# 自然语言处理

## Natural Language Processing

权小军 教授
中山大学数据科学与计算机学院
quanxj3@mail.sysu.edu.cn

# 课程回顾

# 中文分词

- 中文分词作业

# 中文分词

- 中文分词作业
  - 156位同学提交；

✓ 有的同学标了词性

✓ 有的同学没有分析问题

✓ 有的同学对比了多种分词工具的异同（Hanlp，Jieba，snownlp，pkuseg，thulac，nlpir，百度NLP）

✓ 有的同学对分词准确性进行了分析

✓ 有的同学直接使用在线分词工具进行分词

✓ 有的同学对空格进行了处理

✓ 人名，机构名，特殊词汇分不好

✓ 一些奇怪的分词错误

✓ 空格和特殊字符的处理

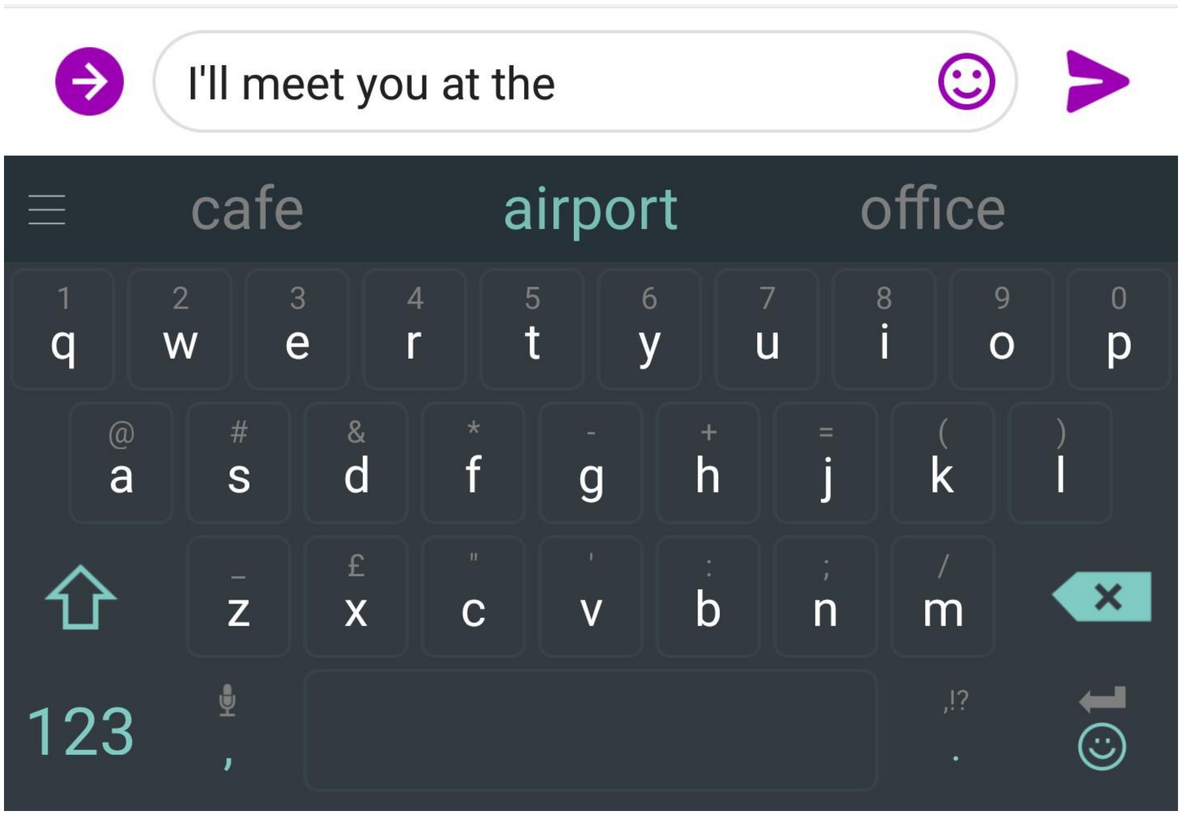
# 语言模型

语言模型

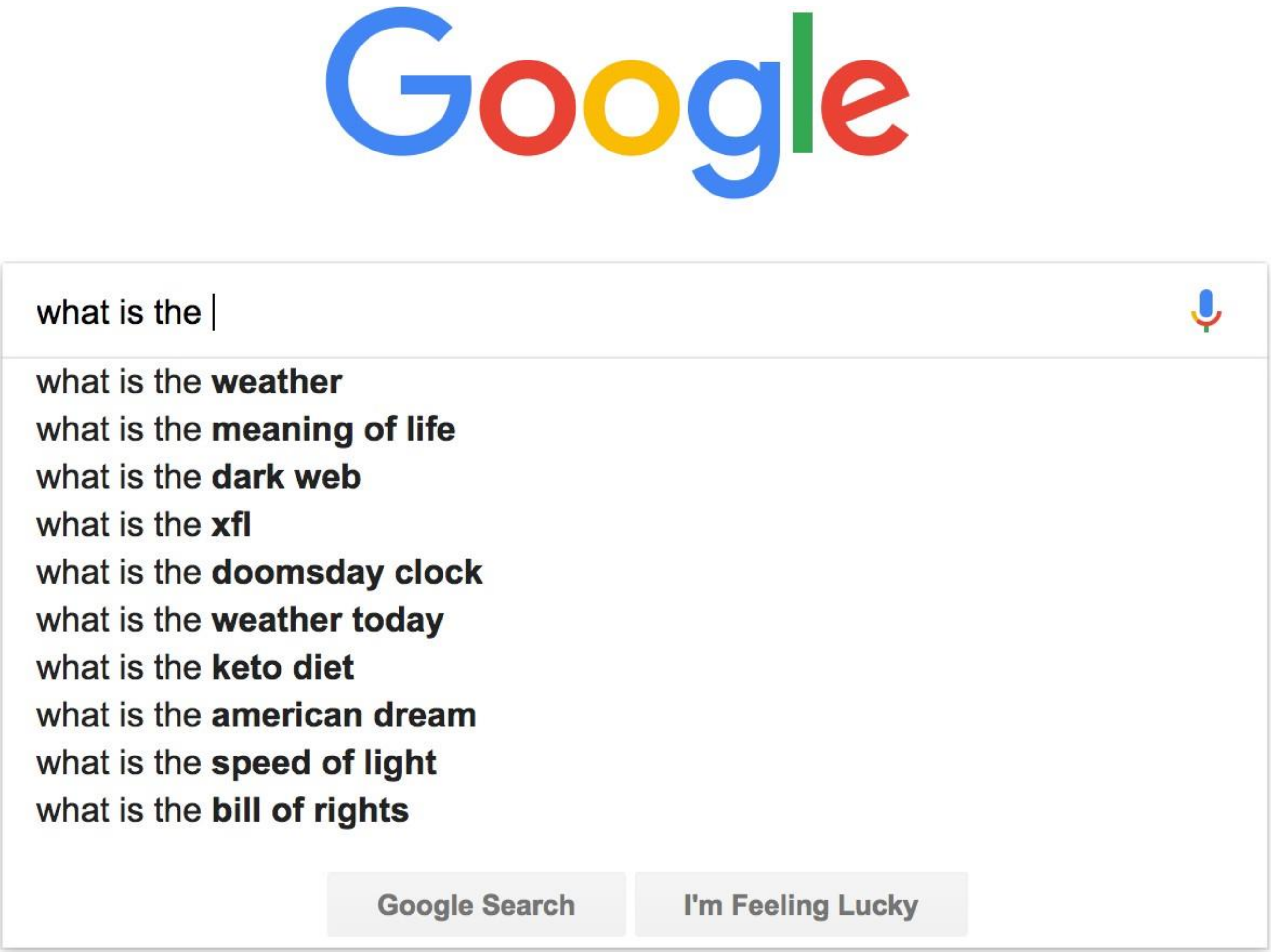
1. 统计语言模型

2. 神经语言模型

# 语言模型应用



# 语言模型应用


Google
what is the
what is the weather
what is the meaning of life
what is the dark web
what is the xfl
what is the doomsday clock
what is the weather today
what is the keto diet
what is the american dream
what is the speed of light
what is the bill of rights
Google Search
I'm Feeling Lucky

# 基于语言模型的文本生成

- You can also use a Language Model to generate words.

例句： *today the price of* ___

# 基于语言模型的文本生成

- You can also use a Language Model to generate words.

*today the price of* ________

$\underbrace{\textit{the price of}}_{\text{基于}}$

计算概率

| | |
|---|---|
| the | 0.072 |
| 18 | 0.043 |
| oil | 0.043 |
| its | 0.036 |
| gold | 0.078 |
| … | |

sample

# 基于语言模型的文本生成

- You can also use a Language Model to generate text.

*today the price of gold per ton , while production of shoe lasts and shoe industry , the bank intervened just after it considered and rejected an imf demand to rebuild depleted european stocks , sept 30 end primary 76 cts a share .*

# 基于语言模型的文本生成

- You can also use a Language Model to generate text.

*today the price of gold per ton , while production of shoe lasts and shoe industry , the bank intervened just after it considered and rejected an imf demand to rebuild depleted european stocks , sept 30 end primary 76 cts a share .*

**Surprisingly grammatical!**

**...but incoherent.**

# *Neural* Language Model

- Window-based neural model

*as the proctor started the clock.*

*The students opened their______*

# *neural* Language Model

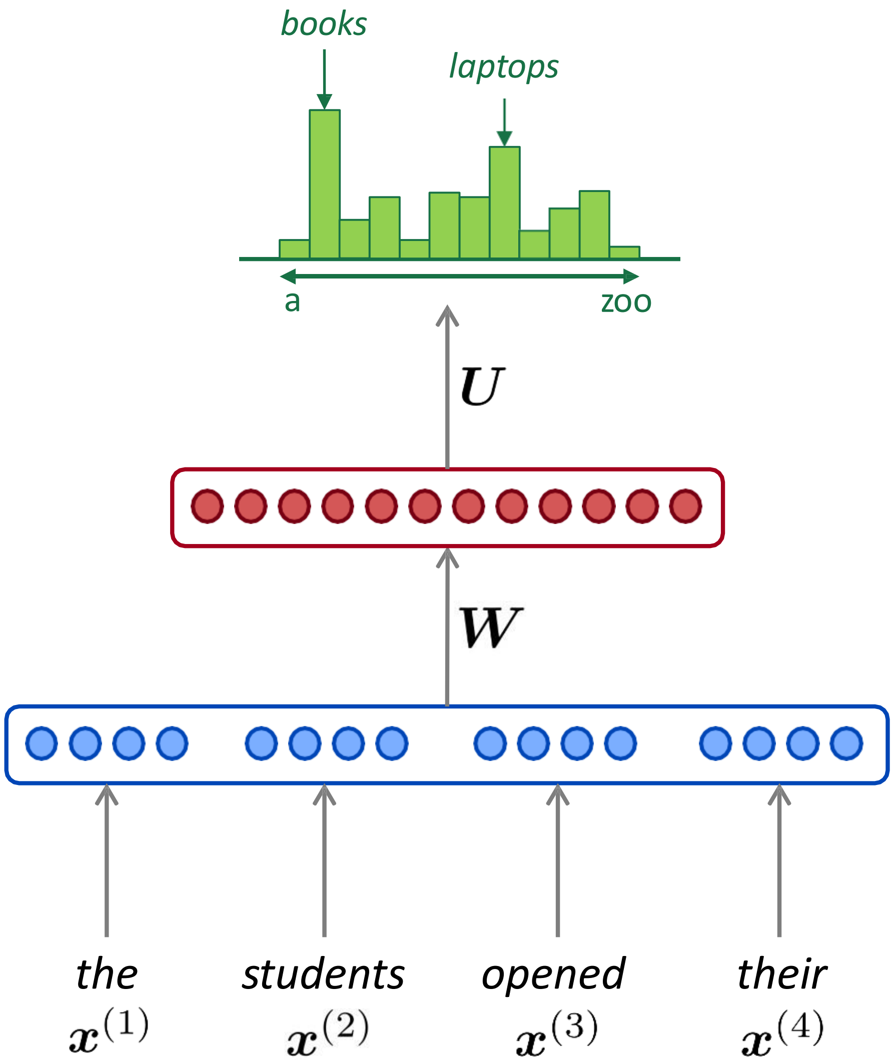
- Window-based neural model?

