自然语言处理

4. 语言模型

虞剑飞

南京大学智能科学与技术学院

2025.3.19

本章内容

- 传统语言模型
- 神经语言模型

本章内容

- 传统语言模型
  - n元文法
  - 参数估计
  - 数据平滑方法
- 神经语言模型

本章内容

- 传统语言模型
  - n元文法
  - 参数估计
  - 数据平滑方法
- 神经语言模型

n元文法

- 大规模语料库的出现为自然语言统计处理方法的实现提供了可能，统计方法的成功应用推动了语料库语言学的发展。

- 基于大规模语料库的统计方法可以
  - 发现语言使用的普遍规律
  - 通过机器学习模型自动获取语言知识
  - 对未知语言现象进行推测

n元文法

- 如何计算一段文字(句子)的概率？

> 阳春三月春意盎然，少先队员脸上荡漾着喜悦的笑容，鲜艳的红领巾在他们的胸前迎风飘扬。

- 以一段文字(句子)为单位统计相对频率？

- 根据句子构成单位的概率计算联合概率?

$$p(s) = p(w_1) \times p(w_2) \times \cdots \times p(w_m)$$

n元文法

- 语句 $s = w_1 w_2 \ldots w_m$的先验概率：

$$p(s) = p(w_1)$$

n元文法

- 语句 $s = w_1 w_2 \ldots w_m$的先验概率：

$$p(s) = p(w_1) \times p(w_2|w_1)$$

n元文法

- 语句 $s = w_1 w_2 \ldots w_m$的先验概率：

$$p(s) = p(w_1) \times p(w_2|w_1) \times p(w_3|w_1 w_2)$$

n元文法

- 语句 $s = w_1 w_2 \ldots w_m$的先验概率：

$$p(s) = p(w_1) \times p(w_2|w_1) \times p(w_3|w_1 w_2) \times \cdots \times p\,(w_m|w_1 \ldots w_{m-1})$$

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

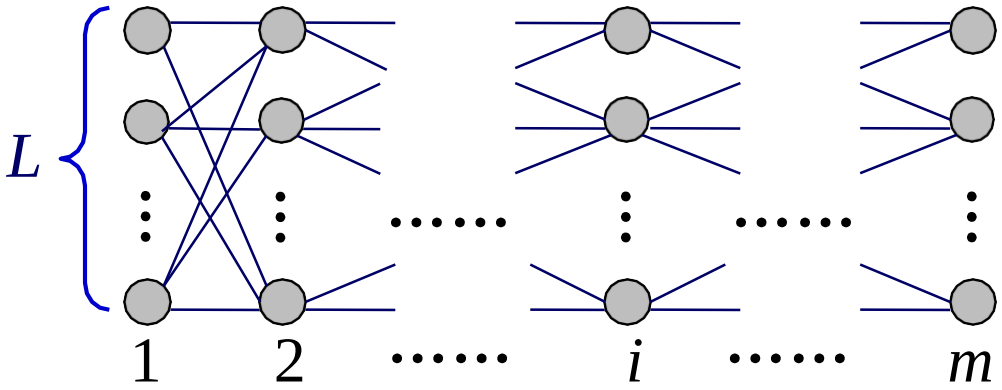
这里当$i = 1$时, $p(w_1|w_0) = p(w_1)$。

说明：(1) w_i 可以是字、词、短语或词类等，统称为统计基元。通常以“词”(token) 代之；(2) w_i 的概率取决于 $w_1, \ldots, w_{i-1}$，条件序列 $w_1, \ldots, w_{i-1}$ 称为 w_i 的历史(history)。

n元文法

- 问题
  - 随着历史基元数量的增加，不同的“历史”组合构成的路径数量指数级增长。对于第i (i > 1) 个统计基元，历史基元的个数为$i-1$，如果每个统计基元上共有 L 个不同的基元，如词汇表，理论上每一个单词都有可能出现在 1 到$i-1$的每一个位置上，那么, i 基元就有 L^{i-1} 种不同的历史组合。我们必须考虑在所有 L^{i-1} 种不同的历史条件下产生第i 个基元的概率。那么，对于长度为 m 的句子，模型中有 L^m 个自由参数 $p(w_m|w_1 \ldots w_{m-1})$。

如果L=6763, m=3,
自由参数的数目为
3.09×10^{11}!



n元文法

- 问题的解决方法

设法减少历史基元的个数，将 $w_1, w_2, \ldots, w_{i-1}$ 映射到等价类 $S(w_1, w_2, \ldots, w_{i-1})$，使等价类的数目远远小于原来不同历史基元的数目。从而使得

$$p(w_i|w_1, \ldots, w_{i-1}) = p(w_i|S(w_1, w_2, \ldots, w_{i-1}))$$

n元文法

- 如何划分等价类

将两个历史映射到同一个等价类，当且仅当这两个历史中的最近 $n-1$ 个基元相同，即：

$$H_1: w_1 w_2 \dots\dots \underbrace{w_{i-n+1} w_{i-n+2} \dots w_{i-1}} w_i \dots\dots$$

$$n-1$$

$$H_2: v_1 v_2 \dots\dots \underbrace{v_{k-n+1} v_{k-n+2} \dots v_{k-1}} v_k \dots\dots$$

$$S(w_1, w_2, \dots, w_i) = S(v_1, v_2, \dots, v_k)$$

$$iff \quad H_1: (w_{i-n+1}, \dots, w_i) = H_2: (v_{k-n+1}, \dots, v_k)$$

n元文法

- 这种计算语言（通常指句子）概率的模型称为语言模型(language model)。由于通常只考虑历史基元与当前词构成 n 元词序列（即只考虑前 $n-1$ 个词的历史情况），因此这种模型又称为n元文法模型 (n-gram model)。n 为整数，通常 $n=1\sim5$。

  - 当$n=1$ 时，出现在第i 位置上的基元w_i 独立于历史，称为一元文法，记作uni-gram 或monogram；
  - 当$n=2$ 时，出现在第i 位置上的基元w_i 只与$i-1$位置上的基元相关，称为2元文法(2-gram 或bi-gram)。2元文法序列又被称为1阶马尔可夫链；
  - 当$n=3$ 时，出现在第i 位置上的基元w_i 与 $i-2$ 和$i-1$ 位置上的基元相关，称为三元文法(3-gram 或tri-gram)。三元文法序列又被称为2阶马尔可夫链。
  - 依次类推。

n元文法

- n元文法就是n个临近词构成的n元词序列，例如：

I came from New York, USA and working in Beijing now.

- **bi-grams**: I came, came from, from New, New York, York ,, , USA, USA and, and working, working in, in Beijing, Beijing now, now .

今天是个灿烂的日子。

今天　是　个　灿烂　的　日子。

- **tri-grams**: 今天是个，是个灿烂，个灿烂的，灿烂的日子，的日子。

n元文法

为了保证条件概率在 $i = 1$ 时有意义，同时保证句子内所有字符串的概率为1，即$\sum_s p(s) = 1$，可以在句子首尾两端增加两个标志: $< \text{BOS} > w_1 w_2 \cdots w_m < \text{EOS} >$。不失一般性，对于 $n > 2$ 的n-grams，$p(s)$ 可以分解为：

$$p(s) = \prod_{i=1}^{m+1} p(w_i | w_{i-n+1}^{i-1})$$

$$p(s) = p(w_1|w_0) \times p(w_2|w_1 w_0) \times \cdots \times p\,(w_m|w_{m-n+1} \cdots w_{m-1}) \times p\,(w_{m+1}|w_{m-n+2} \cdots w_m)$$

其中，w 表示词序列 $w_i \cdots w_j$, w_{i-n+1} 从 w_0 开始， w_0 为 <BOS>，w_{m+1}为 <EOS>。

n元文法

- 应用示例：音字转换问题

输入拼音串：ta shi yan jiu sheng wu de　　*Py*

可能的汉字：*CStri*
- 踏实研究生物的
- 他实验救生物的
- 他使烟酒生物的
- 他是研究生物的
- ……

$$\tilde{C}Stri = \arg\max_{CStri} p(CStri|Py)$$

$$= \arg\max_{CStri} \frac{p(Py|CStri) \times p(CStri)}{p(Py)}$$

$$\simeq \arg\max_{CStri} p(Py|CStri) \times p(CStri)$$

$$\approx \arg\max_{CStri} p(CStri)$$

n元文法

- 应用示例：音字转换问题

CStri ={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，……}

如果使用 2-gram：

$p(CStri_1)$ =p(踏实|<BOS>) × p(研究|踏实) × p(生物|研究) ×
　　p(的|生物) × p(<EOS>|的)

$p(CStri_2)$ =p(他|<BOS>) × p(实验|他) × p(救生|实验) ×
　　p(物|救生) × p(的|物) × p(<EOS>|的)

……

如果汉字的总数为N，使用一元文法时搜索空间为N，只选择使用频率最高的汉字；使用2元文法时搜索空间为N^2，效果比一元文法明显提高；对于汉字而言，4元文法效果会好一些。智能狂拼、微软拼音输入法都是基于n-gram 实现的。

本章内容

- 传统语言模型
  - n元文法
  - 参数估计
  - 数据平滑方法
- 神经语言模型

参数估计

- 基本思路
  - 收集、标注大规模样本，我们称其为训练数据/语料 (training data / corpus)。
  - 利用最大似然估计 (maximum likelihood evaluation, MLE) 方法计算概率。

参数估计

对于 *n*-gram，参数通过最大似然估计进行计算：

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

其中，$f(w_i|w_{i-n+1}^{i-1})$是在给定 w_{i-n+1}^{i-1} 的条件下 w_i 出现的相对频度。分子为 w_{i-n+1}^{i-1} 与 w_i 同现的次数，分母$\sum_{w_i} c(w_{i-n+1}^{i})$是历史串 w_{i-n+1}^{i-1} 在给定语料中出现的次数。

参数估计

- 课堂练习
  - 例如，给定训练语料：

John read Moby Dick,

Mary read a different book,

She read a book by Cher

- 根据2元文法求句子“John read a book”的概率如下：

<BOS>John read Moby Dick<EOS>
<BOS>Mary read a different book<EOS>
<BOS>She read a book by Cher<EOS>

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

参数估计

- 根据2元文法求句子“Cher read a book”的概率如下：

<BOS>John read Moby Dick<EOS>
<BOS>Mary read a different book<EOS>
<BOS>She read a book by Cher<EOS>

