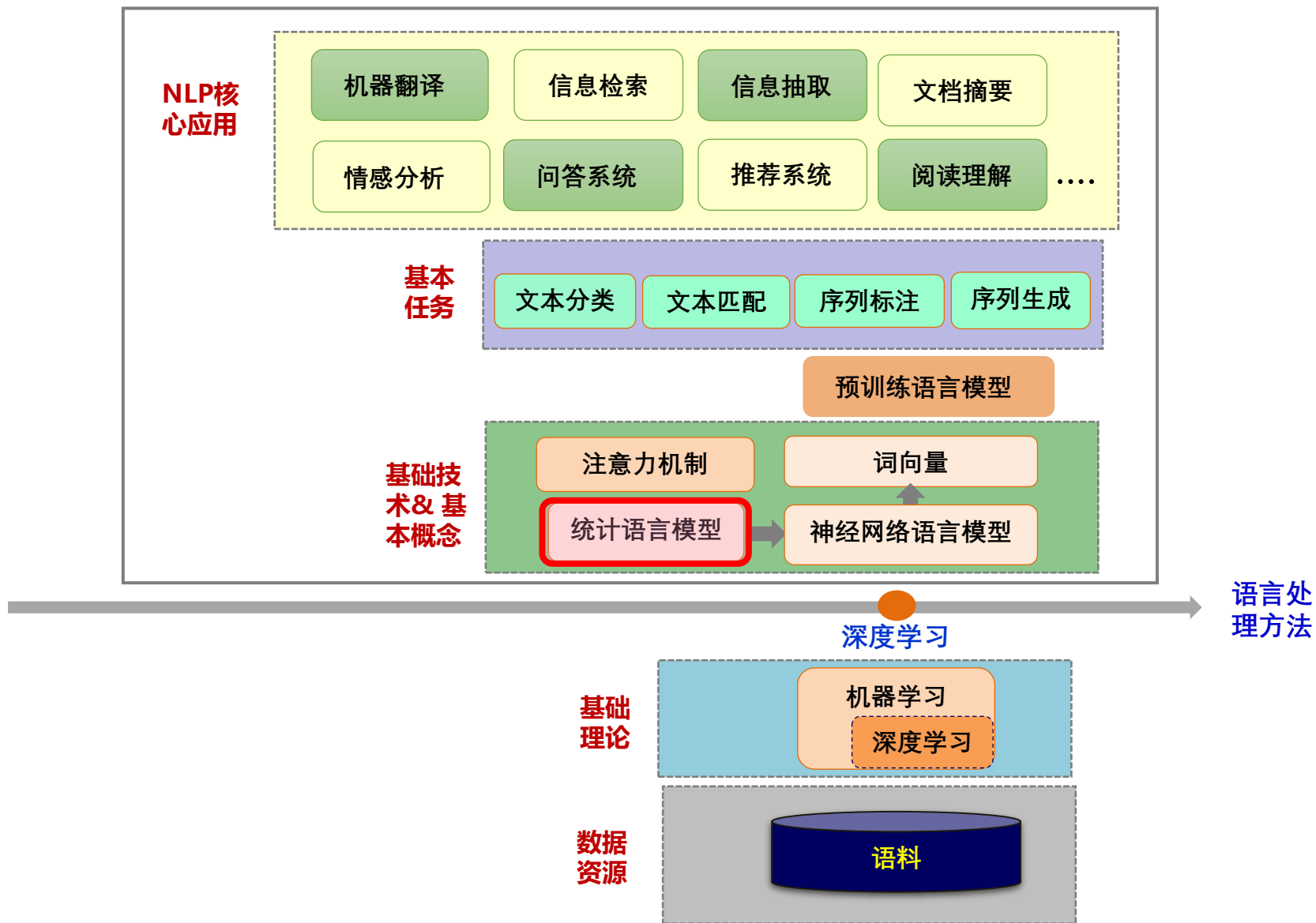
自然语言处理
Natural Language Processing

第 3 章 统计语言模型

授课教师：胡玥

授课时间：2020.9

基于深度学习的自然语言处理课程内容



3.1 语言模型基本概念

重点:

1. 语言模型的定义
2. 概率语言模型的学习方法
3. 概率语言模型存在问题
 - 由于参数数量问题需要对词 i 的历史简化 n -gram
 - 需要数据平滑

3.1 语言模型基本概念

语言模型思想

用句子 $S = w_1, w_2, \dots, w_n$ 的概率 $p(S)$ 刻画句子的合理性。
(用数学的方法描述语言规律)

句子概率 $p(S)$ 定义：

语句 $s = w_1 w_2 \dots w_n$ 的概率 $p(S)$ 定义为：

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

其中：当 $i=1$ 时, $p(w_1|w_0) = p(w_1)$

1. 语言模型

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

输入： 句子 S

输出： 句子概率 $p(S)$

参数： $p(w_i|w_1, \dots, w_{i-1})$

3.1 语言模型基本概念

- 由于参数数量问题需要对词 i 的历史简化 n -gram

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

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

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

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

3.2.1 参数估计

2. 参数学习的方法

对于 n -gram, 参数 $p(w_i | w_{i-n+1}^{i-1})$ 可由最大似然估计求得:

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

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。

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

3.2.1 参数估计

- 需要数据平滑

数据匮乏(稀疏) (*Sparse Data*) 引起零概率问题

如求, $p(\text{Cher read a book}) = ?$

$$= p(\text{Cher} | \langle \text{BOS} \rangle) \times p(\text{read} | \text{Cher}) \times p(\text{a} | \text{read}) \times \\ p(\text{book} | \text{a}) \times p(\langle \text{EOS} \rangle | \text{book})$$

$$p(\text{Cher} | \langle \text{BOS} \rangle) = \frac{c(\langle \text{BOS} \rangle \text{ Cher})}{\sum_w c(\langle \text{BOS} \rangle w)} = \frac{0}{3}$$

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

于是, $p(\text{Cher read a book}) = 0$

$\langle \text{BOS} \rangle \text{John read Moby Dick} \langle \text{EOS} \rangle$

$\langle \text{BOS} \rangle \text{Mary read a different book} \langle \text{EOS} \rangle$

$\langle \text{BOS} \rangle \text{She read a book by Cher} \langle \text{EOS} \rangle$

自然语言处理
Natural Language Processing


第 8 章 神经网络语言模型

授课教师：胡玥

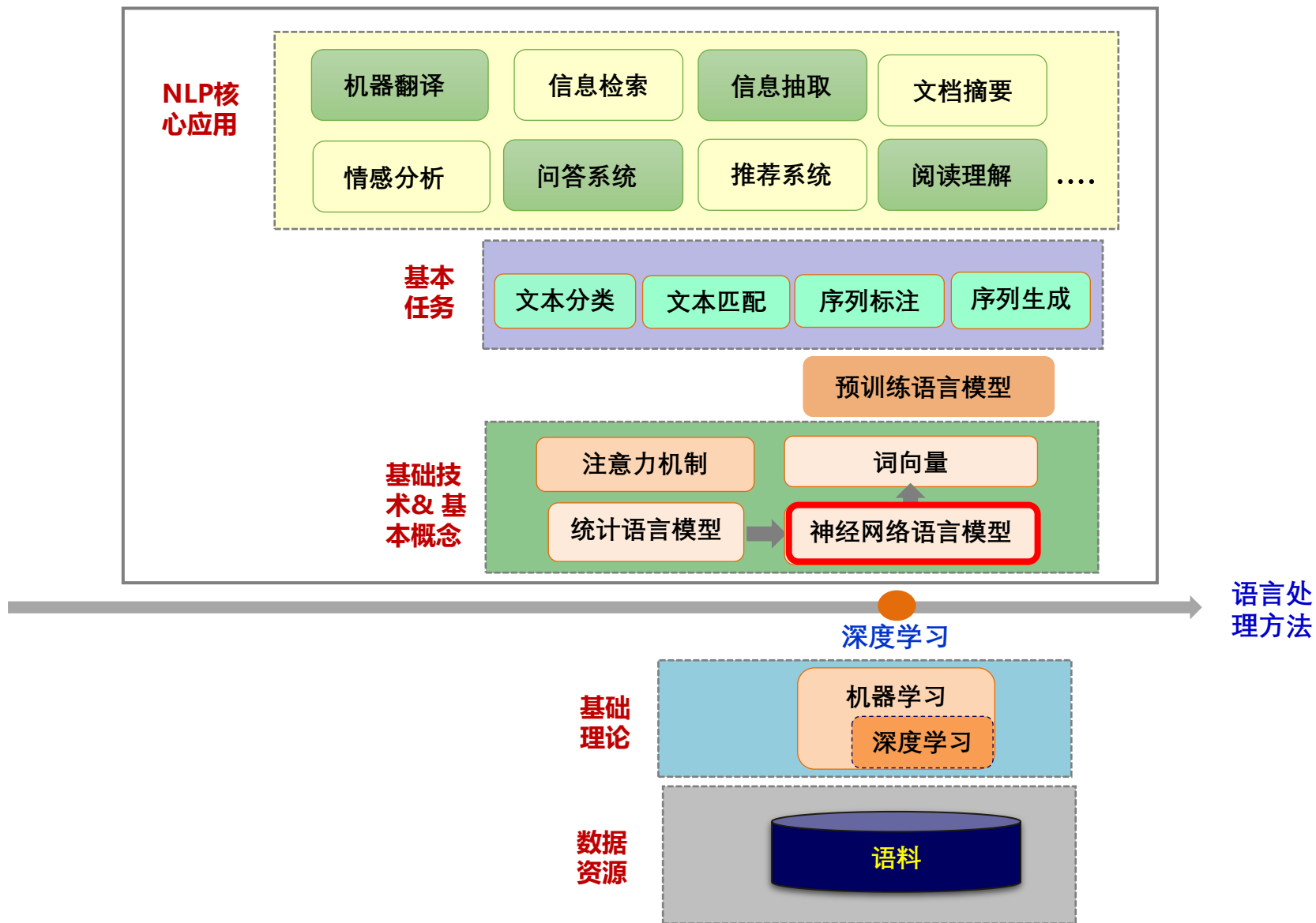
授课时间：2020.9

第 8 章 神经网络语言模型

重点：

1. 神经网络语言模型结构
2. 神经网络语言模型的学习方法 
3. RNN为什么能解决神经网络语言模型的“需历史简化”问题

基于深度学习的自然语言处理课程内容



8.1 概述

神经网络语言模型概念

语言模型：用句子 $S=w_1, w_2, \dots, w_n$ 的概率 $p(S)$ 来定量的刻画句子。

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

输入： 句子 S

输出： 句子概率 $p(S)$

参数： $p(w_i | w_1, \dots, w_{i-1})$

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

根据所用神经网络不同， 分为：

- NNLM 模型 （使用DNN）
- RNNLM 模型 （使用RNN）

内 容 提 要

8.1 概述

8.2 NNLM 模型

8.3 RNNLM 模型

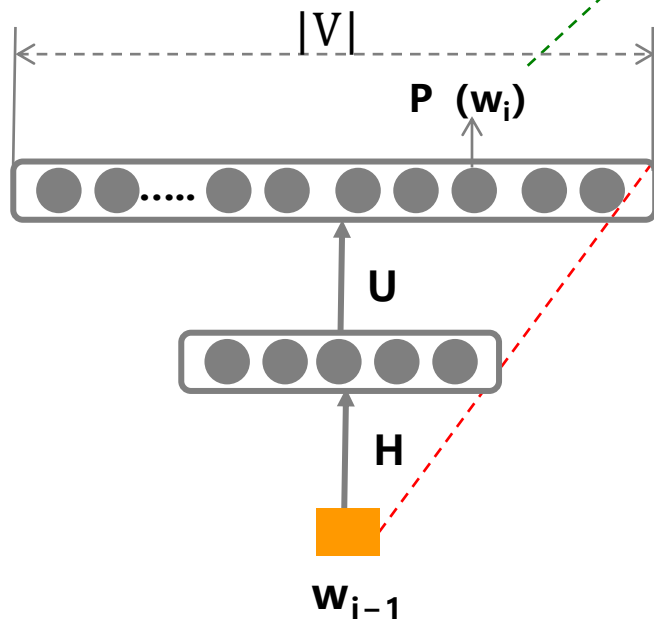
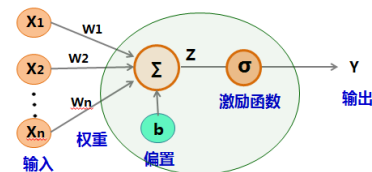
8.2 NNLM模型

(1) NNLM模型结构

2-gram:

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

语言模型参数



$\text{softmax}(y)$

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

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

$$\text{隐藏层: } h = \tanh(XH + b^1)$$

$$\text{输入层: } X: \text{词 } w_{i-1}$$

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

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 $10000 \approx 1000000$ 左右，常见规模70000左右

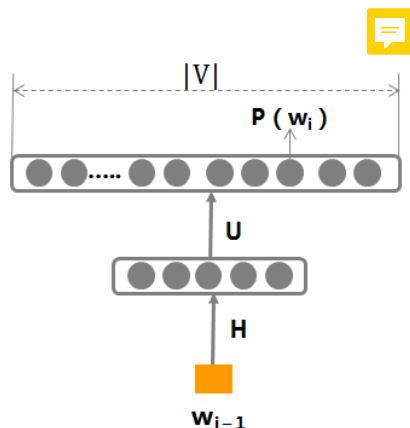
8.2 NNLM模型

(2) NNLM模型学习 (2-gram)

- 目标函数:

采用log损失函数

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



参数: $\theta = \{ H, U, b^1, b^2 \}$

- 语料: (“无监督”)

文本: $S = w_1, w_2, \dots, w_n, \dots$

实例: $X: w_{i-1}$

$\hat{Y}: w_i$

- 参数训练:

(BP) 随机梯度下降法优化训练目标:

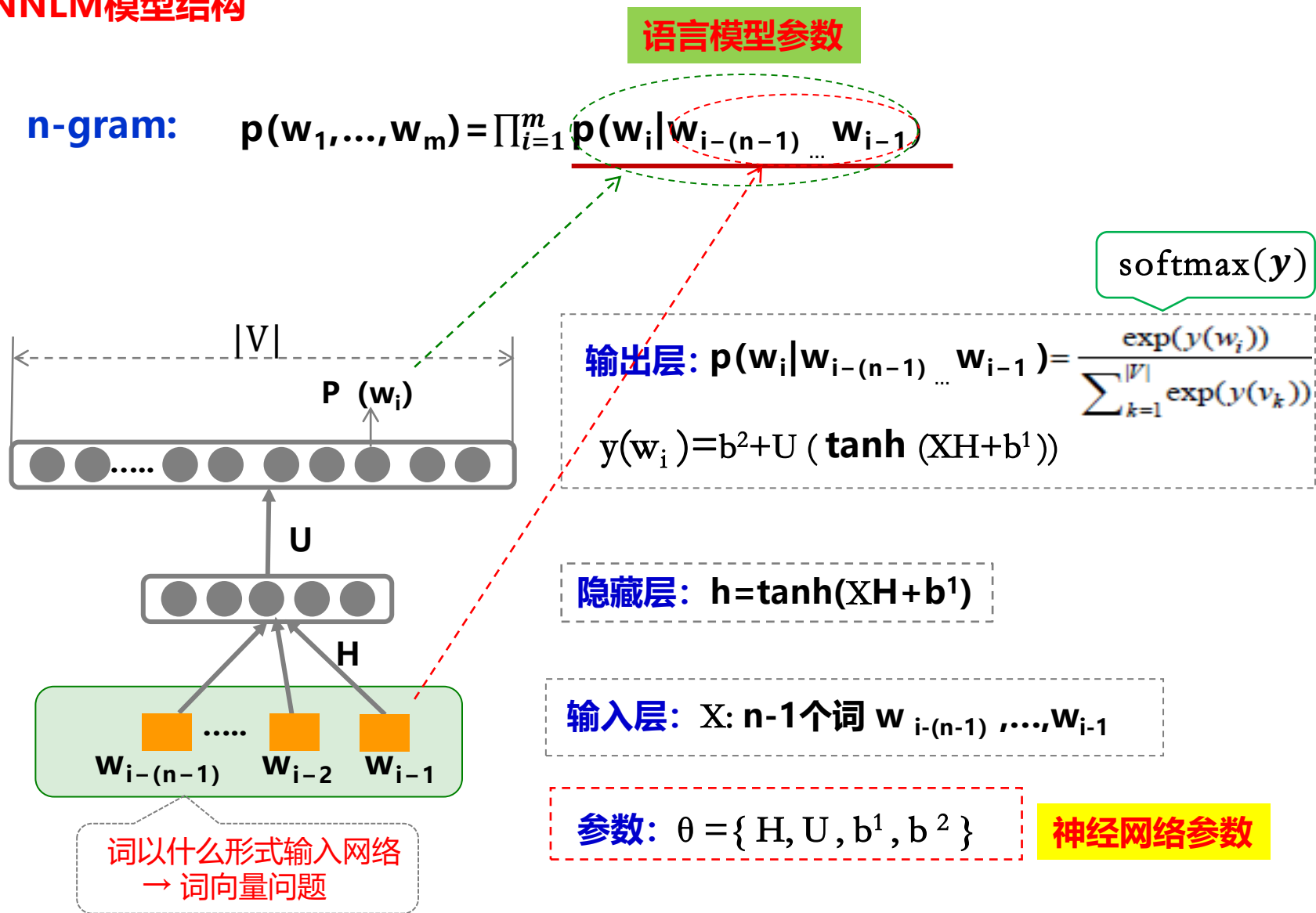
每次迭代, 随机从语料 D 中选取一段文本 $w_{i-(n-1)}, \dots, w_i$ 作为训练样本进行一次梯度迭代

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

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

8.2 NNLM模型

(1) NNLM模型结构



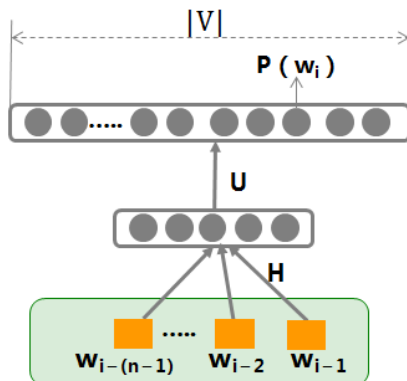
8.2 NNLM模型

(2) NNLM模型学习 (2-gram)

- 目标函数:

采用log损失函数

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



参数: $\theta = \{ H, U, b^1, b^2 \}$

- 语料: (“无监督”)

文本: $S = w_1, w_2, \dots, w_n, \dots$

实例: $X: w_1, w_2, \dots, w_{i-1}$

$\hat{Y}: w_i$

- 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 $w_{i-(n-1)}, \dots, w_i$ 作为训练样本进行一次梯度迭代

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

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

内 容 提 要

8.1 概述

8.2 NNLM 模型

8.3 RNNLM 模型

8.3 RNNLM 模型

(1) RNNLM模型结构

语言模型参数

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

softmax(y)

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

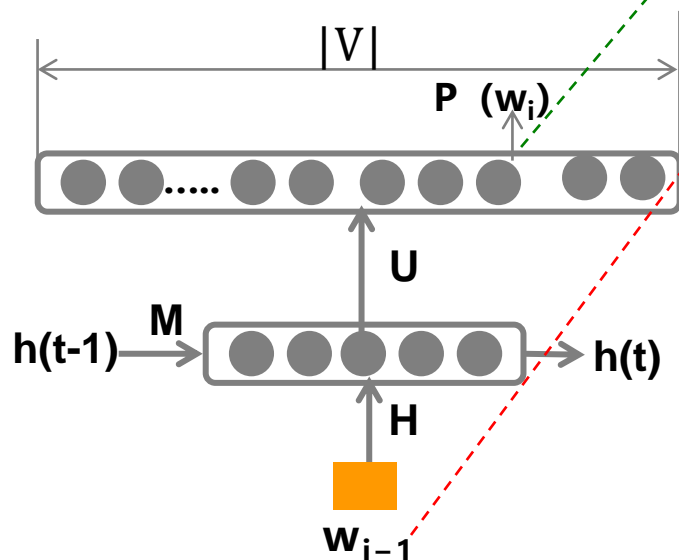
$$y(w_i) = b^2 + U h(t)$$

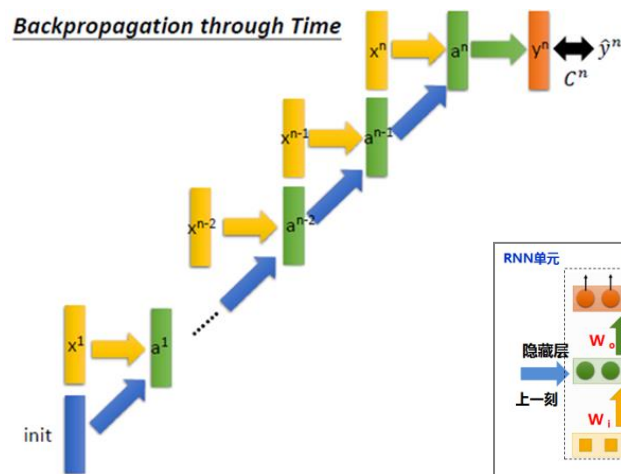
隐藏层: $h(t) = \tanh(XH + Mh(t-1) + b^1)$

输入层: X : 词 w_{i-1}

参数: $\theta = \{H, U, M, b^1, b^2\}$

神经网络参数





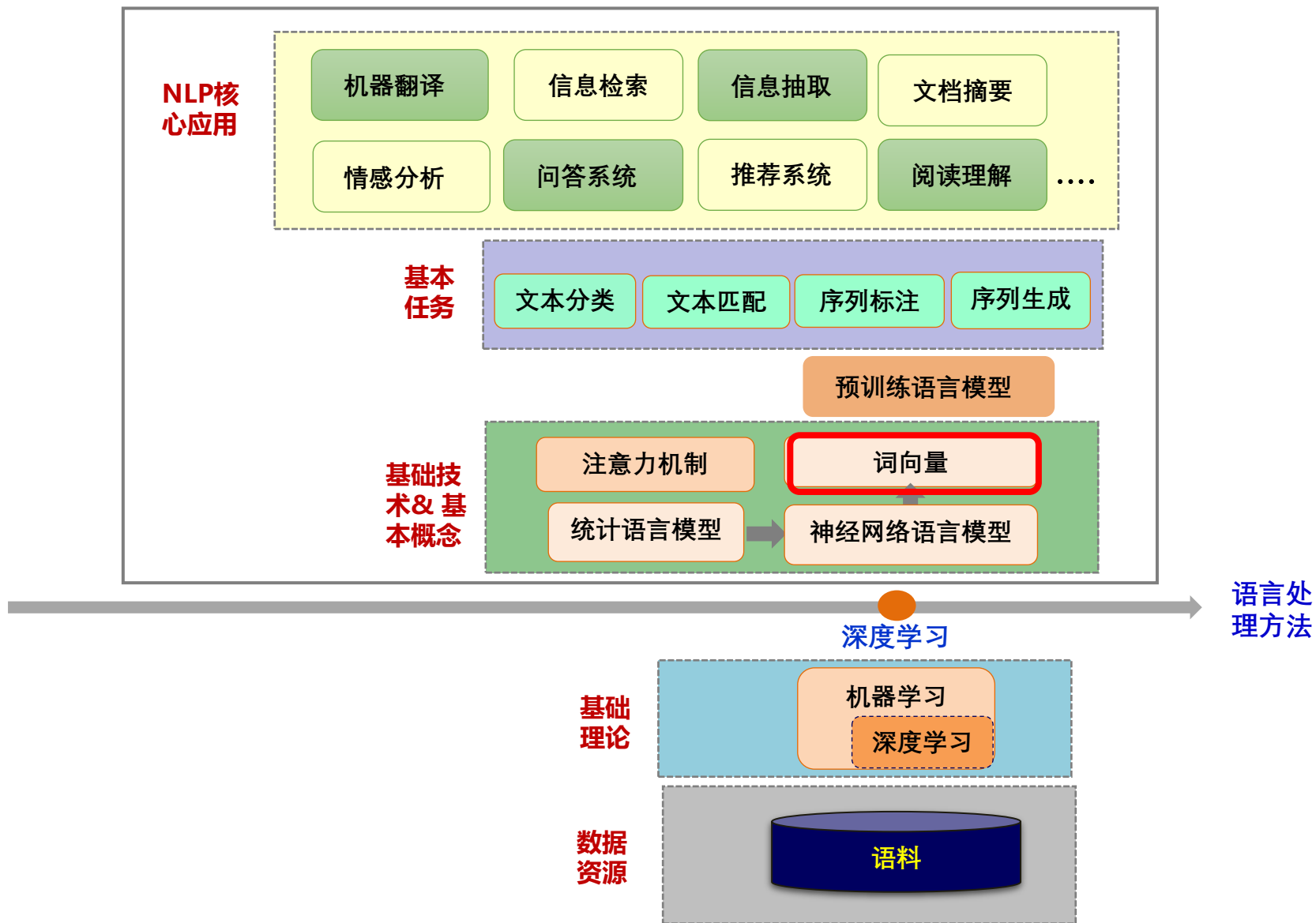
自然语言处理
Natural Language Processing

第 9 章 词向量
(浅层模型)

授课教师：胡玥

授课时间：2020.9

基于深度学习的自然语言处理课程内容



3.1 语言模型基本概念

重点：

1. 掌握以下4典型词向量的基本概念

- NNLM模型词向量
- RNNLM模型词向量
- CBOW 模型词向量
- Skip-gram模型词向量

2. 词向量特征？为什么有这种特征？

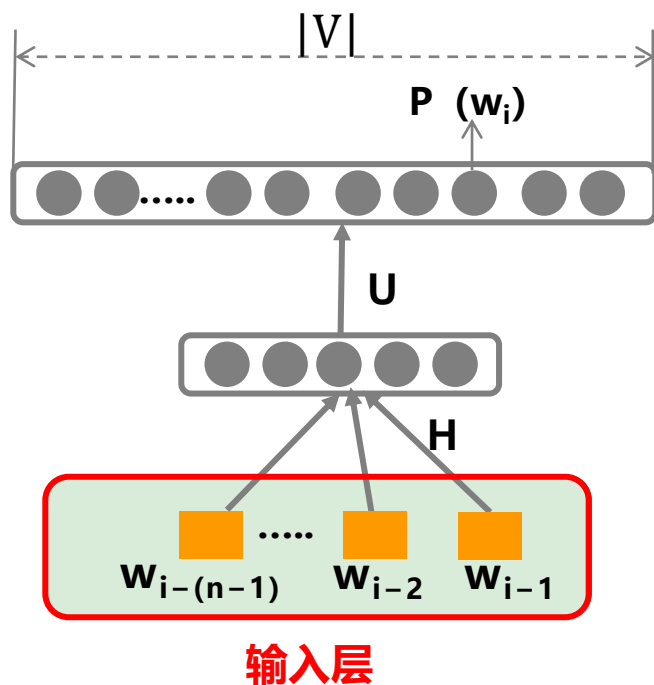
3. 为何神经网络“RNN 语言模型 + 词向量” 可以解决统计语言模型存在的问题

9.2.1 经典词(向量)表示模型

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)