~~*as the proctor started the clock.*~~

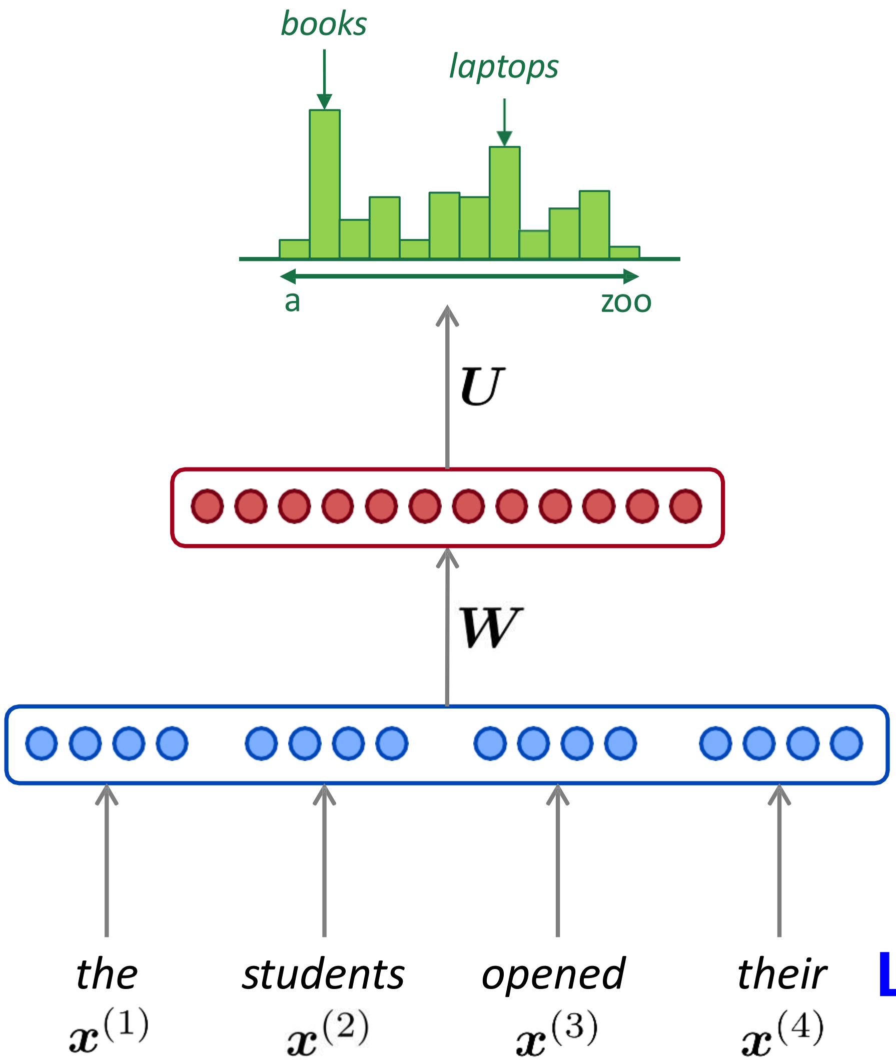
丢弃

$\underbrace{\textit{The students opened their}}_{\text{固定窗口}}$______

# A fixed-window neural Language Model



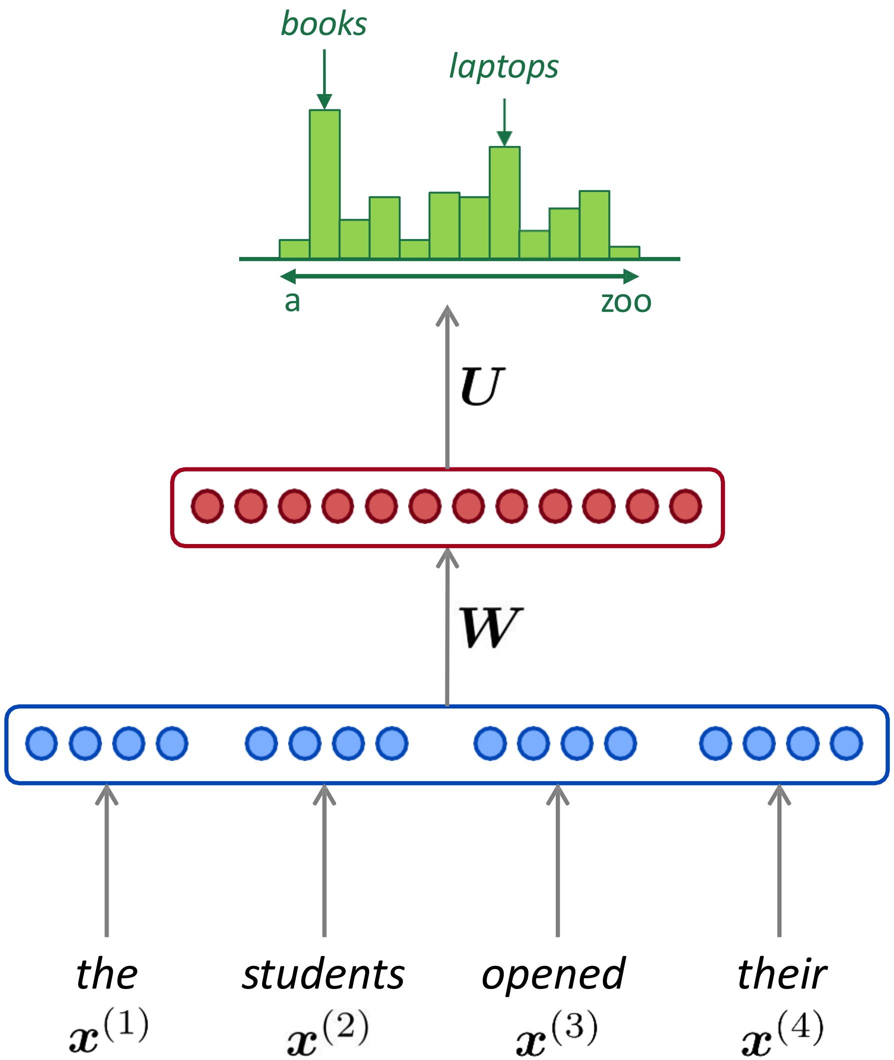
# A fixed-window neural Language Model




**Level 1:** words / one-hot vectors

$$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \boldsymbol{x}^{(4)}$$

# A fixed-window neural Language Model



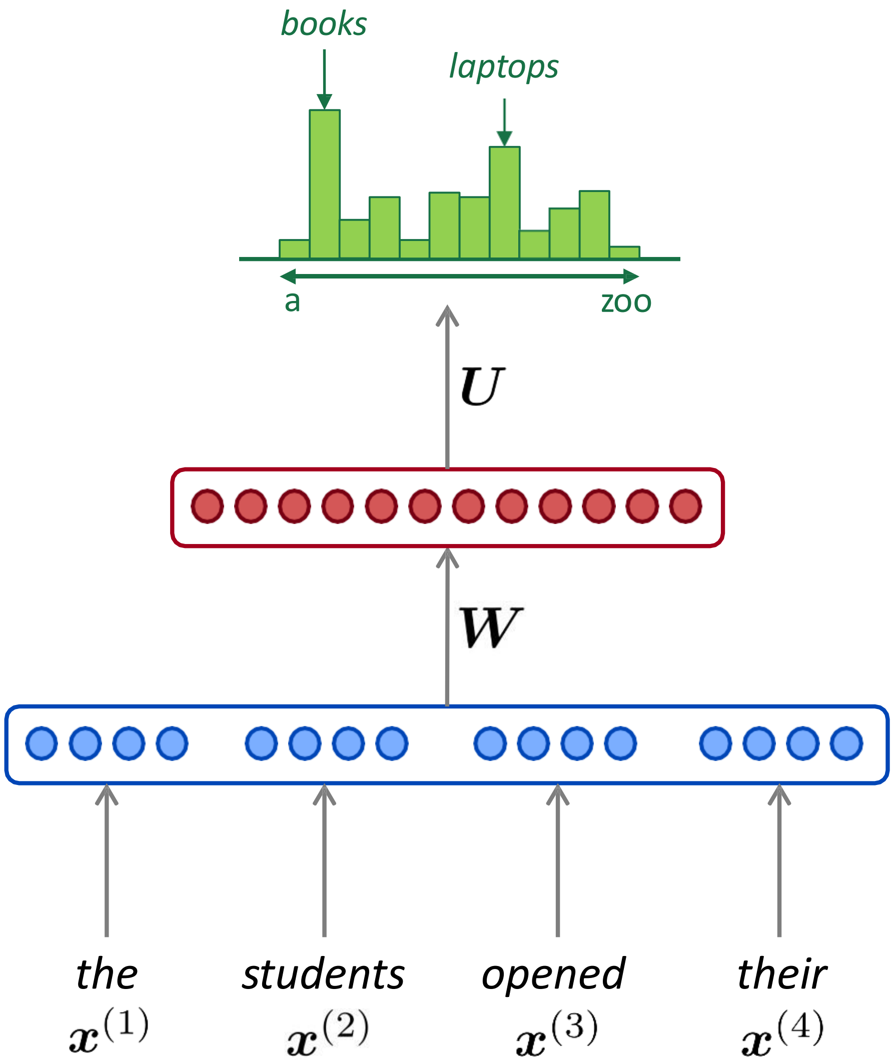

**Level 2:** concatenated word embeddings

$$\boldsymbol{e} = [\boldsymbol{e}^{(1)}; \boldsymbol{e}^{(2)}; \boldsymbol{e}^{(3)}; \boldsymbol{e}^{(4)}]$$

**Level 1:** words / one-hot vectors

$$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \boldsymbol{x}^{(4)}$$

# A fixed-window neural Language Model




**Level 3:** hidden layer

$$\boldsymbol{h} = f(\boldsymbol{W}\boldsymbol{e} + \boldsymbol{b}_1)$$

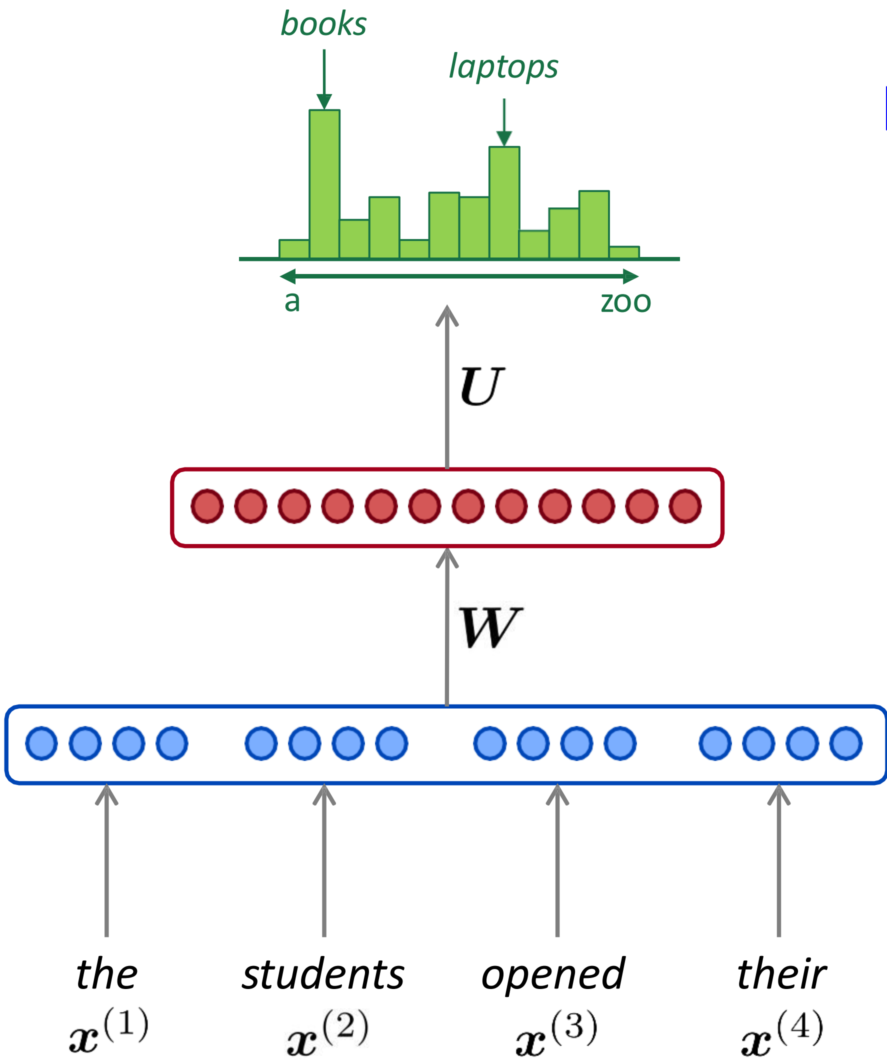
**Level 2:** concatenated word embeddings

$$\boldsymbol{e} = [\boldsymbol{e}^{(1)}; \boldsymbol{e}^{(2)}; \boldsymbol{e}^{(3)}; \boldsymbol{e}^{(4)}]$$

**Level 1:** words / one-hot vectors

$$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \boldsymbol{x}^{(4)}$$

# A fixed-window neural Language Model




**Level 4:** output distribution

$$\hat{\boldsymbol{y}} = \mathrm{softmax}(\boldsymbol{U}\boldsymbol{h} + \boldsymbol{b}_2) \in \mathbb{R}^{|V|}$$

**Level 3:** hidden layer

$$\boldsymbol{h} = f(\boldsymbol{W}\boldsymbol{e} + \boldsymbol{b}_1)$$

**Level 2:** concatenated word embeddings

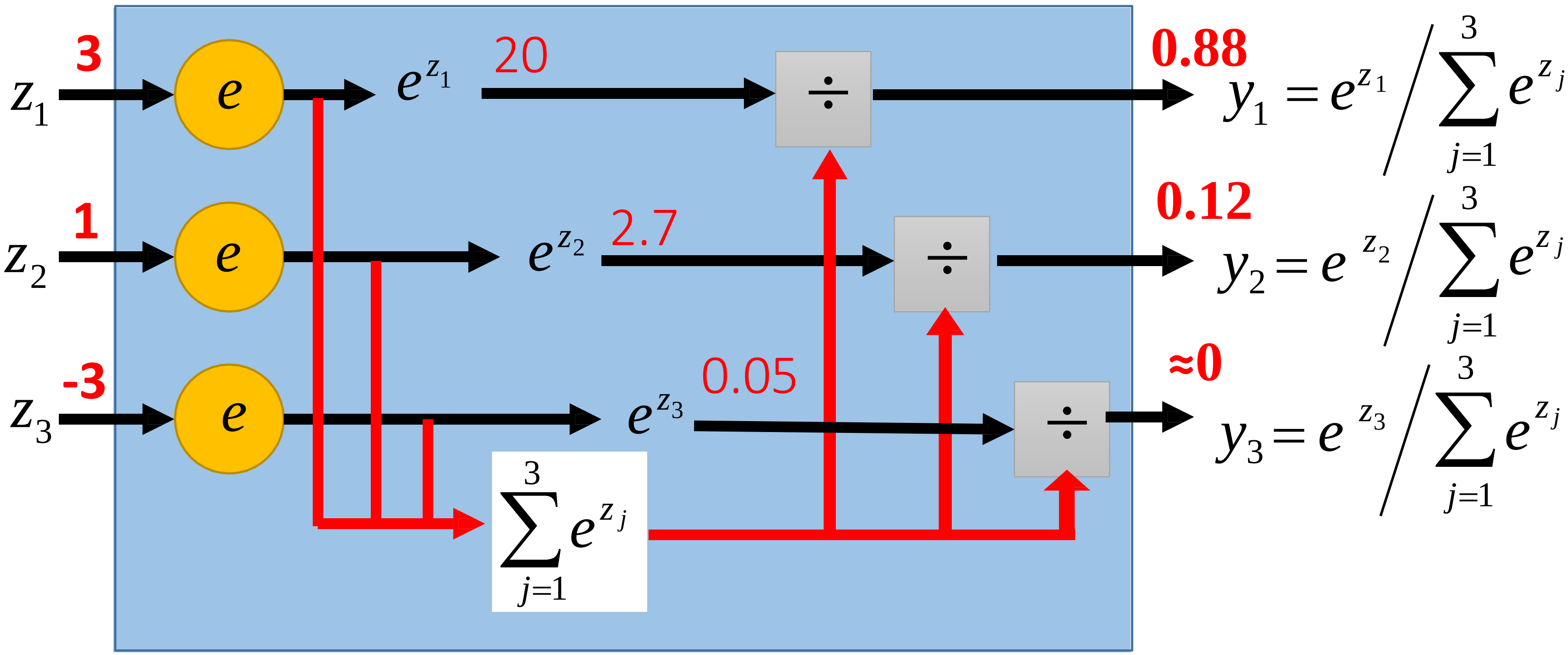
$$\boldsymbol{e} = [\boldsymbol{e}^{(1)}; \boldsymbol{e}^{(2)}; \boldsymbol{e}^{(3)}; \boldsymbol{e}^{(4)}]$$

**Level 1:** words / one-hot vectors

$$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(3)}, \boldsymbol{x}^{(4)}$$