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

本章内容

- 传统语言模型
  - n元文法
  - 参数估计
  - 数据平滑方法
- 神经语言模型

数据平滑方法

- 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。
  - 目标：测试样本的语言模型**困惑度越小越好**。
  - 约束：

$$\sum_{w_i} p\left(w_i \middle| w_{i-n+1}^{i-1}\right) = 1$$

数据平滑方法

- 困惑度
  - 平滑的*n*-gram概率为 $p(w_i|w_{i-n+1}^{i-1})$，句子 s 的概率:

$$p(s) = \prod_{i=1}^{m+1} p(w_i|w_{i-n+1}^{i-1})$$

  - 假定测试语料 T 由l_T个句子构成：$(s_1, s_2, \cdots, s_{l_T})$，共含$w_T$个词，那么，整个测试集的概率为：

$$p(T) = \prod_{i=1}^{l_T} p(s_i)$$

困惑度：$PP_P(T) = 2^{-\frac{1}{w_T}\log_2 p(T)}$

n-gram 对于英语文本的困惑度范围一般为10~1000，语言模型设计的任务就是寻找困惑度最小的模型，使其最接近真实的语言。

数据平滑方法

- 数据平滑方法：
  - 加1法 (additive)
  - 减值法/折扣法 (discounting)
  - 删除插值法 (deleted interpolation)

数据平滑方法

- 数据平滑方法：
  - 加1法 (additive)
  - 减值法/折扣法 (discounting)
  - 删除插值法 (deleted interpolation)

数据平滑方法

- 加1法 (additive)
  - 基本思想：每一种情况出现的次数加1。
  - 例如，对于 *uni*-gram，设w_1, w_2, w_3 三个词， w_1出现1次， w_2出现0次，w_3出现两次，概率分别为：1/3, 0, 2/3，加1后情况？

2/6, 1/6, 3/6

  - 对于2-gram 有：

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

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

    - 其中, V 为被考虑语料的词汇表(全部可能为历史基元)。

数据平滑方法

- 加1法 (additive)
  - 在前面 3 个句子的例子中,

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

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

<BOS>John read Moby Dick<EOS>
<BOS>Mary read a different book<EOS>
<BOS>She read a book by Cher<EOS>

原来：

$$p(\text{Cher}|\text{<BOS>}) = 0/3$$

$$p(\text{read}|\text{Cher}) = 0/1$$

$$p(\text{a}|\text{read}) = 2/3$$

$$p(\text{book}|\text{a}) = 1/2$$

$$p(\text{<EOS>}|\text{book}) = 1/2$$

数据平滑方法

- 加1法 (additive)

  词汇量：$|V|=13$

<BOS>John read Moby Dick<EOS>
<BOS>Mary read a different book<EOS>
<BOS>She read a book by Cher<EOS>

平滑以后：

$$p(\text{Cher}|\text{<BOS>}) = (0+1)/(13+3) = 1/16$$

$$p(\text{read}|\text{Cher}) = (0+1)/(13+1) = 1/14$$

$$p(\text{a}|\text{read}) = (1+2)/(13+3) = 3/16$$

$$p(\text{book}|\text{a}) = (1+1)/(13+2) = 1/15$$

$$p(\text{<EOS>}|\text{book}) = (1+1)/(13+2) = 2/15$$

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

$$p(\text{Cher read a book}) = \frac{1}{16} \times \frac{1}{14} \times \frac{3}{16} \times \frac{2}{15} \times \frac{2}{15} \approx 0.00001$$

数据平滑方法

- 加1法 (additive)

  词汇量：$|V|=13$

<BOS>John read Moby Dick<EOS>
<BOS>Mary read a different book<EOS>
<BOS>She read a book by Cher<EOS>

同理，其它 bi-grams 的概率变为：

$$p(\text{John}|\text{<BOS>}) = 2/16$$
$$p(\text{read}|\text{John}) = 2/14$$
$$p(\text{a}|\text{read}) = 3/16$$
$$p(\text{book}|\text{a}) = 2/15$$
$$p(\text{<EOS>}|\text{book}) = 2/15$$

于是：

$$p(\text{John read a book})$$
$$= p(\text{John}|\text{<BOS>}) \times$$
$$p(\text{read}|\text{John}) \times p(\text{a}|\text{read}) \times$$
$$p(\text{book}|\text{a}) \times p(\text{<EOS>}|\text{book})$$
$$\approx 0.00006$$

本章内容

- 传统语言模型
  - n元文法
  - 参数估计
  - 数据平滑方法
- 神经语言模型
  - 问题的提出
  - 前馈神经网络语言模型
  - 循环神经网络语言模型

本章内容

- 传统语言模型
  - n元文法
  - 参数估计
  - 数据平滑方法
- 神经语言模型
  - 问题的提出
  - 前馈神经网络语言模型
  - 循环神经网络语言模型

问题的提出

- 回顾*n*-gram

句子s的概率：
$$p(s) = \prod_{t=1}^{m} p(w_t | w_1 \dots w_{t-1})$$
$$= \prod_{t=1}^{m} p(w_t | w_{t-n+1}^{t-1})$$

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

问题的提出

- 回顾*n*-gram

这 本 小说 很 **枯燥** ， 读 起来 很 **乏味** 。

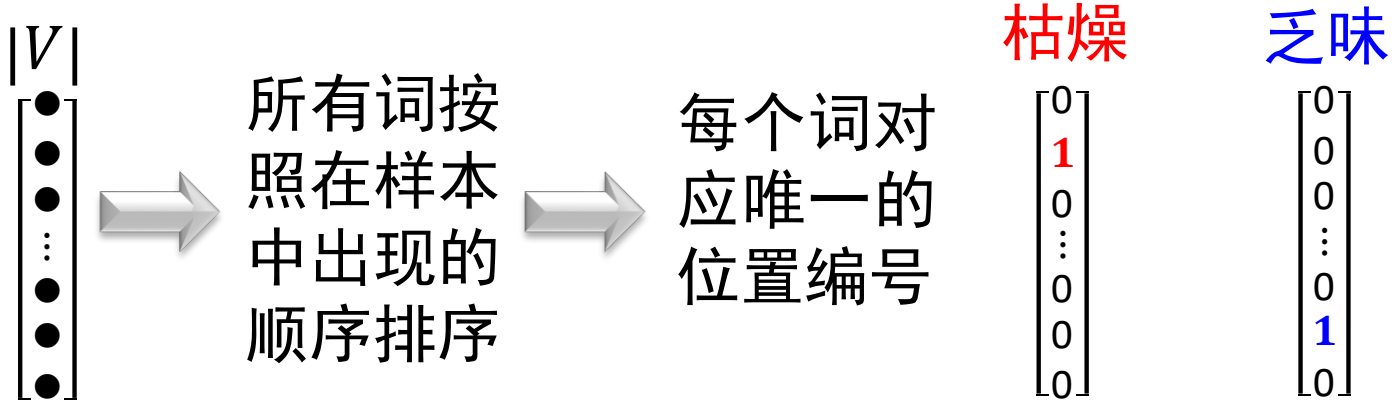
$$p(\text{枯燥}|\text{很}) = \frac{count(\text{很枯燥})}{count(\text{很})} \qquad p(\text{乏味}|\text{很}) = \frac{count(\text{很乏味})}{count(\text{很})}$$

