



南京大學
NANJING UNIVERSITY



自然语言处理

神经网络和语言模型

吴震

wuz@nju.edu.cn

2023年6月

- 神经网络基础
- 循环神经网络和语言模型
- 高级循环神经网络



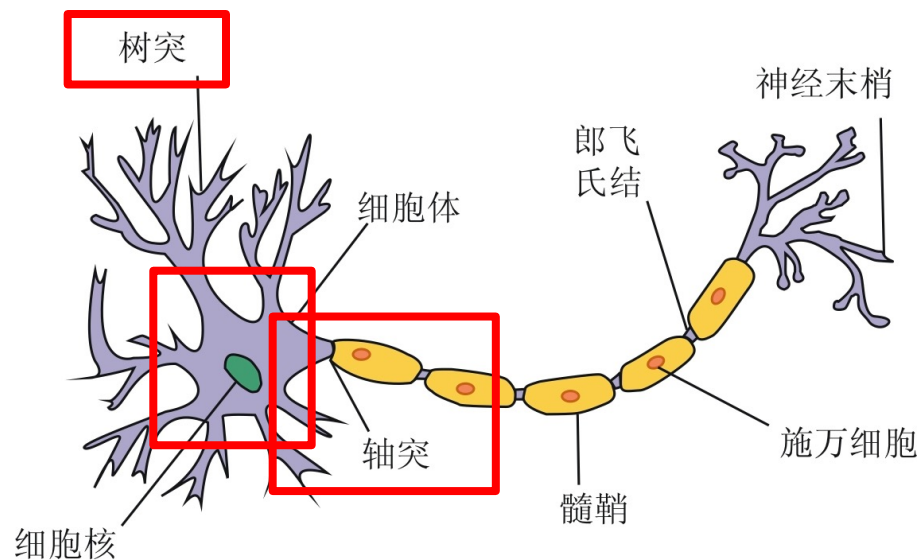
01



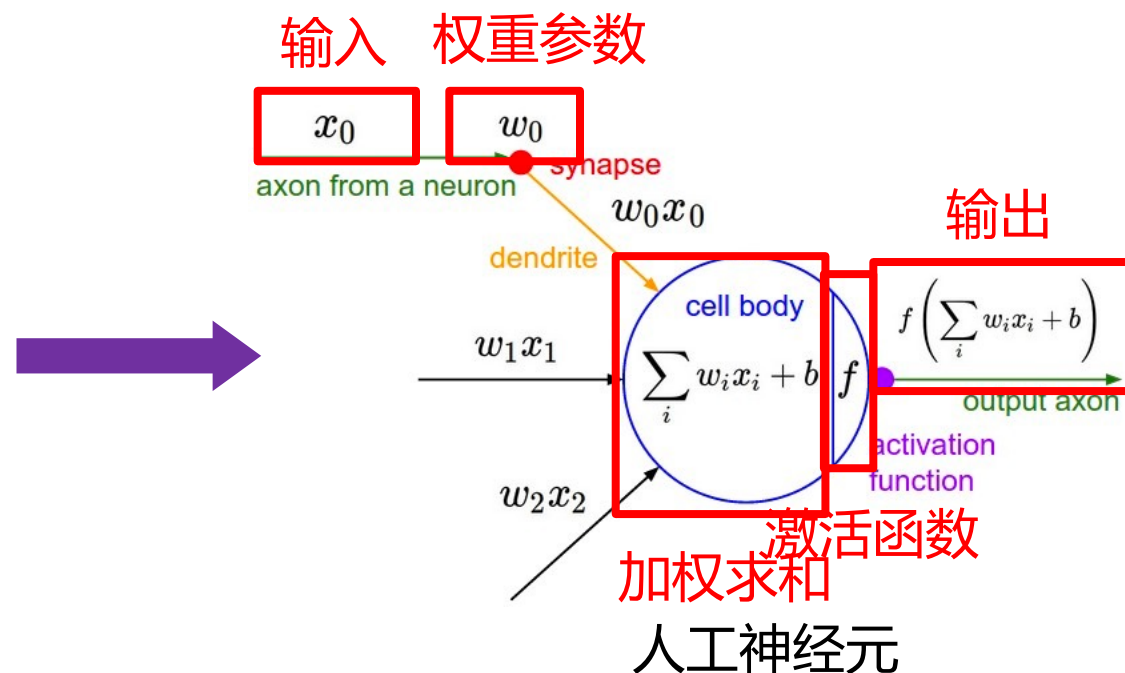
神经网络基础

BASICS OF NEURAL NETWORK

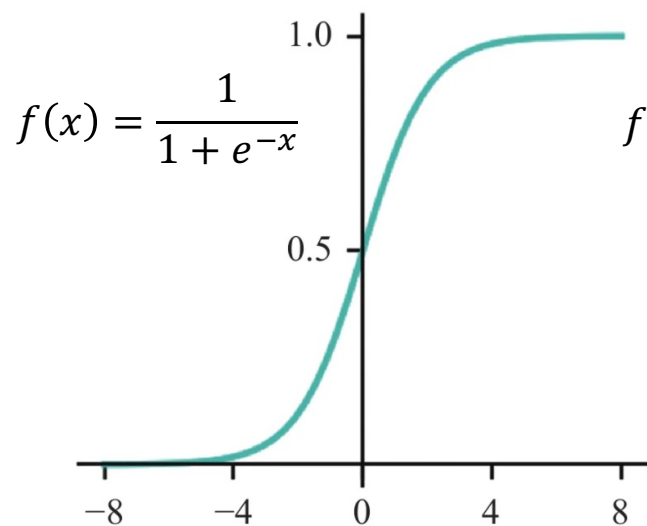
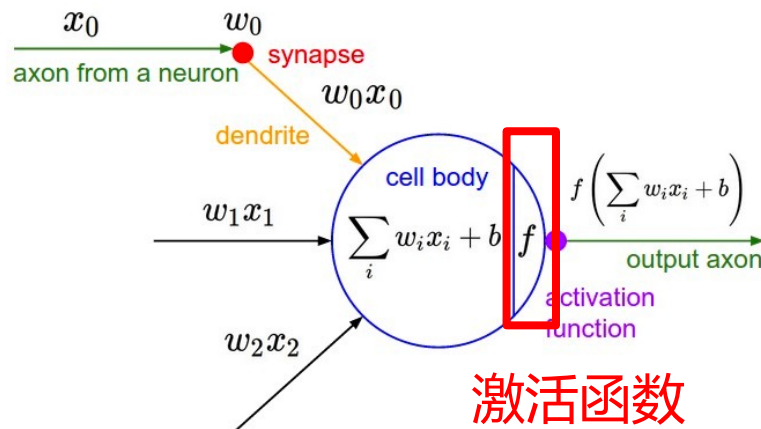
- 人工神经网络 (Artificial Neural Network, ANN) 从信息处理角度对人脑神经元网络进行抽象，建立某种算法数学模型，从而达到处理信息的目的。
 - 神经元之间进行连接，组成网络
 - 网络通常是某种算法或者函数的逼近，也可能是一种逻辑策略的表达



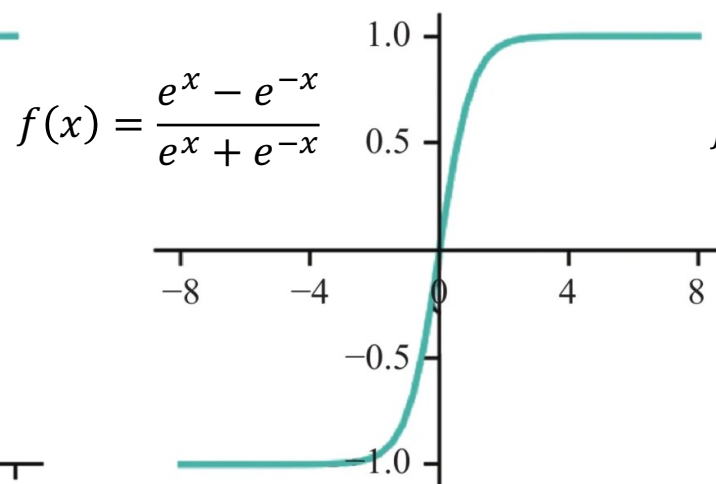
大脑神经元



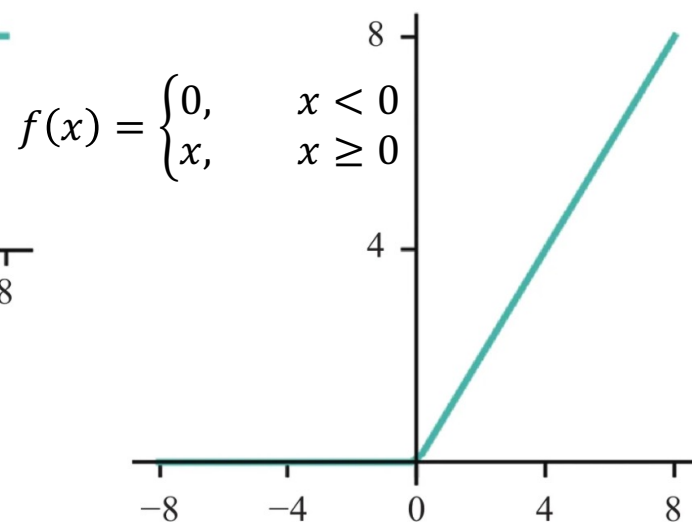
- 为什么需要非线性激活函数？
 - Sigmoid、Tanh、ReLU...



(a) Sigmoid

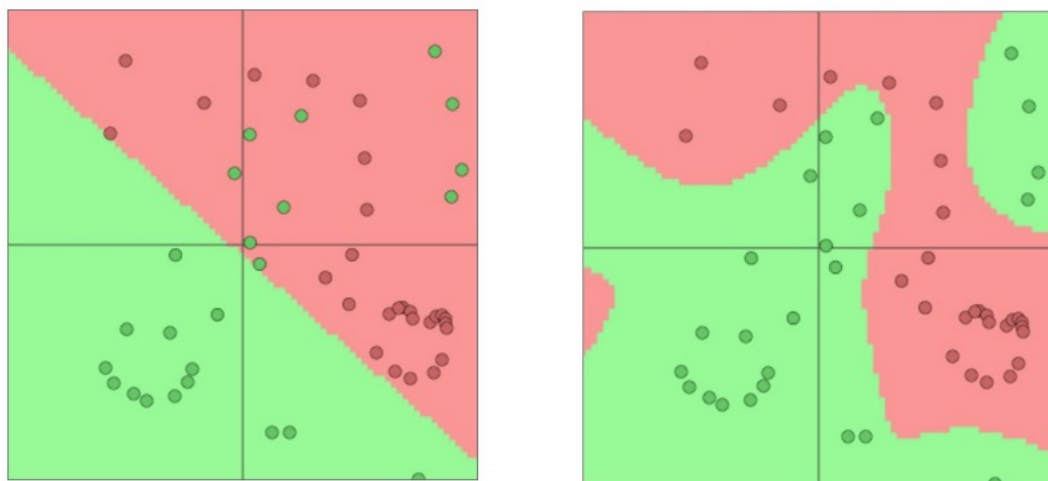


(b) tanh

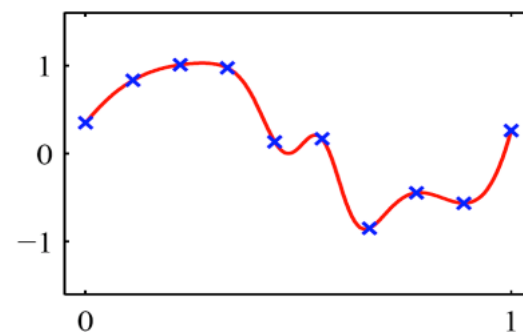
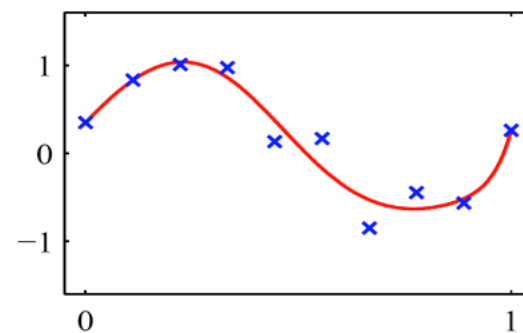
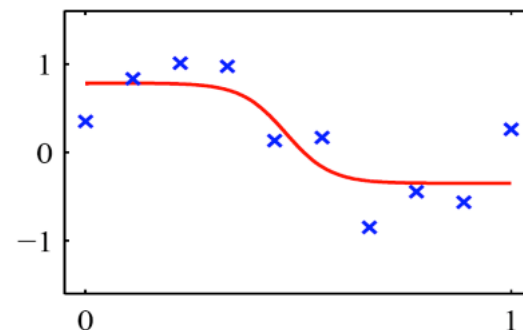