# Softmax

- Softmax 层作为一个输出层



# 典型应用：Word2vec

# Word2vec

***"A word is known by the company it keeps"***

# Word2vec

## Word Representations

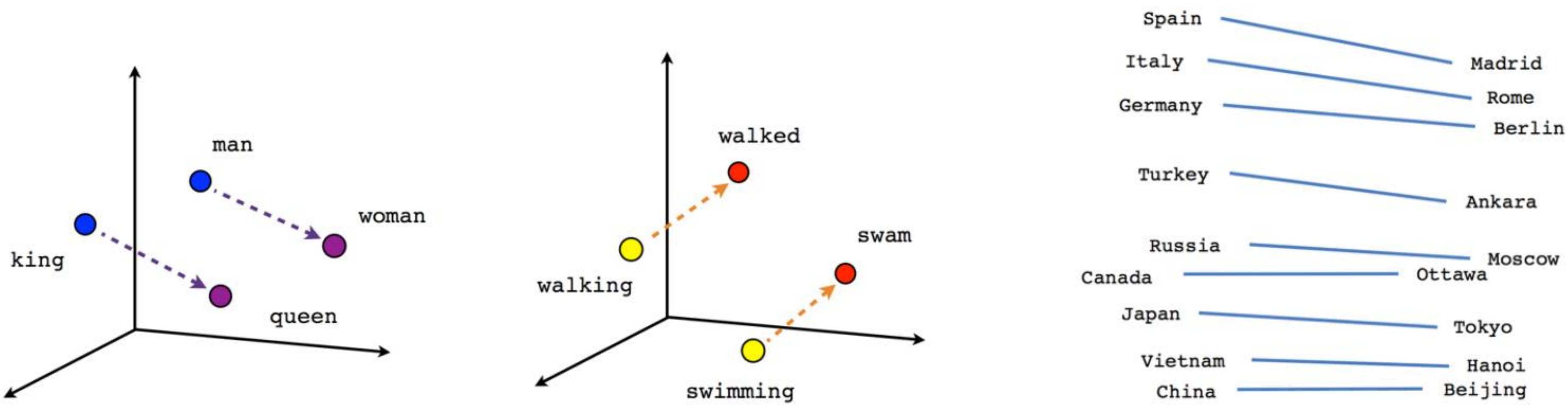
| Traditional Method - Bag of Words Model | Word Embeddings |
|---|---|
| • Uses one hot encoding<br>• Each word in the vocabulary is represented by one bit position in a HUGE vector.<br>• For example, if we have a vocabulary of 10000 words, and "Hello" is the $4^{th}$ word in the dictionary, it would be represented by: 0 0 0 1 0 0 . . . . . . . . 0 0 0 0<br>• Context information is not utilized | • Stores each word in as a point in space, where it is represented by a vector of fixed number of dimensions (generally 300)<br>• Unsupervised, built just by reading huge corpus<br>• For example, "Hello" might be represented as : [0.4, -0.11, 0.55, 0.3 . . . 0.1, 0.02]<br>• Dimensions are basically projections along different axes, more of a mathematical concept. |

# Word2vec

- 词向量给NLP问题提供一个全新的视角
- Word2vec通过一种无监督的方式获取词向量

# Word2vec

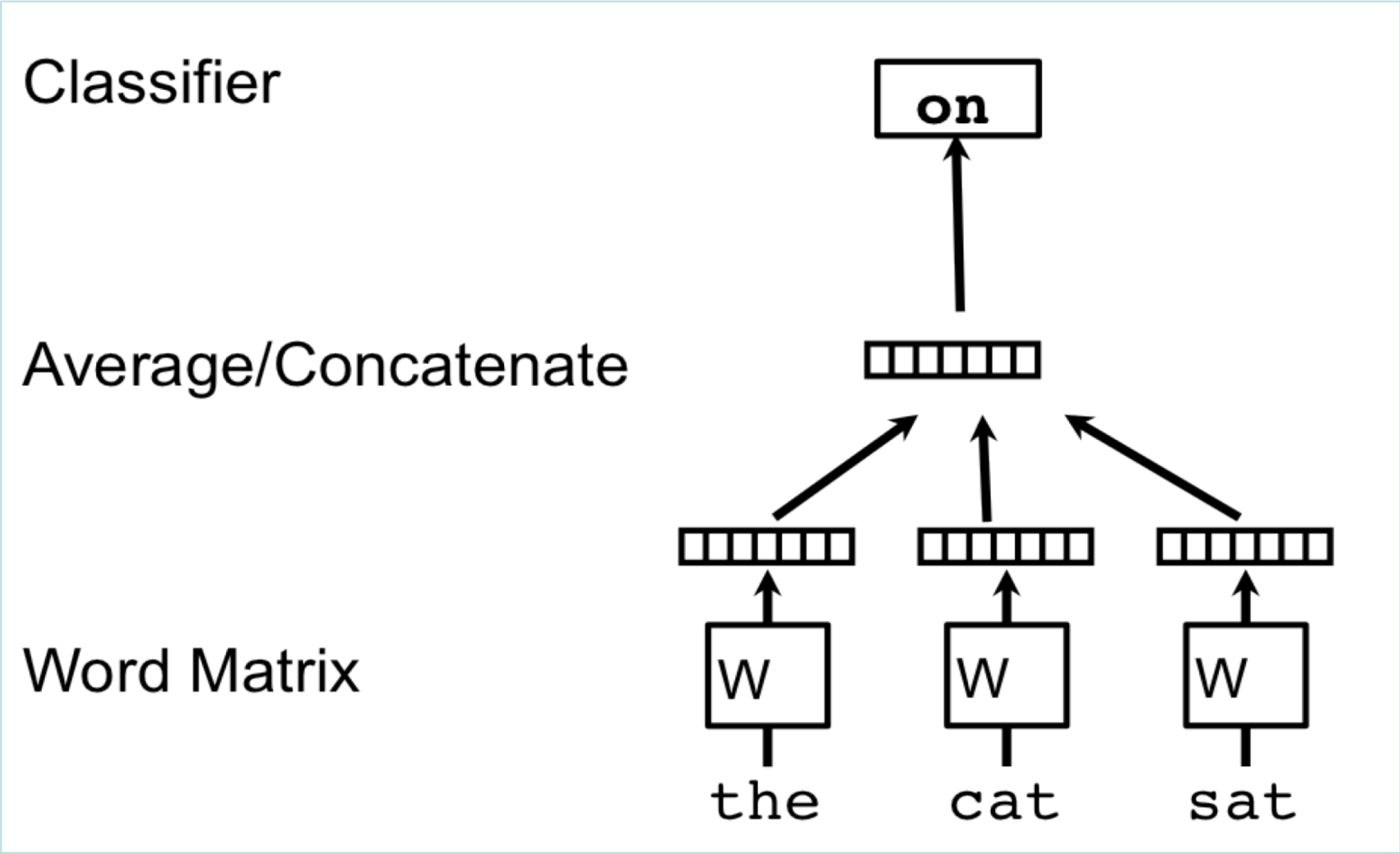
## 典型的例子：




Male-Female　　Verb tense　　Country-Capital

vector[Queen] = vector[King] - vector[Man] + vector[Woman]

# Word2vec原理


Classifier
on
Average/Concatenate
Word Matrix
W
W
W
the
cat
sat

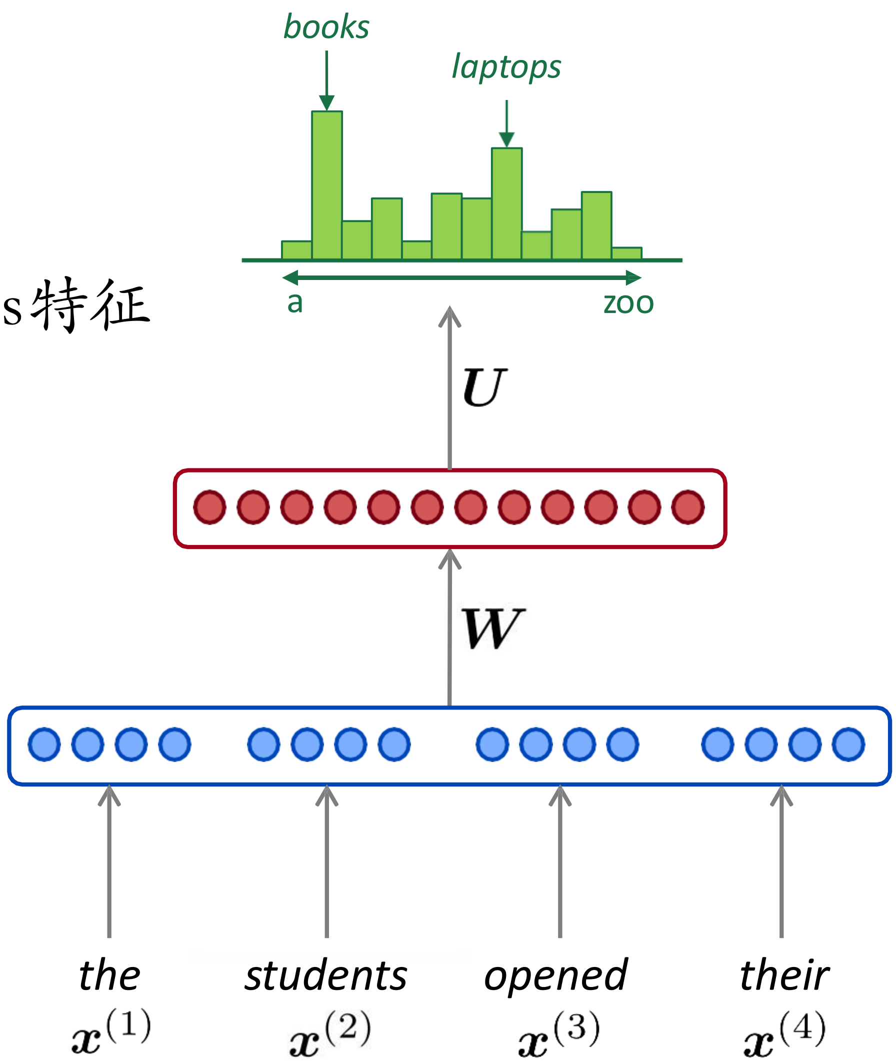
# A fixed-window neural Language Model

和n-gram对比分析

# A fixed-window neural Language Model

和*n*-gram语言模型相比:

- 不存在稀疏性问题
- 不用存储所有已知的*n*-grams特征



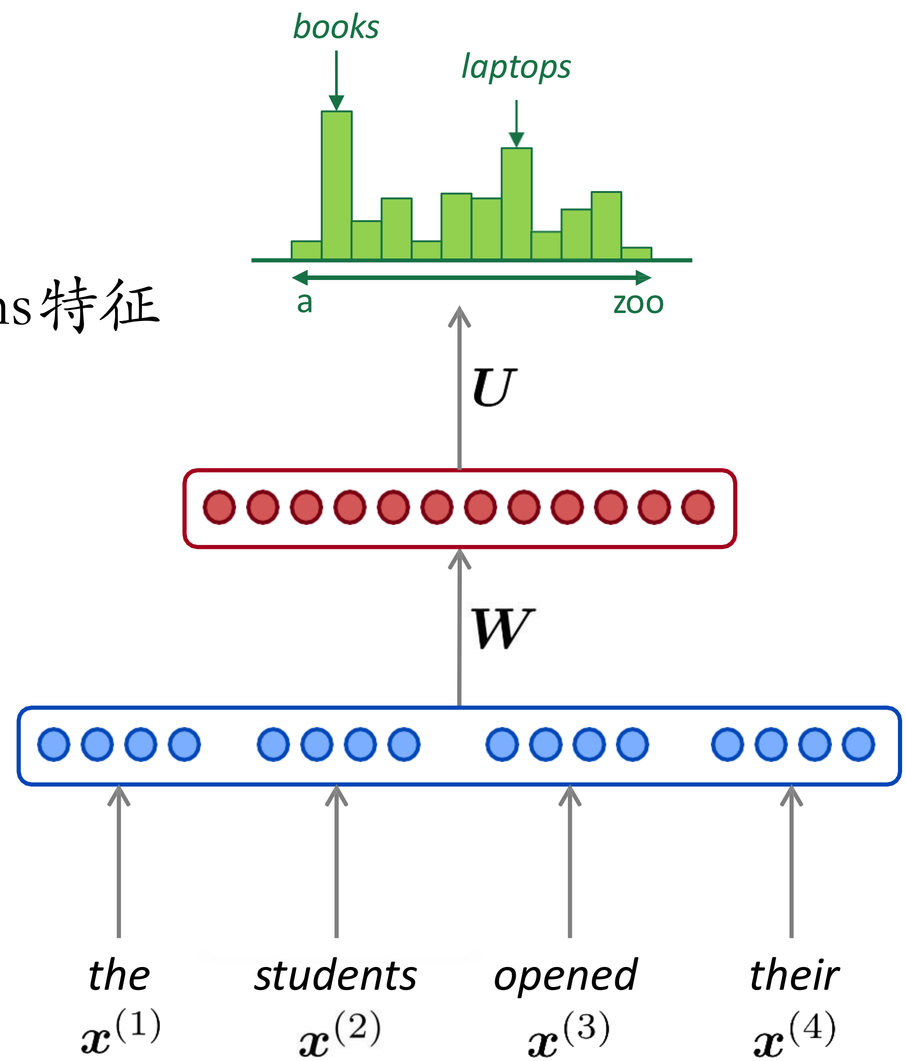
# A fixed-window neural Language Model

和*n*-gram语言模型相比:

- 不存在稀疏性问题
- 不用存储所有已知的*n*-grams特征

存在的问题:

- 窗口太小，效果有限
- 窗口太大，W也会变大



# A fixed-window neural Language Model

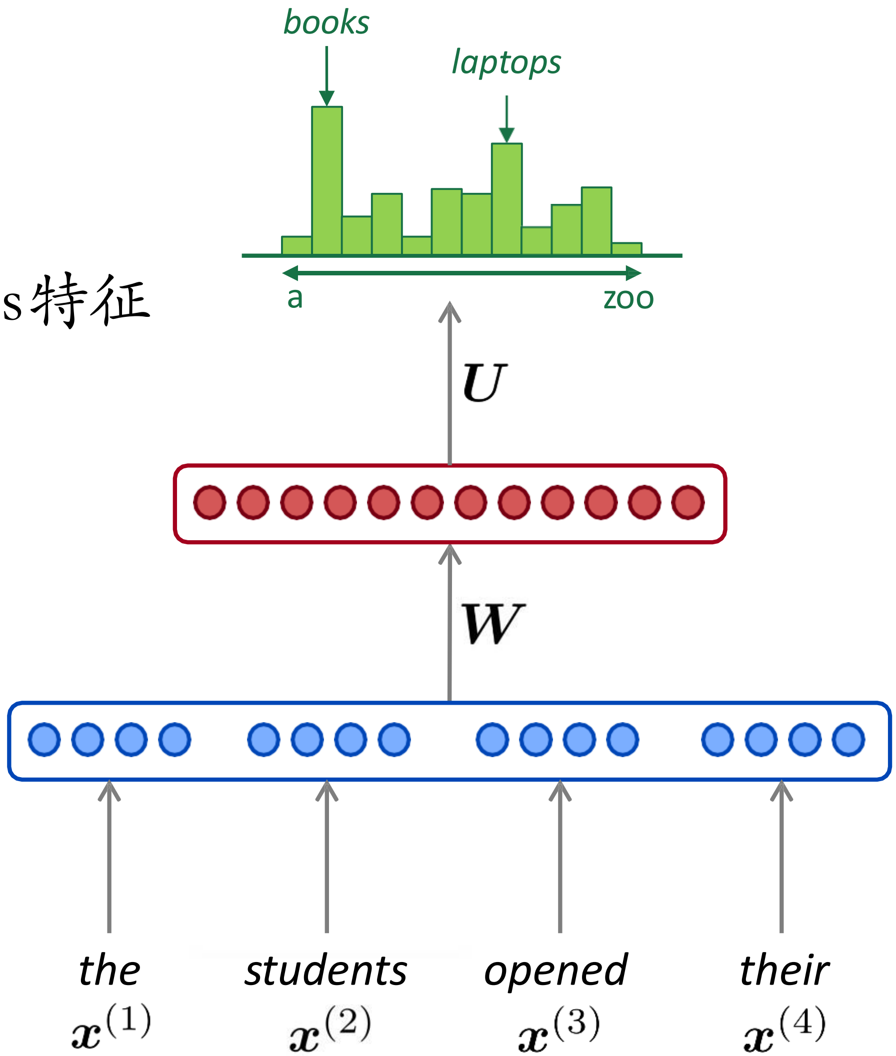
和*n*-gram语言模型相比:

- 不存在稀疏性问题
- 不用存储所有已知的*n*-grams特征

存在的问题:

- 窗口太小
- 窗口太大，W也会变大

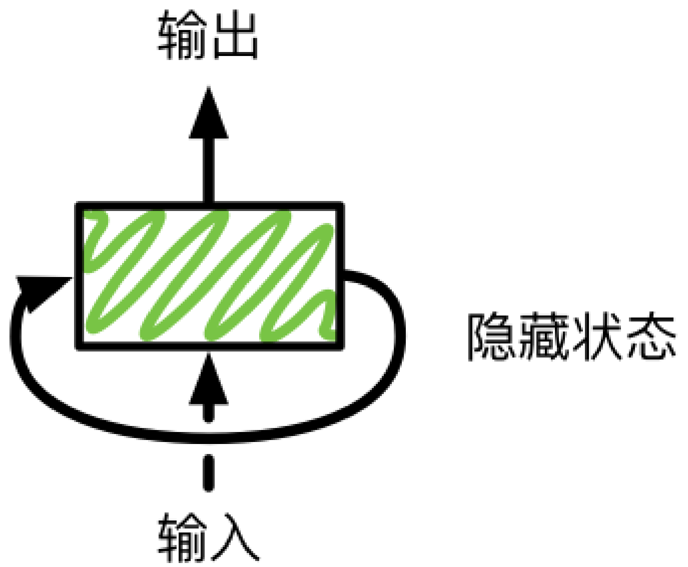
需要一种能处理任意长度输入的架构



# 循环神经网络
# Recurrent Neural Networks

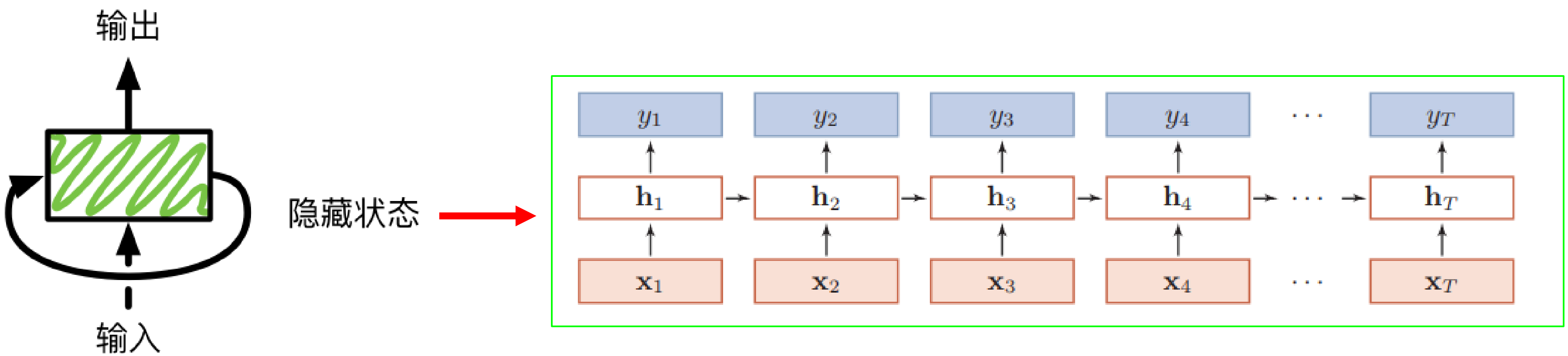
# 循环神经网络

- 循环神经网络主要用于处理（变长）序列数据



# 循环神经网络

- 循环神经网络主要用于处理（变长）序列数据



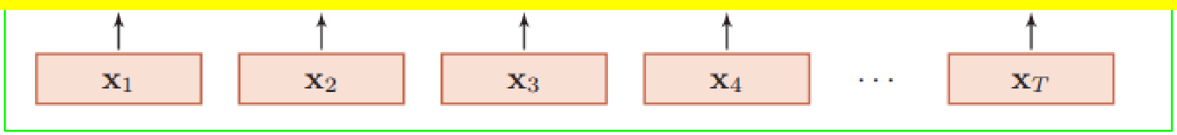

$$\mathbf{h}_t = f(U\mathbf{h}_{t-1} + W\mathbf{x}_t + \mathbf{b}),$$

$$\mathbf{y}_t = V\mathbf{h}_t,$$

# 循环神经网络

- 循环神经网络主要用于处理（变长）序列数据

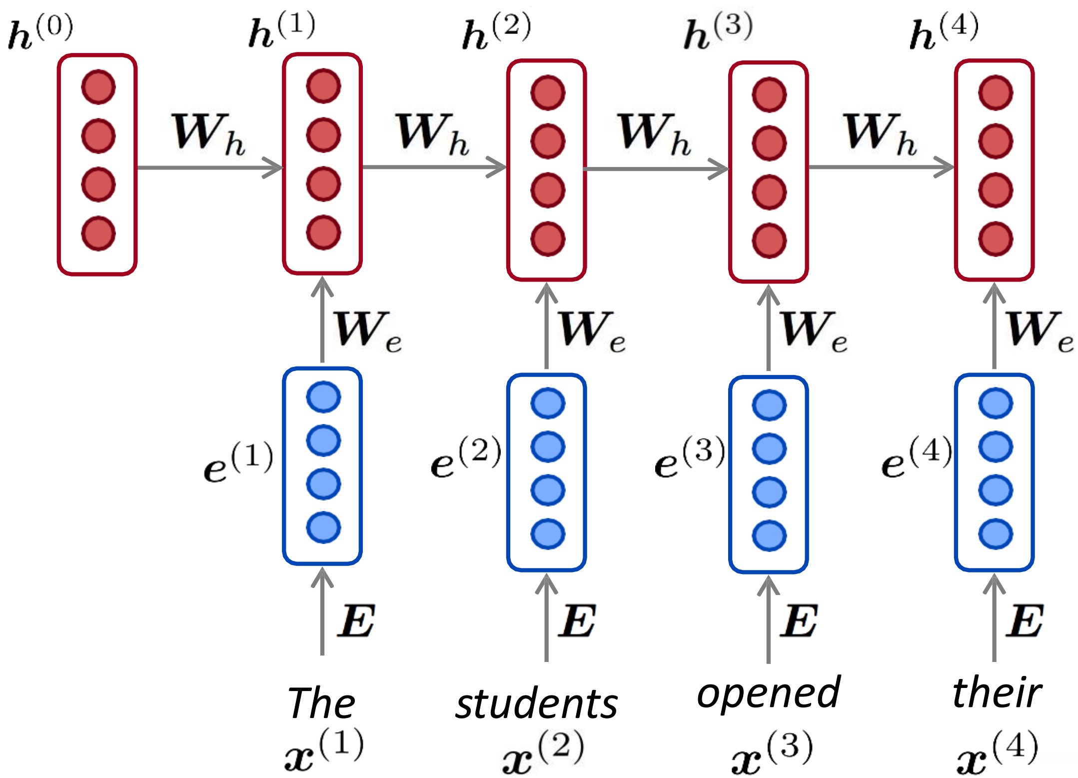

**不同的神经单元共享相同的参数**



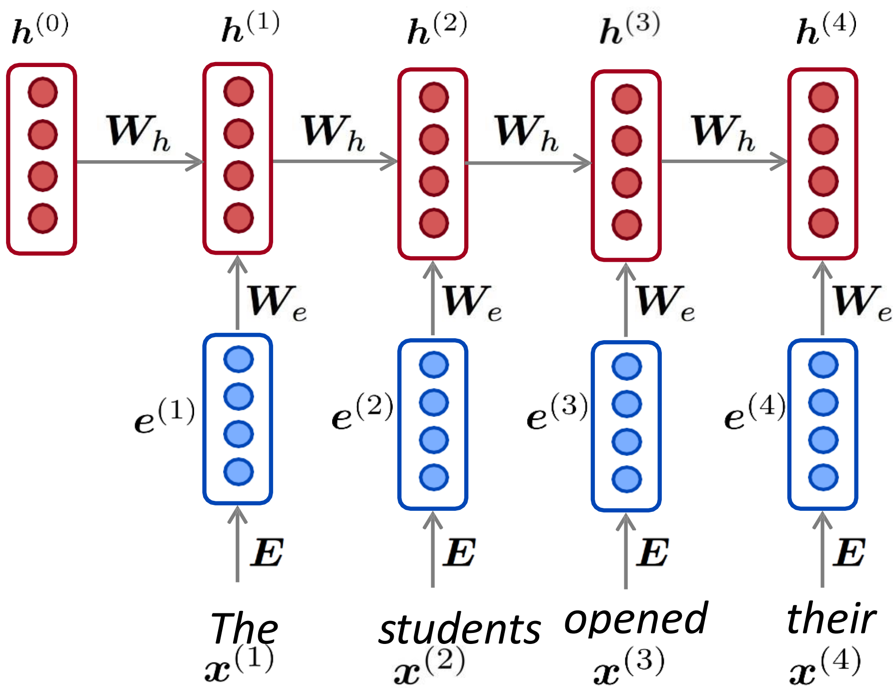

$$\mathbf{h}_t = f(U\mathbf{h}_{t-1} + W\mathbf{x}_t + \mathbf{b}),$$

$$\mathbf{y}_t = V\mathbf{h}_t,$$

# RNN语言模型


$h^{(0)}$
$h^{(1)}$
$h^{(2)}$
$h^{(3)}$
$h^{(4)}$
$W_h$
$W_h$
$W_h$
$W_h$
$W_e$
$W_e$
$W_e$
$W_e$
$e^{(1)}$
$e^{(2)}$
$e^{(3)}$
$e^{(4)}$
$E$
$E$
$E$
$E$
The
$x^{(1)}$
students
$x^{(2)}$
opened
$x^{(3)}$
their
$x^{(4)}$