NNLM模型-输入表示



词的 one-hot 表示

张: 0000100.....00

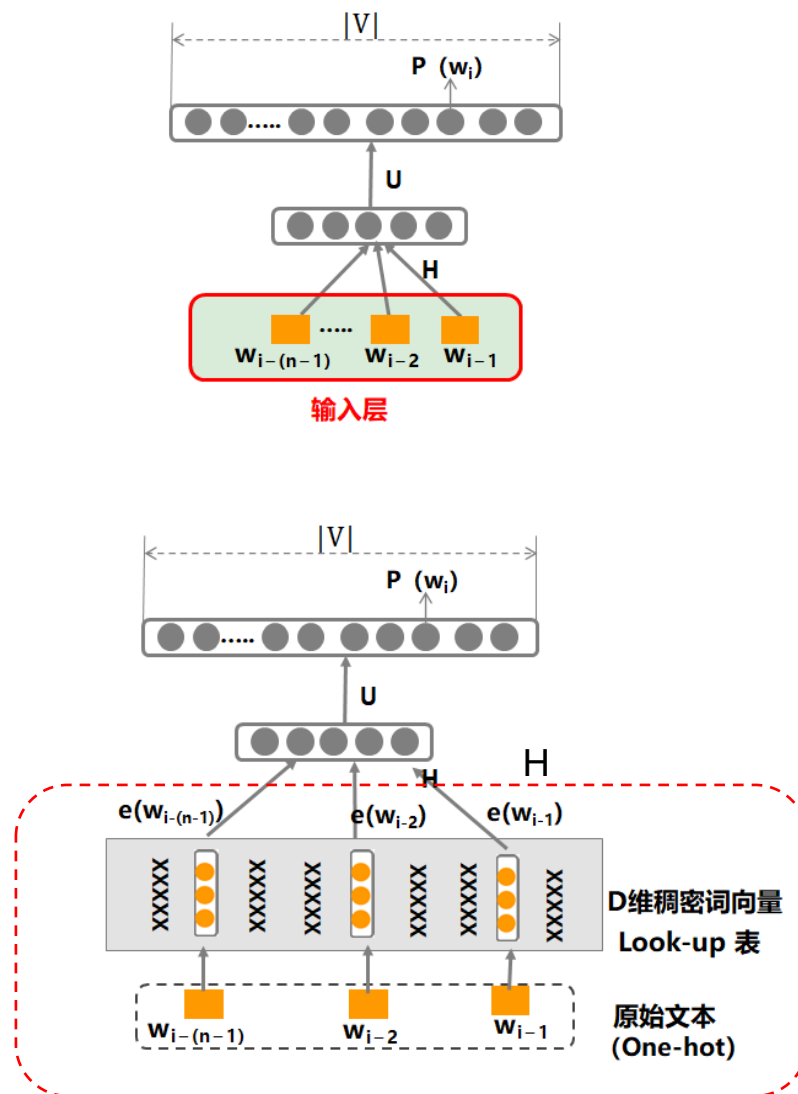
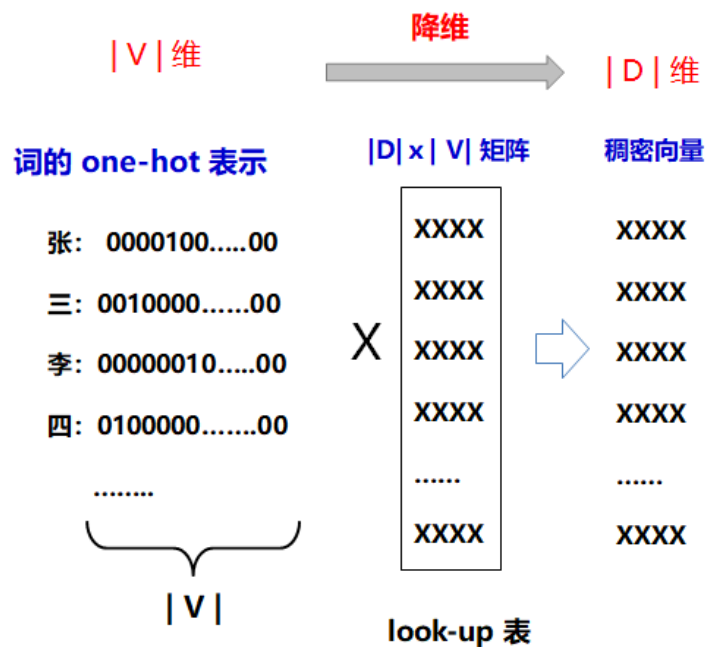
三: 0010000.....00

李: 00000010.....00

四: 0100000.....00

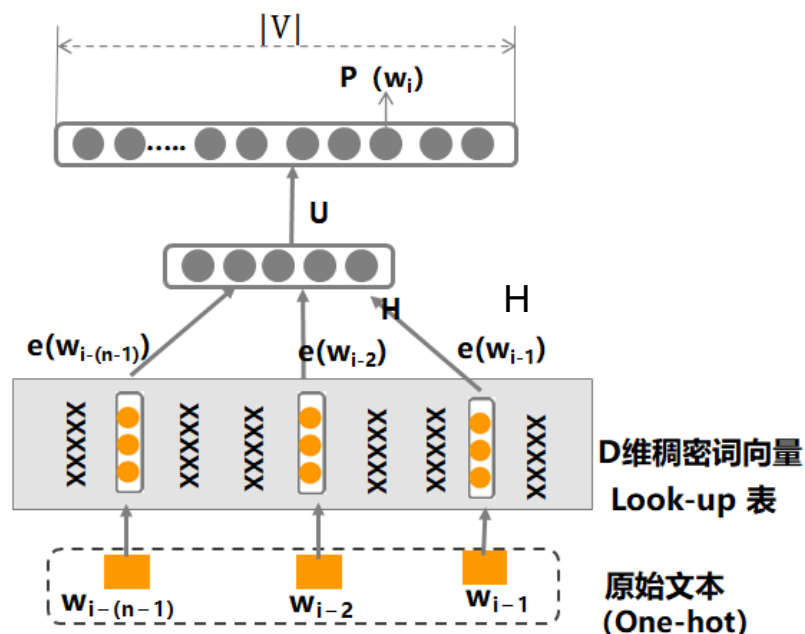


1. NNLM模型(词向量)



1. NNLM模型(词向量)

(1) NNLM模型结构(词向量)



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

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

训练结束 → 训练好的词向量

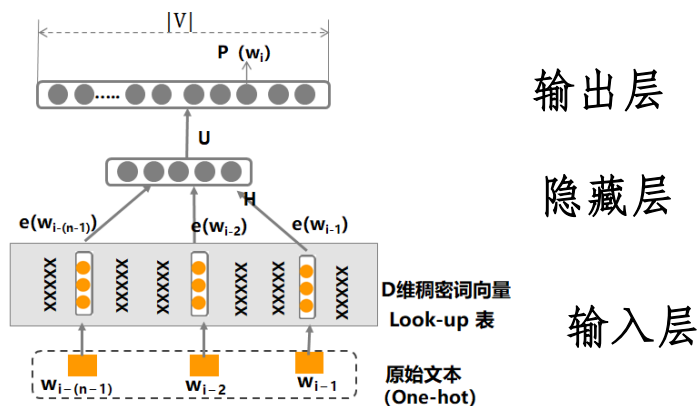
8.2 NNLM模型

(2) NNLM模型学习 (2-gram)

- 目标函数:

采用log损失函数

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



- 语料: (“无监督”)

文本: $S = w_1, w_2, \dots, w_n, \dots$

实例: $X: w_1, w_2, \dots, w_{i-1}$

$\hat{Y}: w_i$

- 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 $w_{i-(n-1)}, \dots, w_i$ 作为训练样本进行一次梯度迭代

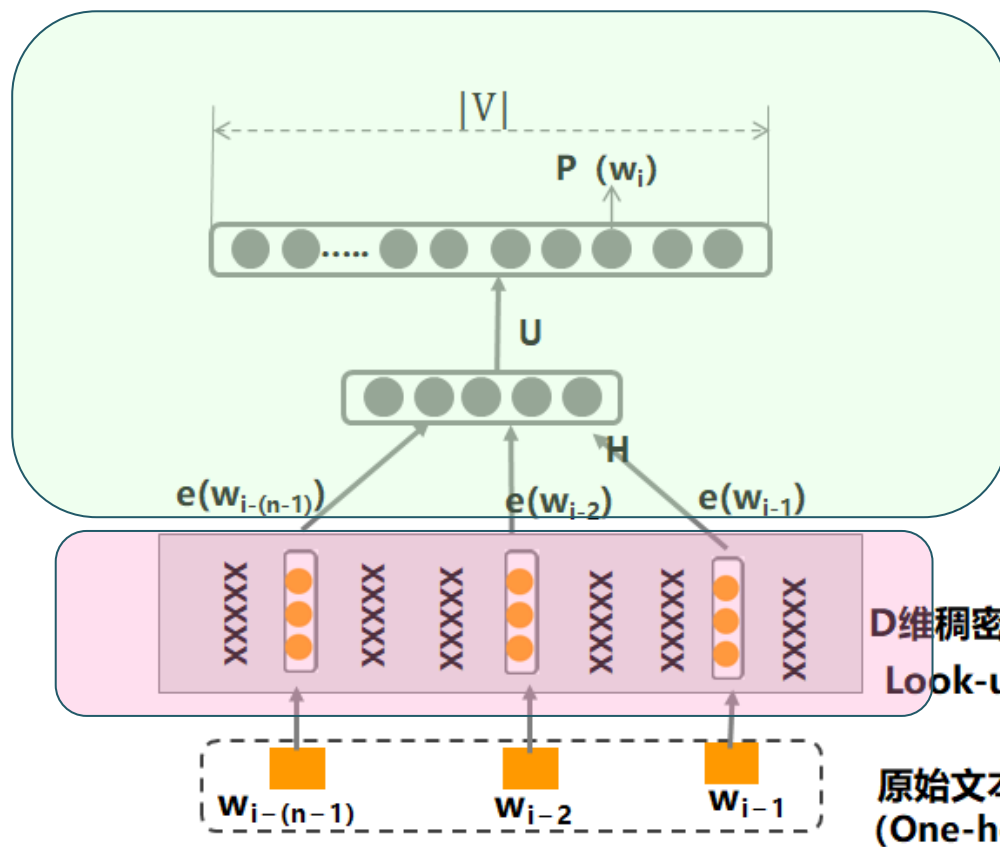
$$\theta \leftarrow \theta + \alpha \frac{\partial \log P(w_i | w_{i-(n-1)}, \dots, w_{i-1})}{\partial \theta}$$

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

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

1. NNLM模型(词向量)

(3) NNLM模型作用



● 语言模型

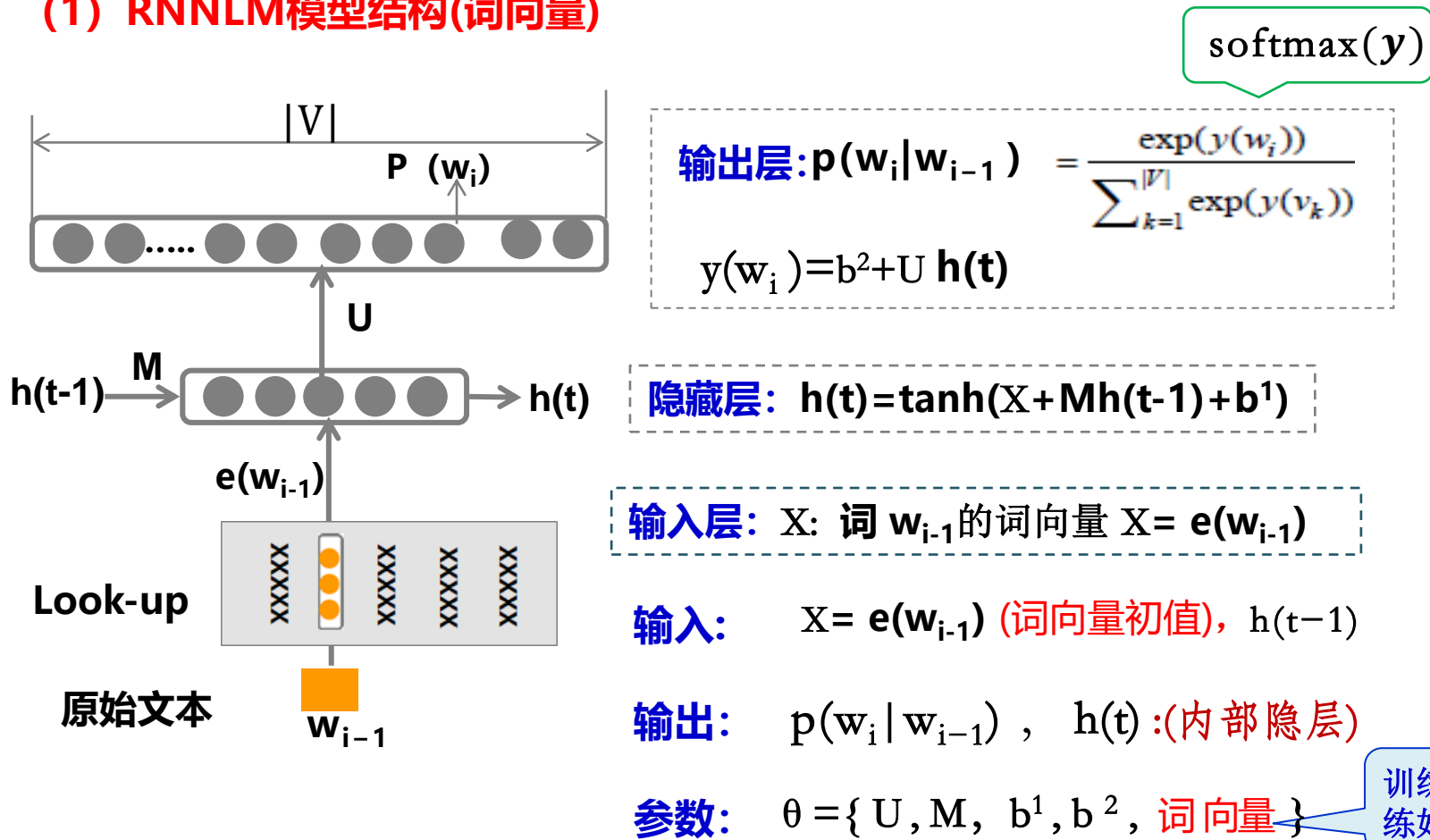
$$p(w_i | w_{i-(n-1)} \dots w_{i-1})$$

● 词向量

$w_{i-(n-1)} \dots w_{i-1}$ 的词向量

2. RNNLM模型(词向量)

(1) RNNLM模型结构(词向量)