(c) ReLU

- 为什么需要非线性激活函数？
 - 使神经网络具有非线性拟合能力

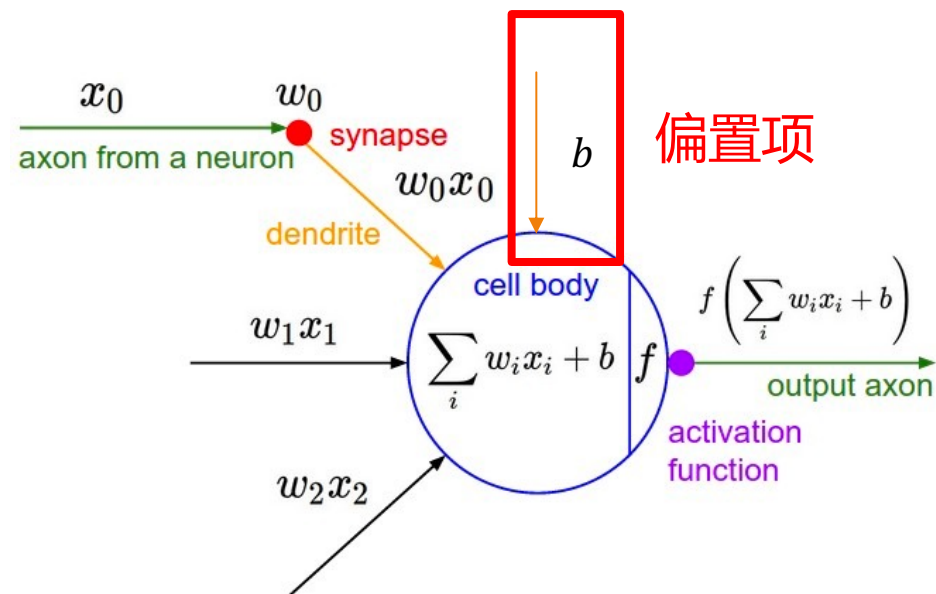
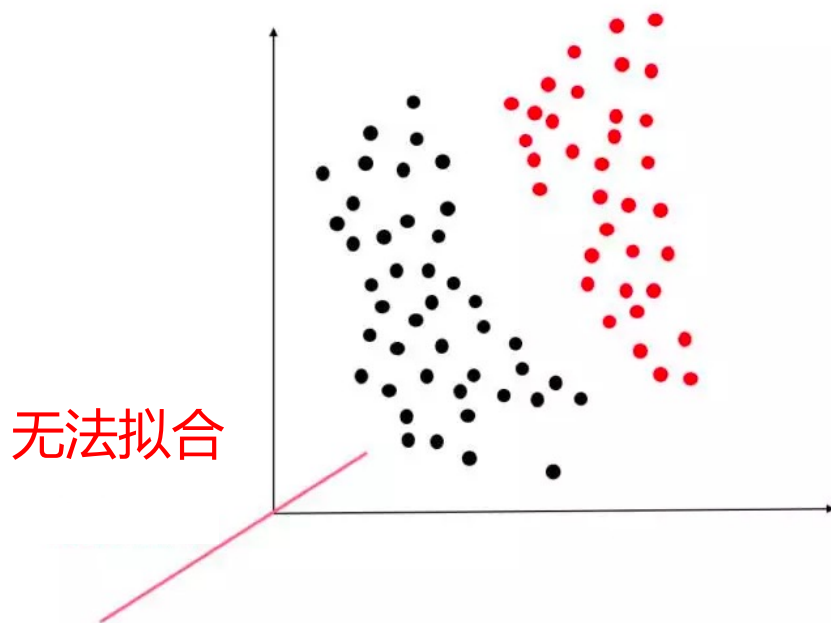


分类

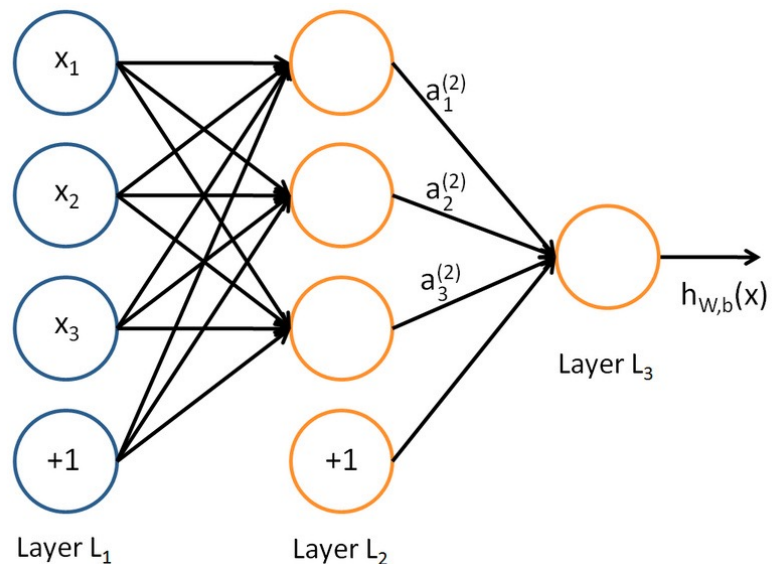


回归

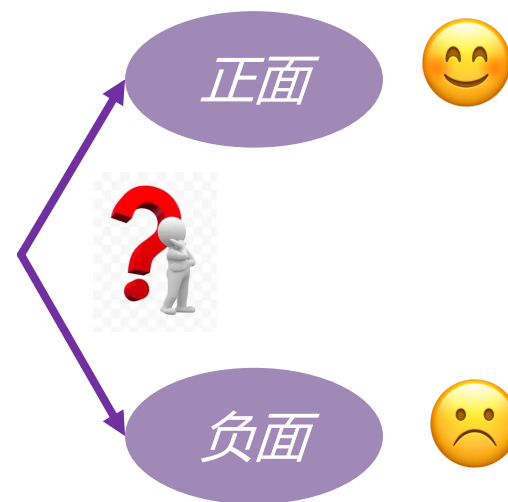
- 为什么需要偏置项？
 - 权重参数 W ：进行缩放拟合
 - 偏置项 b ：进行平移拟合



- 文本分类 (Text Classification) : 将文本打上预定义的标签



I like this book



如何利用神经网络构建文本分类模型？

神经网络文本分类模型



损失函数

$$J(\theta) = -y \log \hat{y} + (1 - y) \log(1 - \hat{y})$$

$$\hat{y} = \frac{1}{1 + e^{-s}}$$

$$s = u^T h$$

$$h = f(Wx + b)$$

$$x = \sum_{i=1}^n v_i / n$$

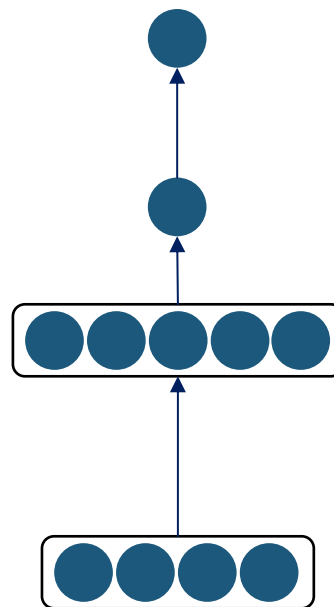
平均池化

词向量

$$v_1, \dots, v_n$$

输入文本

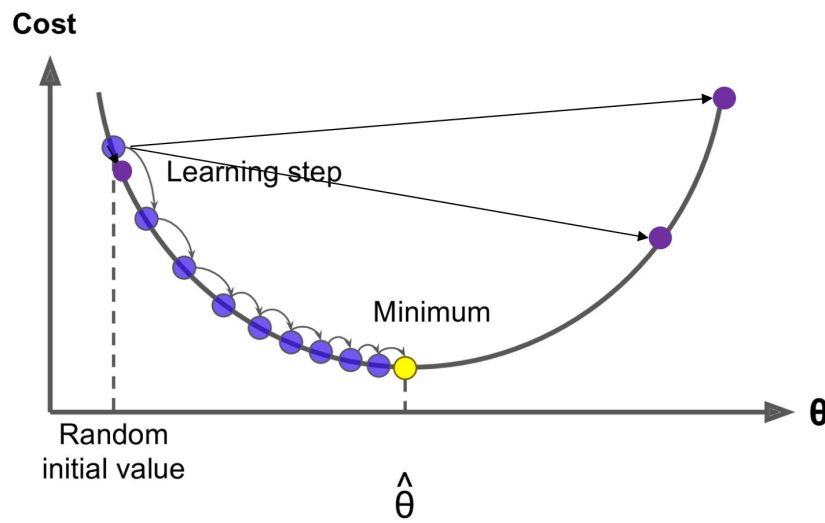
$$w_1, \dots, w_n$$



如何训练模型？

- 梯度下降法

$$\theta^{new} = \theta^{old} - \alpha \nabla_{\theta} J(\theta)$$



如何计算梯度?

- 链式法则

$$\frac{\partial J}{\partial W} = \frac{\partial J}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial s} \frac{\partial s}{\partial h} \frac{\partial h}{\partial W}$$

损失函数

$$J(\theta) = -y \log \hat{y} + (1 - y) \log(1 - \hat{y})$$

$$\hat{y} = \frac{1}{1 + e^{-s}}$$

$$s = u^T h$$

$$h = f(Wx + b)$$

平均池化

$$x = \sum_{i=1}^n v_i / n$$

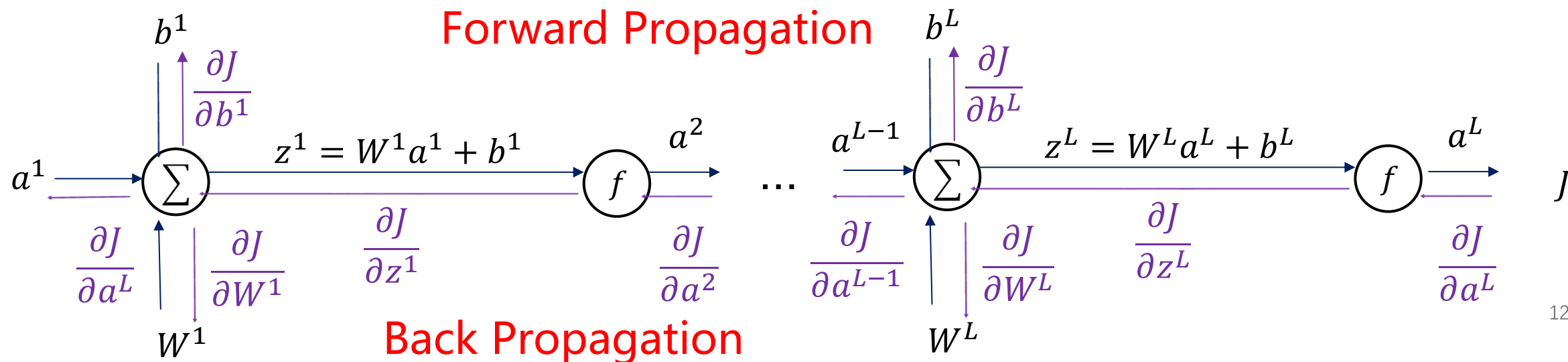
词向量

$$v_1, \dots, v_n$$

输入文本

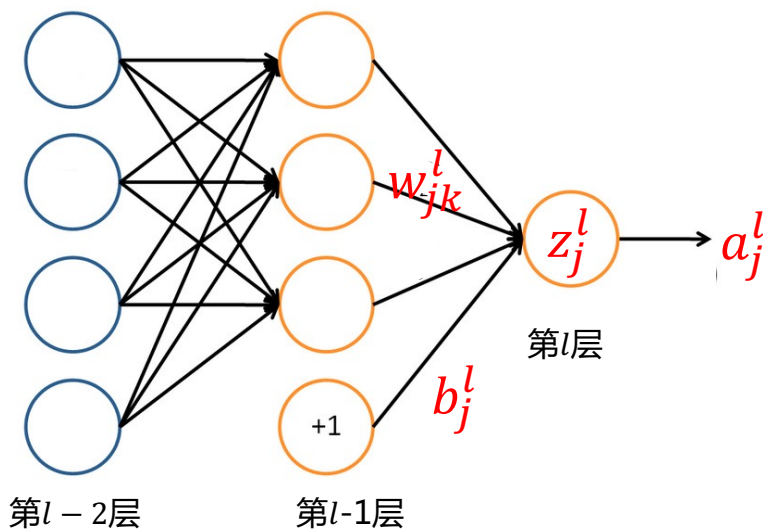
$$w_1, \dots, w_n$$

- 目前训练人工神经网络最常用且有效的算法
 - 前向传播 (Forward Propagation) : 将训练集数据输入到网络的输入层, 经过隐藏层, 最后达到输出层并输出结果;
 - 反向传播 (Back Propagation) : 计算预测值与实际值之间的误差, 并将该误差 (梯度) 从输出层向隐藏层反向传播, 直至传播到输入层, 并更新参数;
 - 迭代上述过程, 直至网络收敛。



● 变量定义

- w_{jk}^l : 第 $l-1$ 层的第 k 个神经元连接到第 l 层的第 j 个神经元的权重
- b_j^l : 第 l 层第 j 个神经元的偏置项
- z_j^l : 第 l 层第 j 个神经元的输入, 即 $z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$
- a_j^l : 第 l 层第 j 个神经元的输出, 即 $a_j^l = f(z_j^l) = f(\sum_k w_{jk}^l a_k^{l-1} + b_j^l)$



第 l 层第 j 个神经元产生的错误（实际值和预测值之间的误差）定义如下：

$$\delta_j^l \equiv \frac{\partial J}{\partial z_j^l}$$

- 计算最后一层神经网络产生的误差 δ^L ：

$$\delta_j^L = \frac{\partial J}{\partial z_j^L} = \frac{\partial J}{\partial a_j^L} \frac{\partial a_j^L}{\partial z_j^L} = \frac{\partial J}{\partial a_j^L} f'(z_j^L)$$

$$a_j^l = f(z_j^l)$$

$$\delta^L = \nabla_{a^L} J \odot f'(z^L)$$

- 由后往前，计算每一层神经网络产生的误差 δ^l ：

$$\begin{aligned}\delta_j^l &= \frac{\partial J}{\partial z_j^l} = \sum_k \frac{\partial J}{\partial z_k^{l+1}} \cdot \frac{\partial z_k^{l+1}}{\partial a_j^l} \cdot \frac{\partial a_j^l}{\partial z_j^l} \\ &= \sum_k \delta_k^{l+1} \cdot \frac{\partial (w_{kj}^{l+1} a_j^l + b_k^{l+1})}{\partial a_j^l} \cdot f'(z_j^l) \\ &= \sum_k \delta_k^{l+1} \cdot w_{kj}^{l+1} \cdot f'(z_j^l)\end{aligned}$$

$$\begin{aligned}z_j^l &= \sum_k w_{jk}^l a_k^{l-1} + b_j^l \\ a_j^l &= f(z_j^l)\end{aligned}$$