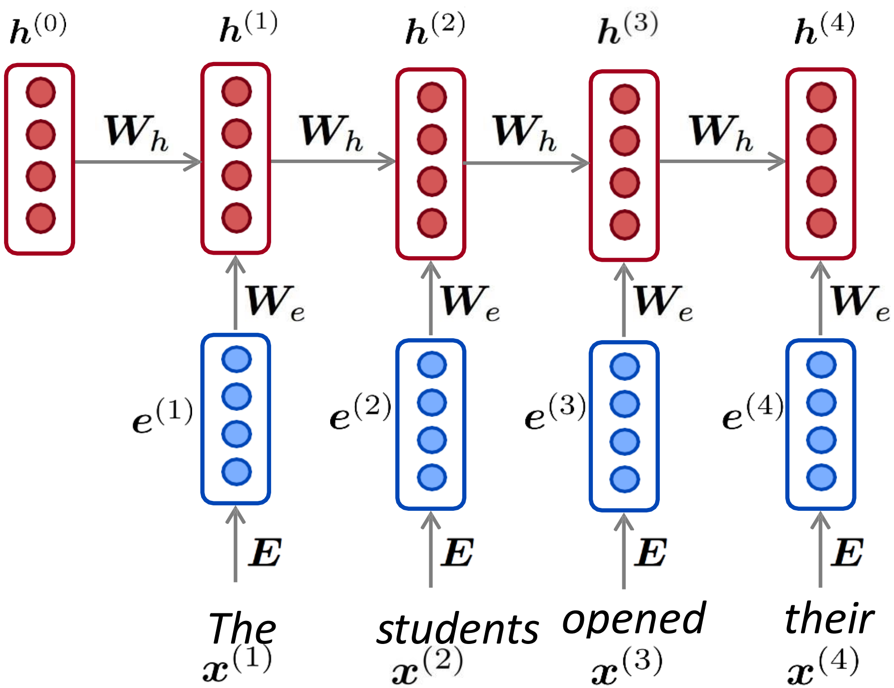
# RNN语言模型




**Level 1:** words / one-hot vectors

$$x^{(t)} \in \mathbb{R}^{|V|}$$

# RNN语言模型



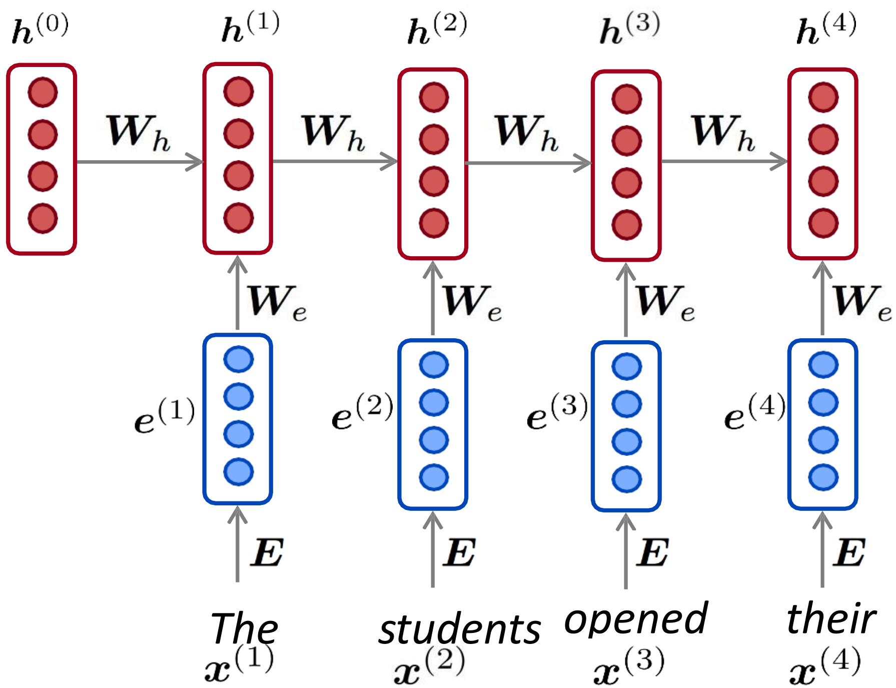

**Level 2:** word embeddings

$$e^{(t)} = Ex^{(t)}$$

**Level 1:** words / one-hot vectors

$$x^{(t)} \in \mathbb{R}^{|V|}$$

# RNN语言模型




**Level 3:** hidden states

$$h^{(t)} = \sigma\left(W_h h^{(t-1)} + W_e e^{(t)} + b_1\right)$$

$h^{(0)}$ is the initial hidden state

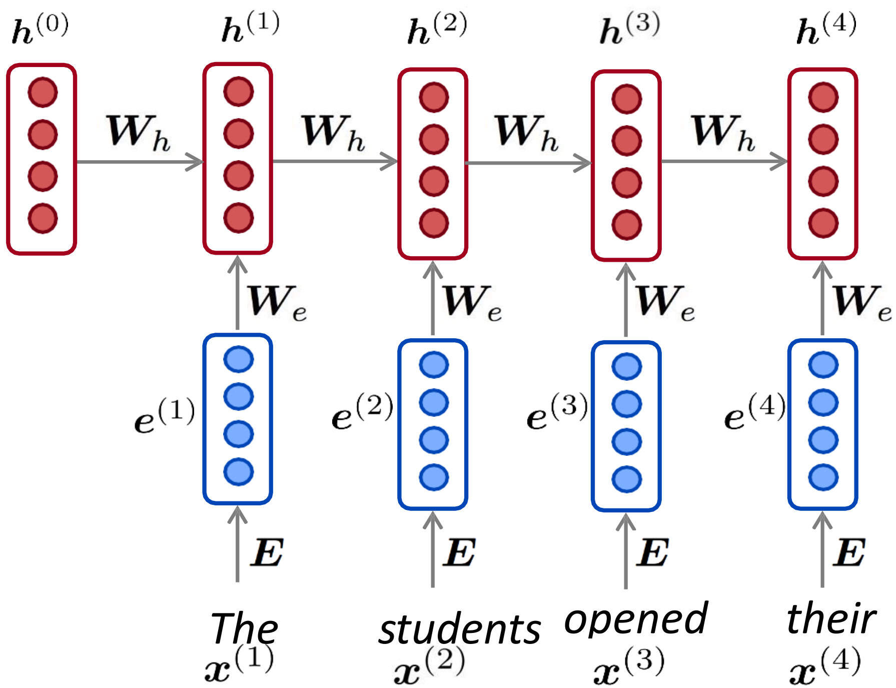
**Level 2:** word embeddings

$$e^{(t)} = E x^{(t)}$$

**Level 1:** words / one-hot vectors

$$x^{(t)} \in \mathbb{R}^{|V|}$$

# RNN语言模型




**Level 4:** output distribution

$$\hat{y}^{(t)} = \mathrm{softmax}\left(Uh^{(t)} + b_2\right) \in \mathbb{R}^{|V|}$$

**Level 3:** hidden states

$$h^{(t)} = \sigma\left(W_h h^{(t-1)} + W_e e^{(t)} + b_1\right)$$

$h^{(0)}$ is the initial hidden state

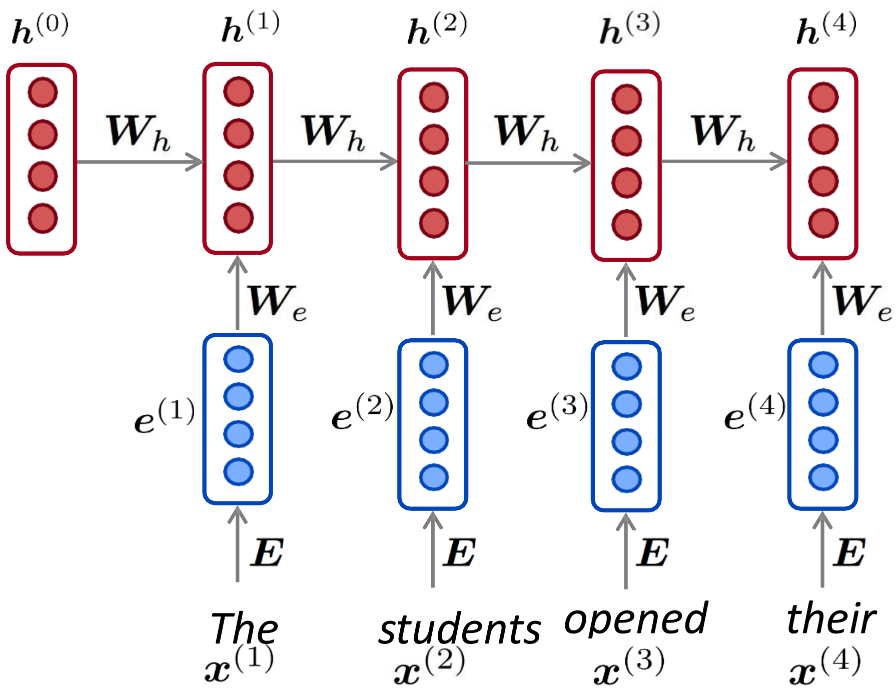
**Level 2:** word embeddings

$$e^{(t)} = E x^{(t)}$$

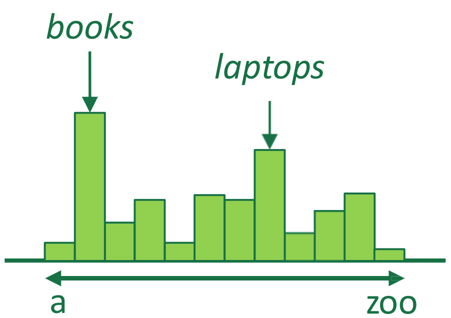
**Level 1:** words / one-hot vectors

$$x^{(t)} \in \mathbb{R}^{|V|}$$

# RNN语言模型




$$\hat{y}^{(4)} = P(x^{(5)}|\text{the students opened their})$$



# RNN Language Model

RNN **的优点**:

- 能够处理任意长度的输入;

# RNN Language Model

RNN **的优点**:

- 能够处理任意长度的输入;
- $t$时刻可以访问之前任意时刻的信息;

# RNN Language Model

RNN **的优点**:

- 能够处理任意长度的输入;
- $t$时刻可以访问之前任意时刻的信息;
- 对于较长的输入，模型的大小不变;

# RNN Language Model

RNN **的优点**:

- 能够处理任意长度的输入;
- $t$时刻可以访问之前任意时刻的信息;
- 对于较长的输入，模型的大小不变;
- 权重共享

# RNN Language Model

RNN **的不足**:

- 循环计算过程较慢;

# RNN Language Model

RNN **的不足**:

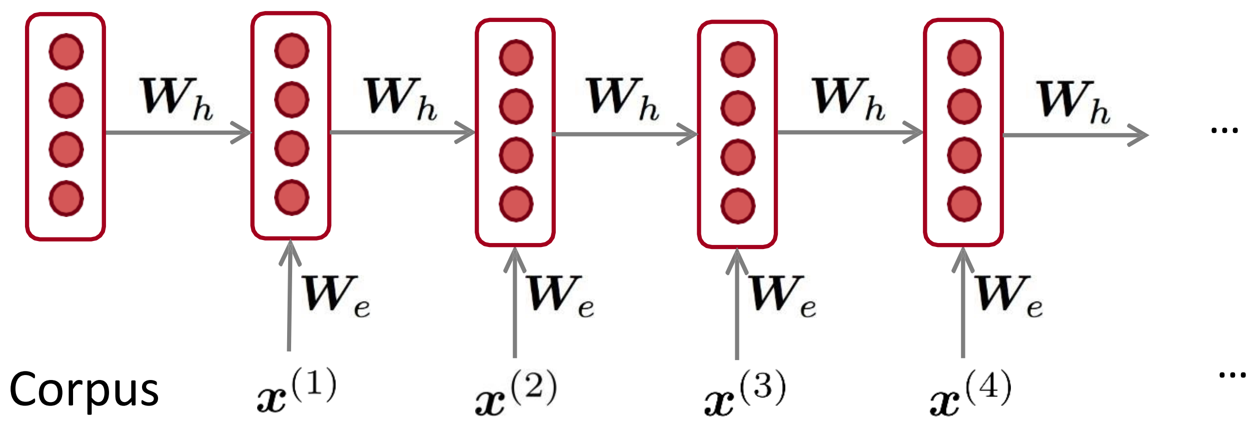
- 循环计算过程较慢;
- 实际运用中，很难访问距当前时刻较远的信息;

# RNN语言模型的训练

- Step 1: 输入文本数据集，每个文本表示为： $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$ ;

# RNN语言模型的训练

- Step 1: 输入文本数据集，每个文本表示为：$\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$；

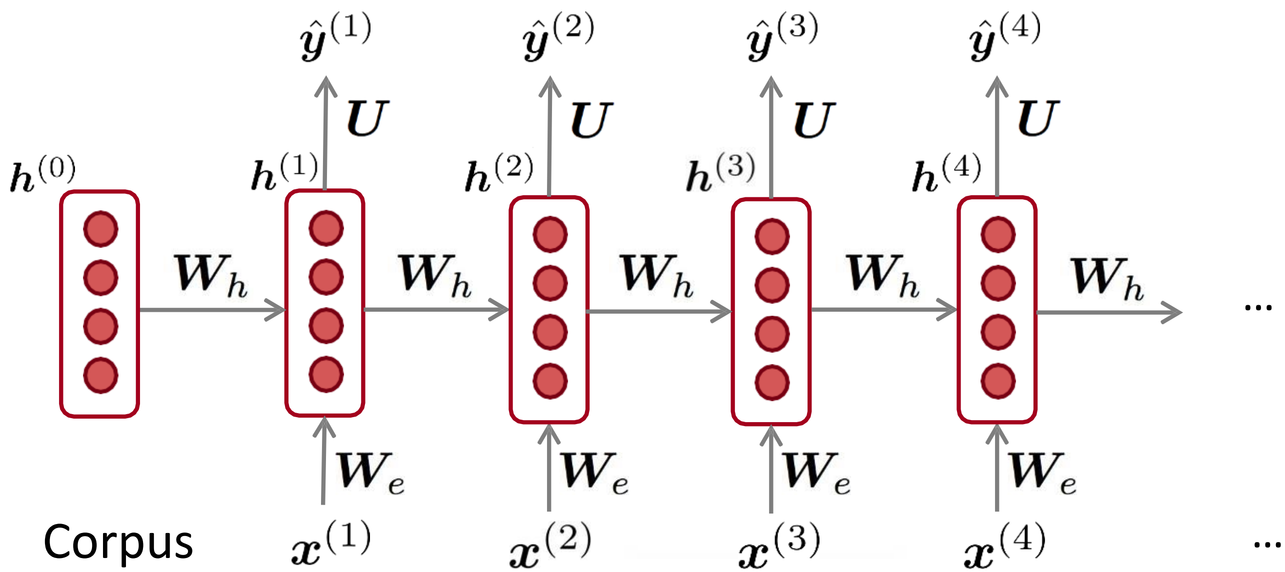


# RNN语言模型的训练

- Step 1: 输入文本数据集，每个文本表示为：$\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$；
- Step 2: 逐词把每条文本输入给RNN语言模型，计算每一时刻的输出概率 $\hat{\boldsymbol{y}}^{(t)}$；

# RNN语言模型的训练

- Step 1: 输入文本数据集，每个文本表示为：$\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$；
- Step 2: 逐词把每条文本输入给RNN语言模型，计算每一时刻的输出概率 $\hat{\boldsymbol{y}}^{(t)}$；



# RNN语言模型的训练

- Step 1: 输入文本数据集，每个文本表示为：$\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$；
- Step 2: 逐词把每条文本输入给RNN语言模型，计算每一时刻的输出概率 $\hat{\boldsymbol{y}}^{(t)}$；
- Step 3: 用交叉熵(cross-entropy)计算每一时刻的损失(loss)

# RNN语言模型的训练

- Step 1: 输入文本数据集，每个文本表示为： $\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$；
- Step 2: 逐词把每条文本输入给RNN语言模型，计算每一时刻的输出概率 $\hat{\boldsymbol{y}}^{(t)}$；
- Step 3:用交叉熵(cross-entropy)计算每一时刻的损失(loss)
  - 方法：使用时刻$t$的预测分布 $\hat{\boldsymbol{y}}^{(t)}$ 和下一个真实词 $\boldsymbol{x}^{(t+1)}$ 的表示 $\boldsymbol{y}^{(t)}$：

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

# RNN语言模型的训练

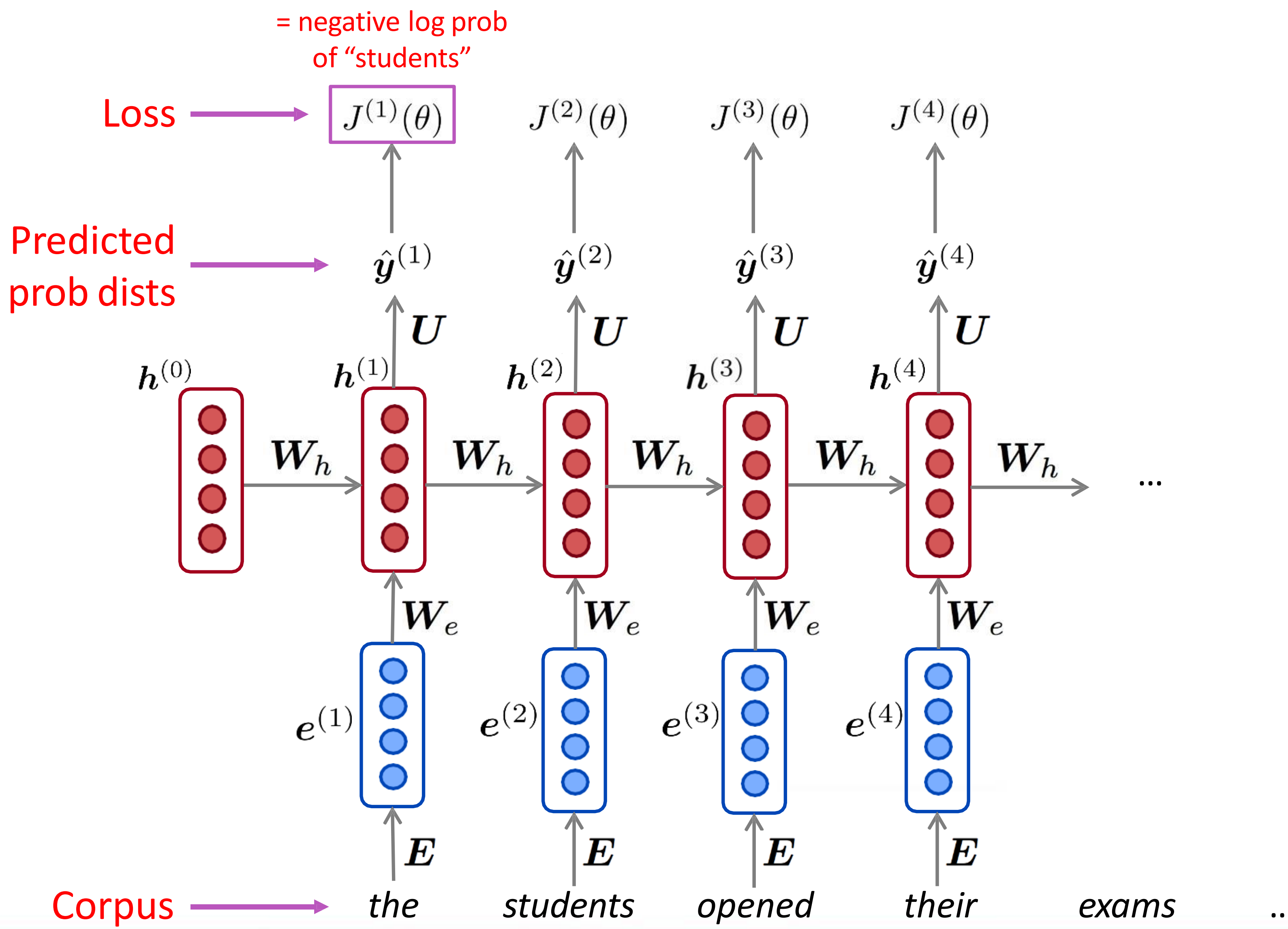
- Step 1: 输入文本数据集，每个文本表示为：$\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(T)}$；
- Step 2: 逐词把每条文本输入给RNN语言模型，计算每一时刻的输出概率 $\hat{\boldsymbol{y}}^{(t)}$；
- Step 3:用交叉熵(cross-entropy)计算每一时刻的损失(loss)
  - 方法：使用时刻$t$的预测分布 $\hat{\boldsymbol{y}}^{(t)}$ 和下一个真实词 $\boldsymbol{x}^{(t+1)}$ 的表示 $\boldsymbol{y}^{(t)}$：