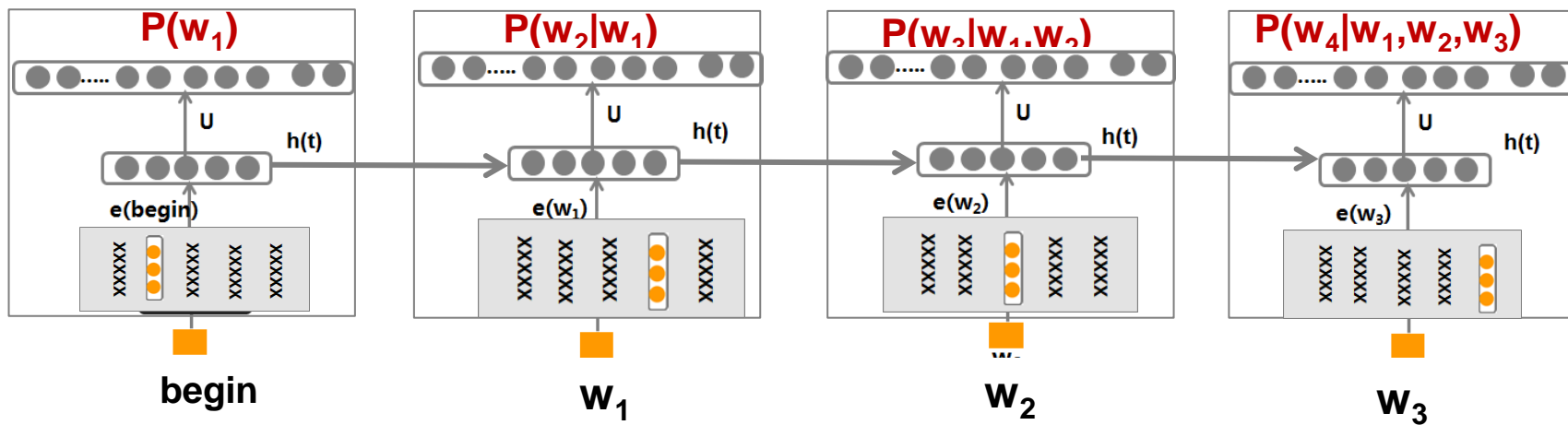
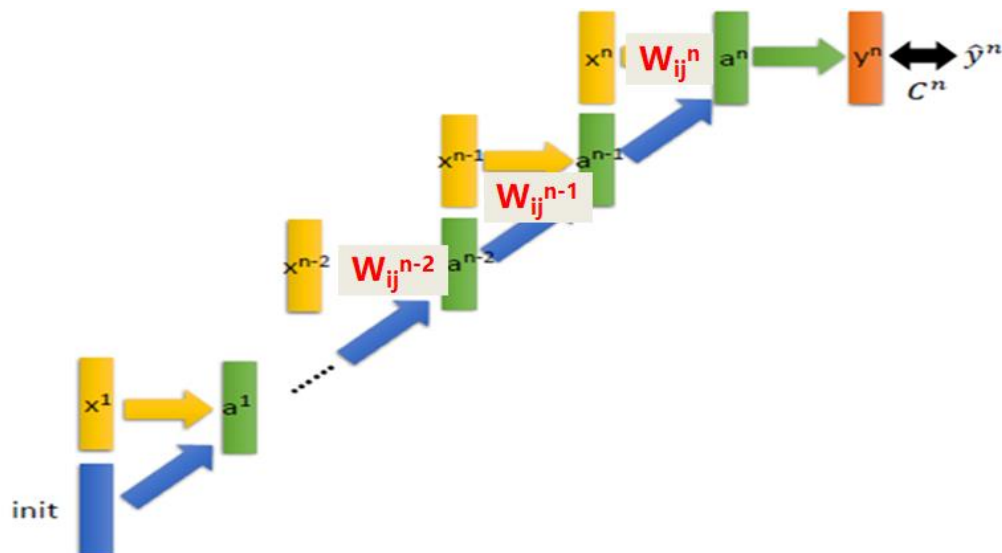


- Tomas Mikolov, et.al. Statistical language models based on neural networks. 2012
- Tomas Mikolov, et.al. Recurrent neural network based language model. 2010

2. RNNLM模型(词向量)

(2) RNNLM模型训练

- 语言模型

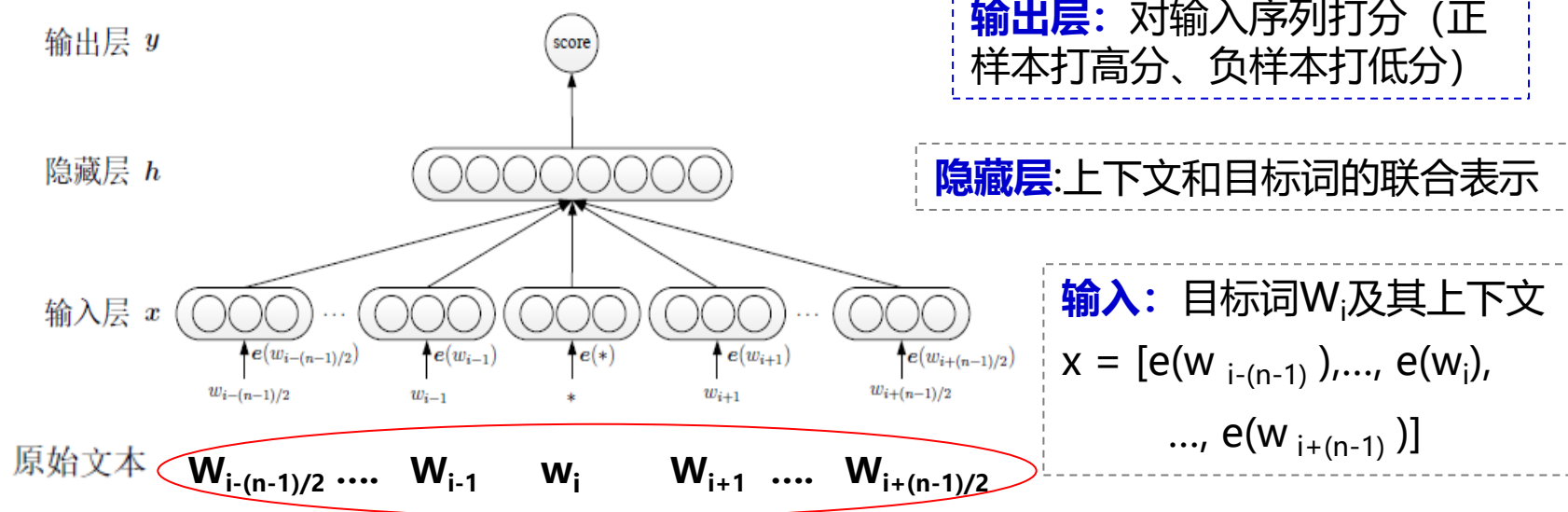


- 词向量

(BPTT) 随机梯度下降法优化训练目标:

3. C&W 模型

(1) C&W模型结构

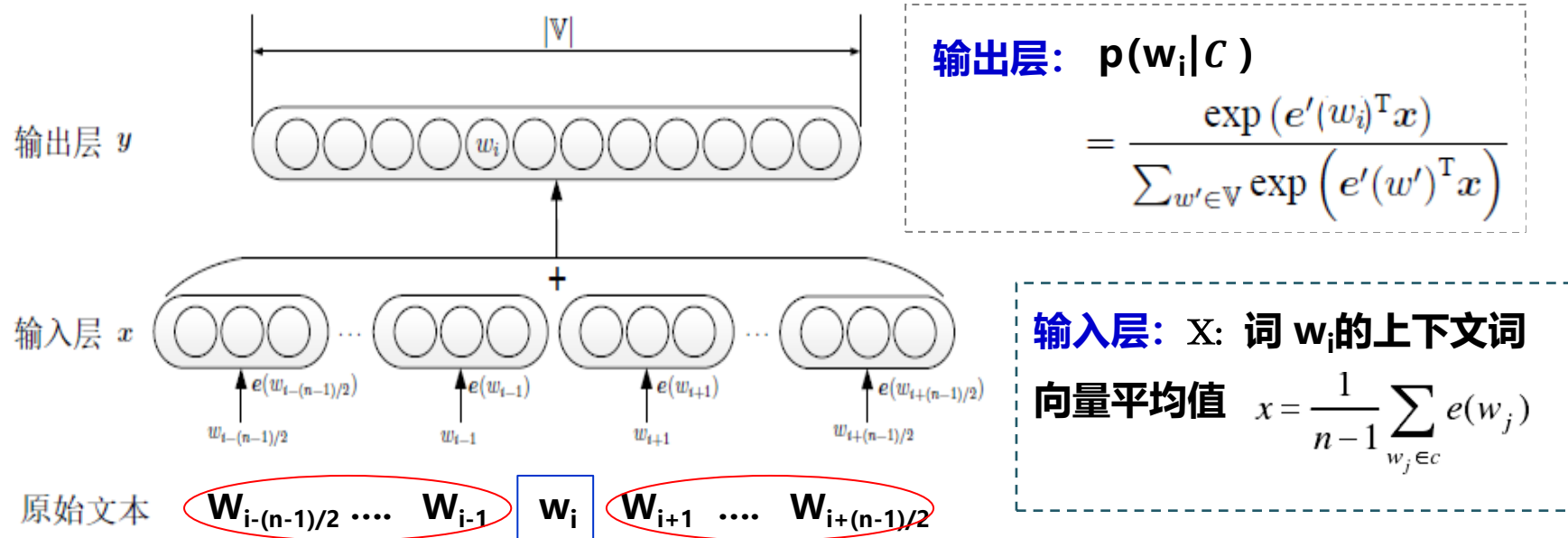


为从语料中选出的一个 n 元短语 $w_{i-(n-1)/2}, \dots, w_i, \dots, w_{i+(n-1)/2}$ 一般 n 为奇数, 以保证上文和下文的词数一致; w_i 为目标词(序列中间的词) $x = [e(w_{i-(n-1)/2}), \dots, e(w_i), \dots, e(w_{i+(n-1)/2})]$

4. CBOW 模型

■ CBOW 模型

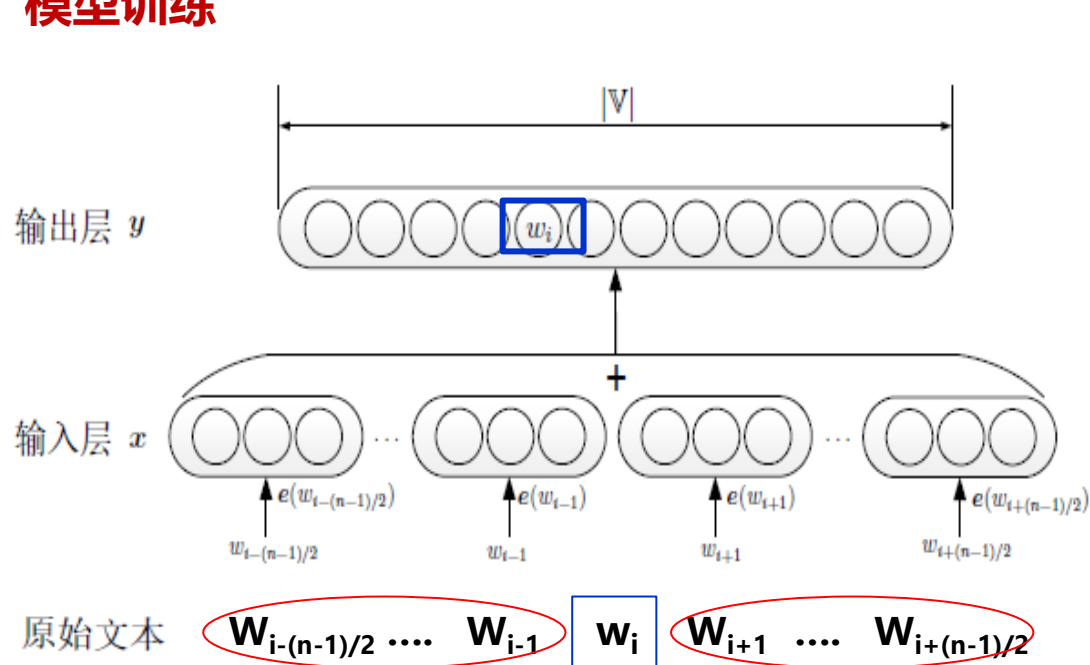
模型结构



为从语料中选出的一个 n 元短语 $w_{i-(n-1)/2}, \dots, w_i, \dots, w_{i+(n-1)/2}$ 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