$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot f'(z^l)$$

- 计算权重的梯度 $\frac{\partial J}{\partial w_{jk}^l}$:

$$\begin{aligned}\frac{\partial J}{\partial w_{jk}^l} &= \frac{\partial J}{\partial z_j^l} \frac{\partial z_j^l}{\partial w_{jk}^l} \\ &= \delta_j^l \frac{\partial f(w_{jk}^l a_k^{l-1} + b_j^l)}{\partial w_{jk}^l} \\ &= a_k^{l-1} \delta_j^l\end{aligned}$$

$$z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$$

- 计算偏置项的梯度 $\frac{\partial J}{\partial b_j^l}$:

$$\begin{aligned}\frac{\partial J}{\partial b_j^l} &= \frac{\partial J}{\partial z_j^l} \frac{\partial z_j^l}{\partial b_j^l} \\ &= \delta_j^l \frac{\partial f(w_{jk}^l a_k^{l-1} + b_j^l)}{\partial b_j^l} \\ &= \delta_j^l\end{aligned}$$

$$z_j^l = \sum_k w_{jk}^l a_k^{l-1} + b_j^l$$

利用反向传播算法训练网络

- 前向传播：
$$z^l = w^l a^{l-1} + b^l$$
$$a^l = f(z^l)$$
- 计算输出层的误差：
$$\delta^L = \nabla_{a^L} J \odot f'(z^L)$$
- 反向传播误差：
$$\delta^l = ((w^{l+1})^T \delta^{l+1}) \odot f'(z^l)$$
- 梯度下降，更新参数：
$$w^l \leftarrow w^l - \alpha \cdot \delta^l (a^{l-1})^T$$
$$b^l \leftarrow b^l - \alpha \cdot \delta^l$$



02

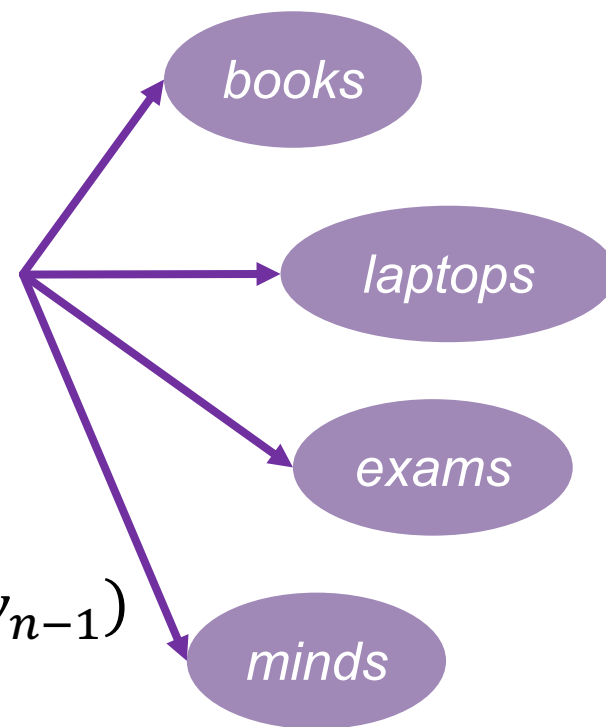


循环神经网络和语言模型

RECURRENT NEURAL NETWORK AND LANGUAGE MODEL

- 语言模型 (Language Model , LM) : 衡量一句话出现概率的模型

The students opened their _____



$$P(s) = P(w_1, \dots, w_n)$$

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

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

- 马尔可夫假设：当前词出现的概率只和它前面的k个词相关

$$P(w_i | w_1, \dots, w_{i-1}) = P(w_i | w_{i-k}, \dots, w_{i-1})$$

- 缺点：
 - N-gram模型的稀疏性问题
 - N-gram模型的存储问题

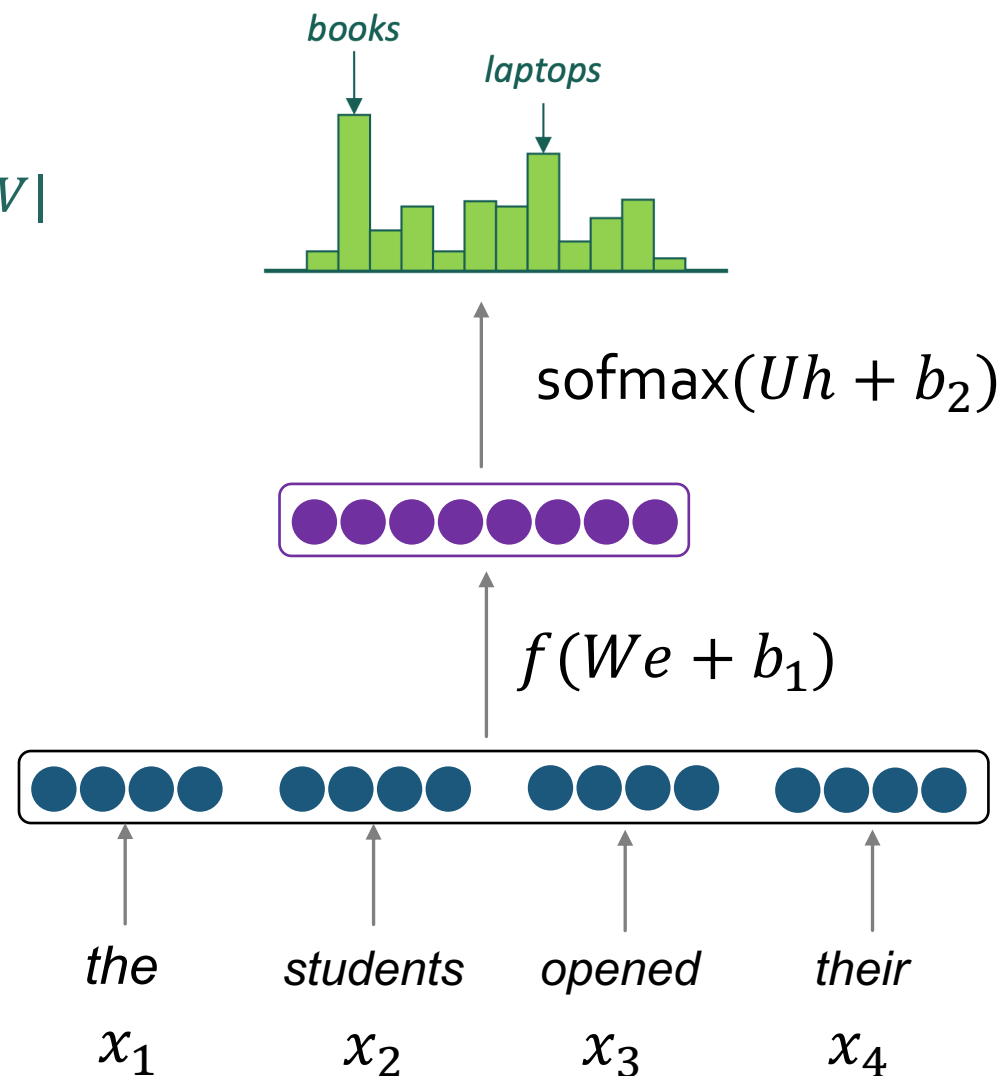
基于4-GRAM的神经网络语言模型

预测概率： $\hat{y} = \text{softmax}(Uh + b_2) \in \mathbb{R}^{|V|}$

隐层表示： $h = f(We + b_1)$

拼接词向量： $e = [e_1, e_2, e_3, e_4]$

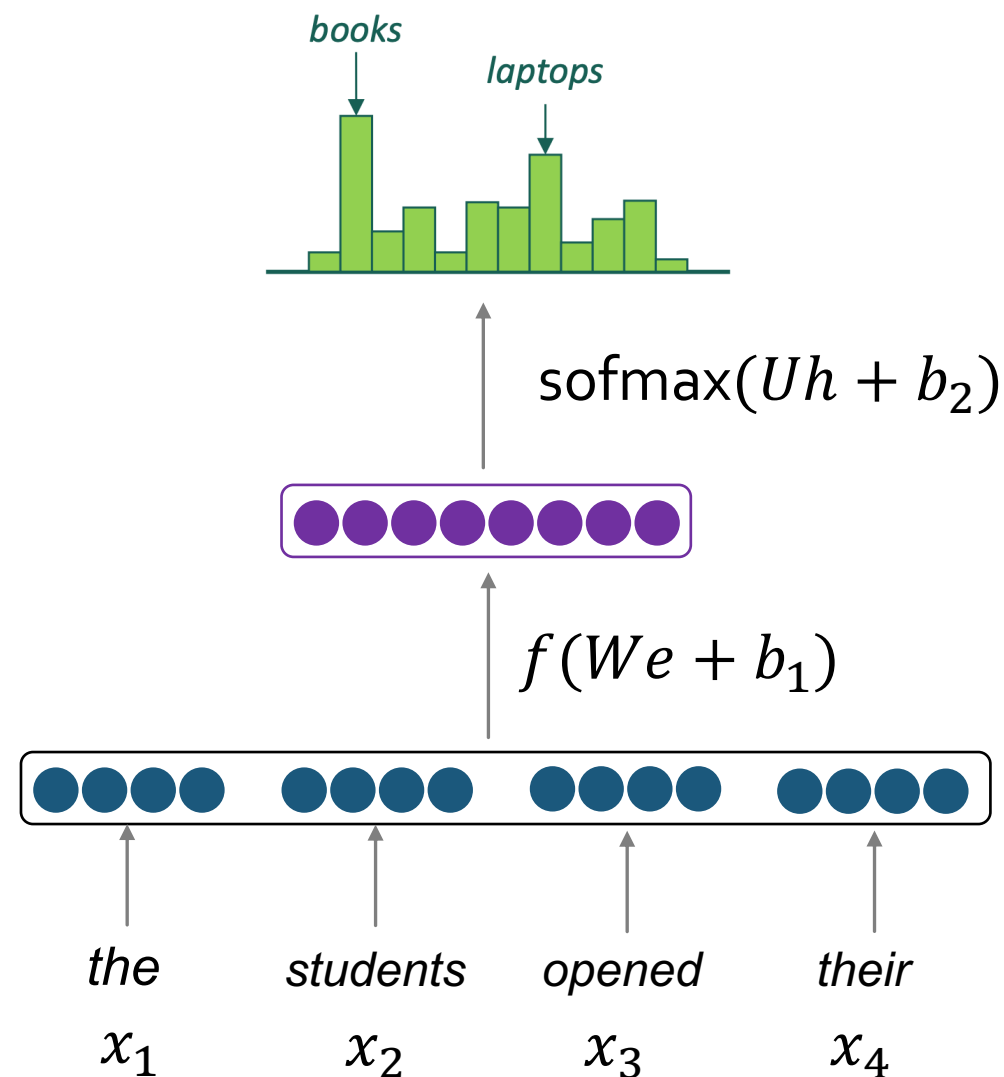
输入单词序列： x_1, x_2, x_3, x_4



基于N-GRAM的神经网络语言模型

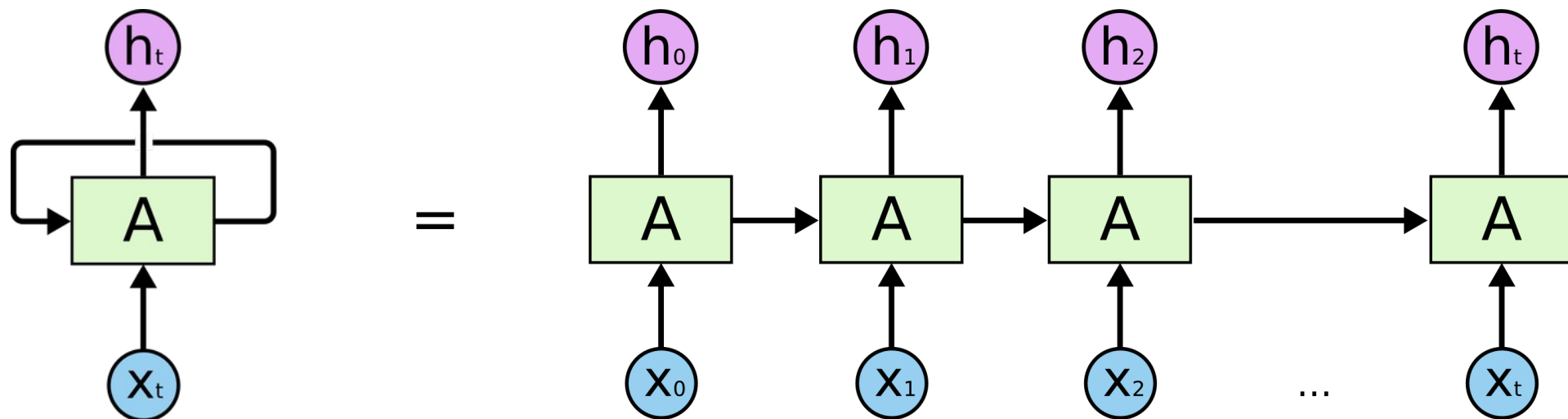
- 优点
 - 不会有稀疏性问题
 - 不需要存储所有的n-grams
- 不足
 - 视野有限，无法建模长距离语义
 - 窗口越大，参数规模越大

能否构建处理任意长度的神经网络模型？



- 循环神经网络 (Recurrent Neural Network , RNN)