问题①：数据稀疏
n-gram“很 乏味”有可能未出现在训练样本中。

问题②：忽略语义相似性
“枯燥”与“乏味”虽然语义相似，但无法共享信息。

问题的提出

- 分析原因
  - “词”以词形本身表示，是离散的符号。
  - 等价的表示：one-hot 向量

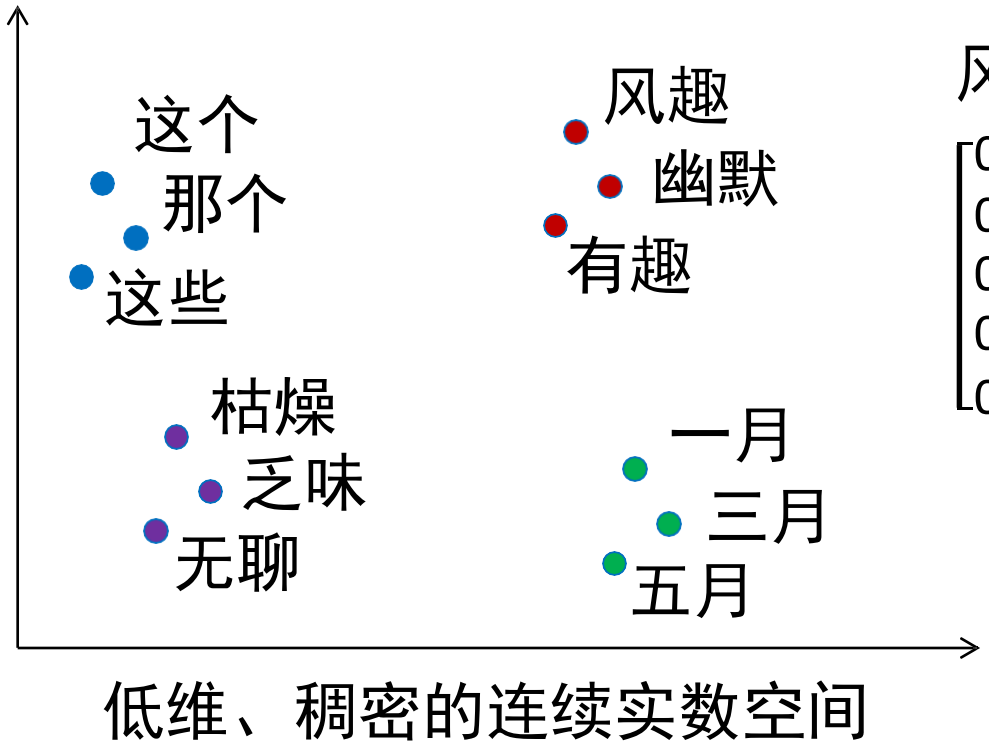


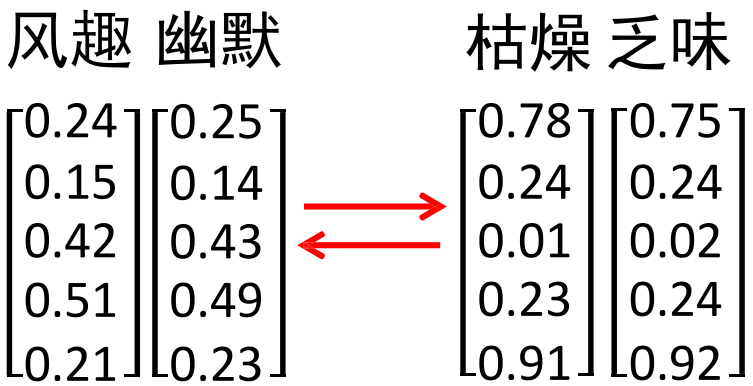
问题：枯燥⊗乏味= 0，任意两个词之间的相似度都为0。

维度太高！

问题的提出

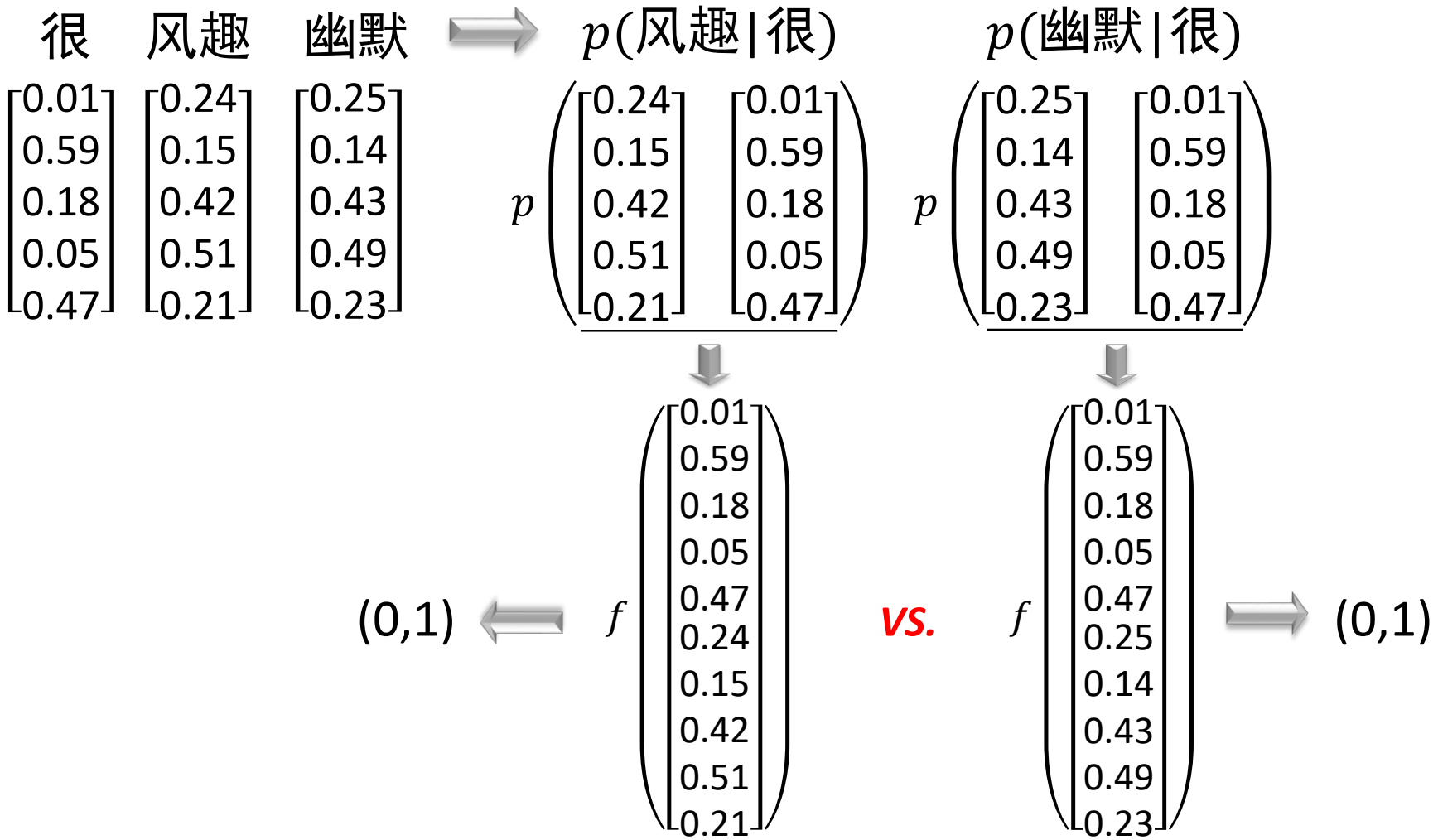
- 解决方式
  - 探索：是否可以用连续空间的分布式表示赋予词向量呢？





问题的提出

- 解决方式

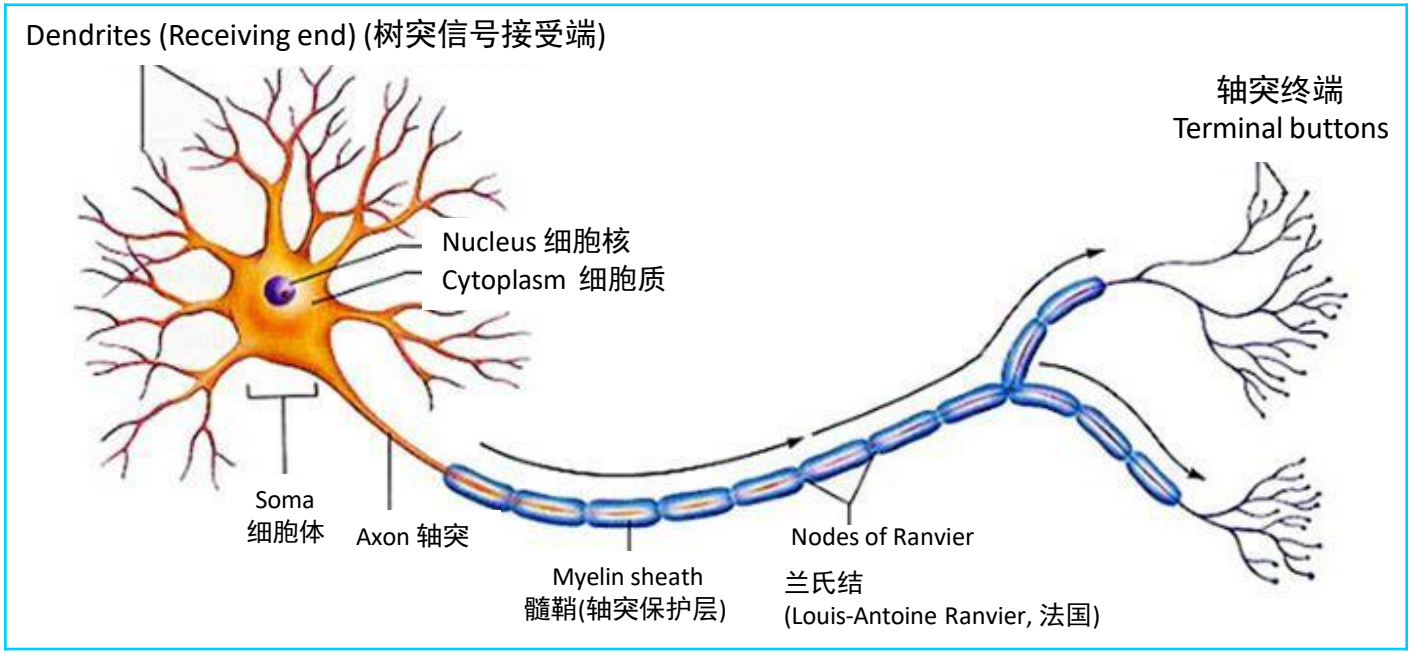

很
风趣
幽默
p(风趣|很)
p(幽默|很)
0.01 0.59 0.18 0.05 0.47
0.24 0.15 0.42 0.51 0.21
0.25 0.14 0.43 0.49 0.23
p
0.24 0.15 0.42 0.51 0.21
0.01 0.59 0.18 0.05 0.47
0.25 0.14 0.43 0.49 0.23
0.01 0.59 0.18 0.05 0.47 0.24 0.15 0.42 0.51 0.21
f
(0,1)
VS.
0.01 0.59 0.18 0.05 0.47 0.25 0.14 0.43 0.49 0.23
f
(0,1)

本章内容

- 传统语言模型
  - n元文法
  - 参数估计
  - 数据平滑方法
- 神经语言模型
  - 问题的提出
  - 前馈神经网络语言模型
  - 循环神经网络语言模型

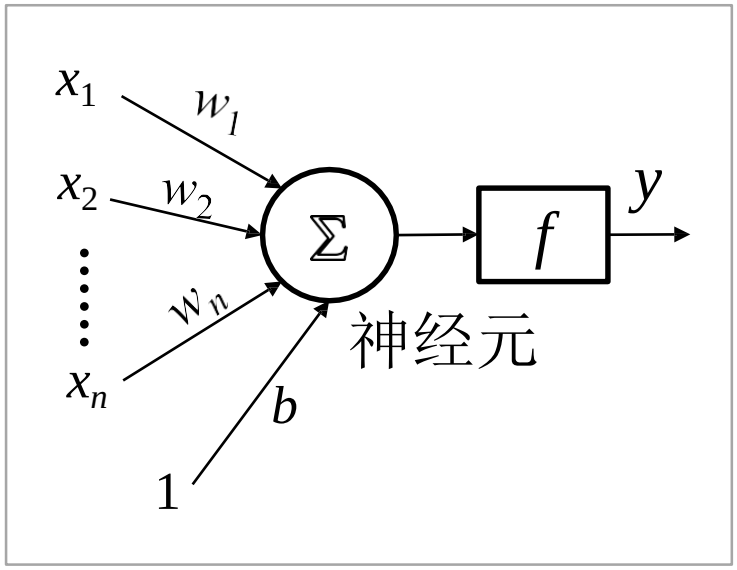
前馈神经网络语言模型

- 人工神经网络 (artificial neural networks, ANN) 是1943年心理学家沃伦•麦卡洛克(W. McCulloch)和数理逻辑学家 W. Pitts 建立了神经网络的数学模型，提出了神经元的形式化数学描述和网络结构方法。



前馈神经网络语言模型

- 神经元数学描述
  - $x_1 \sim x_n$为输入向量的各分量；
  - $w_1 \sim w_n$为权值系数(传递效率)；
  - b为偏置；
  - f为传递函数(激发/激活函数)，通常是非线性的；
  - $\sum$是神经元的阈值；
  - y为神经元的输出。