4. CBOW 模型

模型训练



w_i

$\overline{Y}: 0 \ 0 \ \dots \ 1 \ \dots 0 \ 0 \ 0$

Y : 所有词的概率

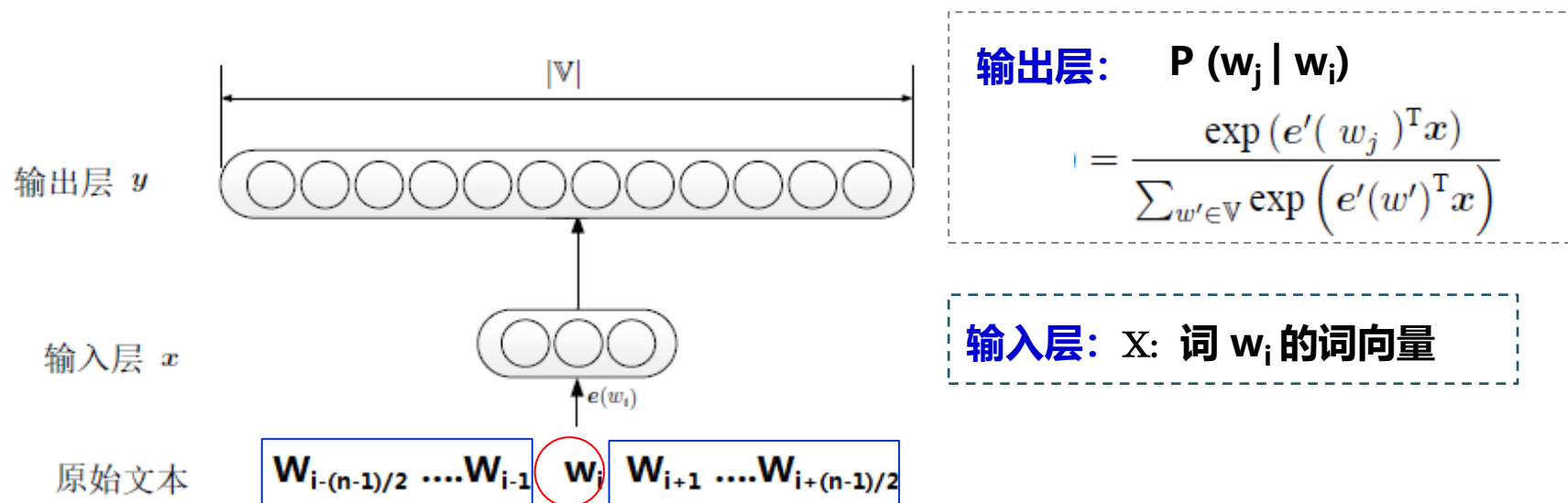
X : w_i 上下文

- 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

5. Skip-gram模型

■ Skip-gram模型

模型结构

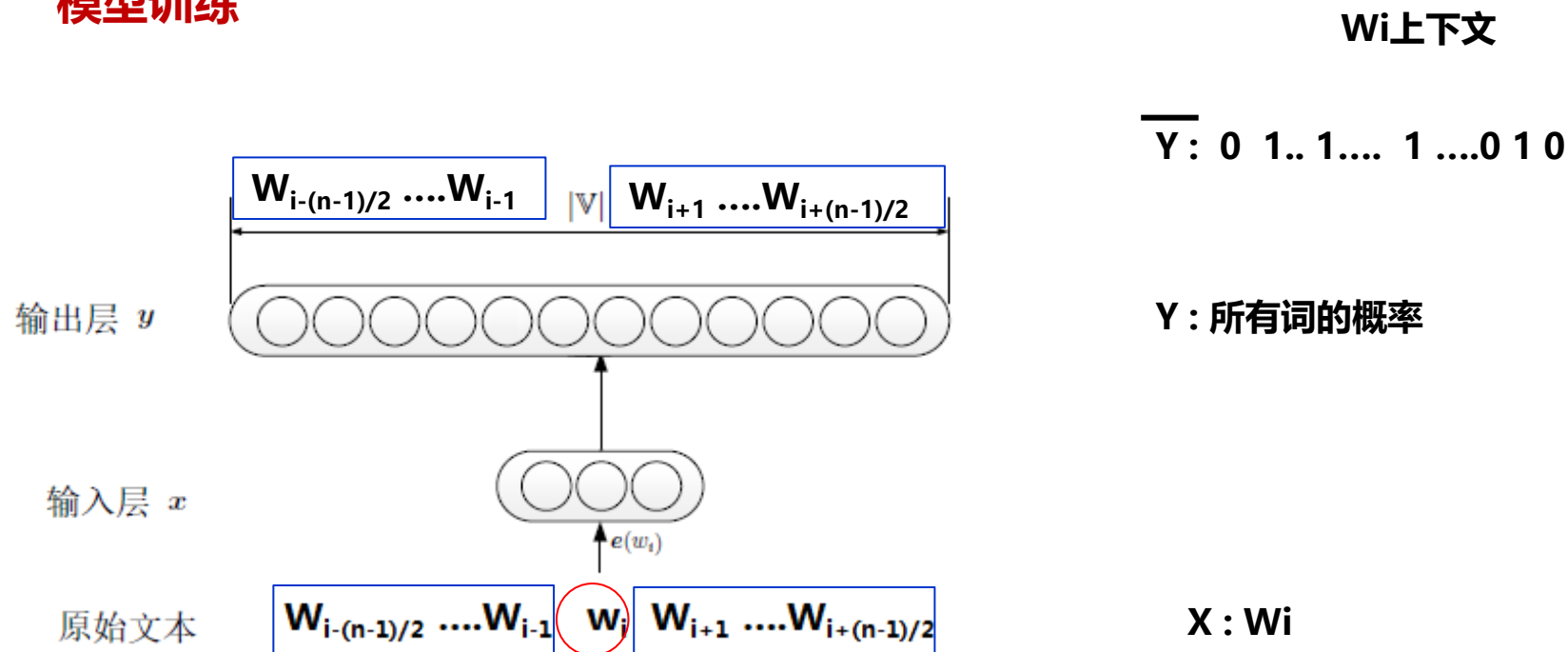


将目标词 w_i 的词向量作为输入，每次从 w_i 的上下文 C 中选一个词作为预测词进行预测。目标词 w_i 及上下文 C 定义同CBOW模型

5. Skip-gram模型

■ Skip-gram模型

模型训练



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

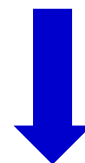
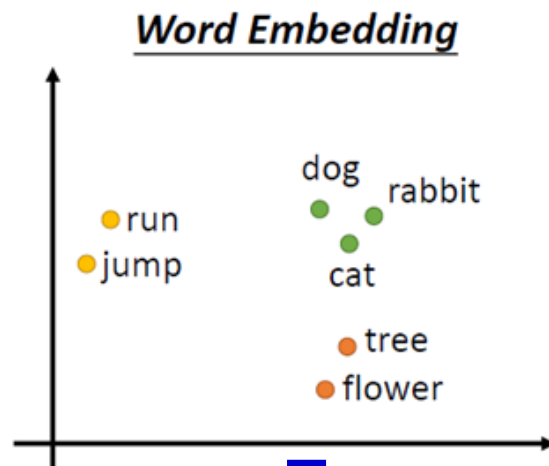
9.2.2 词向量特性及应用

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 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

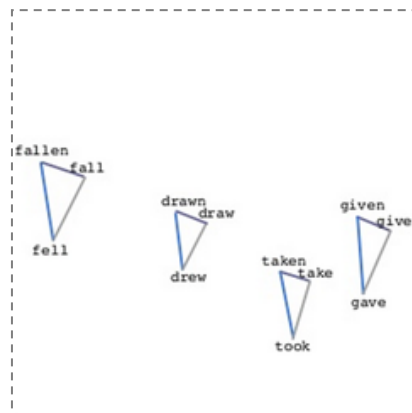
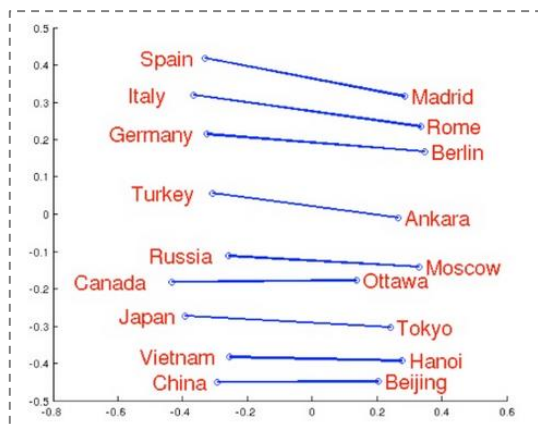
9.2.2 词向量特性及应用

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

$$V(\text{king}) - V(\text{queen}) \approx V(\text{uncle}) - V(\text{aunt})$$

$$V(\text{hotter}) - V(\text{hot}) \approx V(\text{bigger}) - V(\text{big})$$



9.2.2 词向量特性及应用

第3章： 概率语言模型存在问题

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

神经网络 “RNN 语言模型 + 词向量” 可以解决以上问题

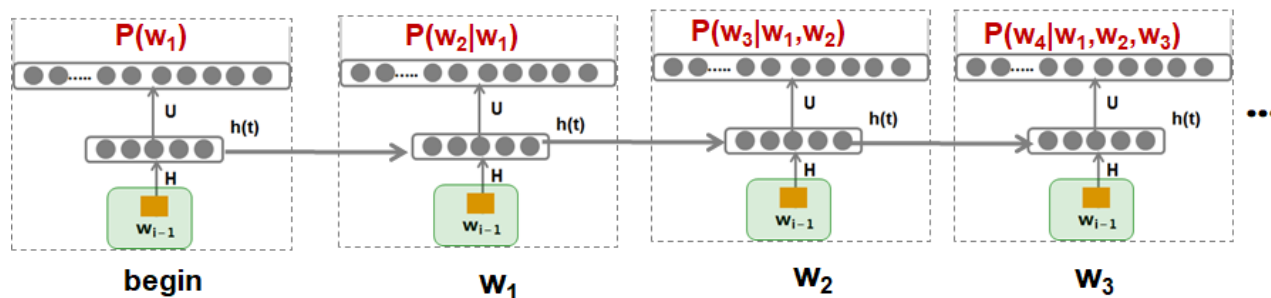
9.2.2 词向量特性及应用

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**

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

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



RNNLM模型可以保留每个词的所有历史信息

9.2.2 词向量特性及应用

问题2：需要数据平滑

平滑问题： $p(\textit{Cher read a book}) = ?$

$$= p(\textit{Cher} | \langle \textit{BOS} \rangle) \times p(\textit{read} | \textit{Cher}) \times p(\textit{a} | \textit{read}) \times p(\textit{book} | \textit{a}) \times p(\langle \textit{EOS} \rangle | \textit{book})$$

统计语言模型：

$$p(\textit{Cher} | \langle \textit{BOS} \rangle) = \frac{c(\langle \textit{BOS} \rangle \textit{Cher})}{\sum_w c(\langle \textit{BOS} \rangle w)} = \frac{0}{3}$$

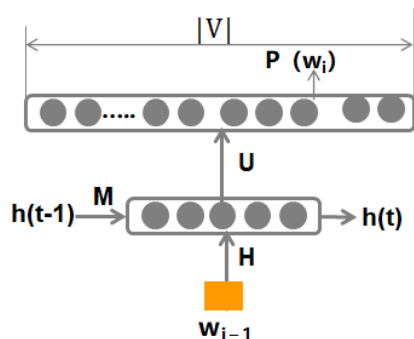
$$p(\textit{read} | \textit{Cher}) = \frac{c(\textit{Cher} \textit{read})}{\sum_w c(\textit{Cher} w)} = \frac{0}{1}$$

$\langle \textit{BOS} \rangle \textit{John read Moby Dick} \langle \textit{EOS} \rangle$
 $\langle \textit{BOS} \rangle \textit{Mary read a different book} \langle \textit{EOS} \rangle$
 $\langle \textit{BOS} \rangle \textit{She read a book by Cher} \langle \textit{EOS} \rangle$

$p(\textit{Cher read a book}) = 0 \rightarrow$ 需要数据平滑

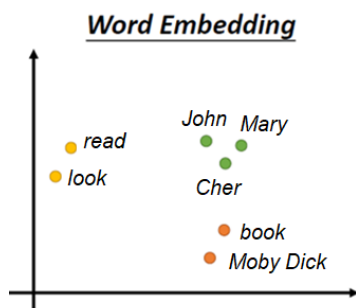
9.2.2 词向量特性及应用

神经网络语言模型：

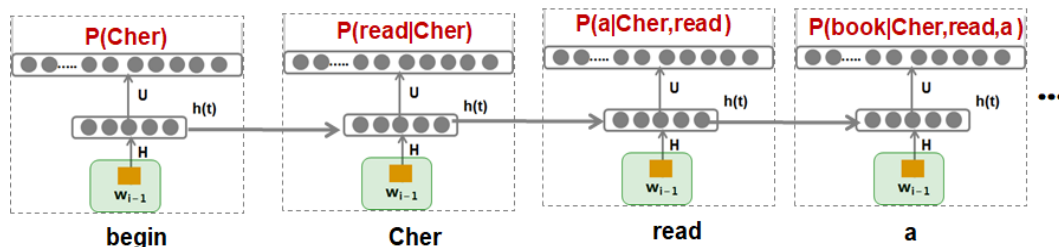


当语言模型训练好后，模型网络参数固定，
这时给任意的 w_{i-1} $P(w_i)$ 不会为 0

词向量特征：



Cher 和 John 的
词向量比较接近



所以，采用预训练的词向量做输入，不需要数据平滑且效果好

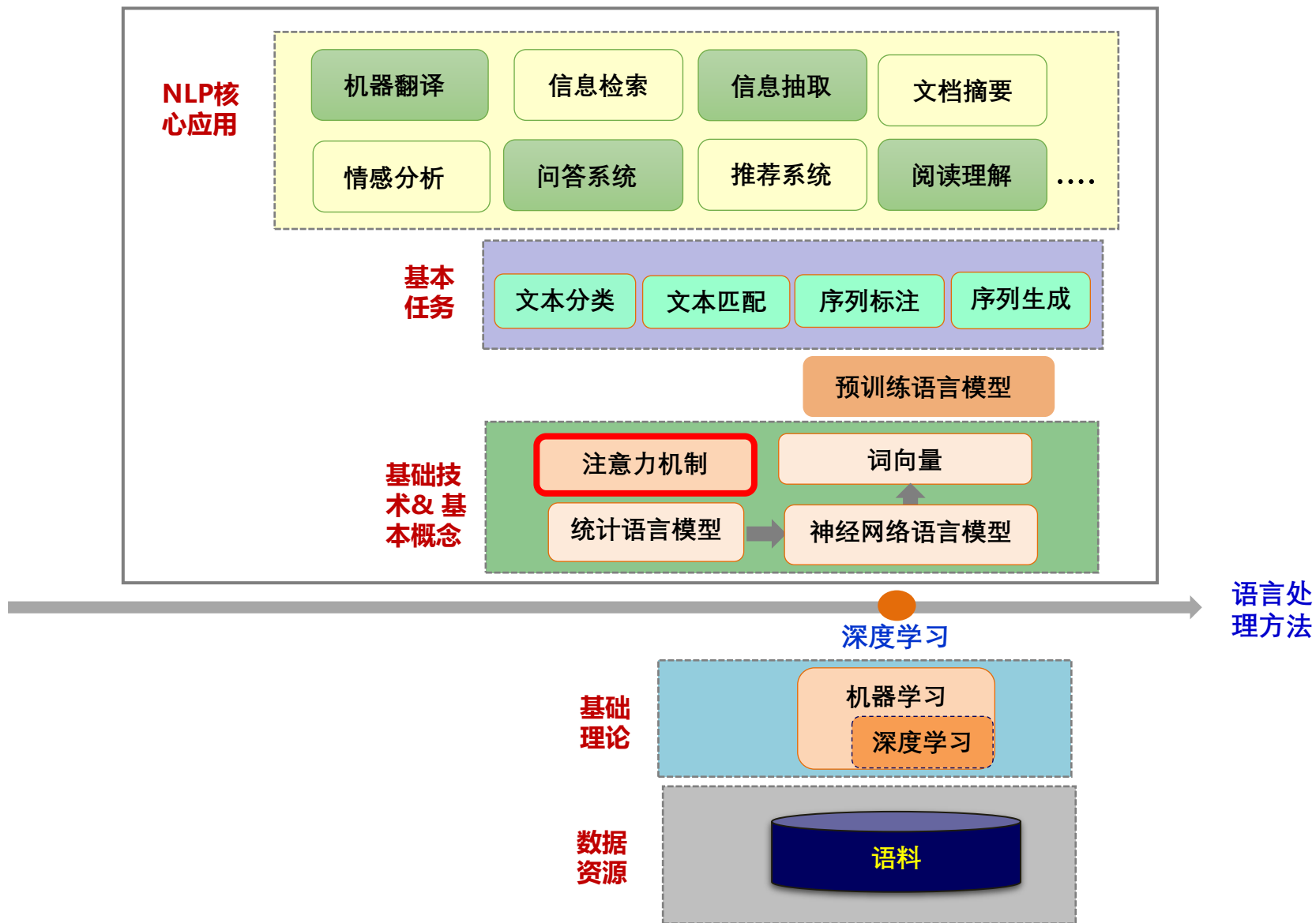
自然语言处理
Natural Language Processing

第 10 章 NLP中的注意力机制

授课教师：胡玥

授课时间：2020.9

基于深度学习的自然语言处理课程内容



内 容 提 要

10.1 传统注意力机制

10.2 注意力编码机制

10. 注意力机制

重点：

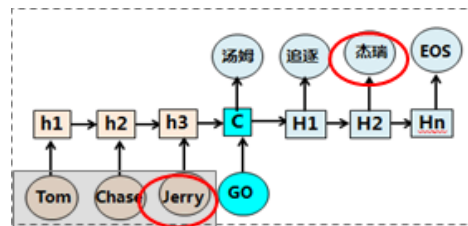
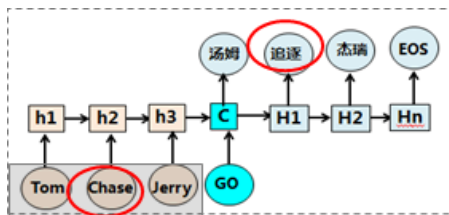
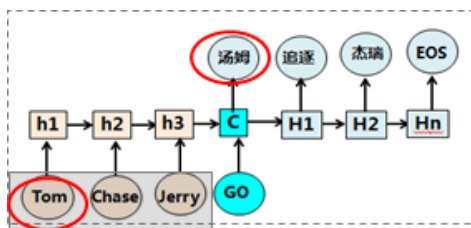
1. 注意力机制的基本概念
2. 注意力机制的模块定义
3. 何为软/硬/局部/全局注意力
4. 运用注意力机制的好处有哪些？