- 重复使用隐层参数
- 可处理任意序列长度



$$h_t = f(W_h h_{t-1} + W_x x_t + b)$$

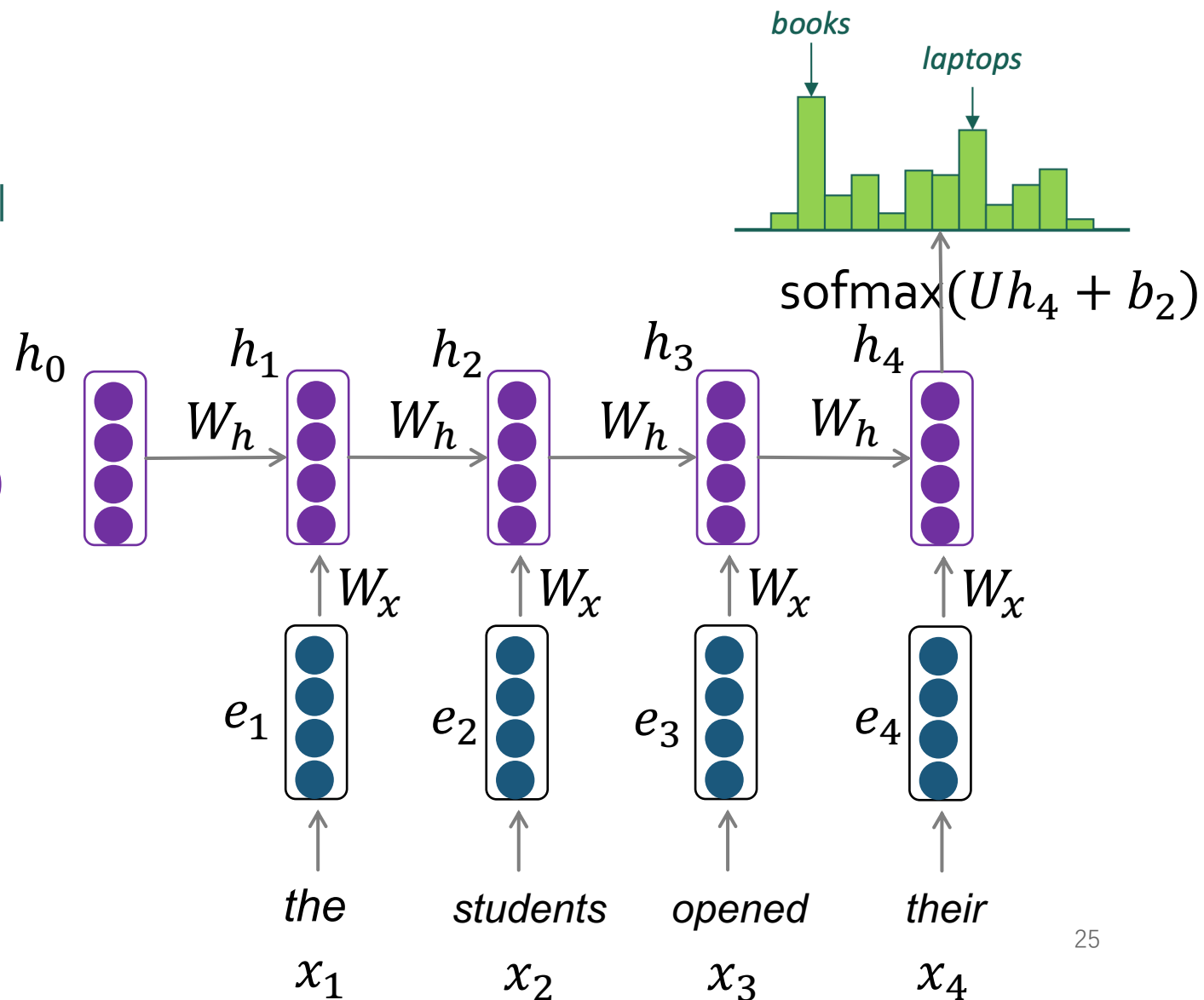
循环神经网络语言模型

预测概率： $\hat{y} = \text{softmax}(Uh + b_2) \in \mathbb{R}^{|V|}$

隐层表示： $h_t = f(W_h h_{t-1} + W_x e_t + b)$

输入词向量序列： e_1, e_2, e_3, e_4

输入单词序列： x_1, x_2, x_3, x_4

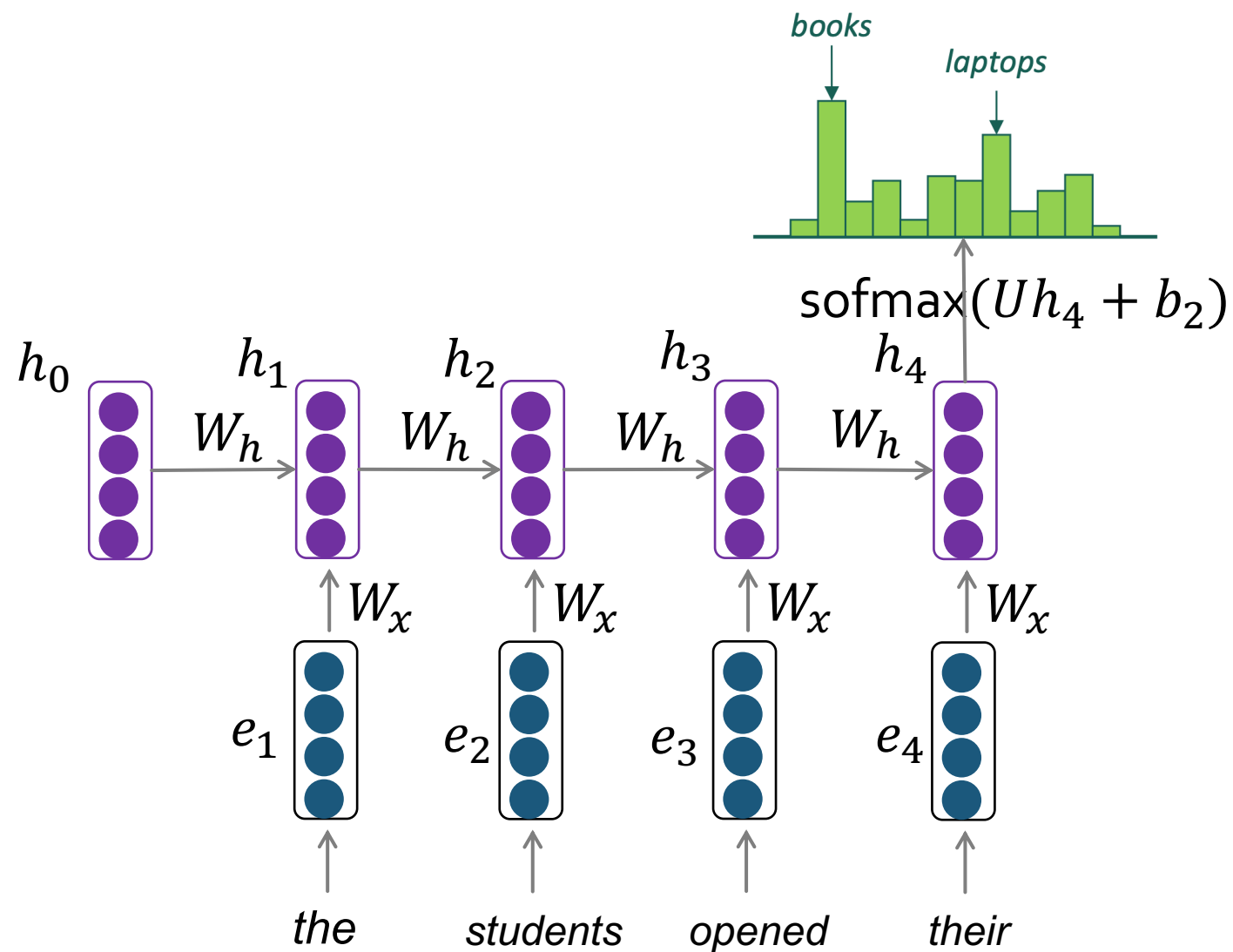


- 优点

- 能够处理任意长度序列
- 能够使用历史信息
- 模型参数量不随序列长度增加

- 不足

- 逐步计算，速度较慢
- 长期依赖问题



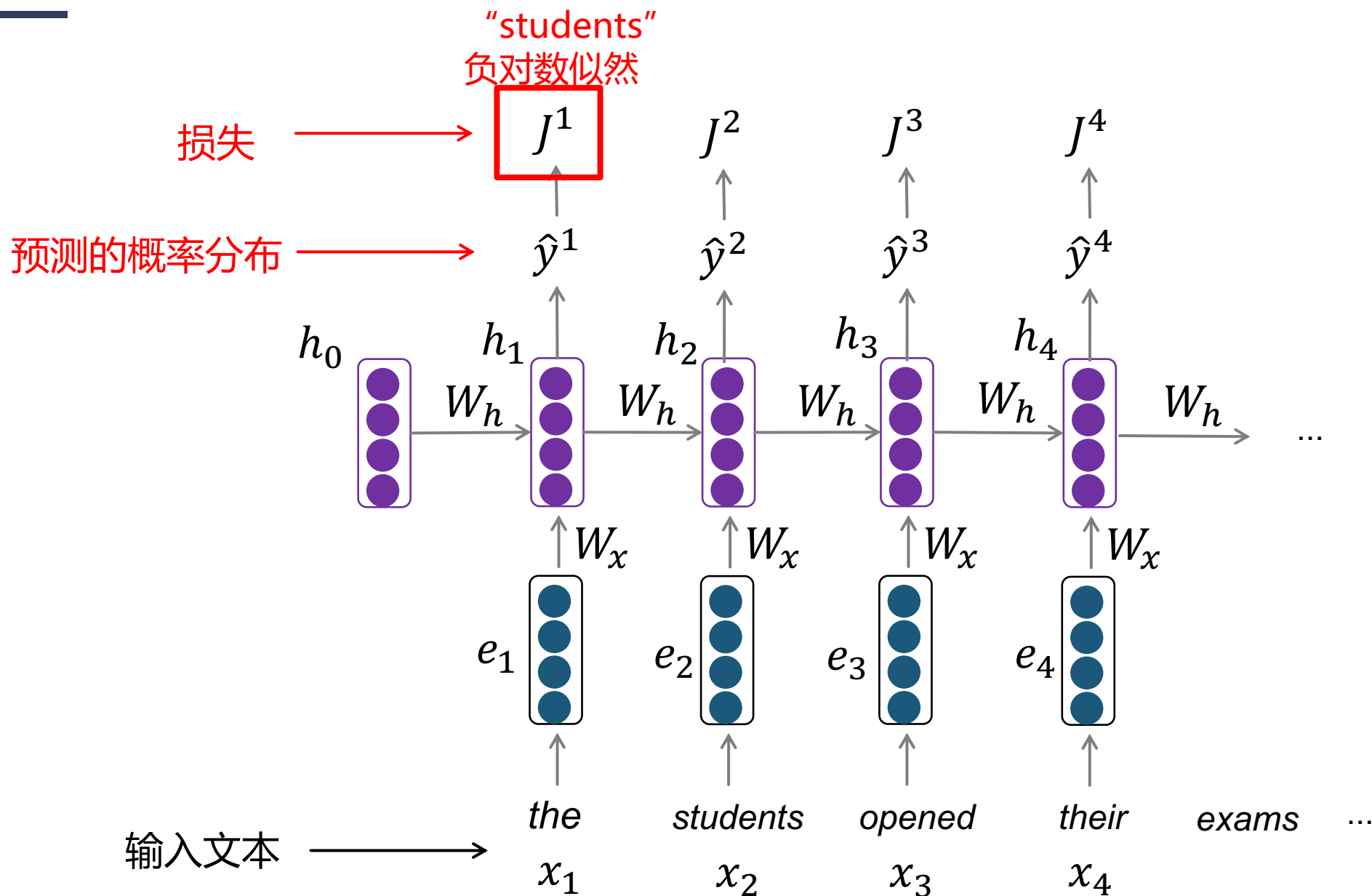
- 给定长度为 T 的输入文本： x_1, \dots, x_T ；
- 将文本输入到RNN-LM，计算每一步预测的单词分布 \hat{y}^t ；
- 计算每一步预测单词的概率分布 \hat{y}^t 和真实单词 y^t （one-hot向量）之间的交叉熵：

$$J^t(\theta) = \text{CE}(y^t, \hat{y}^t) = - \sum_{w \in V} y_w^t \log \hat{y}_w^t = - \log \hat{y}_{x_{t+1}}^t$$

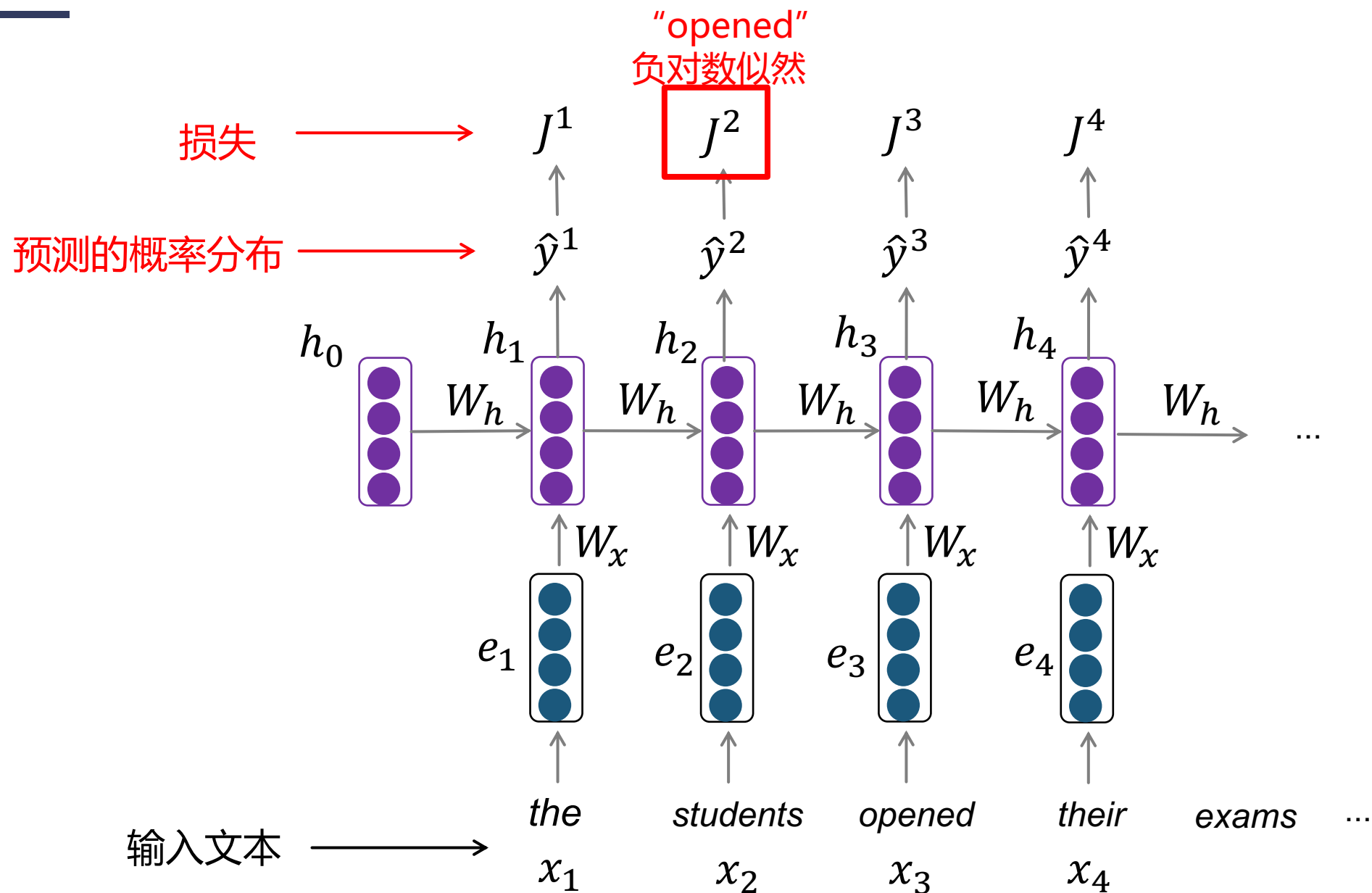
- 模型在输入文本上的训练损失为：

$$J(\theta) = \frac{1}{T} \sum_{t=1}^T J^t(\theta) = \frac{1}{T} \sum_{t=1}^T - \log \hat{y}_{x_{t+1}}^t$$