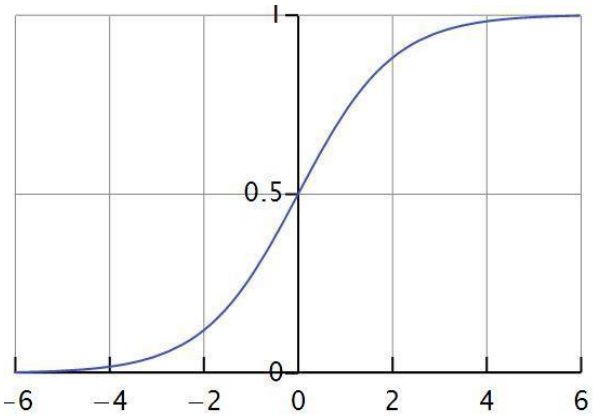


前馈神经网络语言模型

- 神经元数学描述
  - 常用的激活函数：

①Logistic/Sigmoid函数：

$$\sigma(x) = \frac{1}{1 + \exp(-x)}$$

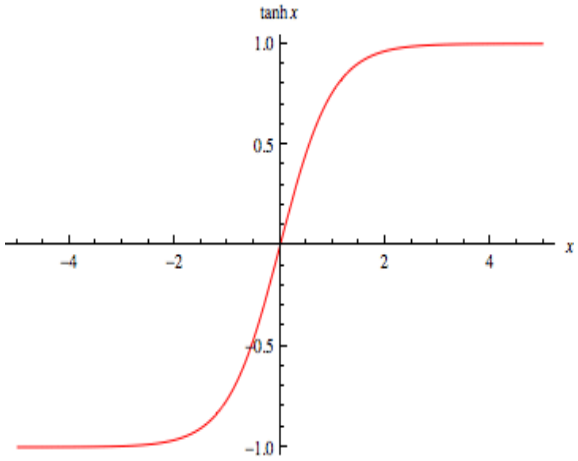


Logistic函数可以看成是一个“挤压”函数，把一个实数域的输入“挤压”到(0, 1)。当输入值在0附近时，似为线性函数；当输入值靠近两端时，对输入进行抑制。输入越小，越接近于0；输入越大，越接近于1。这种特点与生物神经元类似，对某些输入会产生兴奋（输出为1），对另一些输入产生抑制（输出为0）。

前馈神经网络语言模型

- 神经元数学描述
  - 常用的激活函数：

②Tanh 函数



$$\tanh = \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)}$$

或者：

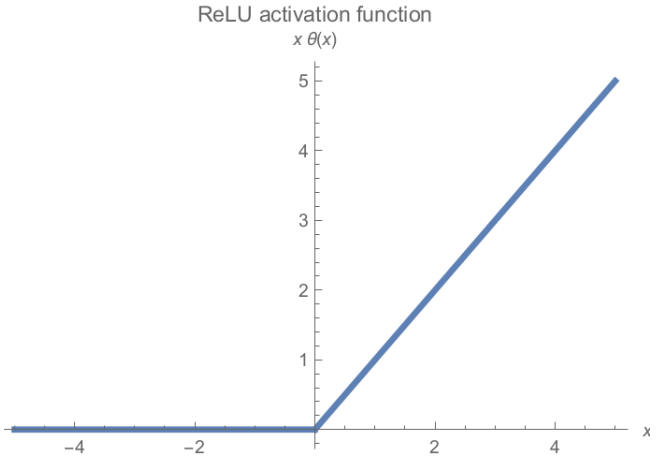
$$\tanh(x) = 2\sigma(2x) - 1$$

Tanh 函数可以看作是放大并平移的Logistic 函数，其值域是(−1, 1)。

前馈神经网络语言模型

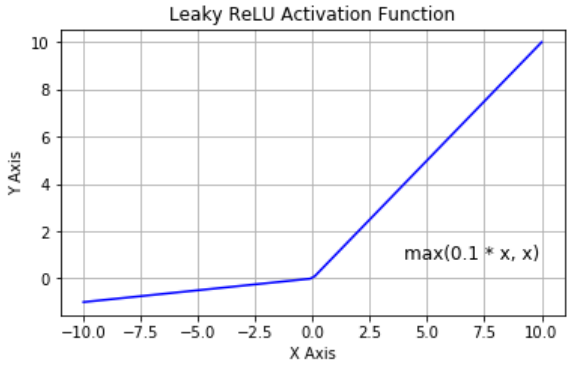
- 神经元数学描述
  - 常用的激活函数：

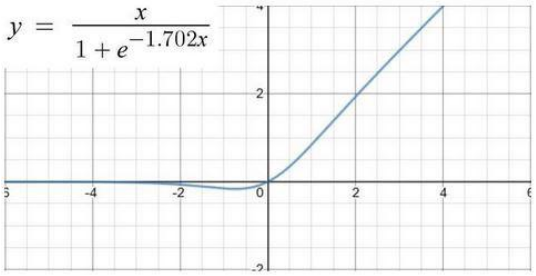
③ReLU (rectified linear unit) 函数

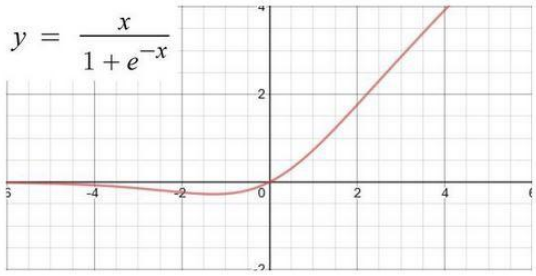


$$\mathrm{ReLU}(x) = max(0, x)$$

  - ReLU 函数有很多变种，比如LeakyReLU，GELU。

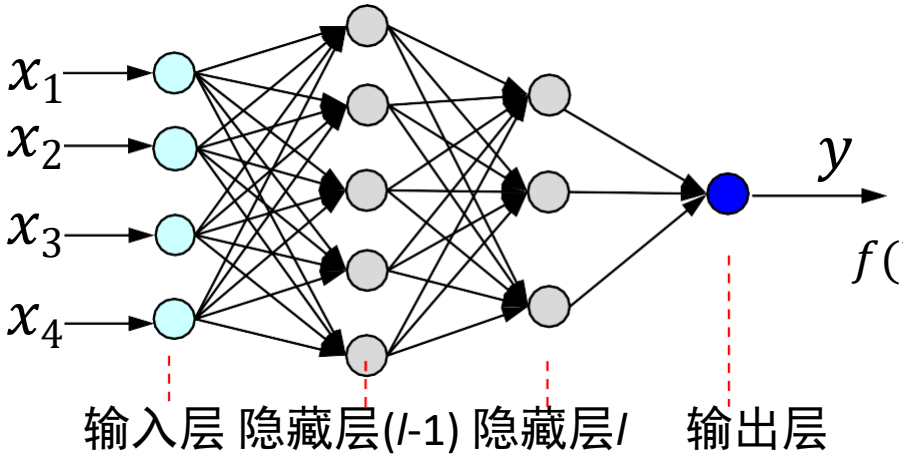






前馈神经网络语言模型

- 前馈神经网络 (Feedforward Neural Network，FNN)
  - FNN是最早发明的简单人工神经网络，也经常被称为多层感知器(Multi-Layer Perceptron, MLP)。前馈网络中各个神经元按接收信息的先后分为不同的组，每一组可以看作一个神经层，每一层中的神经元接收前一层神经元的输出，并输出到下一层神经元，整个网络中的信息是朝一个方向传播，没有反向的信息传播，可以看作是一个有向的无环图。

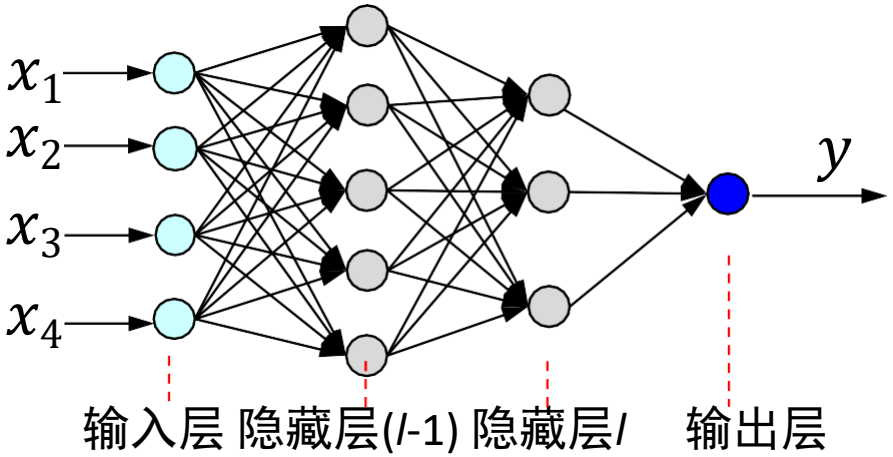


$$f(\boldsymbol{W}^3\big(f(\boldsymbol{W}^2 f(\boldsymbol{W}^1 \cdot \boldsymbol{x} + b^1) + b^2)\big) + b^3)\cdots$$

前馈神经网络语言模型

- 前馈神经网络 (Feedforward Neural Network，FNN)
  - 网络表示：L为神经网络的层数；M_l为第l层的神经元个数；$f_l(\cdot)$为第l层神经元的激活函数；

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \quad \boldsymbol{W}^{(l)} = \underbrace{\begin{bmatrix} w_{11} & w_{21} & & w_{(l-1)1} \\ w_{12} & w_{22} & \vdots & w_{(l-1)2} \\ \vdots & \vdots & \vdots & \vdots \\ w_{1M_l} & w_{2M_l} & & w_{(l-1)M_l} \end{bmatrix}}_{M_{l-1}} \in \mathbb{R}^{M_l \times M_{l-1}} \Big\} M_l$$



$$\boldsymbol{b}^{(l)} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_{M_l} \end{bmatrix}$$

前馈神经网络语言模型

- 前馈神经网络 (Feedforward Neural Network，FNN)
  - 参数学习：确定网络中所有的 $\boldsymbol{W}$ 和 $\boldsymbol{b}$

前馈神经网络语言模型

- 前馈神经网络 (Feedforward Neural Network，FNN)
  - 基于反向传播(Back Propagation)算法的随机梯度下降参数训练过程

输入： 训练集 $\mathcal{D}=\left\{\left(\boldsymbol{x}^{(n)},\boldsymbol{y}^{(n)}\right)\right\}_{n=1}^{N}$，验证集 $\mathcal{V}$，学习率 η，正则化系数 λ，网络层数 L，神经元数量 M_l，$1\le l\le L$。