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

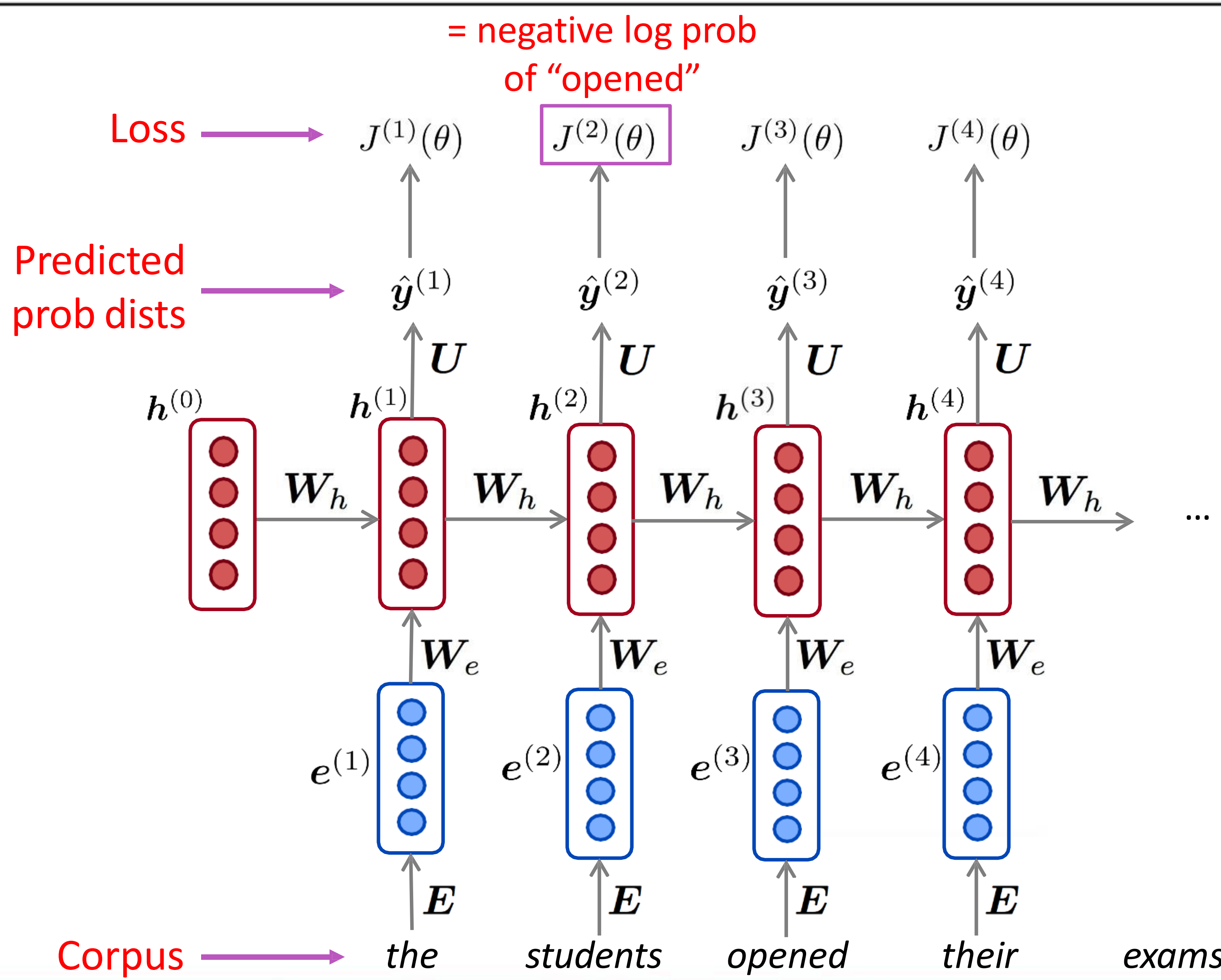
- Step 4: 计算所有文本的平均损失：

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

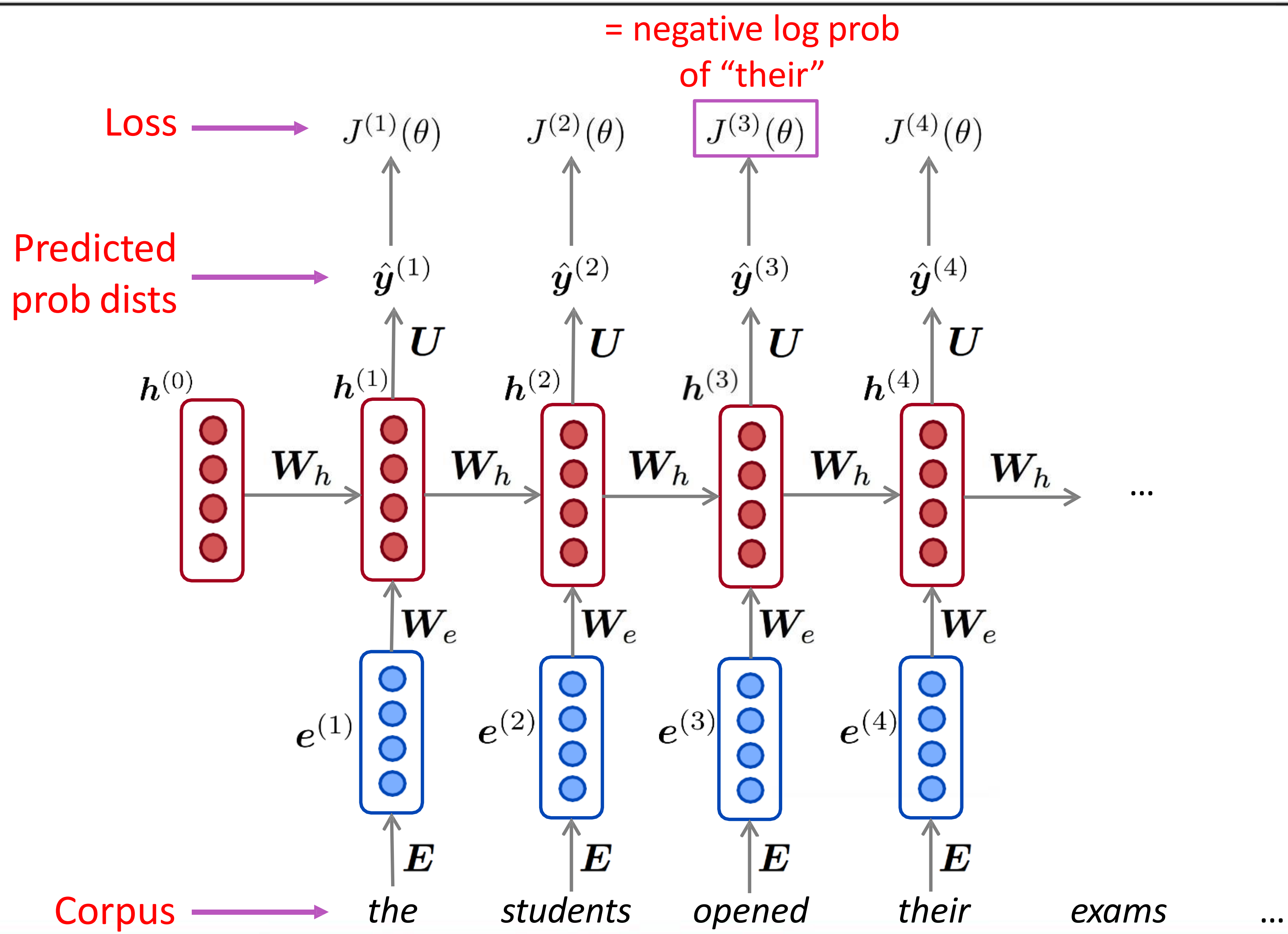
# RNN语言模型的训练


= negative log prob
of "students"
Loss
$J^{(1)}(\theta)$
$J^{(2)}(\theta)$
$J^{(3)}(\theta)$
$J^{(4)}(\theta)$
Predicted
prob dists
$\hat{\boldsymbol{y}}^{(1)}$
$\hat{\boldsymbol{y}}^{(2)}$
$\hat{\boldsymbol{y}}^{(3)}$
$\hat{\boldsymbol{y}}^{(4)}$
$\boldsymbol{U}$
$\boldsymbol{h}^{(0)}$
$\boldsymbol{h}^{(1)}$
$\boldsymbol{h}^{(2)}$
$\boldsymbol{h}^{(3)}$
$\boldsymbol{h}^{(4)}$
$\boldsymbol{W}_h$
...
$\boldsymbol{W}_e$
$\boldsymbol{e}^{(1)}$
$\boldsymbol{e}^{(2)}$
$\boldsymbol{e}^{(3)}$
$\boldsymbol{e}^{(4)}$
$\boldsymbol{E}$
Corpus
the
students
opened
their
exams
...

# RNN语言模型的训练

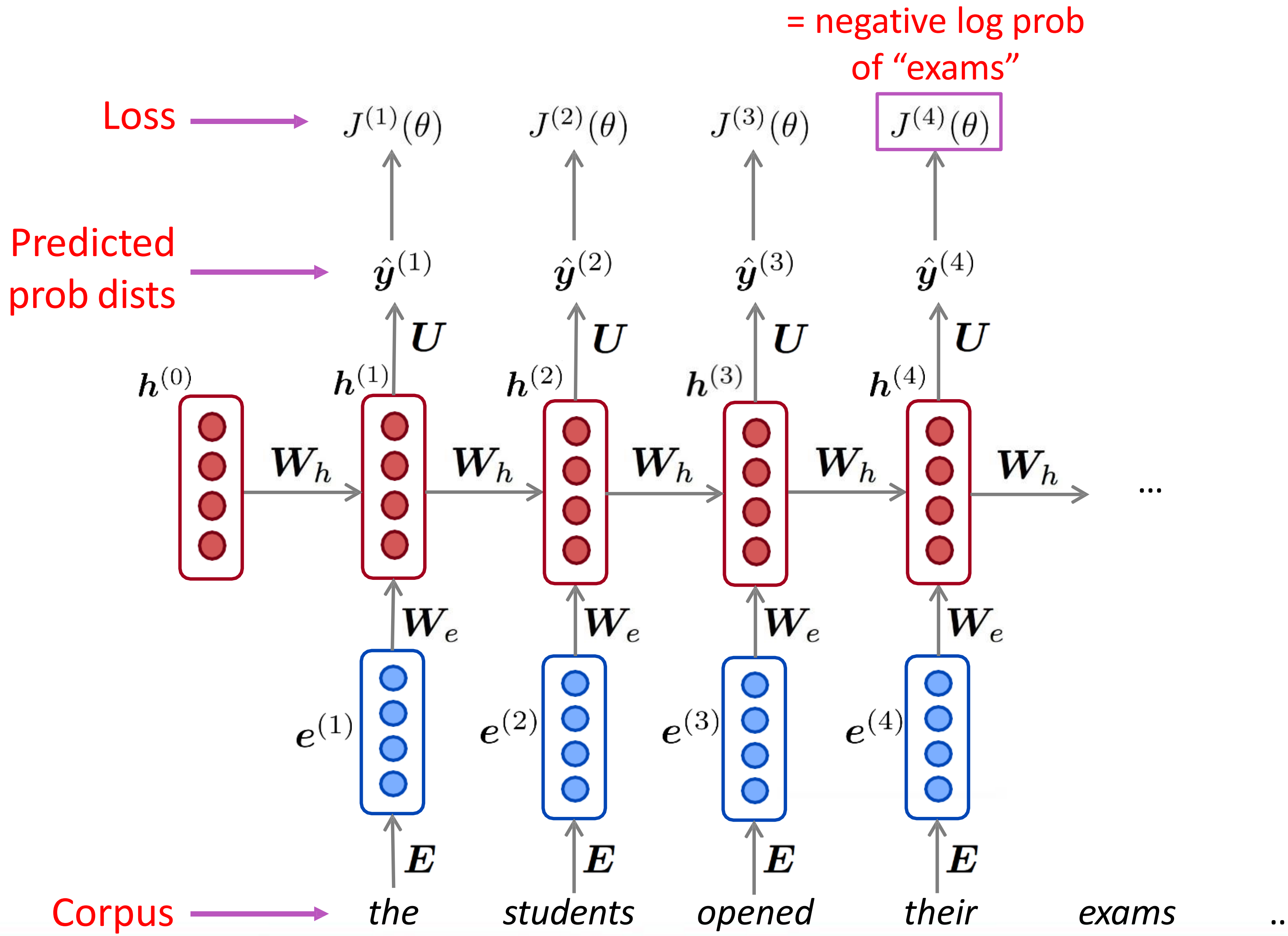

= negative log prob
of “opened”
Loss
$J^{(1)}(\theta)$
$J^{(2)}(\theta)$
$J^{(3)}(\theta)$
$J^{(4)}(\theta)$
Predicted
prob dists
$\hat{y}^{(1)}$
$\hat{y}^{(2)}$
$\hat{y}^{(3)}$
$\hat{y}^{(4)}$
$U$
$h^{(0)}$
$h^{(1)}$
$h^{(2)}$
$h^{(3)}$
$h^{(4)}$
$W_h$
...
$W_e$
$e^{(1)}$
$e^{(2)}$
$e^{(3)}$
$e^{(4)}$
$E$
Corpus
the
students
opened
their
exams
...