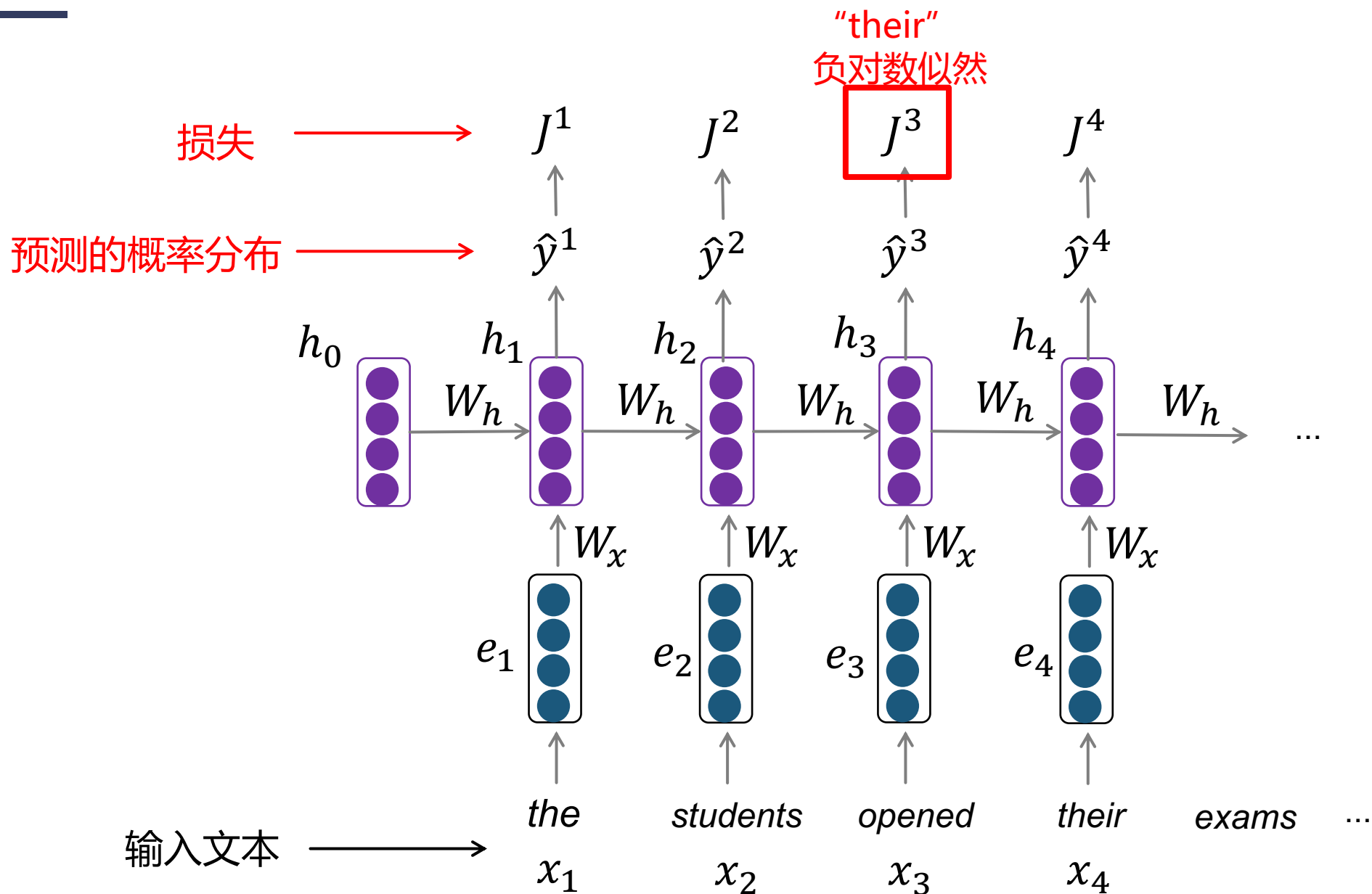
RNN-LM模型训练



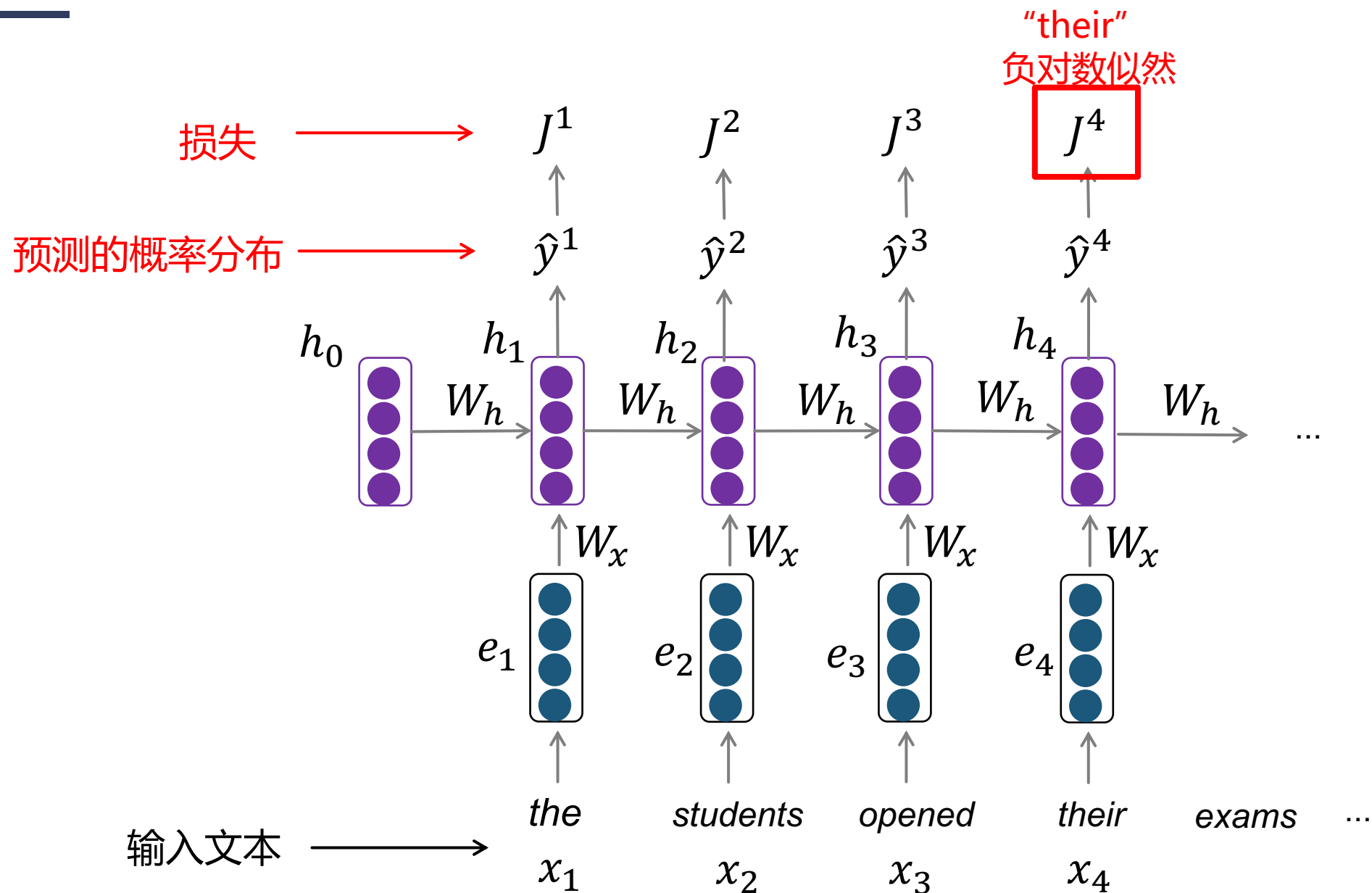
RNN-LM模型训练



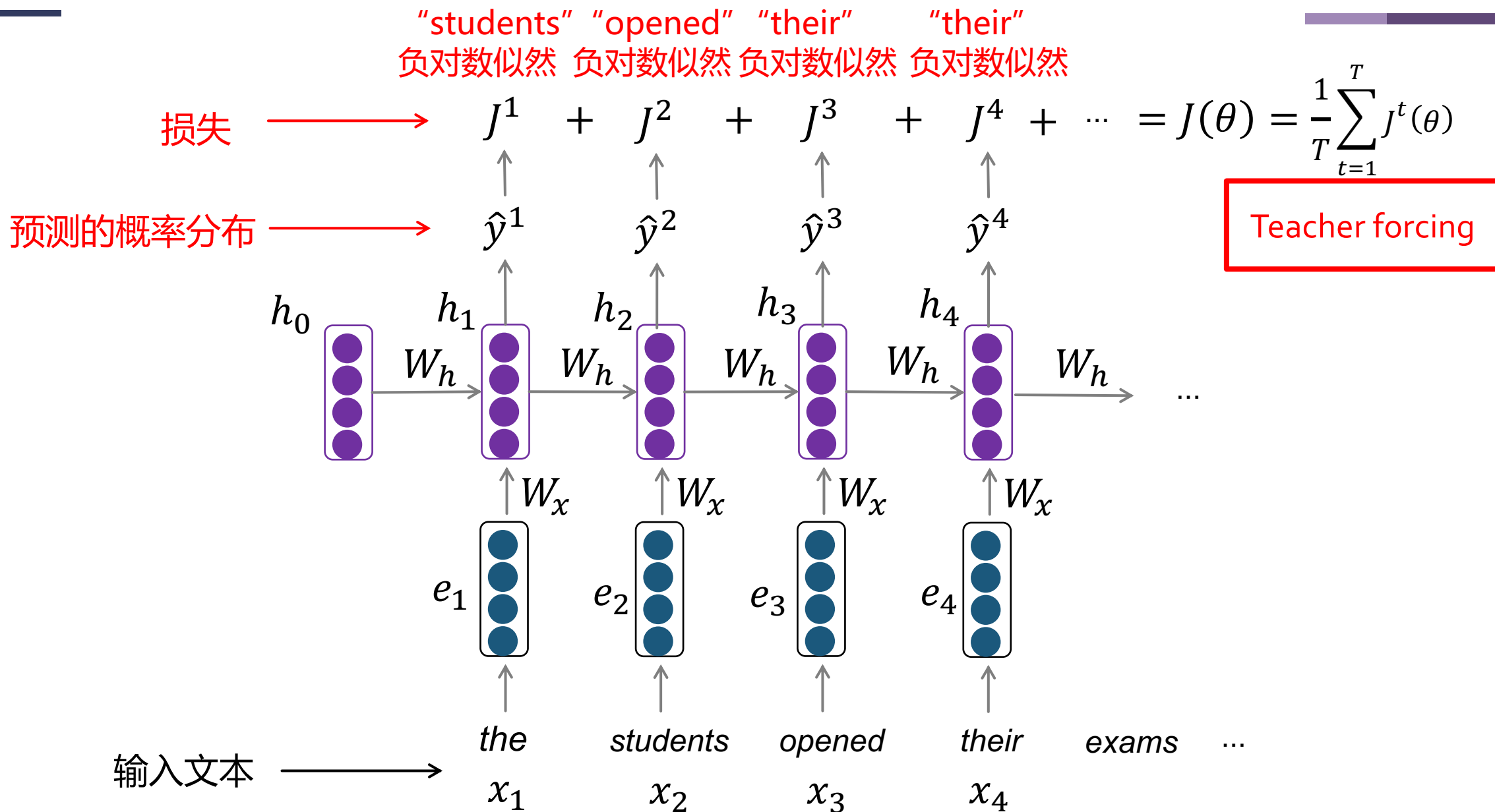
RNN-LM模型训练



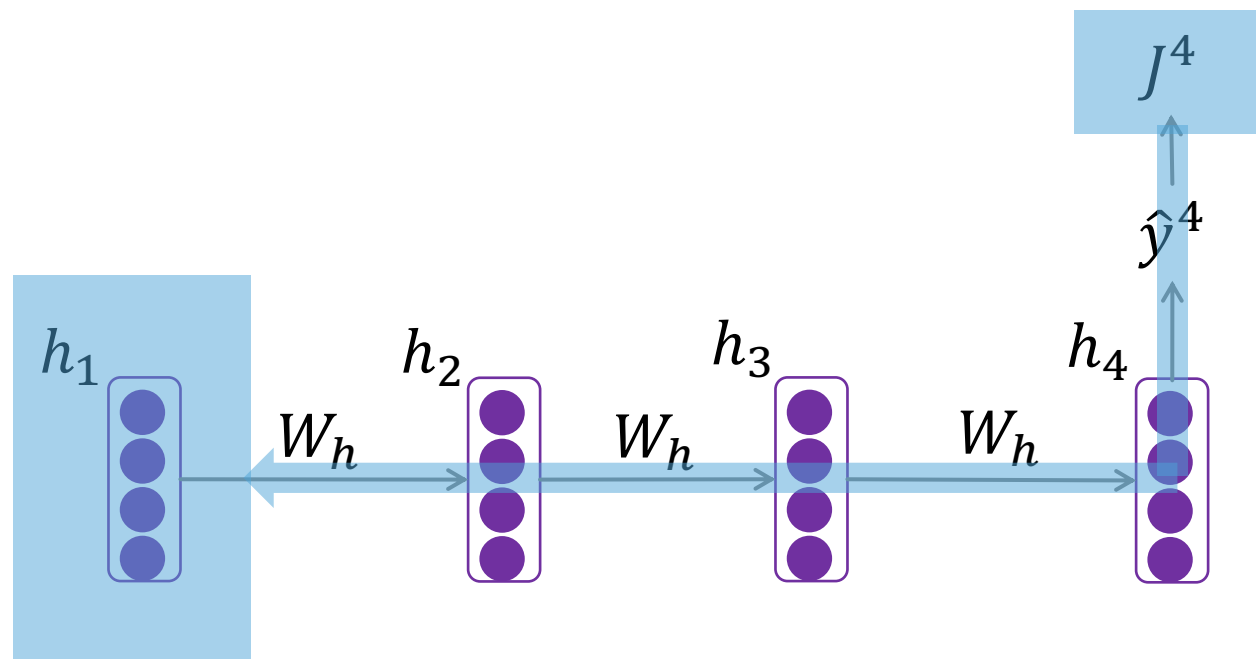
RNN-LM模型训练



RNN-LM模型训练

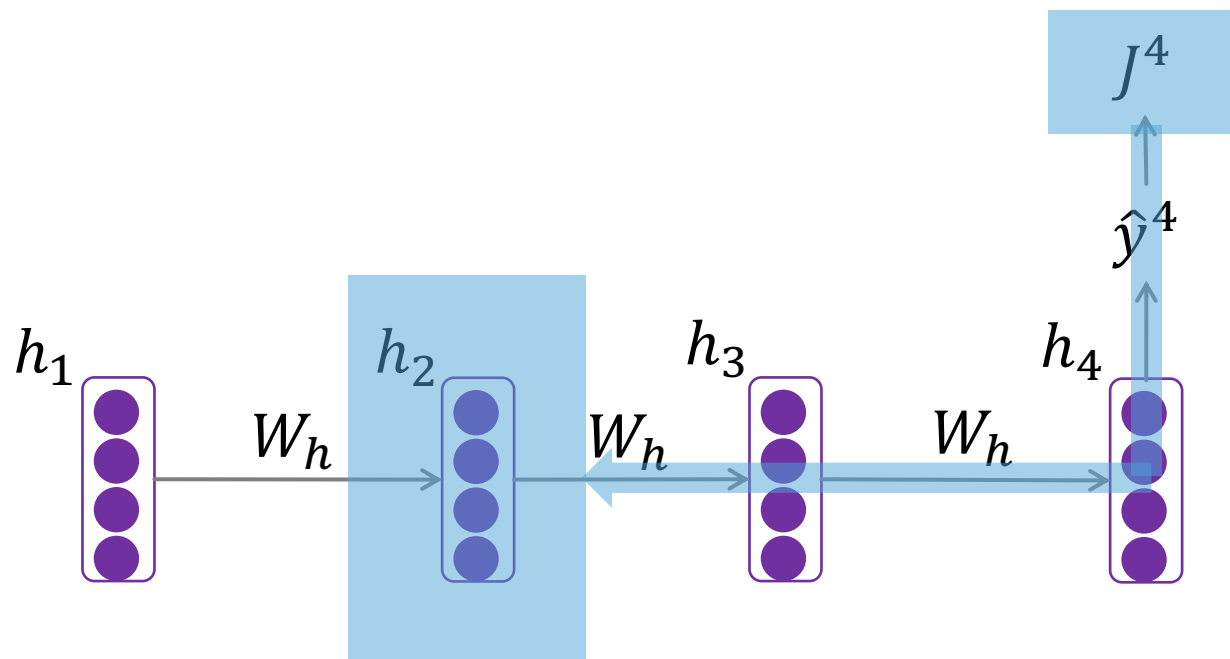


RNN：梯度爆炸、梯度消失



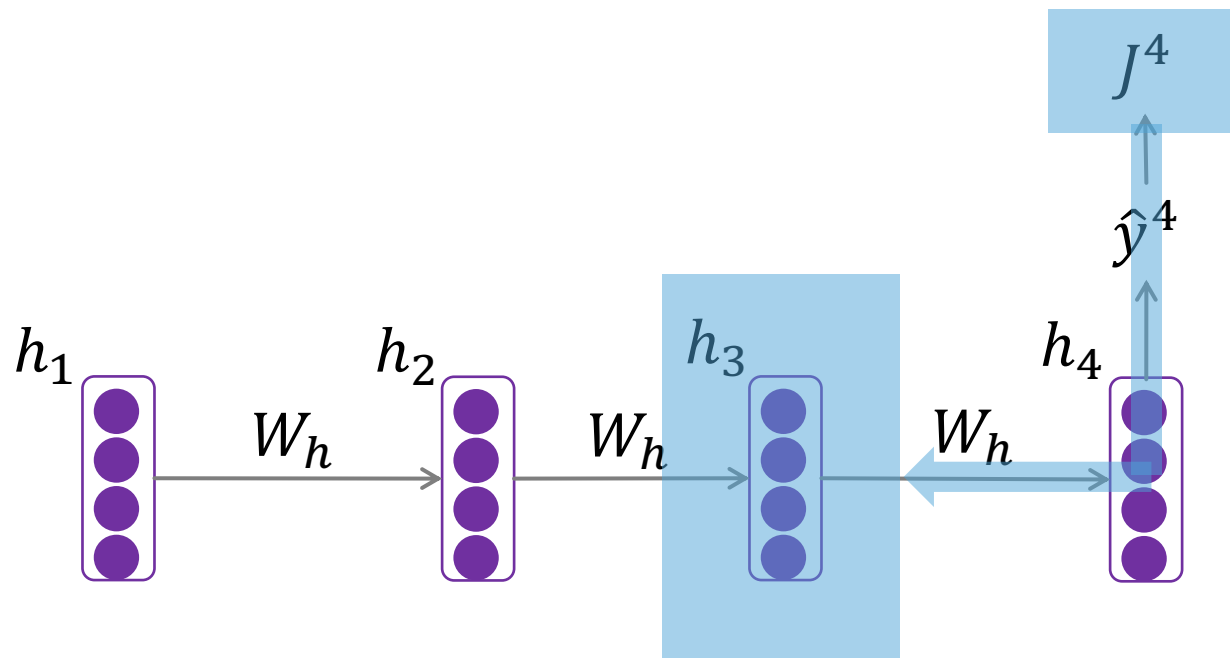
$$\frac{\partial J^4}{\partial h^1}$$

RNN：梯度爆炸、梯度消失



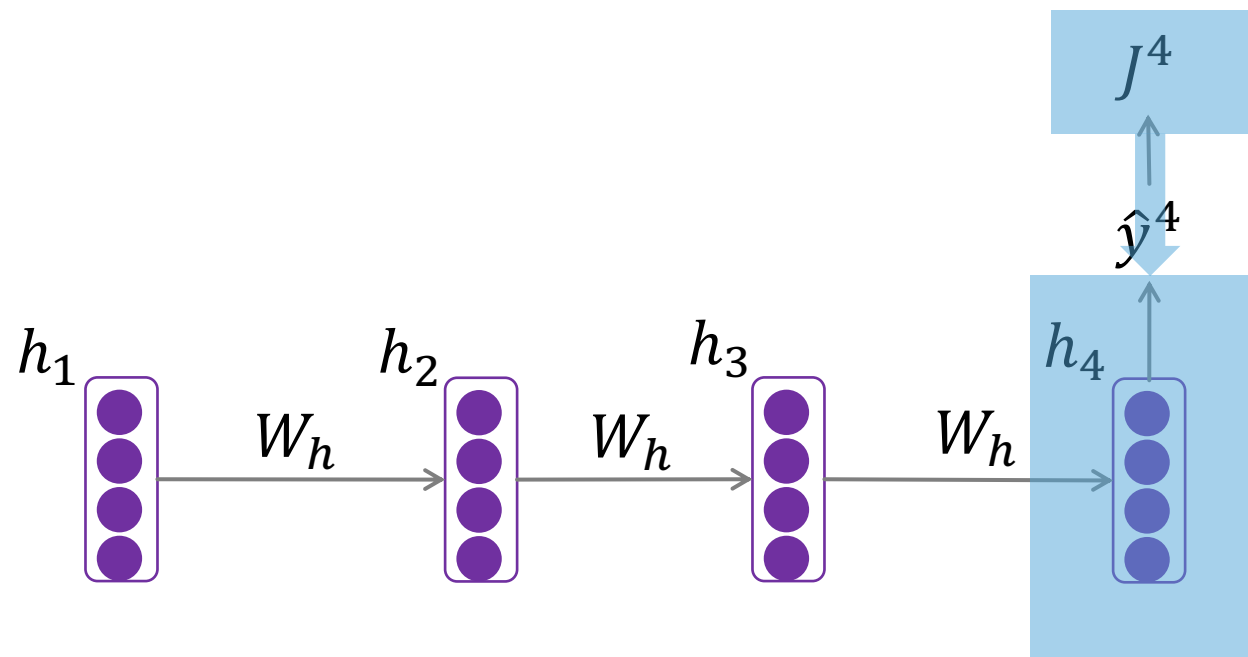
$$\frac{\partial J^4}{\partial h^1} = \frac{\partial h^2}{\partial h^1} \times \frac{\partial J^4}{\partial h^2}$$

RNN：梯度爆炸、梯度消失



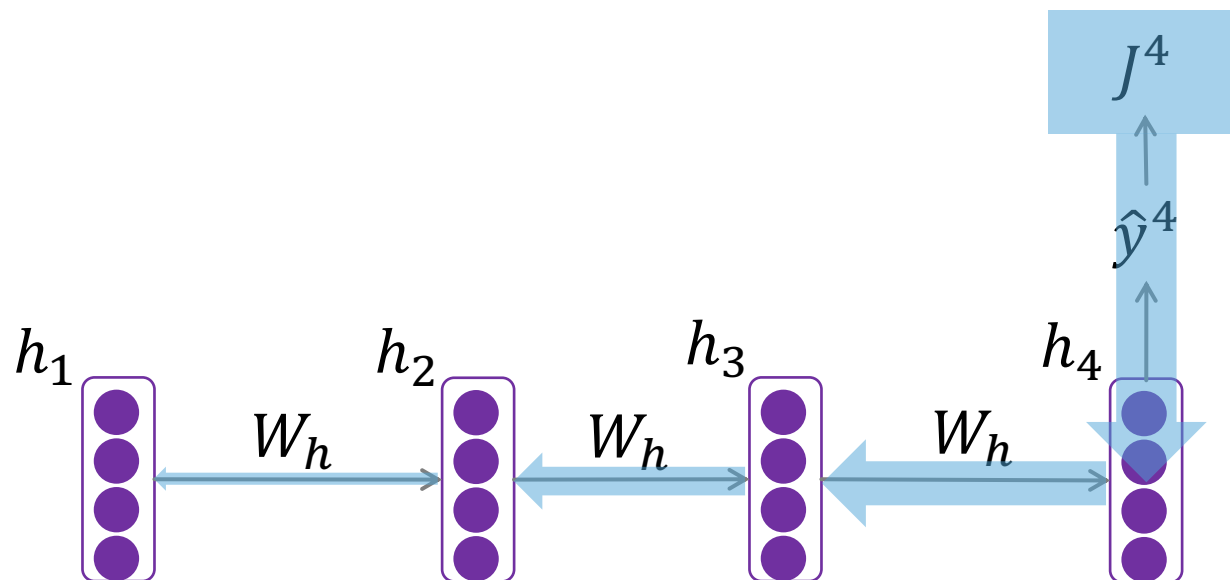
$$\frac{\partial J^4}{\partial h^1} = \frac{\partial h^2}{\partial h^1} \times \frac{\partial h^3}{\partial h^2} \times \frac{\partial J^4}{\partial h^3}$$

RNN：梯度爆炸、梯度消失



$$\frac{\partial J^4}{\partial h^1} = \frac{\partial h^2}{\partial h^1} \times \frac{\partial h^3}{\partial h^2} \times \frac{\partial h^4}{\partial h^3} \times \frac{\partial J^4}{\partial h^4}$$

RNN：梯度爆炸、梯度消失



$$\frac{\partial J^4}{\partial h^1} = \frac{\partial h^2}{\partial h^1} \times \frac{\partial h^3}{\partial h^2} \times \frac{\partial h^4}{\partial h^3} \times \frac{\partial J^4}{\partial h^4}$$