10.1 传统注意力机制

注意力机制概念

加权求和模块

结果 $Att-V$



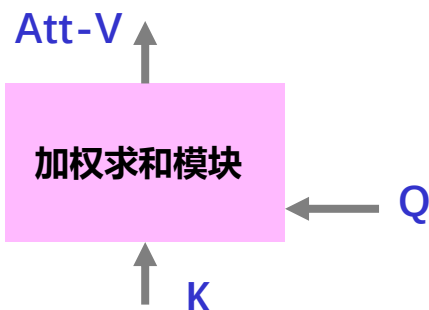
Q

K

各个数的“权重”是针对某个元素而言的。我们将这个元素定义为 Q (Q 也可以是参数)，将需要计算权重的各元素集合定义为 K ，将最后的加权求和结果定义为 $Att-V$

输入: Q, K

输出: $Att-V$



注: Q 可以是变量

10.1 传统注意力机制

(1) 模块结构：运算关系

输入 → 输出 函数关系：

步骤1：计算对于Q 各个 K_i 的权重

步骤2：计算输出 Att-V值（各 K_i 乘以自己的权重，然后求和）

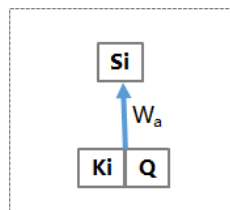
10.1 传统注意力机制

步骤1：计算对于Q 各个 K_i 的权重

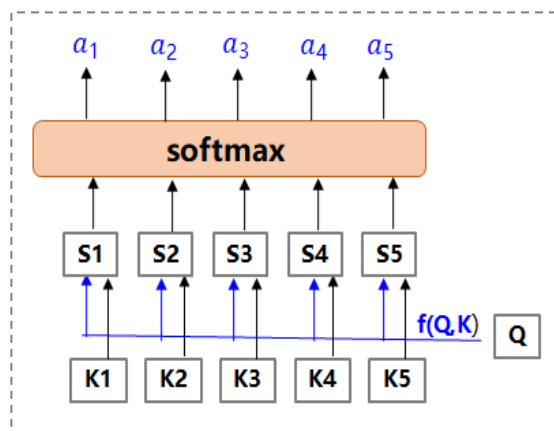
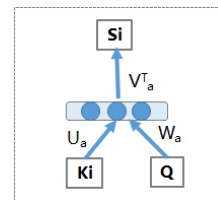
设注意力打分函数 $S = f(Q, K)$

$$S = f(Q, K) = \begin{cases} Q^T K_i & \text{点积模型} \\ \frac{Q^T K_i}{\sqrt{d}} & \text{缩放点积模型} \\ W_a[Q, K_i] & \text{连接模型} \\ Q^T W_a K_i & \text{双线性模型} \\ V_a^T \tanh(W_a Q + U_a K_i) & \text{加性模型} \end{cases}$$

$W_a[Q, K_i]$



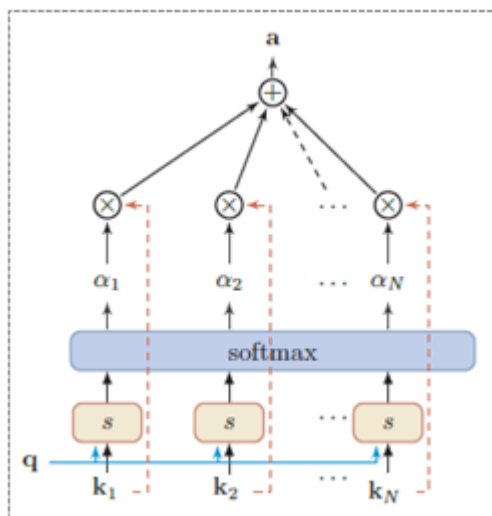
$V_a^T \tanh(W_a Q + U_a K_i)$



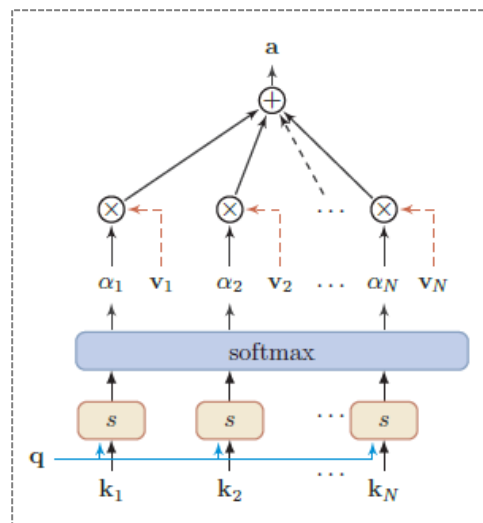
$$a_i = \text{softmax}(f(Q, K_i)) = \frac{\exp(f(Q, K_i))}{\sum_j \exp(f(Q, K_j))}$$

10.1 传统注意力机制

步骤2: 计算输出 Att-V值 (各 K_i 乘以自己的权重, 然后求和)



普通模式



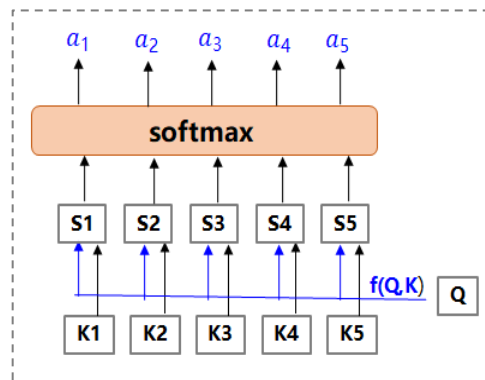
键值对模式

Att-V值

K1	K2	K3	K4	K5
V1	V2	V3	V4	V5

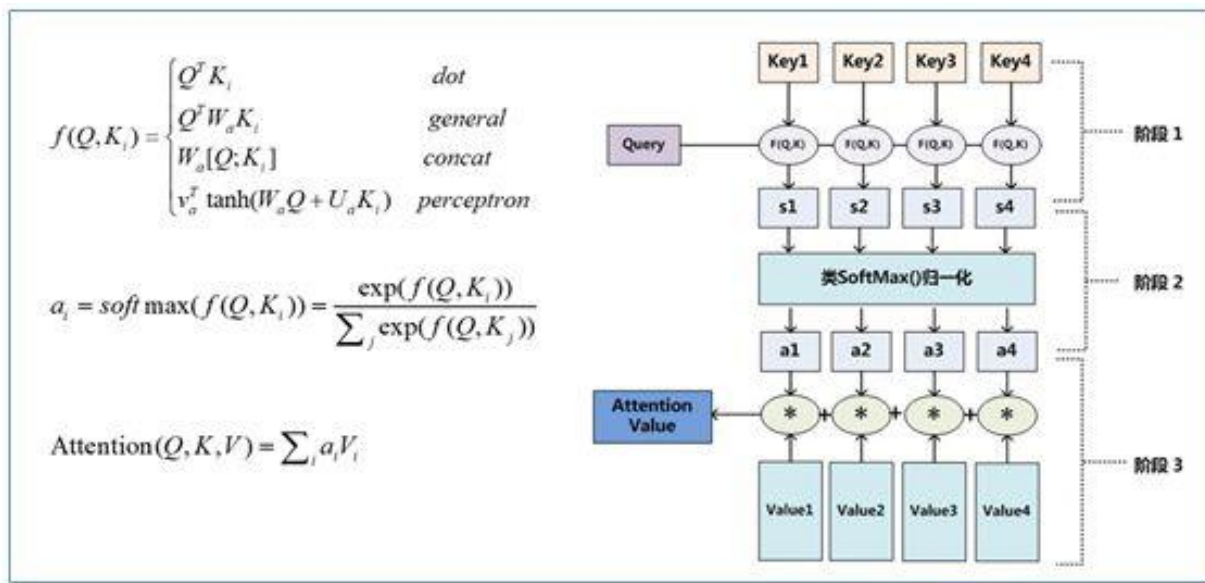
$$\text{Att-V} = a_1 \times K1 + a_2 \times K2 + a_3 \times K3 + a_4 \times K4 + a_5 \times K5$$

$$\text{Att-V} = a_1 \times V1 + a_2 \times V2 + a_3 \times V3 + a_4 \times V4 + a_5 \times V5$$



10.1 传统注意力机制

Attention模型表示:

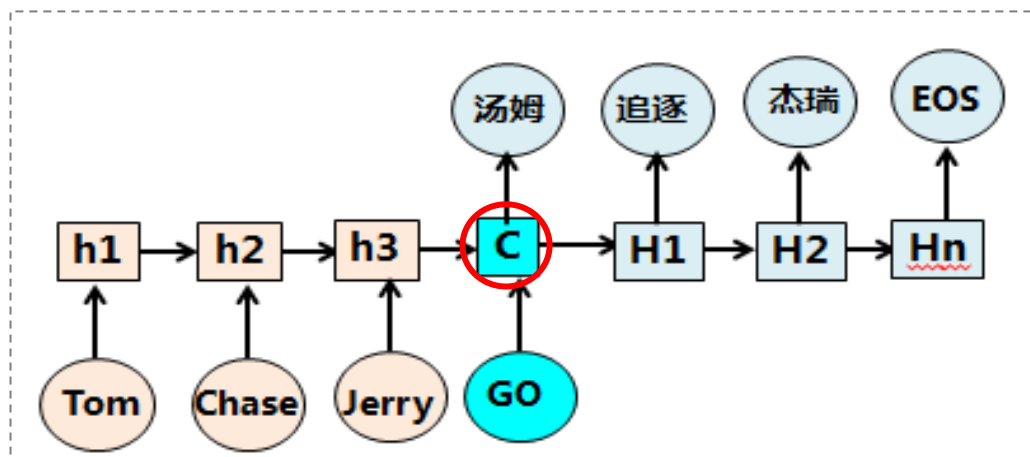


功能: 对于集合 K，求相对 Q 各个元素的权重，然后按权重相加形成 Q 要的结果

10.1 传统注意力机制

问题引入： 机器翻译例

Encoder-Decoder RNN



问题： 对不同的输出 Y_i 中间语义表示 C 相同

实际应该：在翻译“杰瑞”的时候，体现出英文单词对于翻译当前中文单词不同的影响程度，比如 (Tom,0.3) (Chase,0.2) (Jerry,0.5)

问题： 对每个输出的词，如何生成针对它的更准确的中间语义单元？

$$X = \langle x_1, x_2 \dots x_m \rangle$$

$$Y = \langle y_1, y_2 \dots y_n \rangle$$

$$C = \mathcal{F}(x_1, x_2 \dots x_m)$$

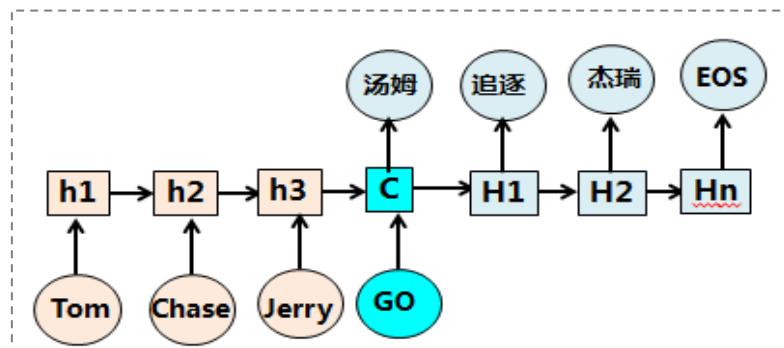
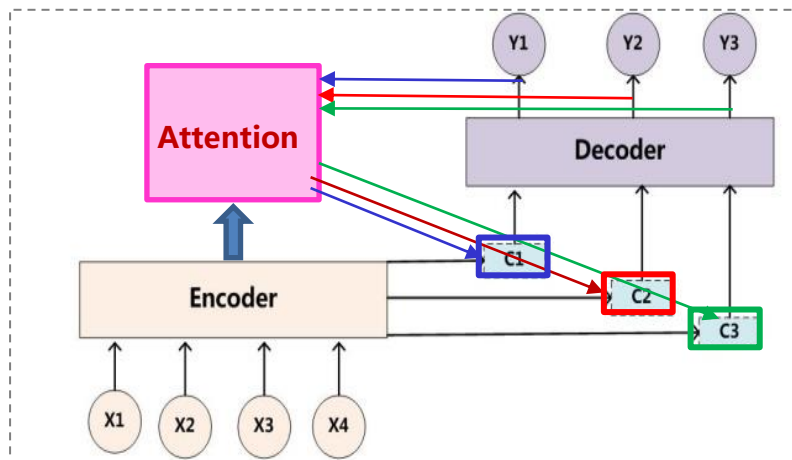
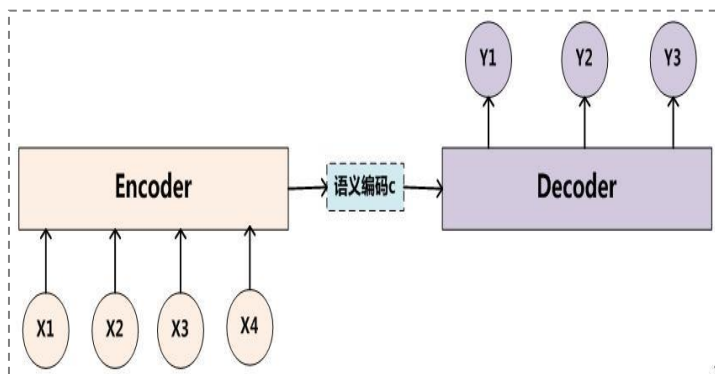
$$y_1 = f(C)$$