# RNN语言模型的训练



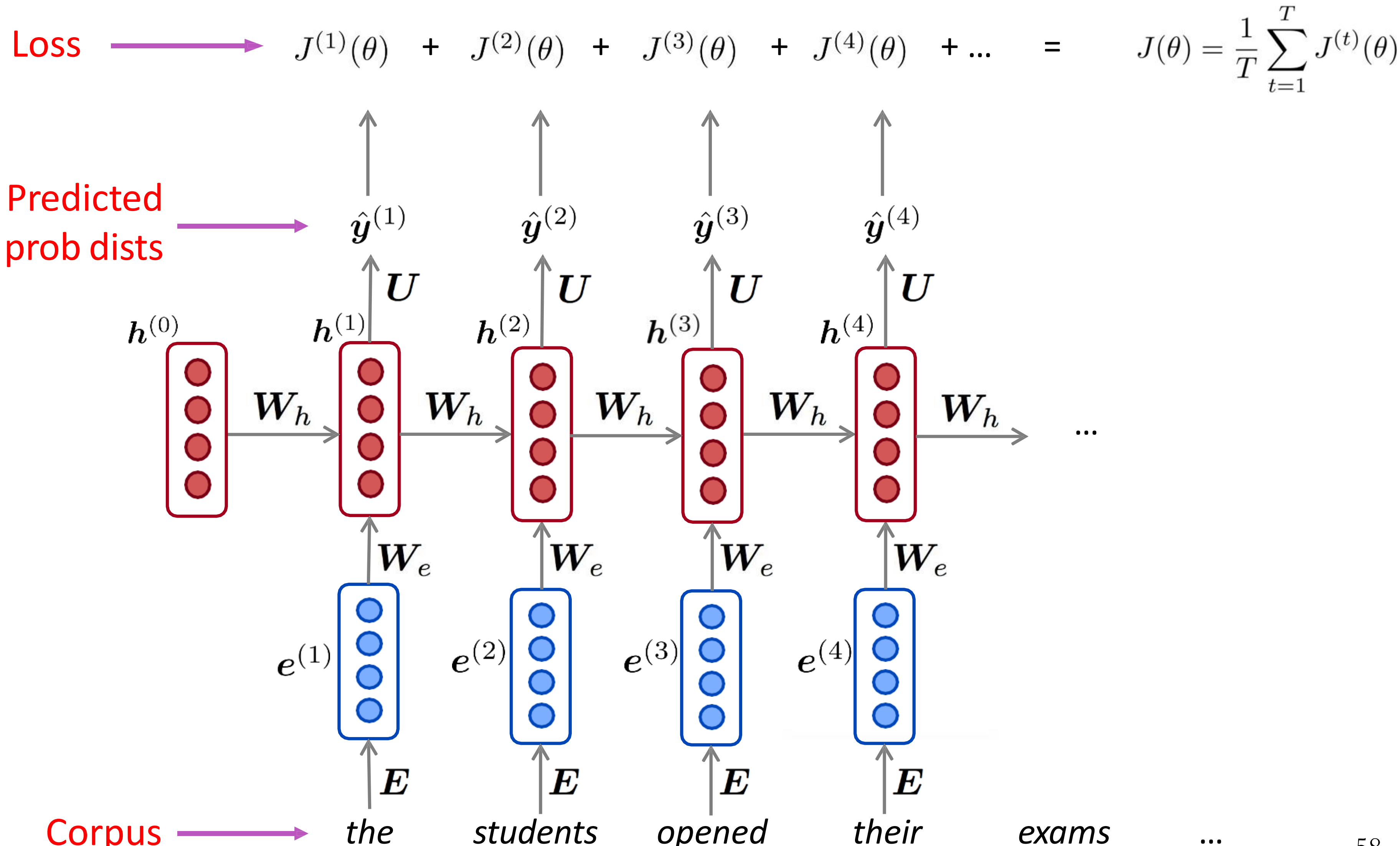
= negative log prob
of “their”
Loss
$J^{(1)}(\theta)$
$J^{(2)}(\theta)$
$J^{(3)}(\theta)$
$J^{(4)}(\theta)$
Predicted
prob dists
$\hat{y}^{(1)}$
$\hat{y}^{(2)}$
$\hat{y}^{(3)}$
$\hat{y}^{(4)}$
$U$
$h^{(0)}$
$h^{(1)}$
$h^{(2)}$
$h^{(3)}$
$h^{(4)}$
$W_h$
...
$W_e$
$e^{(1)}$
$e^{(2)}$
$e^{(3)}$
$e^{(4)}$
$E$
Corpus
the
students
opened
their
exams
...

# RNN语言模型的训练



= negative log prob
of "exams"
Loss
$J^{(1)}(\theta)$
$J^{(2)}(\theta)$
$J^{(3)}(\theta)$
$J^{(4)}(\theta)$
Predicted
prob dists
$\hat{y}^{(1)}$
$\hat{y}^{(2)}$
$\hat{y}^{(3)}$
$\hat{y}^{(4)}$
$U$
$h^{(0)}$
$h^{(1)}$
$h^{(2)}$
$h^{(3)}$
$h^{(4)}$
$W_h$
...
$W_e$
$e^{(1)}$
$e^{(2)}$
$e^{(3)}$
$e^{(4)}$
$E$
Corpus
the
students
opened
their
exams
...

# RNN语言模型的训练

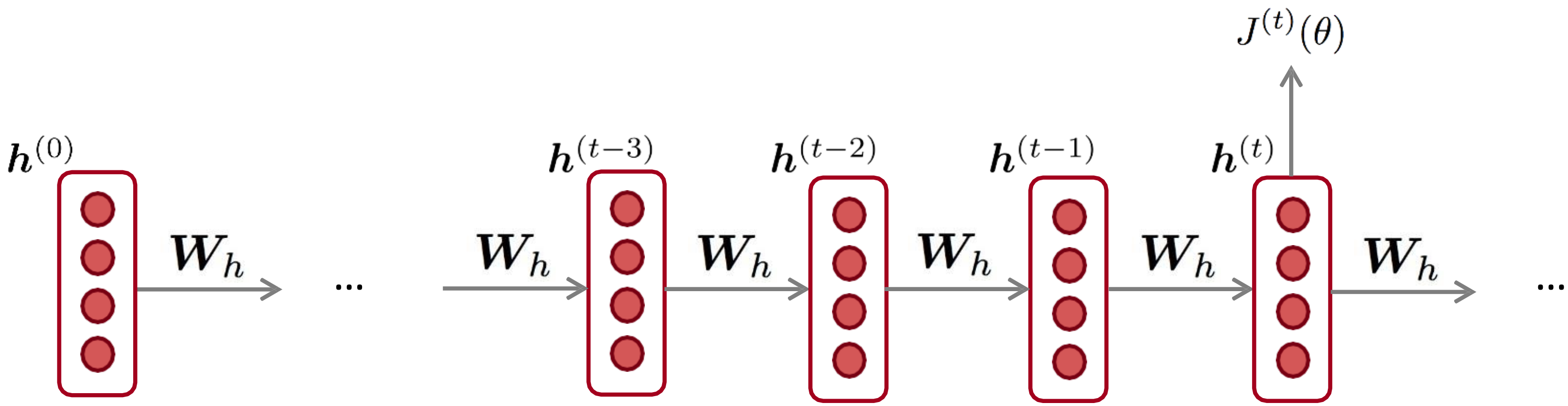


Loss
$J^{(1)}(\theta)$ + $J^{(2)}(\theta)$ + $J^{(3)}(\theta)$ + $J^{(4)}(\theta)$ + ... = $J(\theta) = \frac{1}{T}\sum_{t=1}^{T} J^{(t)}(\theta)$
Predicted prob dists
$\hat{y}^{(1)}$ $\hat{y}^{(2)}$ $\hat{y}^{(3)}$ $\hat{y}^{(4)}$
$U$
$h^{(0)}$ $h^{(1)}$ $h^{(2)}$ $h^{(3)}$ $h^{(4)}$
$W_h$
...
$W_e$
$e^{(1)}$ $e^{(2)}$ $e^{(3)}$ $e^{(4)}$
$E$
Corpus
the students opened their exams ...

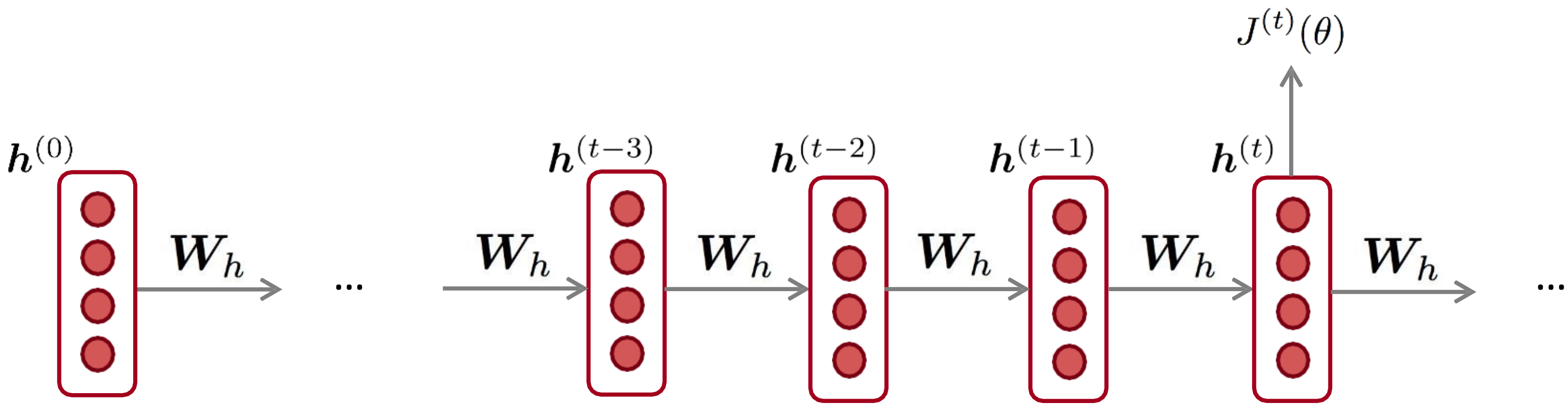
# RNN语言模型的训练

- 通常，使用随机梯度下降(Stochastic Gradient Descent) 计算一小部分随机数据(mini-batch)的损失;
- 根据得到的损失，计算梯度并更新参数;

# RNN的反向传播

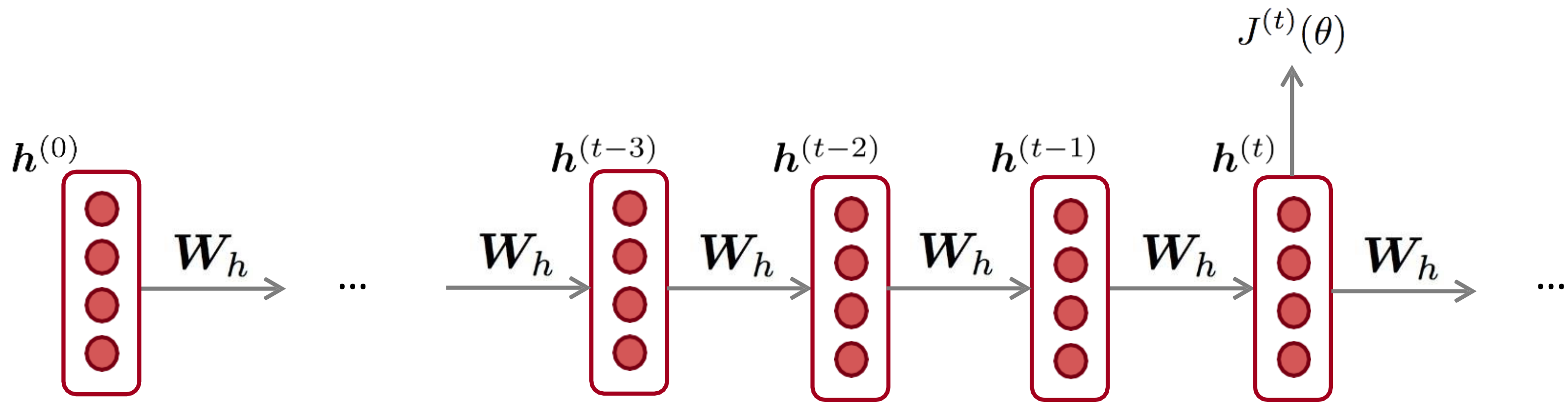

$J^{(t)}(\theta)$
$\boldsymbol{h}^{(0)}$
$\boldsymbol{h}^{(t-3)}$
$\boldsymbol{h}^{(t-2)}$
$\boldsymbol{h}^{(t-1)}$
$\boldsymbol{h}^{(t)}$
$\boldsymbol{W}_h$
…

# RNN的反向传播




**问题:** $J^{(t)}(\theta)$ 对权重矩阵 $\boldsymbol{W}_h$ 的偏导是什么?

# RNN的反向传播



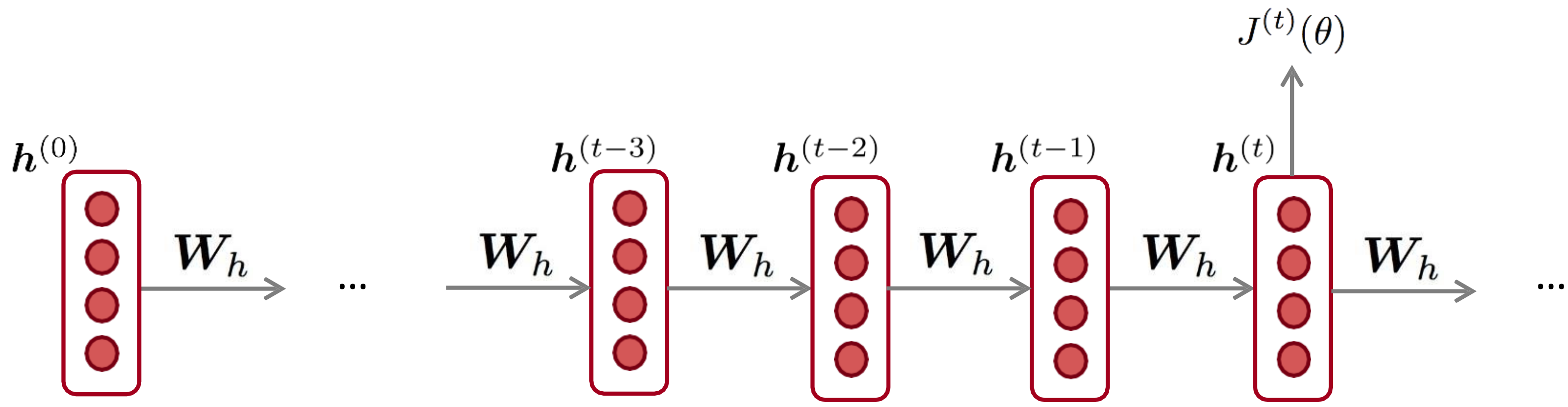

**问题:** $J^{(t)}(\theta)$ 对权重矩阵 $\boldsymbol{W}_h$ 的偏导是什么?

**答案:**

$$\frac{\partial J^{(t)}}{\partial \boldsymbol{W}_{\boldsymbol{h}}} = \sum_{i=1}^{t} \left. \frac{\partial J^{(t)}}{\partial \boldsymbol{W}_{\boldsymbol{h}}} \right|_{(i)}$$

对于重复出现的变量，它的梯度是每次出现的梯度之和。

# RNN的反向传播




**问题:** $J^{(t)}(\theta)$ 对权重矩阵 $W_h$ 的偏导是什么?

**答案:** $$\frac{\partial J^{(t)}}{\partial \boldsymbol{W_h}} = \sum_{i=1}^{t} \left.\frac{\partial J^{(t)}}{\partial \boldsymbol{W_h}}\right|_{(i)}$$

对于重复出现的变量，它的梯度是每次出现的梯度之和。

WHY?