如果这些梯度很大或者很小怎么办？



03



高级循环神经网络

ADVANCED RNN

- 长短期记忆网络 (Long Short-Term Memory , LSTM) : 引入三个门和一个细胞状态来控制神经元的信息流动

- 遗忘门 f_t : 控制哪些信息应该从之前的细胞状态中遗忘 $f_t = \sigma(W_f h_{t-1} + U_f x_t + b_f)$

- 输入门 i_t : 控制哪些信息应该被更新到细胞状态中 $i_t = \sigma(W_i h_{t-1} + U_i x_t + b_i)$

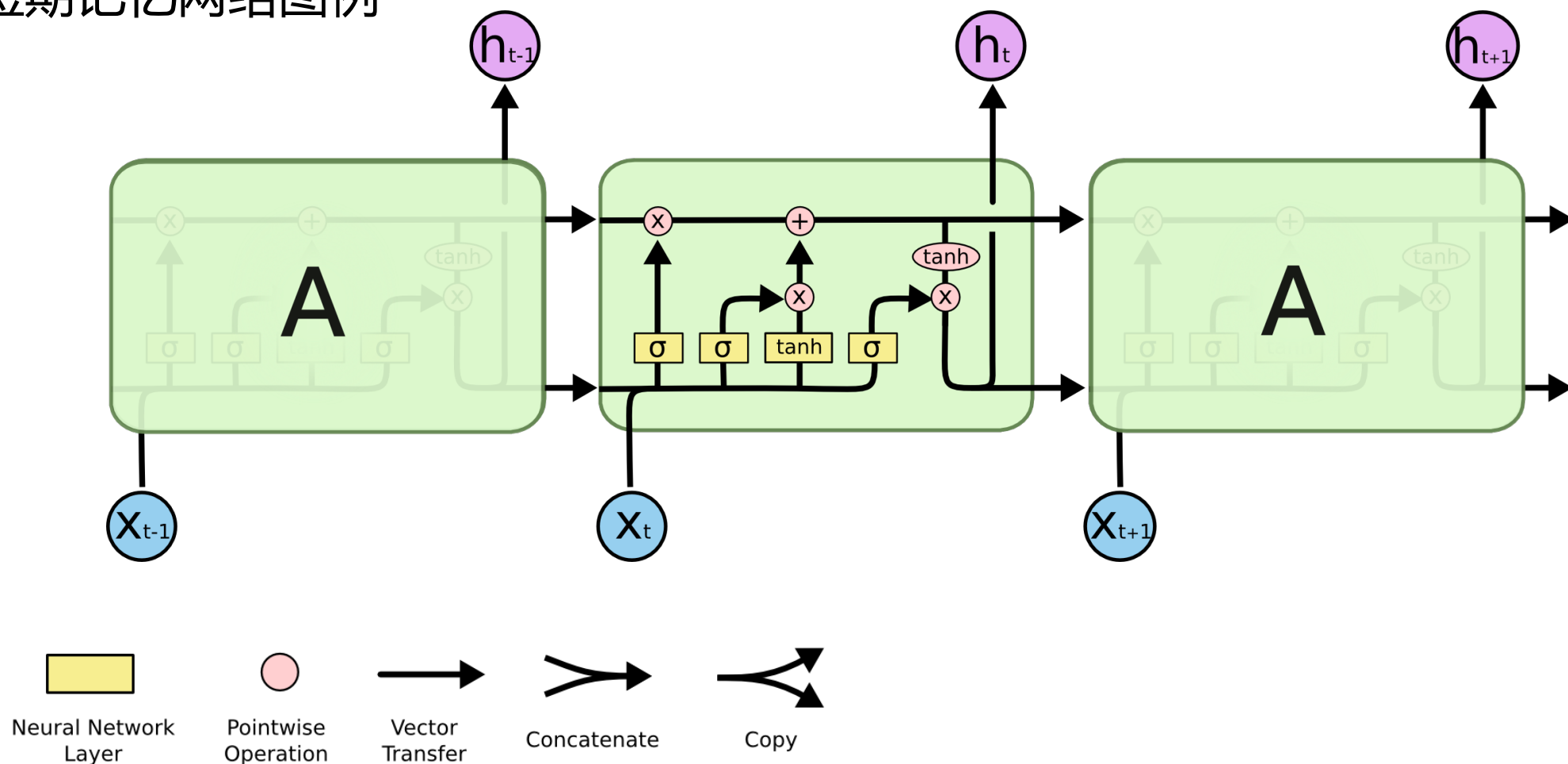
- 输出门 o_t : 控制哪些信息应该被输出到隐层状态中 $o_t = \sigma(W_o h_{t-1} + U_o x_t + b_o)$

- 细胞状态 C_t : 容纳神经元信息 $\hat{C}_t = \tanh(W_c h_{t-1} + U_c x_t + b_c)$

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

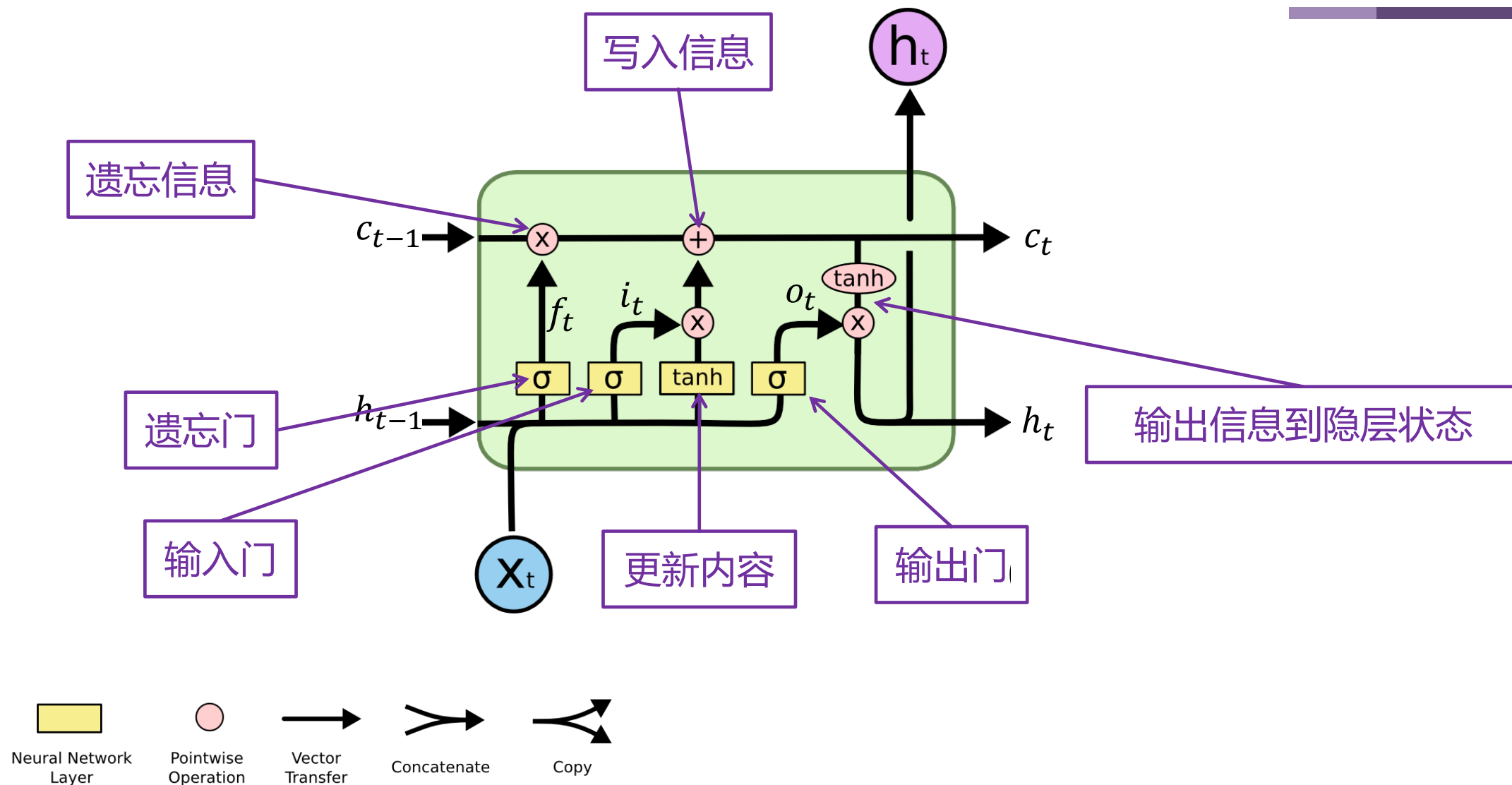
$$h_t = o_t * \tanh(C_t)$$

- 长短期记忆网络图例

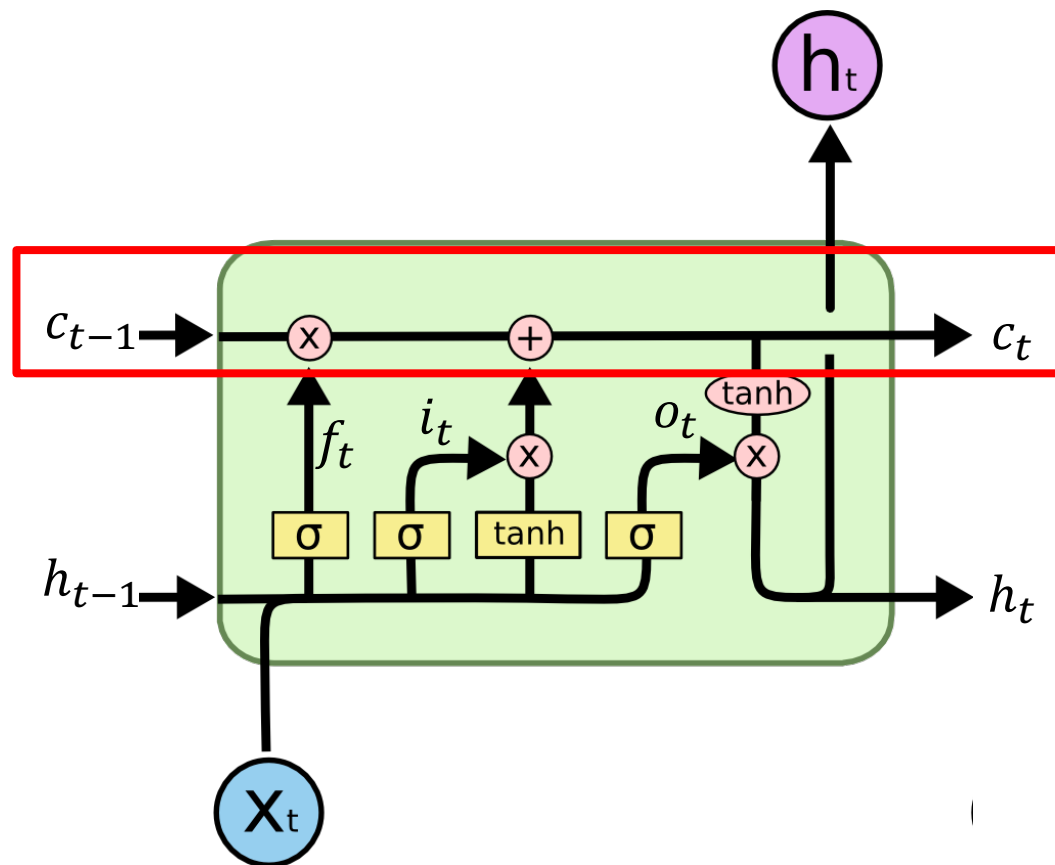


<http://colah.github.io/posts/2015-08-Understanding-LSTMs/>

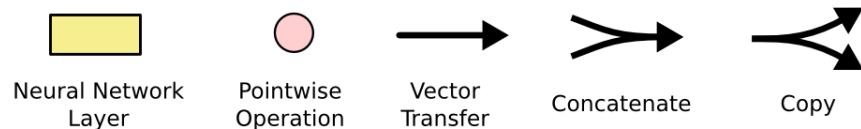
长短期记忆网络



长短期记忆网络



线性变换，缓解远程梯度爆炸和梯度消失问题



- 门控循环单元 (Gated Recurrent Unit , GRU) : 不使用细胞状态

- 更新门 z_t : 控制哪些信息应该被更新
- 重置门 r_t : 控制哪些信息应该被重置
- 更新内容 \tilde{h}_t : 从之前的隐层状态中筛选的信息
- 隐层状态 h_t : 神经元输出的信息

$$z_t = \sigma(W_u h_{t-1} + U_u x_t + b_u)$$

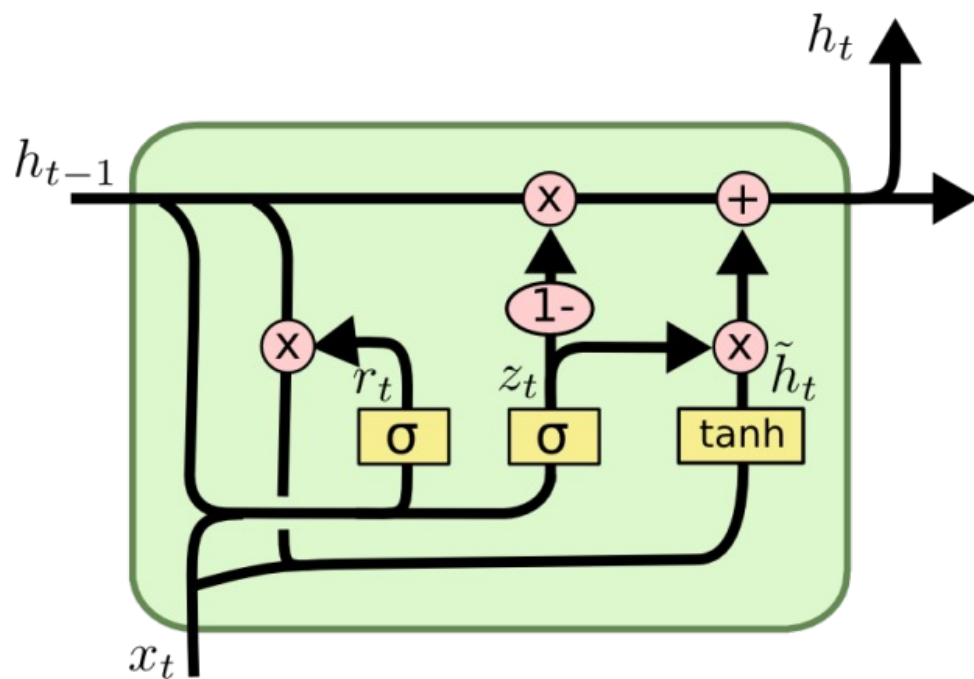
$$r_t = \sigma(W_r h_{t-1} + U_r x_t + b_r)$$