输出： $\boldsymbol{W},\boldsymbol{b}$

重复执行该过程：

(1) 对训练集 $\mathcal{D}$ 中的样本随机重排序；

(2) For n=1 … N do

从训练集 $\mathcal{D}$ 中选取样本 $\left(\boldsymbol{x}^{(n)},\boldsymbol{y}^{(n)}\right)$；

前馈计算每一层的净输入 $\boldsymbol{z}^{(l)}$ 和激活值 $\boldsymbol{\alpha}^{(l)}$，直到最后一层；

反向传播计算每一层的误差 $\delta^{(l)}$；

// 计算每一层参数的导数：
$$\begin{cases}\forall l,\dfrac{\partial\mathcal{L}(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{W}^{(l)}}=\delta^{(l)}(\boldsymbol{\alpha}^{(l-1)})^{T}\\ \forall l,\dfrac{\partial\mathcal{L}(\boldsymbol{y},\hat{\boldsymbol{y}})}{\partial\boldsymbol{b}^{(l)}}=\delta^{(l)}\end{cases}$$

// 更新参数：
$$\begin{cases}\boldsymbol{W}^{(l)}\leftarrow\boldsymbol{W}^{(l)}-\eta\left(\delta^{(l)}\cdot\left(\boldsymbol{\alpha}^{(l-1)}\right)^{T}+\lambda\boldsymbol{W}^{(l)}\right);\\ \boldsymbol{b}^{(l)}\leftarrow\boldsymbol{b}^{(l)}-\eta\delta^{(l)};\end{cases}$$

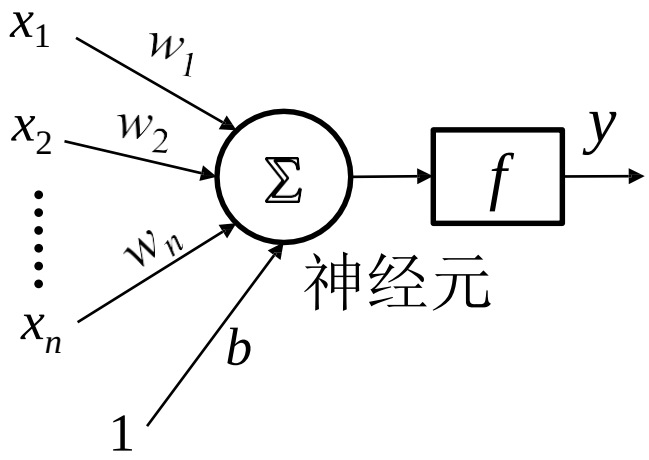
end

直到模型在验证集 $\mathcal{V}$ 上的损失函数值收敛，结束过程，输出 $\boldsymbol{W},\boldsymbol{b}$。

前馈神经网络语言模型

- 基于前馈神经网络的语言模型

$$p(\text{风趣}|\text{很}) = f\left(\begin{bmatrix}0.01\\0.59\\0.18\\0.05\\0.47\\0.24\\0.15\\0.42\\0.51\\0.21\end{bmatrix}\right) \Rightarrow ? \quad (0,1)$$



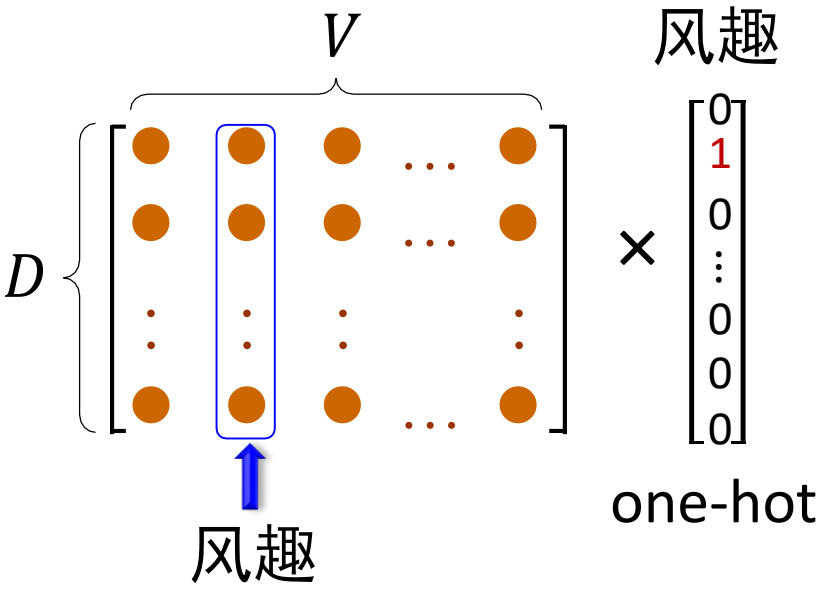
$$p(\text{幽默}|\text{很}) = f\left(\begin{bmatrix}0.01\\0.59\\0.18\\0.05\\0.47\\0.25\\0.14\\0.43\\0.49\\0.23\end{bmatrix}\right) \Rightarrow ? \quad (0,1)$$

词向量？

$\boldsymbol{W}, \boldsymbol{b} = ?$

前馈神经网络语言模型

- 基于前馈神经网络的语言模型
  - 词向量表示（Word Embeddings）



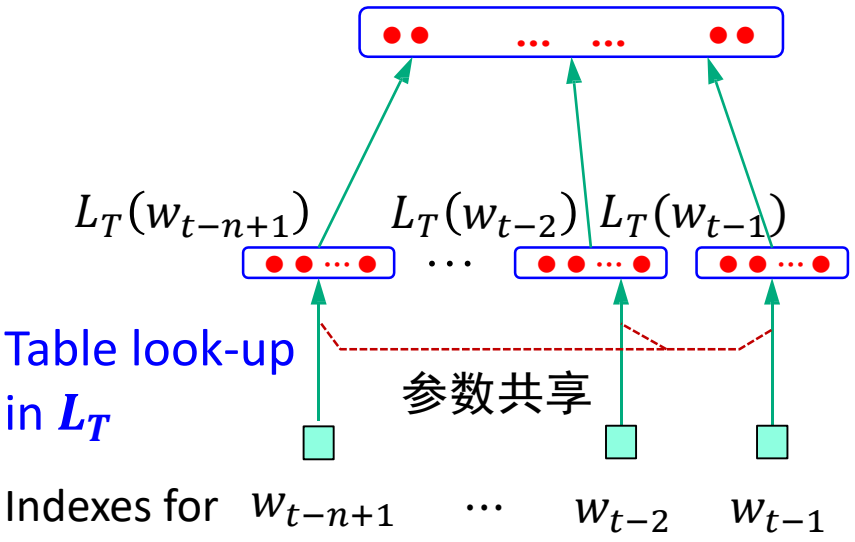
如何学习L_T?

通常先随机初始化，然后通过目标函数优化词的向量表示(如最大化语言模型似然度)。

- V的确定：(1)训练数据中所有词；(2)频率高于某个阈值的所有词；(3)前V个频率最高的词。
- D的确定：超参数，人工设定，一般从几十到几百。

前馈神经网络语言模型

- 基于前馈神经网络的语言模型

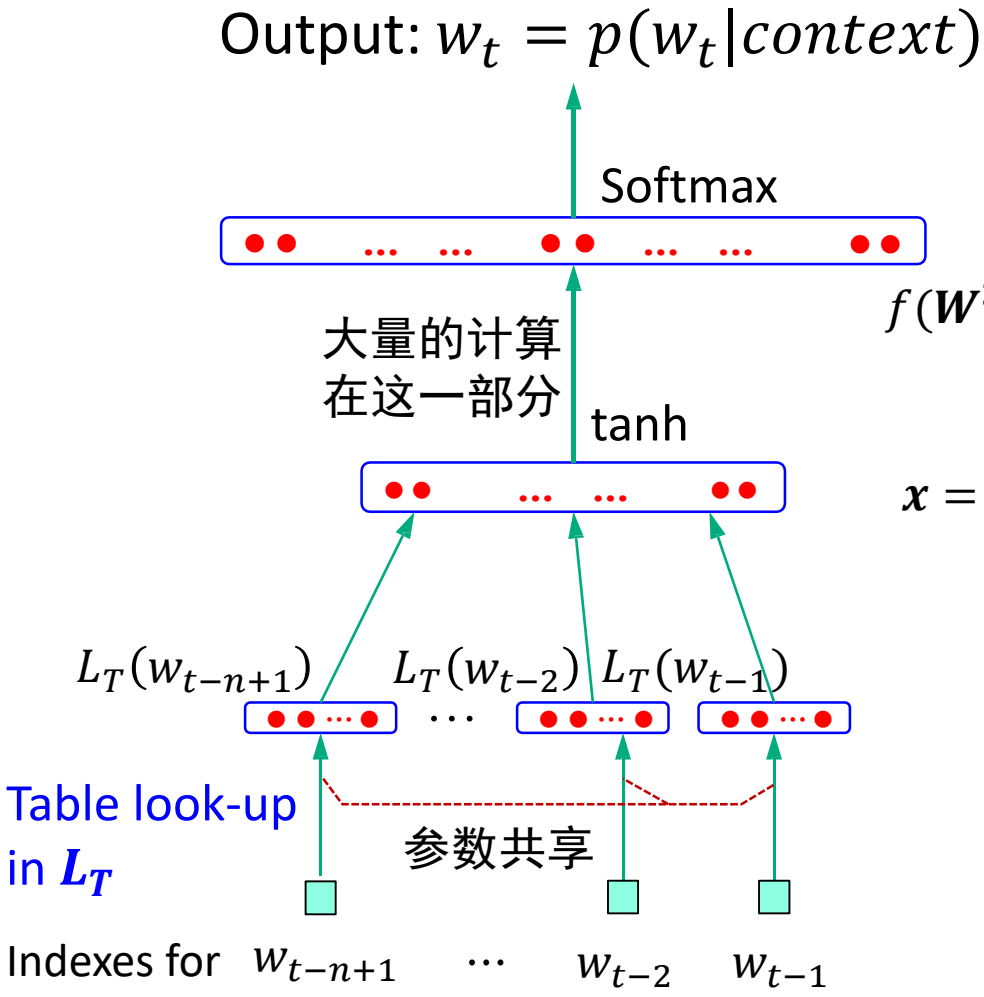


$$\boldsymbol{x} = (L_T(w_{t-n+1}), \dots, L_T(w_{t-2}), L_T(w_{t-1}))$$

(Bengio *et al.*, 2003)