$$y_2 = f(C, y_1)$$

$$y_3 = f(C, y_1, y_2)$$

$$y_i = g(C, y_1, y_2 \dots y_{i-1})$$

10.1 传统注意力机制



注意力机制：神经网络中的一个组件，可以单独使用，但更多地用作网络中的一部分。

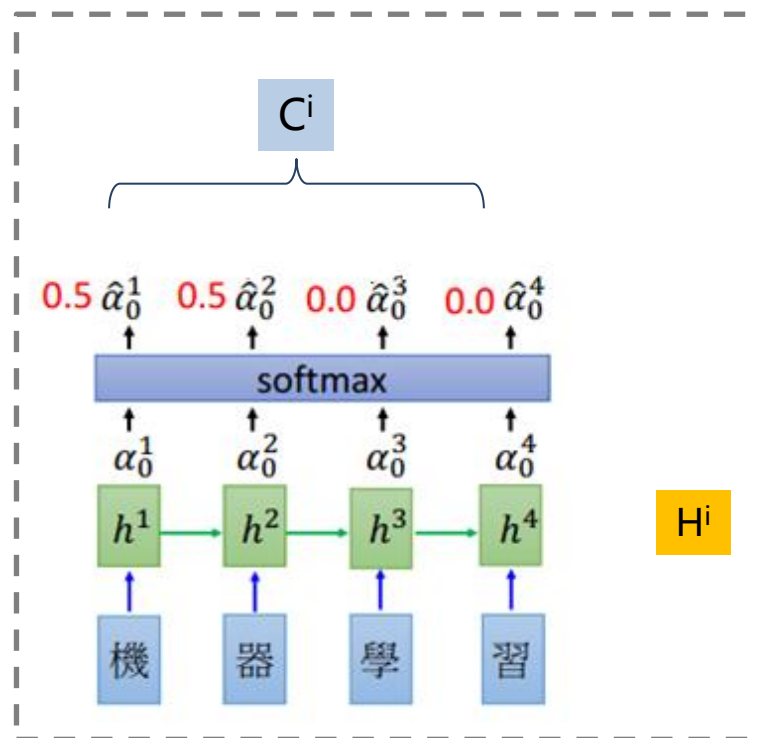
作用：让任务处理系统找到与当前任务相关显著的输入信息，并按重要性进行处理，从而提高输出的质量。

优势：不需要监督信号，可推理多种不同模态数据之间的难以解释、隐蔽性强、复杂映射关系，对于先验认知少的问题，极为有效。

10.1 传统注意力机制

□ 软注意力 Hard Attention

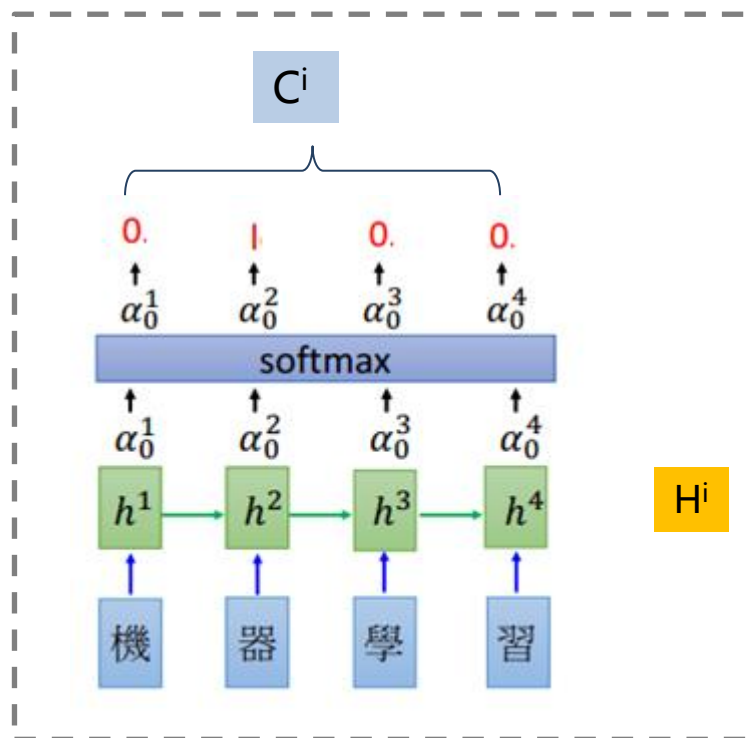
Soft AM: 在求注意力分配概率分布的时候，对于输入句子X中任意一个单词都给出个概率，是个概率分布。



10.1 传统注意力机制

□ 硬注意力 Hard Attention

Hard AM: 直接从输入句子里面找到某个特定的单词，然后把目标句子单词和这个单词对齐，而其它输入句子中的单词硬性地认为对齐概率为0

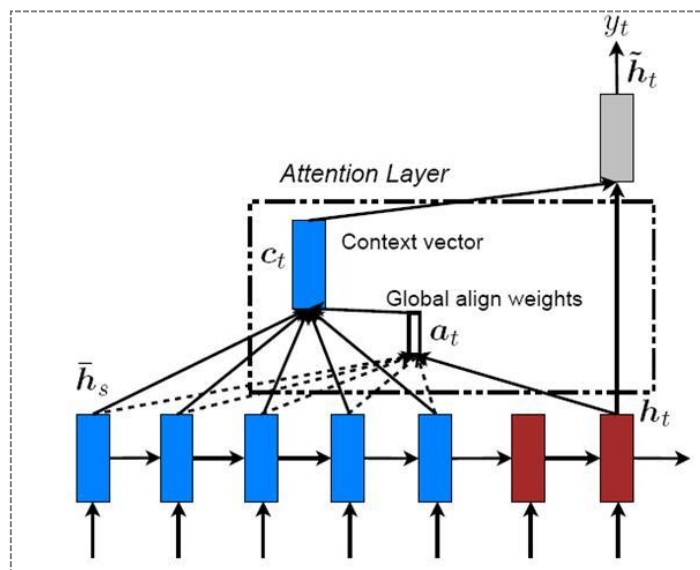


10.1 传统注意力机制

□ 全局注意力 Global Attention

Decode端Attention计算时要考虑Encoder端序列中所有的词

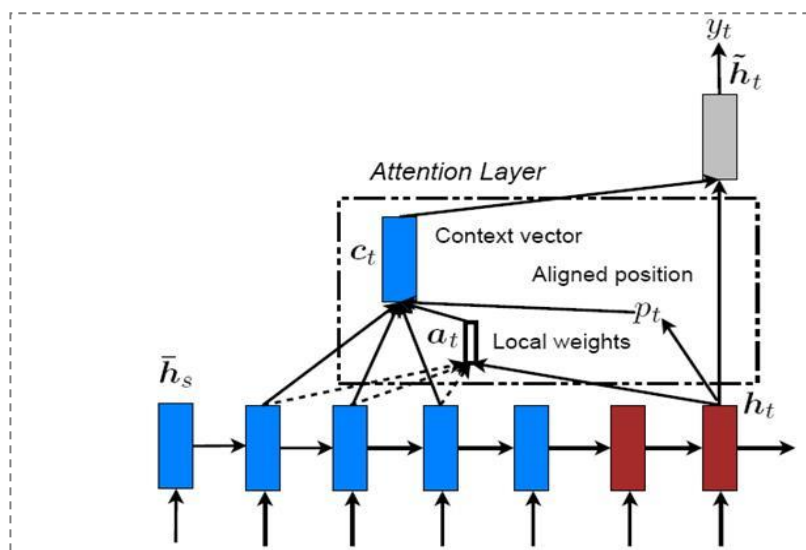
Global Attention Model 是Soft Attention Model



10.1 传统注意力机制

□ 局部注意力 Local Attention

Local Attention Model本质上是Soft AM和 Hard AM的一个混合或折衷。一般首先预估一个对齐位置 p_t ，然后在 p_t 左右大小为 D 的窗口范围来取类似于Soft AM的概率分布。



内 容 提 要

10.1 传统注意力机制

10.2 注意力编码机制

10. 注意力机制

重点：

1. 注意力机制编码的好处有哪些？

10.2 注意力编码机制

Attention用作编码机制

- ◆ **不同序列间编码：**可以将2个序列编码成二者的融合表示，
匹配任务，阅读理解任务常用
- ◆ **同一序列自编码：**利用多头Self-Attention编码对一个句子编码可以起到句法分析器的作用

10.2 注意力编码机制

K 与 Q 序列的融合序列

A-V_i 序列的含义?

Attention层

A-V1

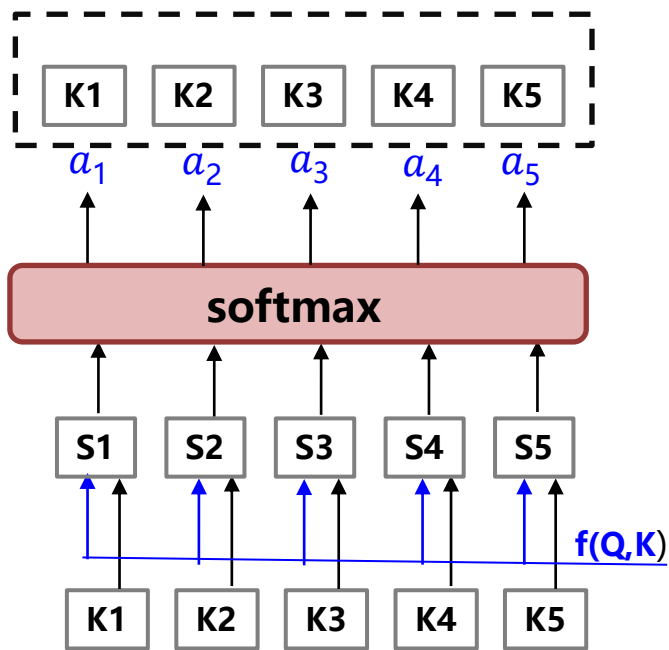
A-V2

A-V3

问题: Attention层是词袋模型

$$\text{Att-V} = a_1 \times K_1 + a_2 \times K_2 + a_3 \times K_3 + a_4 \times K_4 + a_5 \times K_5$$

权重:



Q1 Q2 Q3

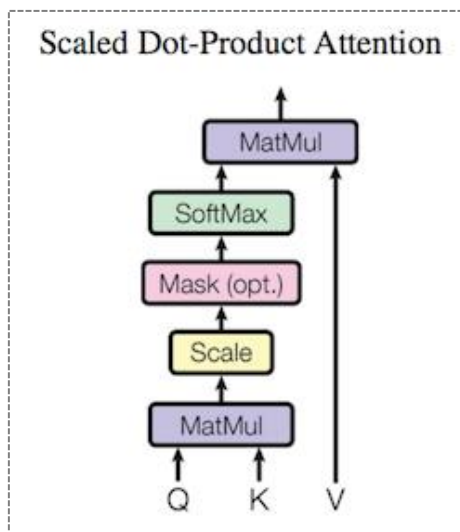
A-V_i 序列元素个数等于 Q 序列元素个数

10.2 注意力编码机制

◆ 同一序列自编码：

自注意力 Self-Attention

其实就是 $\text{Attention}(X, X, X)$, X 为输入序, 其含义为在序列内部做 Attention, 寻找序列内部的联系



$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

其中, $Q=K=V$

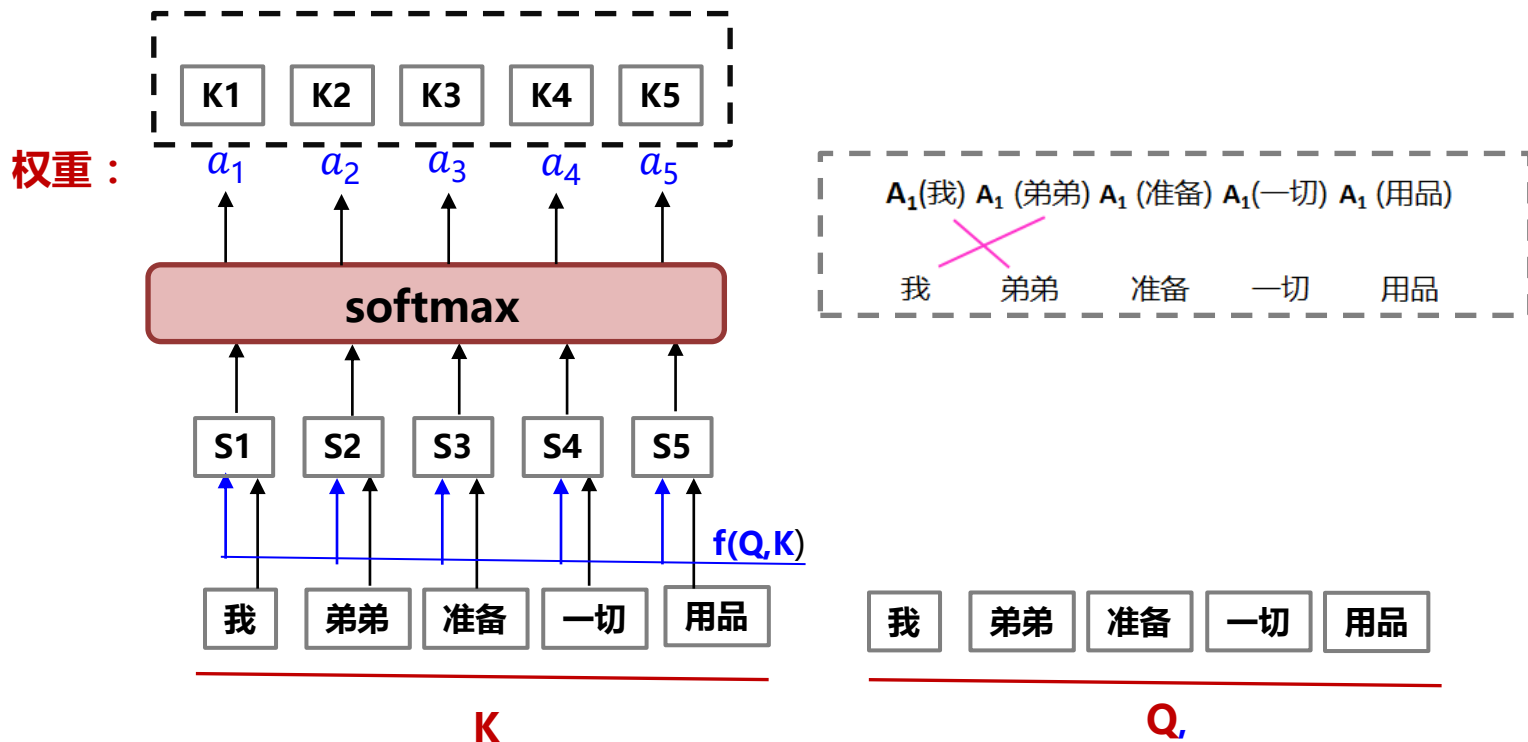
self-attention的特点在于无视词之间的距离直接计算依赖关系, 能够学习一个句子的内部结构, 实现也较为简单并行可以并行计算 (self-attention可以当成一个层和RNN, CNN, FNN等配合使用, 应用于其他NLP任务)