$$\tilde{h}_t = \tanh(W_h(r_t * h_{t-1}) + U_h x_t + b_h)$$

$$h_t = (1 - u_t) * h_{t-1} + z_t * \tilde{h}_t$$

门控循环单元

- 门控循环单元图例



$$z_t = \sigma(W_u h_{t-1} + U_u x_t + b_u)$$

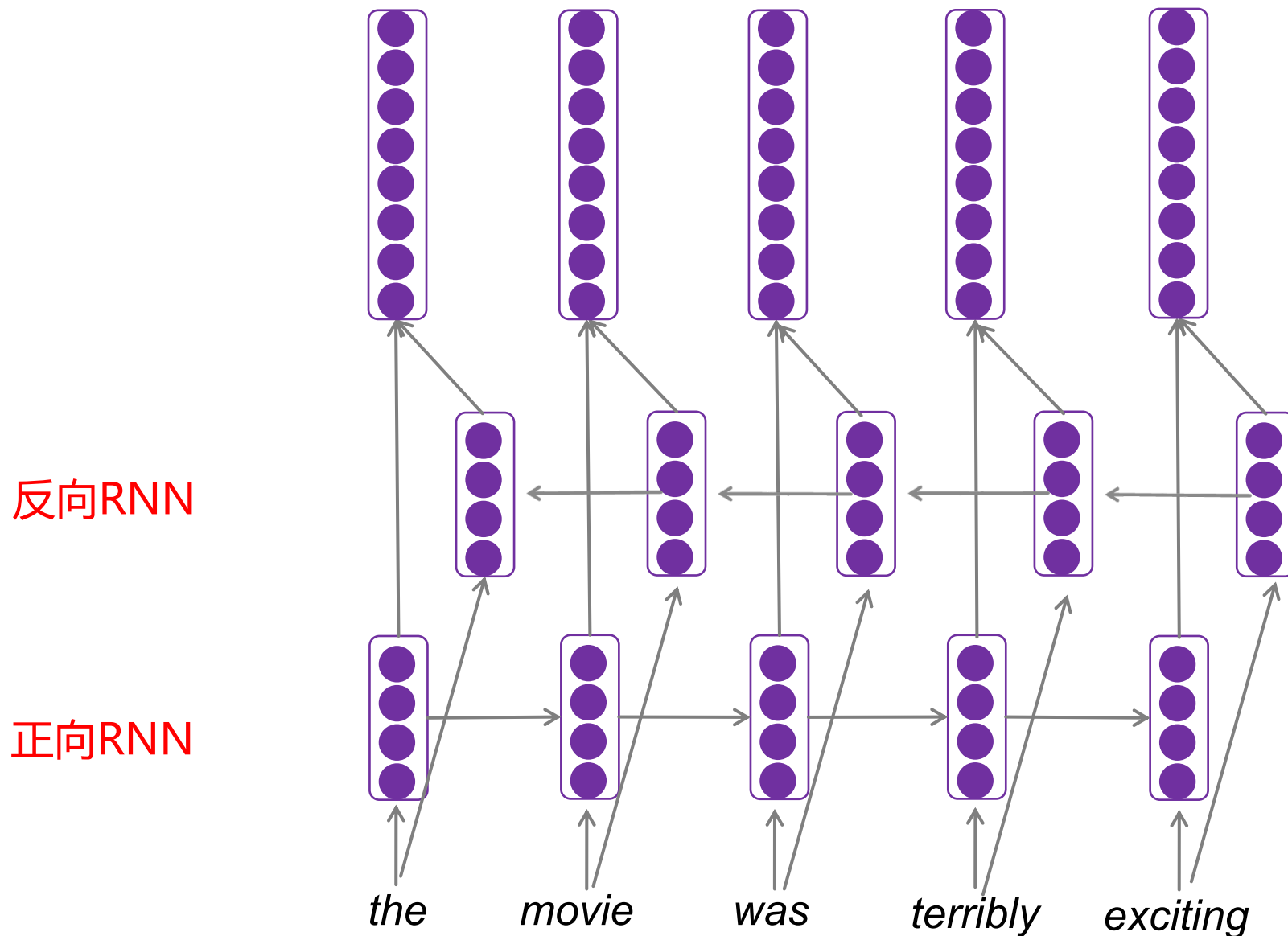
$$r_t = \sigma(W_r h_{t-1} + U_r x_t + b_r)$$

$$\tilde{h}_t = \tanh(W_h(r_t * h_{t-1}) + U_h x_t + b_h)$$

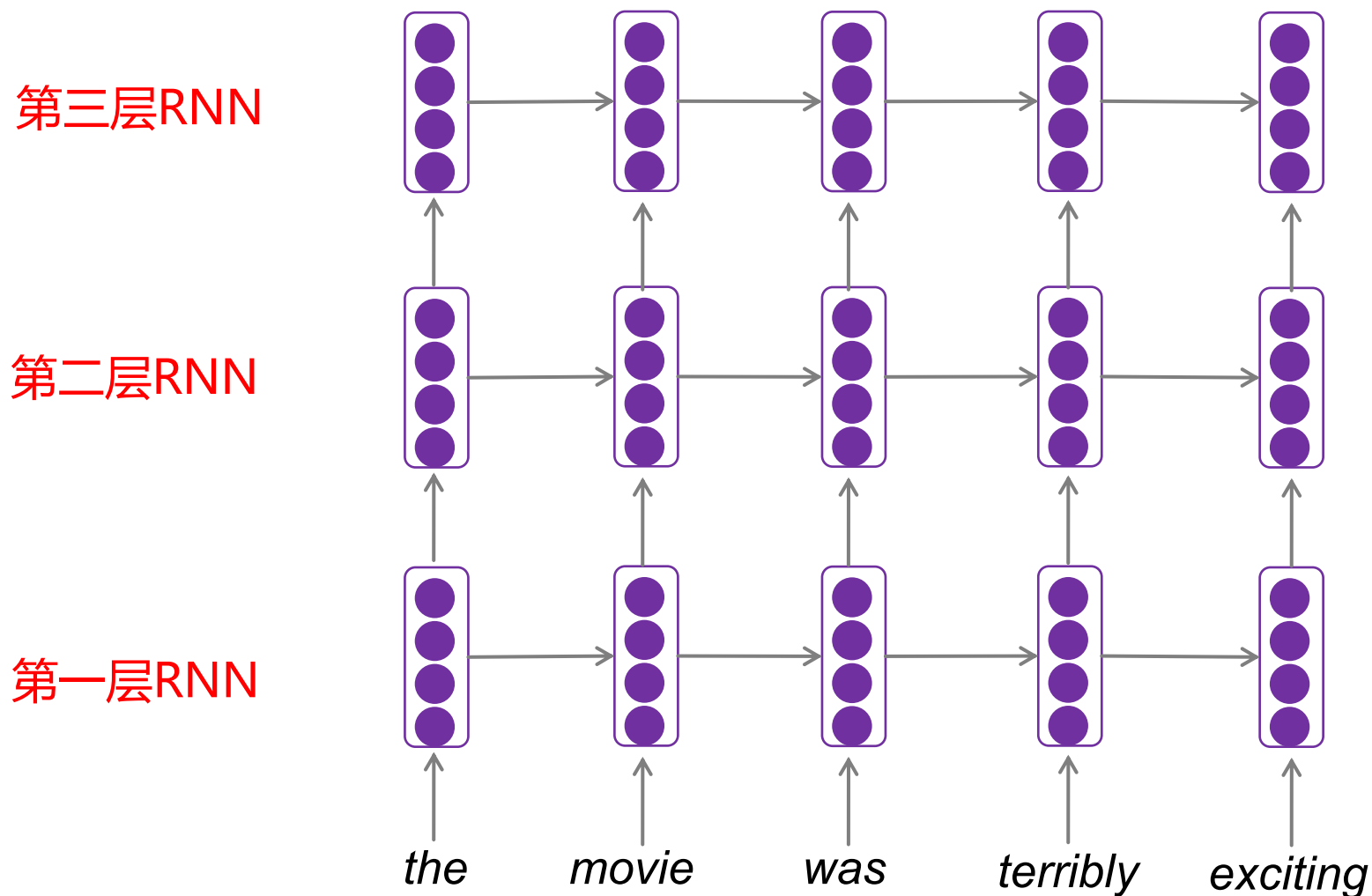
$$h_t = (1 - u_t) * h_{t-1} + z_t * \tilde{h}_t$$

相比于LSTM，参数更少，速度更快

双向循环神经网络 (BI-RNN)



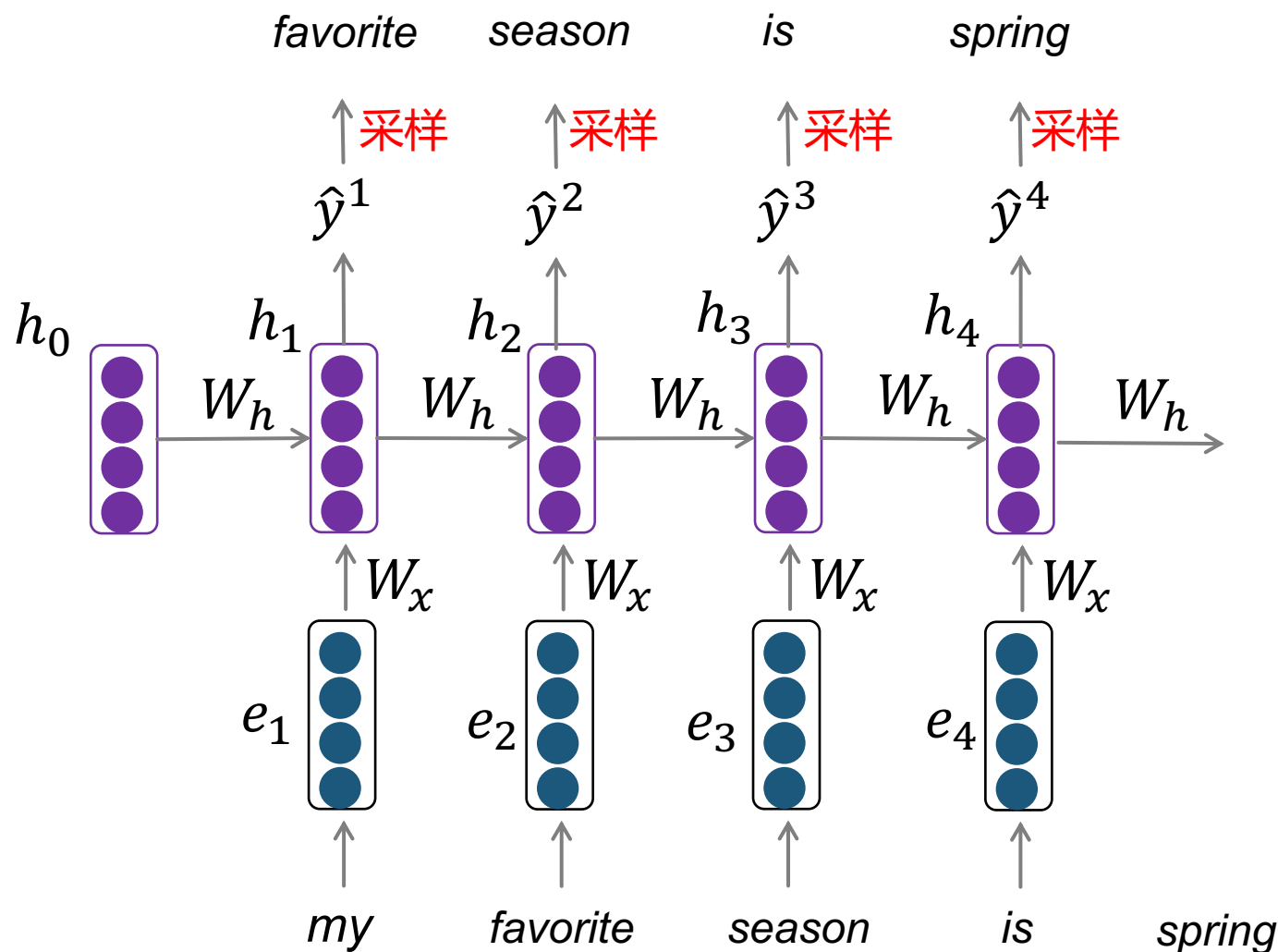
多层循环神经网络 (STACKED-RNN)



捕捉输入序列
的深层语义信息

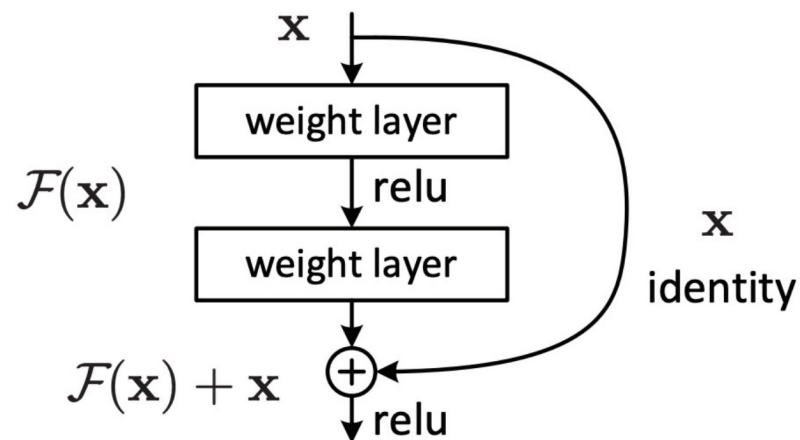
应用：利用RNN-LM生成文本

- 对预测结果进行采样，并作为下一时刻的输入



讨论：仅仅RNN有梯度问题吗？

- 神经网络都有可能会产生梯度问题
 - 前馈神经网络、卷积神经网络、循环神经网络.....
- 深层网络更容易产生梯度问题
 - 梯度连乘后容易接近0（消失）或者爆炸
- 梯度消失常用解决方案
 - ReLU、残差连接...
- 梯度爆炸常用解决方案
 - 梯度裁剪（Clipping）



"Deep Residual Learning for Image Recognition", He et al, 2015. <https://arxiv.org/pdf/1512.03385.pdf>

- Mikolov T, Karafiát M, Burget L, et al. Recurrent neural network based language model. Interspeech. 2010, 2(3): 1045-1048.
- Pascanu R, Mikolov T, Bengio Y. On the difficulty of training recurrent neural networks. International conference on machine learning. PMLR, 2013: 1310-1318.
- Hochreiter S, Schmidhuber J. Long short-term memory. Neural computation, 1997, 9(8): 1735-1780.
- Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014.



南京大學
NANJING UNIVERSITY

Q&A