前馈神经网络语言模型

- 基于前馈神经网络的语言模型



$$f(\boldsymbol{W}^3\big(f(\boldsymbol{W}^2 f(\boldsymbol{W}^1 \cdot \boldsymbol{x} + b^1) + b^2)\big) + b^3)\cdots$$

$$\boldsymbol{x} = (L_T(w_{t-n+1}), \ldots, L_T(w_{t-2}), L_T(w_{t-1}))$$

(Bengio *et al.*, 2003)

前馈神经网络语言模型

- 基于前馈神经网络的语言模型
  - Softmax 回归 (regression) 也称为多项(Multinomial) 或多类(Multi-Class) 的Logistic回归，是Logistic回归在多分类问题上的推广，它也可以看作是一种条件最大熵模型。
  - 对于多类问题，类别标签 $y \in \{1,2,\cdots,C\}$ 可以有 C 个取值，给定一个样本 $\boldsymbol{x}$, Softmax回归预测的属于类别 c 的条件概率为:

$$p(y=c|\boldsymbol{x}) = \text{Softmax}(\boldsymbol{w}_c^T\boldsymbol{x})$$

$$= \frac{\exp(\boldsymbol{w}_c^T\boldsymbol{x})}{\sum_{c'=1}^{C}\exp(\boldsymbol{w}_{c'}^T\boldsymbol{x})}$$

其中，$\boldsymbol{w}_c$是第 c 类的权重向量。

向量表示：

$$\hat{\boldsymbol{y}} = \text{Softmax}(\boldsymbol{W}^T\boldsymbol{x}) = \frac{\exp(\boldsymbol{W}^T\boldsymbol{x})}{1_C^T\exp(\boldsymbol{W}^T\boldsymbol{x})}$$

其中，$\boldsymbol{W} = [\boldsymbol{w}_1,\ldots,\boldsymbol{w}_c]$是由$C$个类的权重向量组成的矩阵，$1_C$为$C$维的全1向量。为所有类别预测的条件概率组成的向量，第c维的值就是第c类的预测条件概率。

前馈神经网络语言模型

- 举例

$$L_T = \begin{bmatrix} \dots 0.1 \dots & 0.5 \dots & 0.3 \dots & 0.4 \dots & 0.2 \dots \\ \dots 0.3 \dots & 0.4 \dots & 0.2 \dots & 0.2 \dots & 0.1 \dots \end{bmatrix} \qquad 2 \times 5000$$

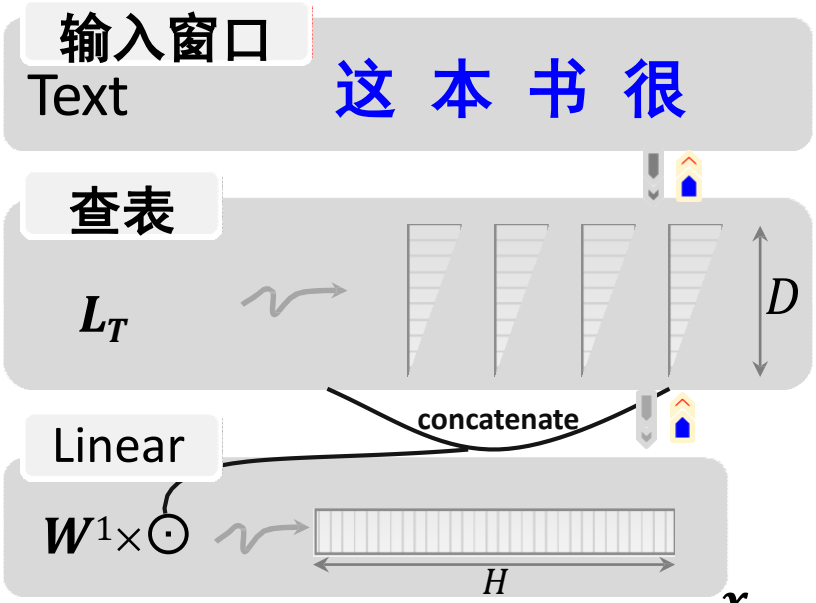
本… 很… 乏味… 书… 这…

输入词串：这 本 书 很 __ ?

$\Longrightarrow p(\_\_ | 这，本，书，很)$

前馈神经网络语言模型

- 举例



$p(\_\_|这，本，书，很)$

①查词表

0.2	0.1	0.4	0.5
0.1	0.3	0.2	0.4
↑	↑	↑	↑
这	本	书	很

②拼接所有的条件词的向量，形成一个向量。

或者加和，取平均

这 本 书 很

$\rightarrow (0.2, 0.1, 0.1, 0.3, 0.4, 0.2, 0.5, 0.4)^T$

③隐藏层：线性映射+非线性变换。

$$\boldsymbol{W}^1 = \begin{pmatrix} 0.1 & 0 & 0.2 & 0.4 & 0.2 & 0.1 & 0 & 0.3 \\ 0.5 & 0.4 & 0.2 & 0 & 0.2 & 0.6 & 0 & 0.2 \end{pmatrix} \times \boldsymbol{x} = \begin{pmatrix} 0.2 \\ 0.1 \\ 0.1 \\ 0.3 \\ 0.4 \\ 0.2 \\ 0.5 \\ 0.4 \end{pmatrix}$$

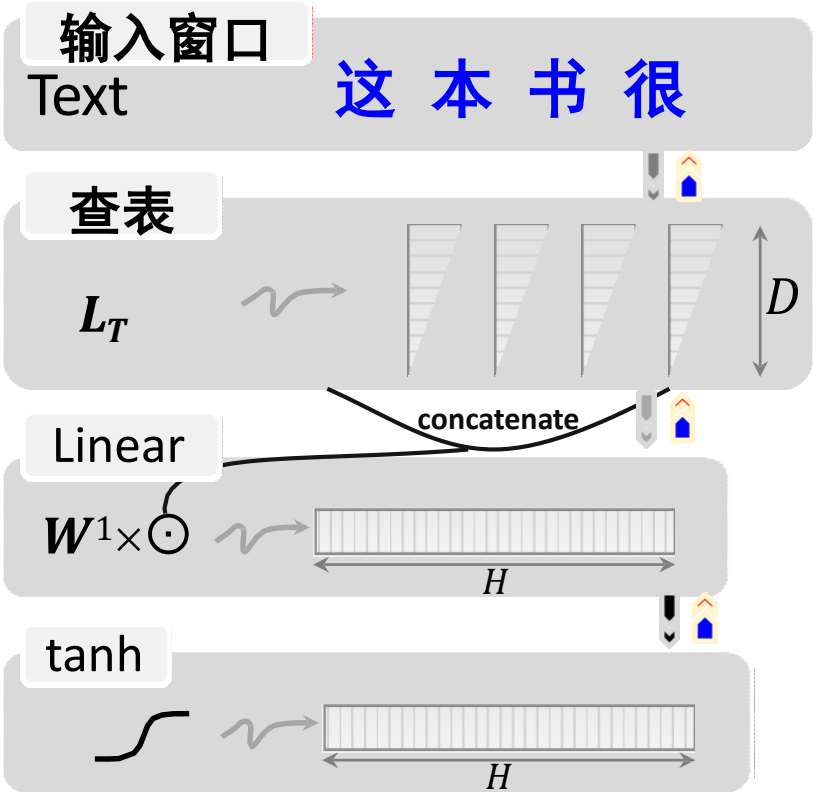
$$\boldsymbol{W}^1 \times \boldsymbol{x} \Rightarrow \begin{pmatrix} 0.38 \\ 0.44 \end{pmatrix}$$

任设的初始值（0~1之间，均匀或高斯分布）

暂不考虑偏置b

前馈神经网络语言模型

- 举例



$p(\_\_|这，本，书，很)$

①查词表

0.2	0.1	0.4	0.5
0.1	0.3	0.2	0.4
↑	↑	↑	↑
这	本	书	很

②拼接所有的条件词的向量，形成一个向量。

或者加和，取平均

这 本 书 很

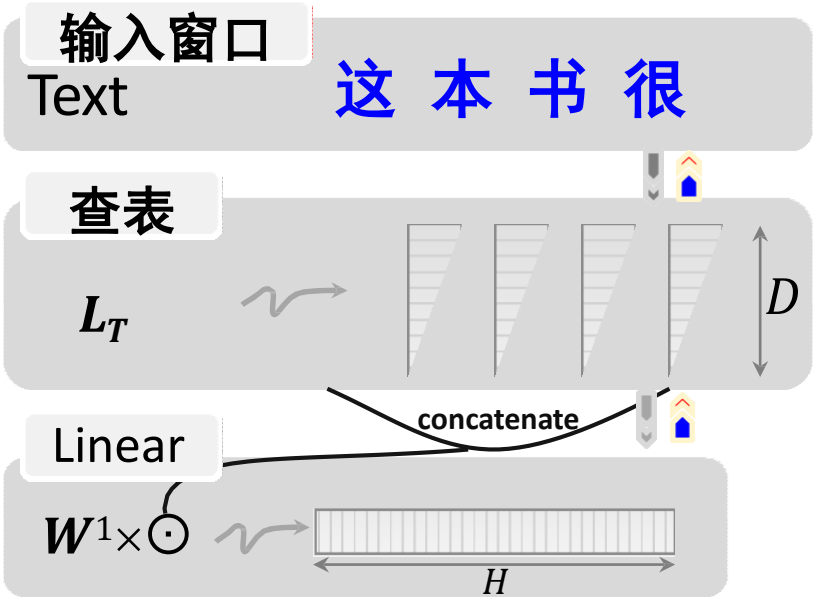
$\rightarrow (0.2, 0.1, 0.1, 0.3, 0.4, 0.2, 0.5, 0.4)^T$

③隐藏层：线性映射+非线性变换。

$$W^1 \times x \rightarrow \begin{pmatrix} 0.38 \\ 0.44 \end{pmatrix} \xrightarrow{\tanh(\bullet)} \begin{pmatrix} 0.36 \\ 0.41 \end{pmatrix}$$