10.2 注意力编码机制

例：对同一序列自注意力编码

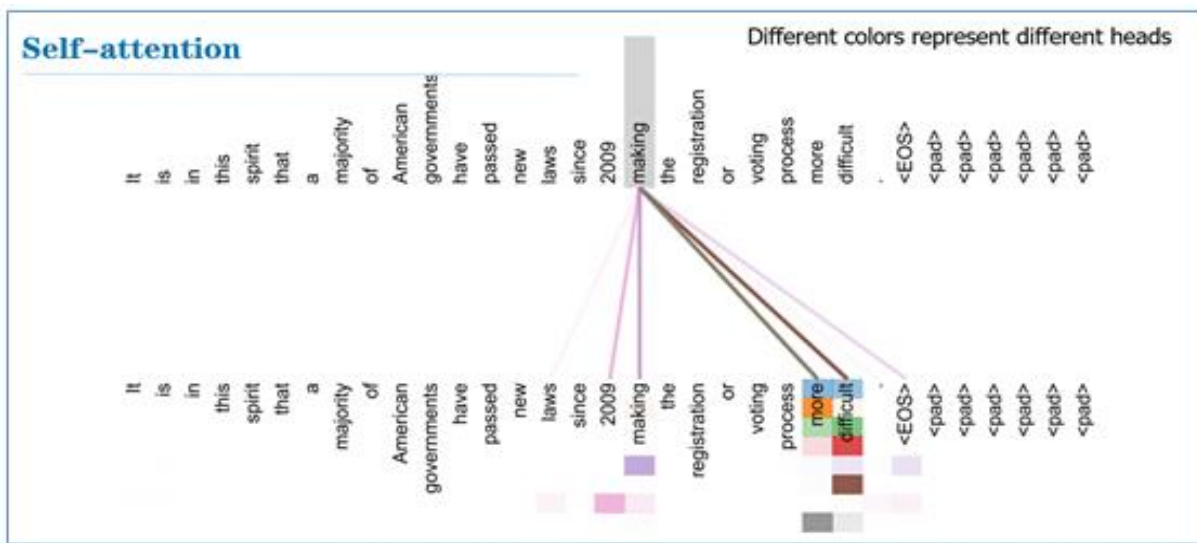
Self-Att 层 A-我 A-弟弟 A-准备 A-一切 A-用品

$$\text{Att-V} = a_1 \times K_1 + a_2 \times K_2 + a_3 \times K_3 + a_4 \times K_4 + a_5 \times K_5$$



10.2 注意力编码机制

Self-Attention可视化的效果

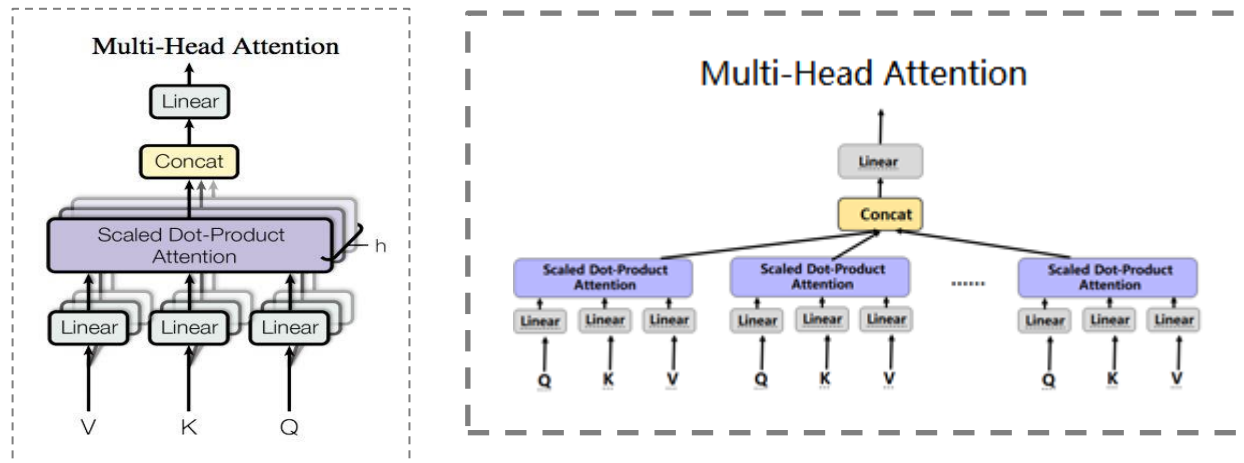


可以看到self-attention在这里可以学习到句子内部长距离依赖
"making.....more difficult"这个短语

10.2 注意力编码机制

多头注意力Multi-Head Attention

多头 (Multi-Head) 就是做多次同样的事情 (参数不共享), 然后把结果拼接
多头attention通过计算多次来捕获不同子空间上的相关信息。



$$\text{Head}_i = \text{Attention}(QW_i^{Q_i}, KW_i^{K_i}, VW_i^{V_i})$$

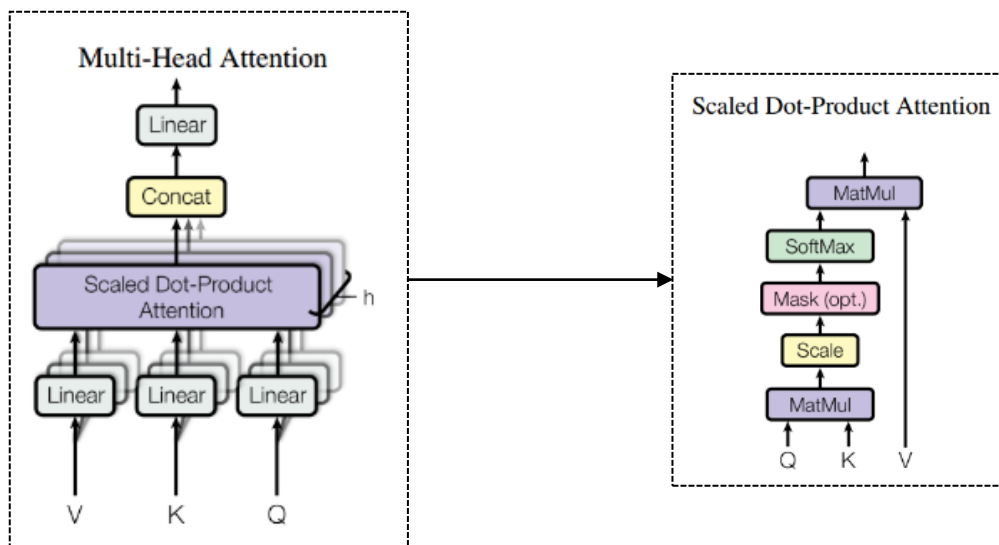
其中, $W_i^{Q_i} \in \mathbb{R}^{d_k \times \sim d_k}, W_i^{K_i} \in \mathbb{R}^{d_k \times \sim d_k}, W_i^{V_i} \in \mathbb{R}^{d_v \times \sim d_v}$

$\text{Multi-Head}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h)$ 最后得到一个 $n \times (h \sim d_v)$ 的序列

10.2 注意力编码机制

多头自注意力 Multi-Head Self Attention (Transformer)

多头自注意力 (Multi-Head Self Attention) 就是多头attention中, 每头均为自注意力 Attention(X, X, X)



$$\text{MultiHead}(H) = W_O[\text{head}_1; \dots; \text{head}_M]$$

$$\text{head}_m = \text{self-att}(Q_m, K_m, V_m).$$

$$Q_m = W_Q^m H, K = W_K^m X, V = W_V^m X$$

$$\text{self-att}(Q, K, V) = \text{softmax}\left(\frac{K^T Q}{\sqrt{d_h}} V\right)$$

10.2 注意力编码机制

例：对同一序列多头自注意力编码

$$Y = \text{MultiHead}(X, X, X)$$

◆ Head1 Self Attention

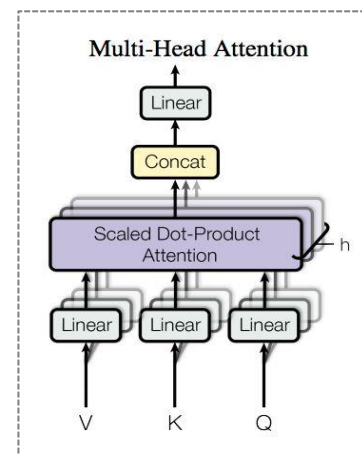
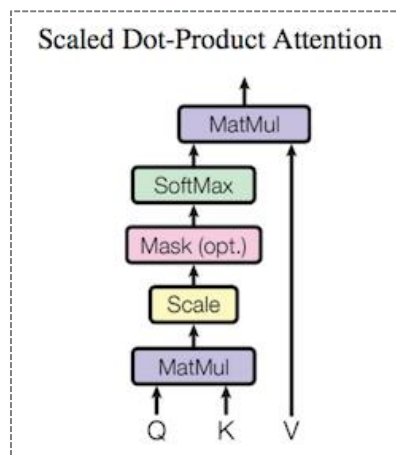
输出： $A_1(\text{我}) A_1(\text{弟弟}) A_1(\text{准备}) A_1(\text{一切}) A_1(\text{用品})$

输入：我 弟弟 准备 一切 用品

◆ Head2 Self Attention

输出： $A_2(\text{我}) A_2(\text{弟弟}) A_2(\text{准备}) A_2(\text{一切}) A_2(\text{用品})$

输入：我 弟弟 准备 一切 用品



◆ Multi-Head Self Attention

输出： $C_{1-2}(\text{我}) C_{1-2}(\text{弟弟}) C_{1-2}(\text{准备}) C_{1-2}(\text{一切}) C_{1-2}(\text{用品})$

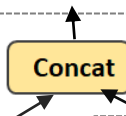
输入：

$A_1(\text{我}) A_1(\text{弟弟}) A_1(\text{准备}) A_1(\text{一切}) A_1(\text{用品})$

我 弟弟 准备 一切 用品

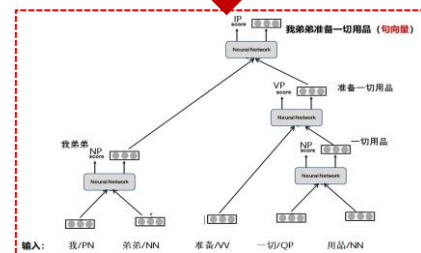
$A_2(\text{我}) A_2(\text{弟弟}) A_2(\text{准备}) A_2(\text{一切}) A_2(\text{用品})$

我 弟弟 准备 一切 用品



$C_{1-2}(\text{我}) C_{1-2}(\text{弟弟}) C_{1-2}(\text{准备}) C_{1-2}(\text{一切}) C_{1-2}(\text{用品})$

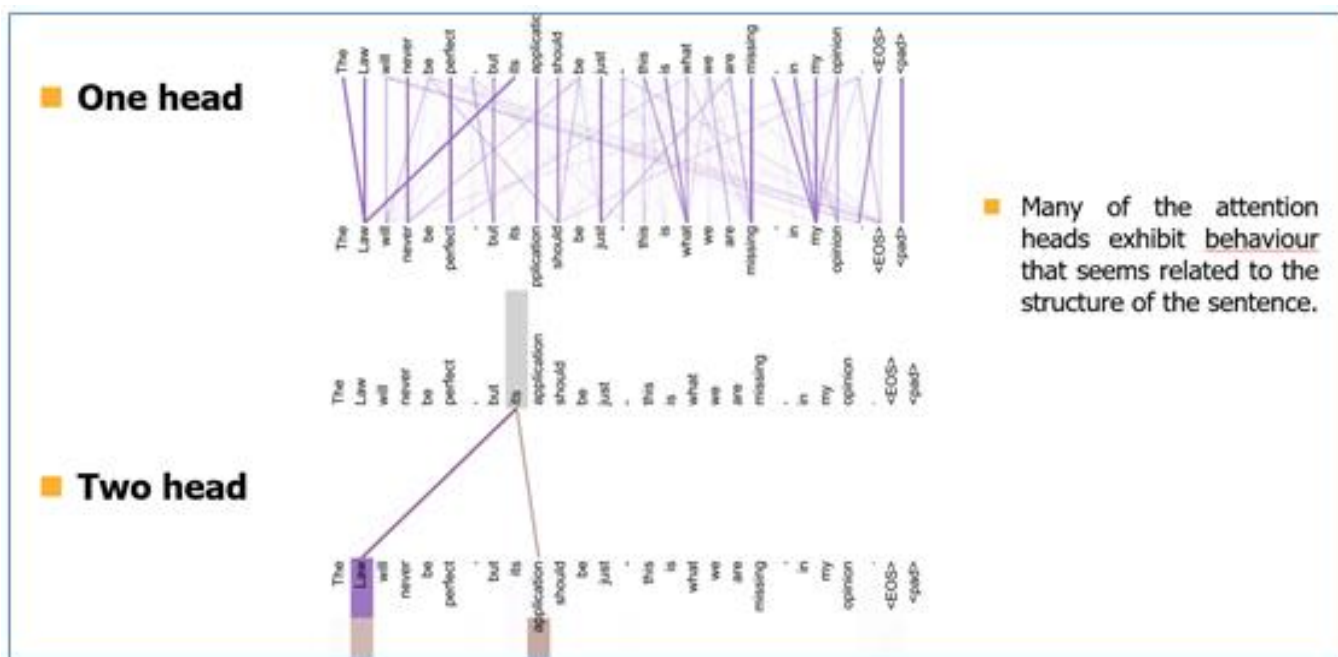
我 弟弟 准备 一切 用品



Multi-Head Self Attention 序列的含义？

10.2 注意力编码机制

多头自注意力的可视化的效果



在两个头和单头的比较中，可以看到单头"its"这个词只能学习到"law"的依赖关系，而两个头"its"不仅学习到了"law"还学习到了"application"依赖关系。多头能够从不同的表示子空间里学习相关信息

期 中 复 习

(上)

完