前馈神经网络语言模型

- 举例



$p(\_\_|这，本，书，很)$

①查词表

0.2	0.1	0.4	0.5
0.1	0.3	0.2	0.4
↑	↑	↑	↑
这	本	书	很

②拼接所有的条件词的向量，形成一个向量。

或者加和，取平均

这 本 书 很

$\rightarrow (0.2, 0.1, 0.1, 0.3, 0.4, 0.2, 0.5, 0.4)^T$

③隐藏层：线性映射+非线性变换。

$$\boldsymbol{h} = \begin{pmatrix}0.36\\0.41\end{pmatrix} \Longrightarrow \boldsymbol{L_T^{\mathrm{T}}} \times \boldsymbol{h} = \begin{pmatrix}0.2\ 0.1\\0.1\ 0.3\\0.4\ 0.2\\0.5\ 0.4\\0.3\ 0.2\\\cdots\ \cdots\end{pmatrix} \times \begin{pmatrix}0.36\\0.41\end{pmatrix} = \begin{pmatrix}0.113\\0.159\\0.226\\0.344\\0.190\\\cdots\cdots\end{pmatrix}$$

乏味

Softmax(•)

前馈神经网络语言模型

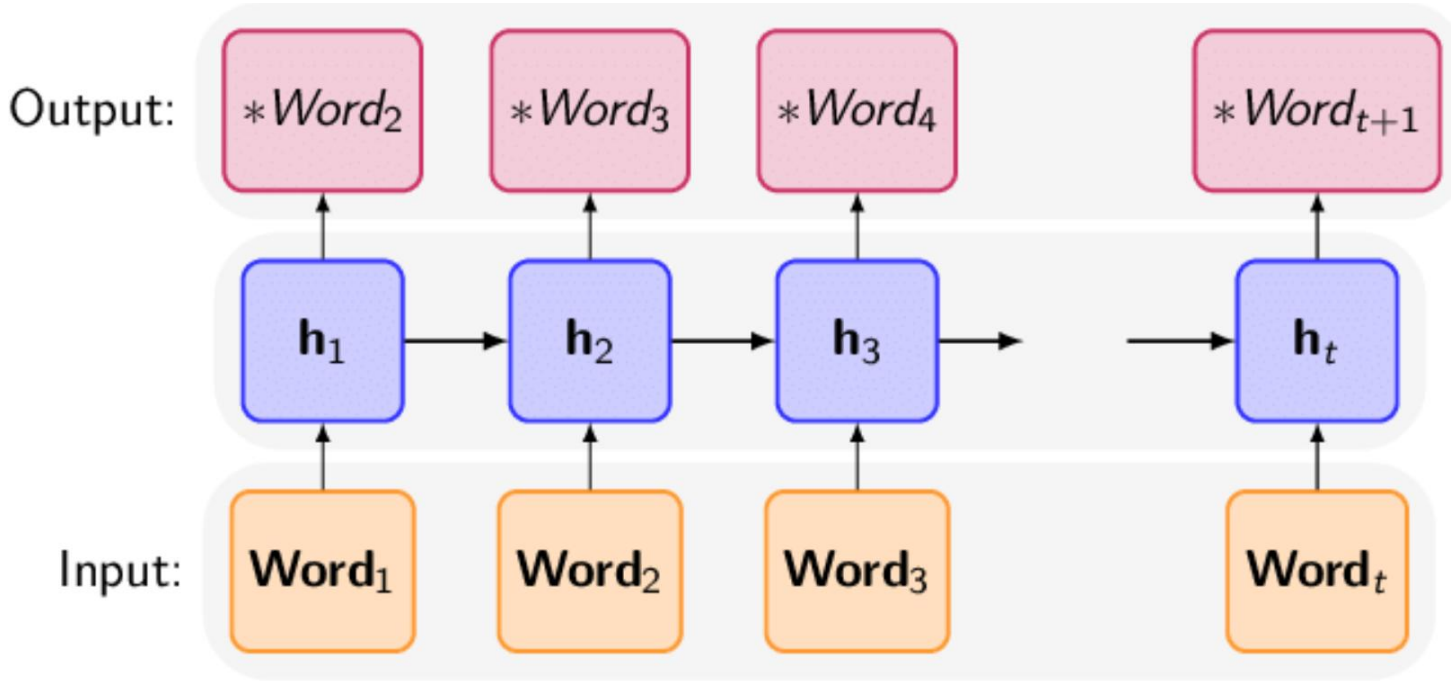
- 问题分析
  - **仅对小窗口的历史信息进行建模**
  - n-gram语言模型仅考虑前面$n-1$个词的历史信息
  - 能否对所有的历史信息进行建模?

本章内容

- 传统语言模型
  - n元文法
  - 参数估计
  - 数据平滑方法
- 神经语言模型
  - 问题的提出
  - 前馈神经网络语言模型
  - 循环神经网络语言模型

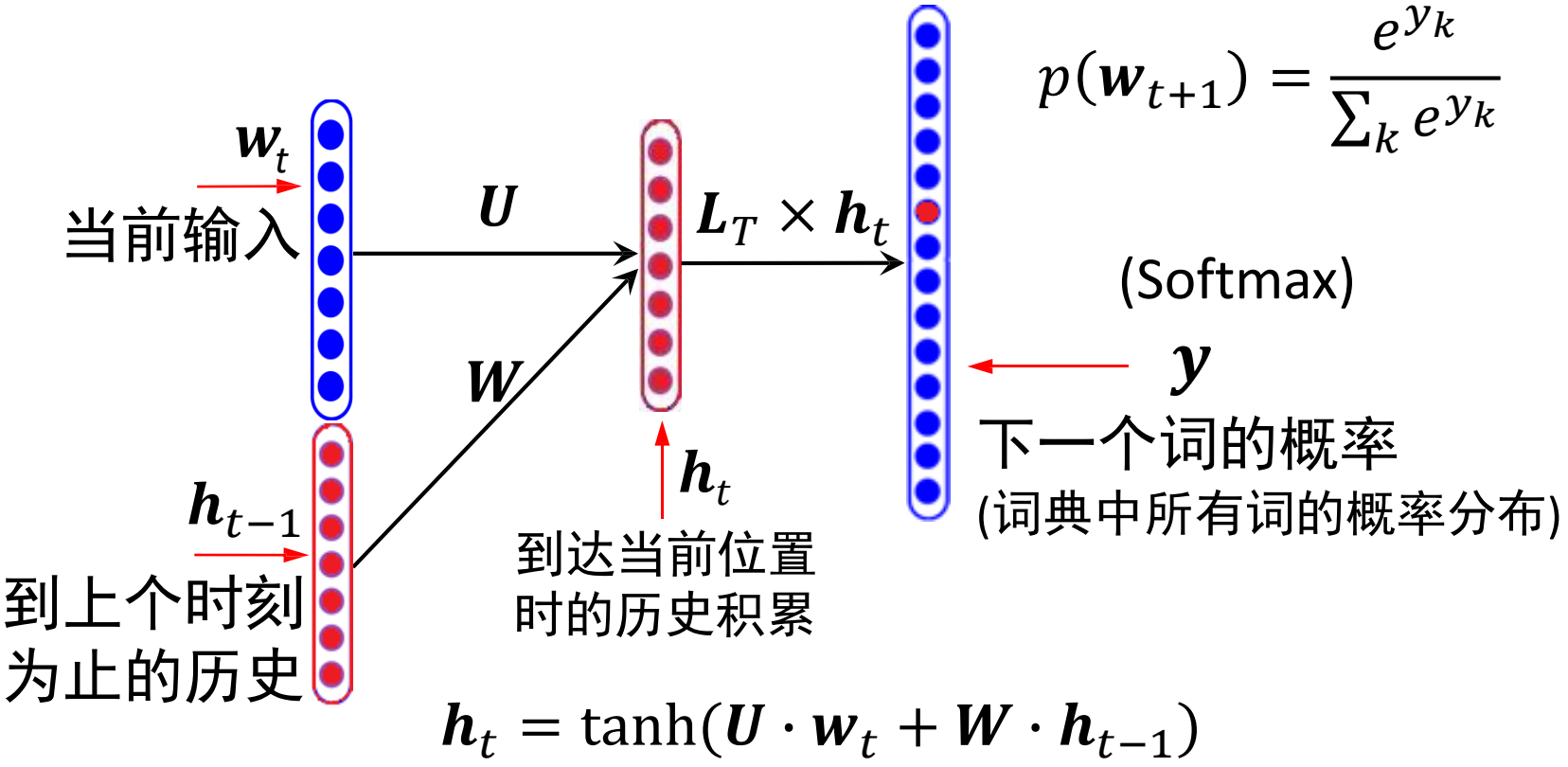
循环神经网络语言模型

- 循环神经网络语言模型结构



循环神经网络语言模型

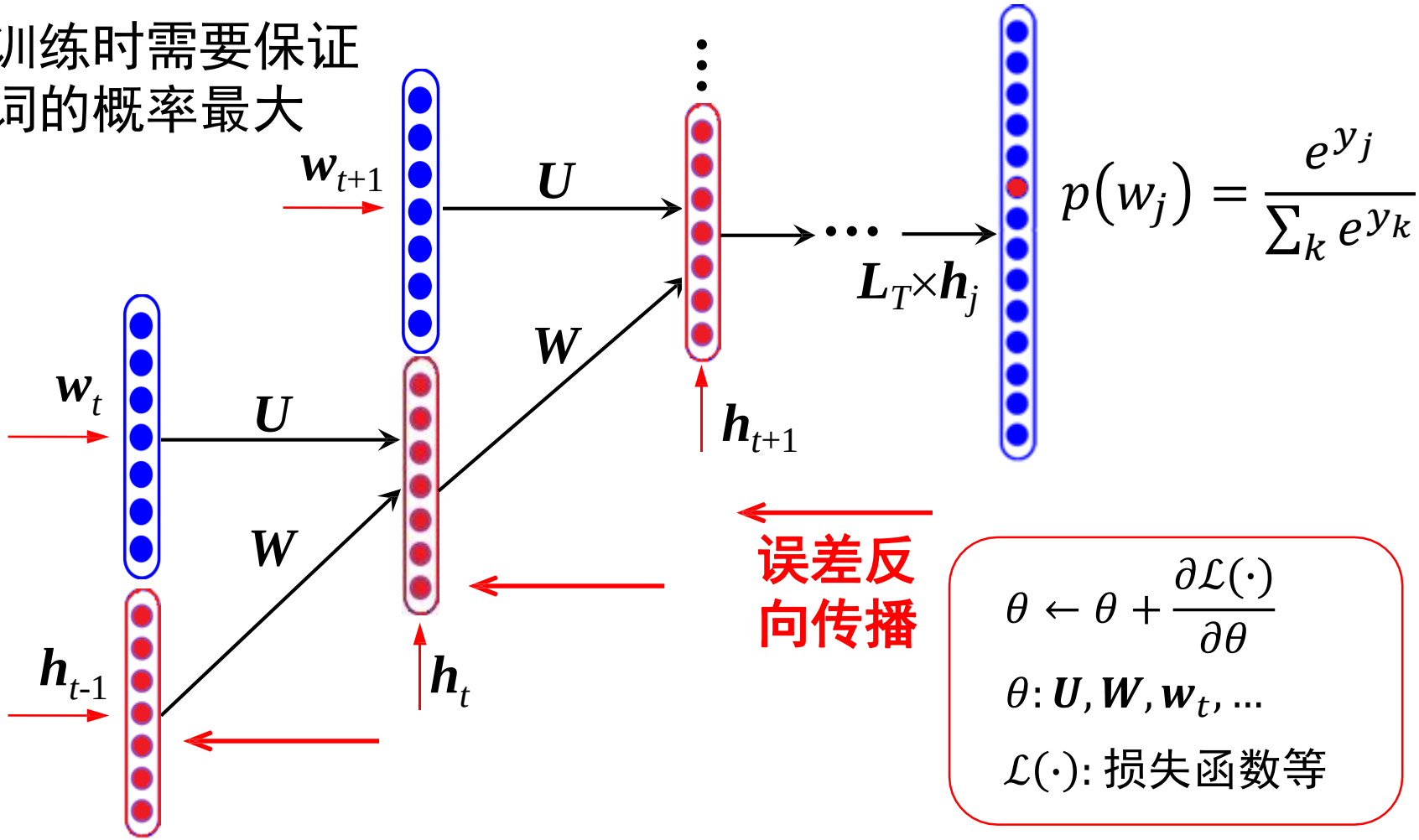
- 循环神经网络
  - 输入:从开始到$t-1$时刻的历史$\boldsymbol{h}_{t-1}$; 当前位置t的词向量$\boldsymbol{w}_t$
  - 输出: 到t位置时的历史积累$\boldsymbol{h}_t$及其该位置上词的概率



循环神经网络语言模型

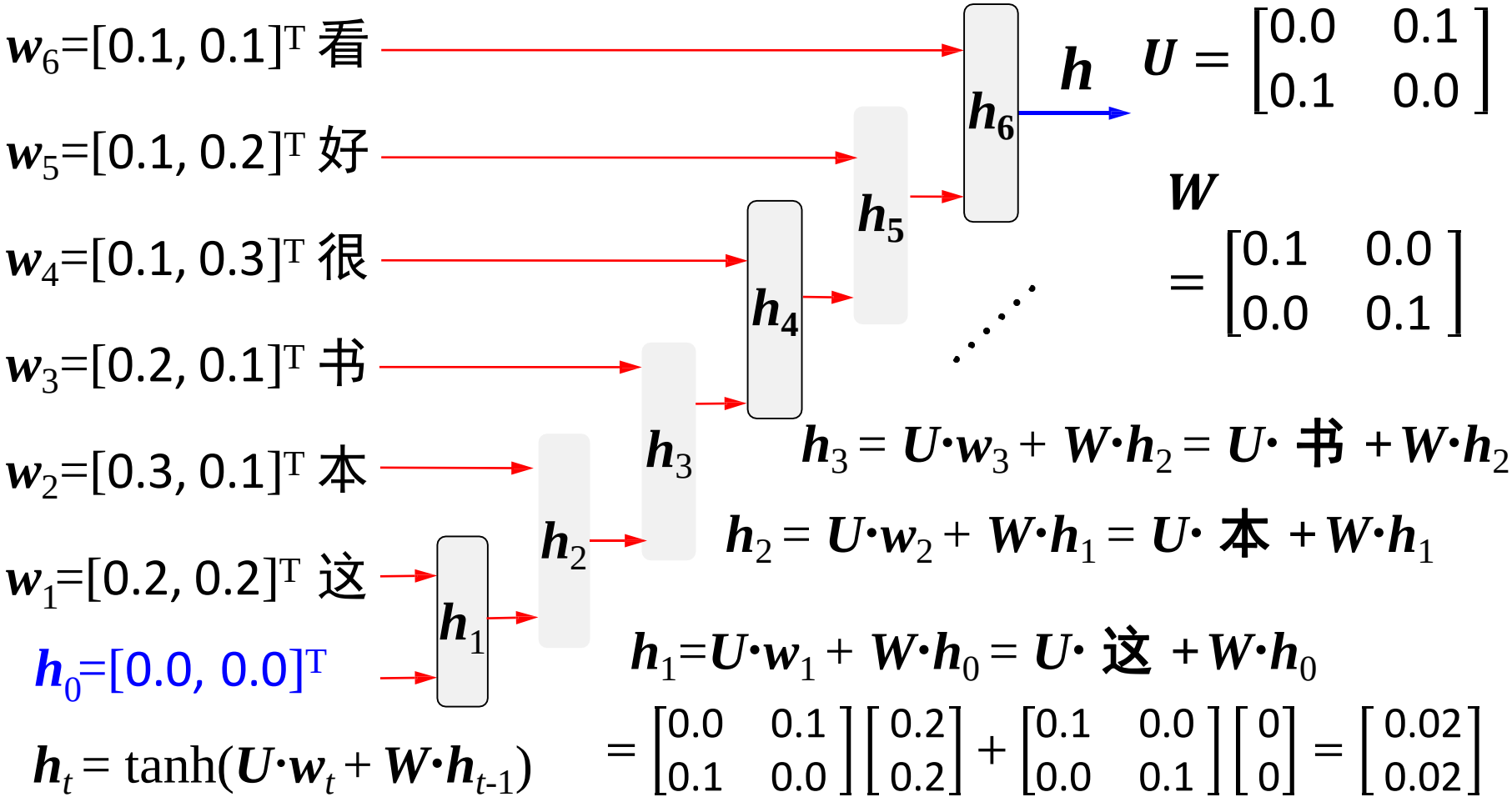
- 循环神经网络

模型训练时需要保证
当前词的概率最大



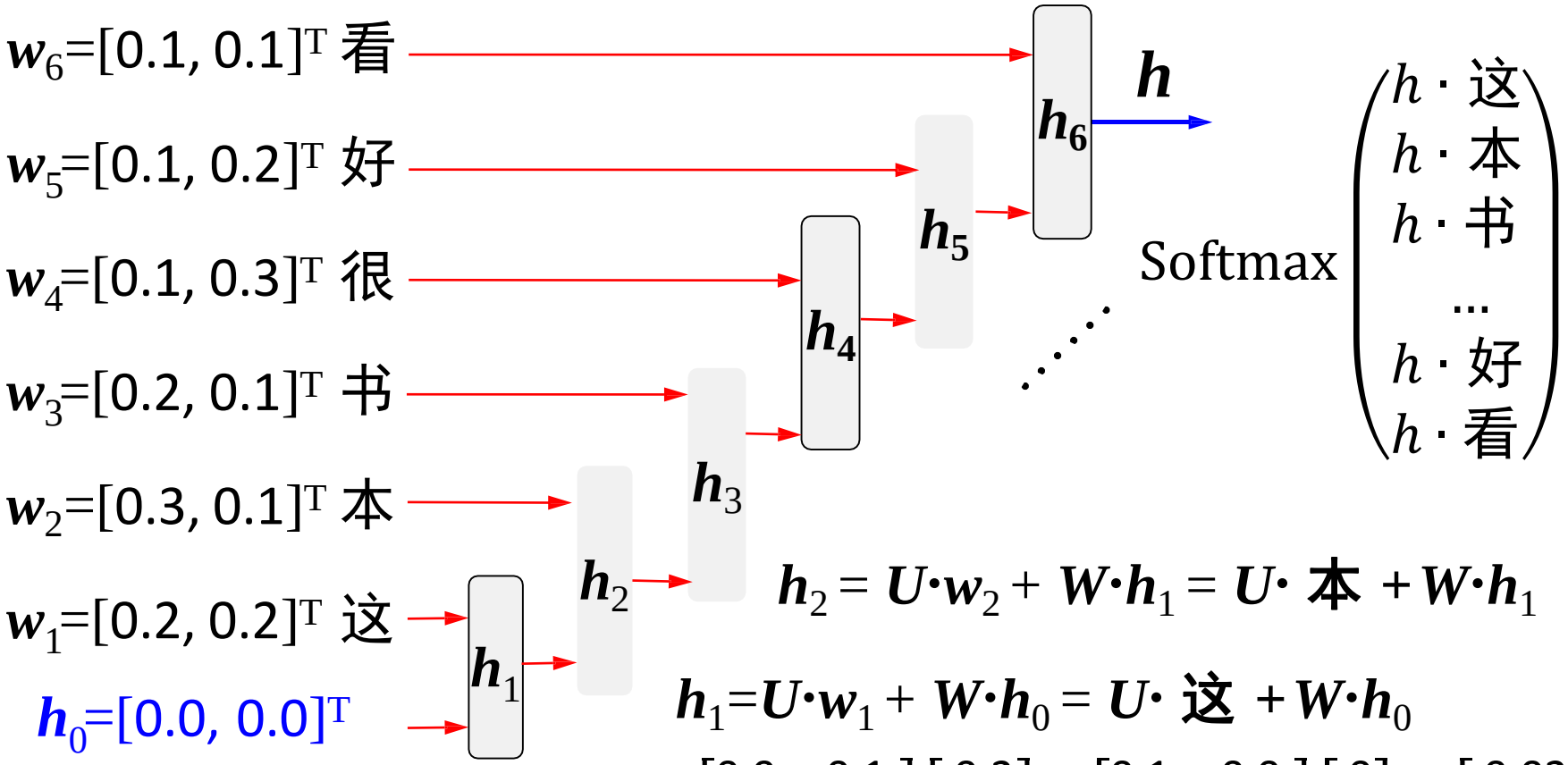
循环神经网络语言模型

- 循环神经网络



循环神经网络语言模型

- 循环神经网络



循环神经网络语言模型

- 循环神经网络
  - 问题分析
    - **梯度消失或爆炸：**参数$\boldsymbol{W}$经过多次传递后有可能导致梯度消失(小于1)或者爆炸(大于1)。
  - 是否能够通过某种策略选择性地保留或者遗忘某些信息？
    - **长短时记忆网络LSTM (Long-Short Term Memory)**
    - **在后续进行详细介绍**

欢迎提问！