# RNN的反向传播

## 多变量链式法则：

- Given a multivariable function $f(x, y)$, and two single variable functions $x(t)$ and $y(t)$, here's what the multivariable chain rule says:

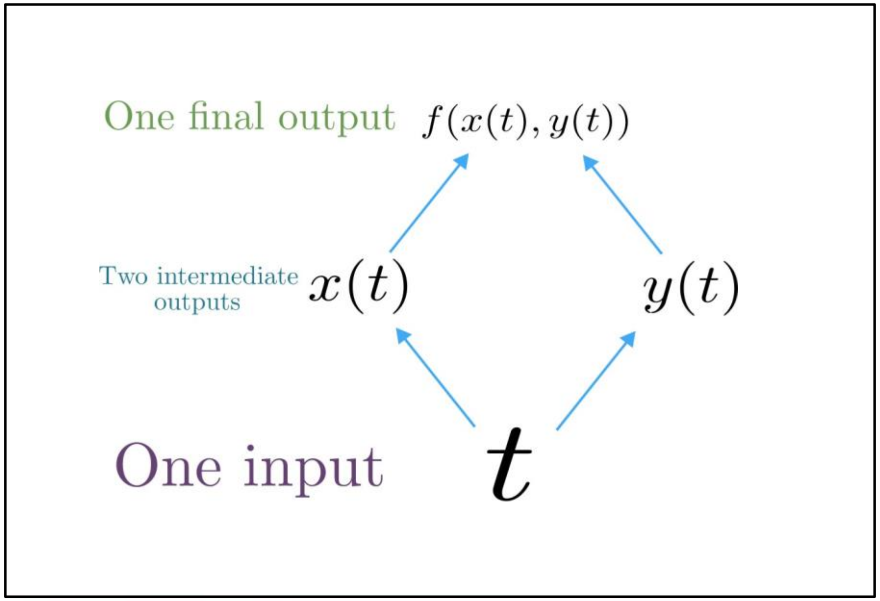
$$\underbrace{\frac{d}{dt} f(x(t), y(t))}_{\text{Derivative of composition function}} = \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt}$$

# RNN的反向传播

## 多变量链式法则：

- Given a multivariable function $f(x, y)$, and two single variable functions $x(t)$ and $y(t)$, here's what the multivariable chain rule says:

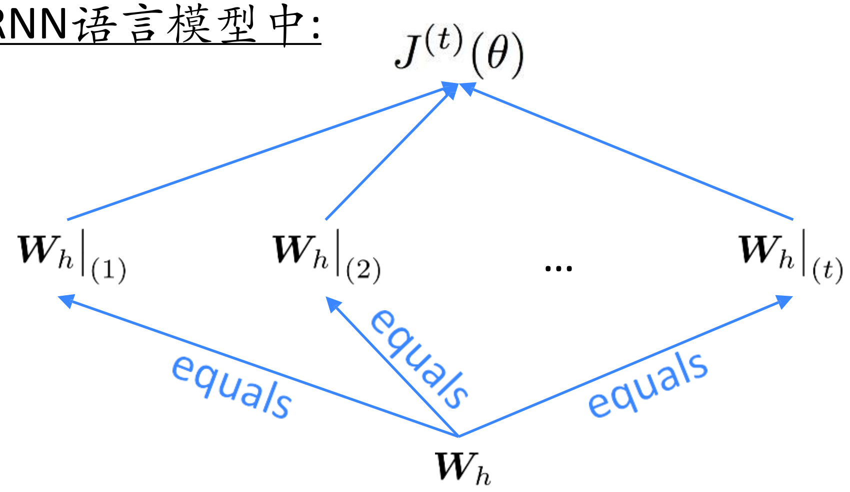
$$\underbrace{\frac{d}{dt} f(x(t), y(t))}_{\text{Derivative of composition function}} = \frac{\partial f}{\partial x}\frac{dx}{dt} + \frac{\partial f}{\partial y}\frac{dy}{dt}$$



# RNN的反向传播

## 多变量链式法则：

RNN语言模型中:




运用多变量链式法则:

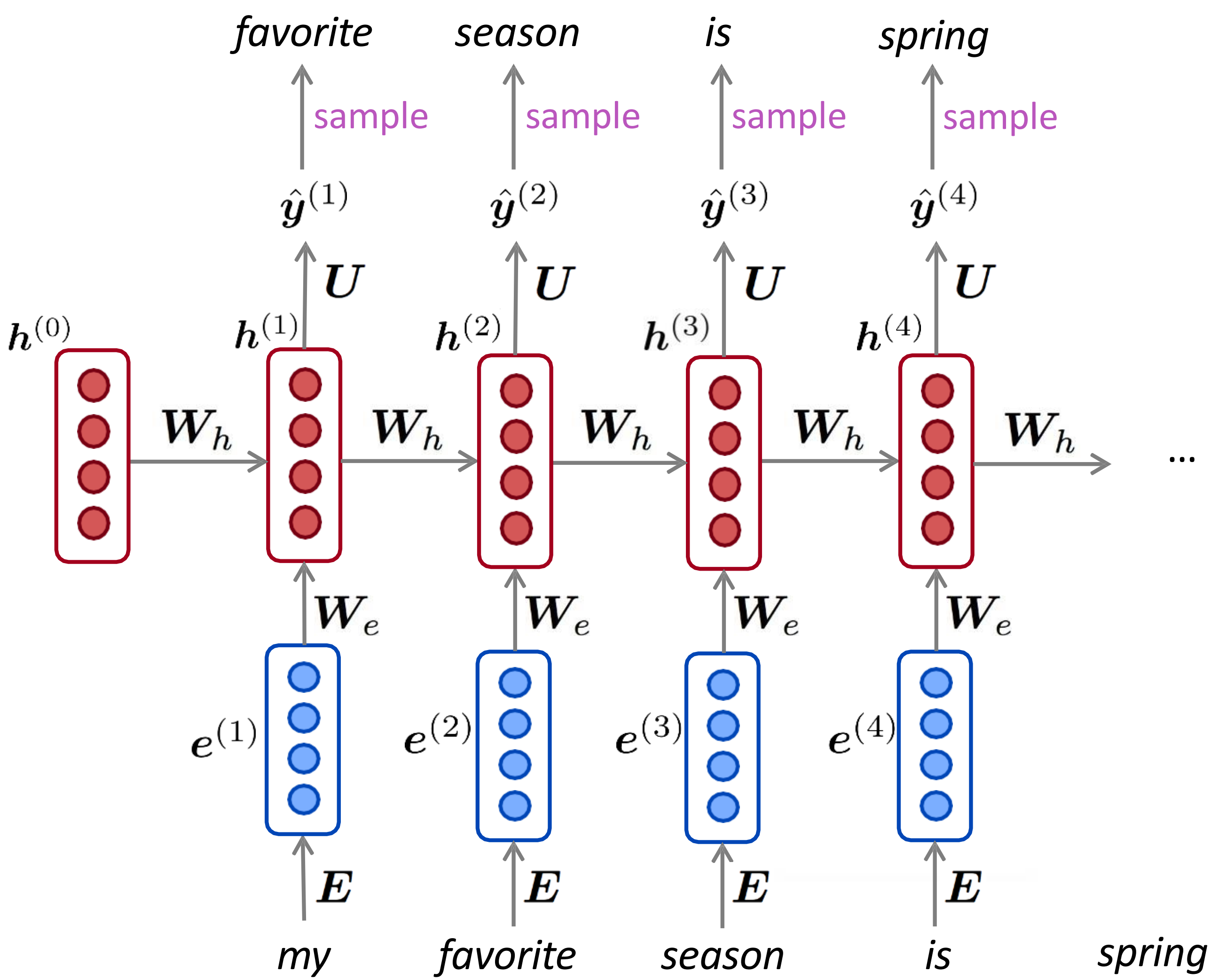
$$\frac{\partial J^{(t)}}{\partial \boldsymbol{W}_h} = \sum_{i=1}^{t} \frac{\partial J^{(t)}}{\partial \boldsymbol{W}_h}\bigg|_{(i)} \underbrace{\boxed{\frac{\partial \boldsymbol{W}_h\big|_{(i)}}{\partial \boldsymbol{W}_h}}}_{=1}$$

$$= \sum_{i=1}^{t} \frac{\partial J^{(t)}}{\partial \boldsymbol{W}_h}\bigg|_{(i)}$$

# RNN语言模型用于文本生成

- 和n-gram 语言模型一样，RNN语言模型可以通过循环采样生成文本：

# RNN语言模型用于文本生成



# RNN语言模型用于文本生成

- 可以使用RNN语言模型在各类文本数据上训练，然后生成相应风格的文本；
- 例如，在奥巴马的演说文本上：

*The United States will step up to the cost of a new challenges of the American people that will share the fact that we created the problem. They were attacked and so that they have to say that all the task of the final days of war that I will not be able to get this done. The promise of the men and women who were still going*

# RNN语言模型用于文本生成

- 可以使用RNN语言模型在各类文本数据上训练，然后生成相应风格的文本；
- 例如，在哈利波特小说上：

"Sorry," Harry shouted, panicking—"I'll leave those brooms in London, are they?"

"No idea," said Nearly Headless Nick, casting low close by Cedric, carrying the last bit of treacle Charms, from Harry's shoulder, and to answer him the common room perched upon it, four arms held a shining knob from when the spider hadn't felt it seemed. He reached the teams too.

# RNN语言模型用于文本生成

- 可以使用RNN语言模型在各类文本数据上训练，然后生成相应风格的文本；
- 例如，在菜谱上：

```
     Title: CHOCOLATE RANCH BARBECUE
Categories: Game, Casseroles, Cookies, Cookies
     Yield: 6 Servings

     2 tb Parmesan cheese -- chopped
     1 c  Coconut milk
     3    Eggs, beaten

 Place each pasta over layers of lumps. Shape mixture into the moderate oven and simmer
 until firm. Serve hot in bodied fresh, mustard, orange and cheese.

 Combine the cheese and salt together the dough in a large skillet; add the ingredients
 and stir in the chocolate and pepper.
```

# 语言模型的评估

❑ 评估语言模型的标准方法是计算困惑度(perplexity)

$$\text{perplexity} = \prod_{t=1}^{T} \underbrace{\left( \frac{1}{P_{\text{LM}}(\boldsymbol{x}^{(t+1)} | \, \boldsymbol{x}^{(t)}, \ldots, \boldsymbol{x}^{(1)})} \right)}_{\text{Inverse probability of corpus, according to Language Model}}^{1/T}$$

Normalized by number of words

Inverse probability of corpus, according to Language Model

# 语言模型的评估

## RNN能够显著提升模型的困惑度：

*n*-gram model → Interpolated Kneser-Ney 5-gram

Increasingly complex RNNs ↓

| Model | Perplexity |
|---|---|
| Interpolated Kneser-Ney 5-gram (Chelba et al., 2013) | 67.6 |
| RNN-1024 + MaxEnt 9-gram (Chelba et al., 2013) | 51.3 |
| RNN-2048 + BlackOut sampling (Ji et al., 2015) | 68.3 |
| Sparse Non-negative Matrix factorization (Shazeer et al., 2015) | 52.9 |
| LSTM-2048 (Jozefowicz et al., 2016) | 43.7 |
| 2-layer LSTM-8192 (Jozefowicz et al., 2016) | 30 |
| **Ours small** (LSTM-2048) | 43.9 |
| **Ours large** (2-layer LSTM-2048) | 39.8 |

Perplexity improves (lower is better)

**Source:** https://research.fb.com/building-an-efficient-neural-language-model-over-a-billion-words/

# 注意！

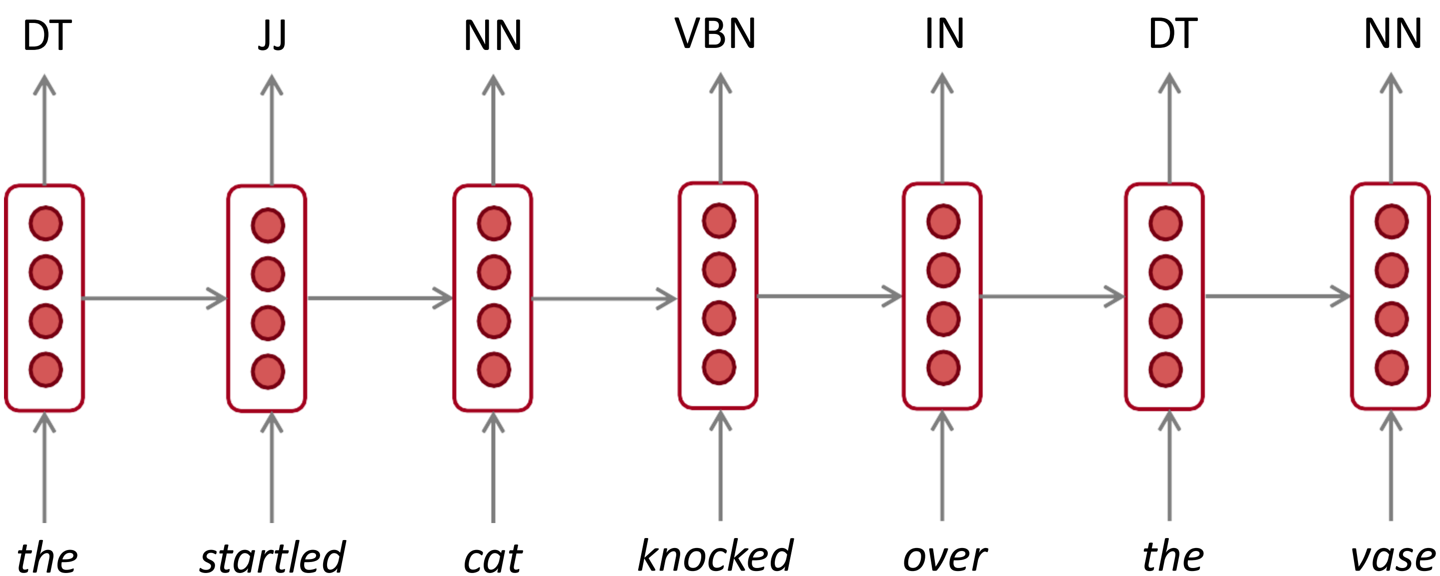
- **语言模型**: 预测下一个词的系统
- **RNN**: 一种神经网络:
  - 输入是任意长度的文本;
  - 共享权重;
  - 每一步都可以产生输出;
- Recurrent Neural Network ≠ Language Model
- RNN可以用来构建语言模型，但RNN还有更多用途;

# 注意!

- RNN可以用于做序列标注，如词性标注

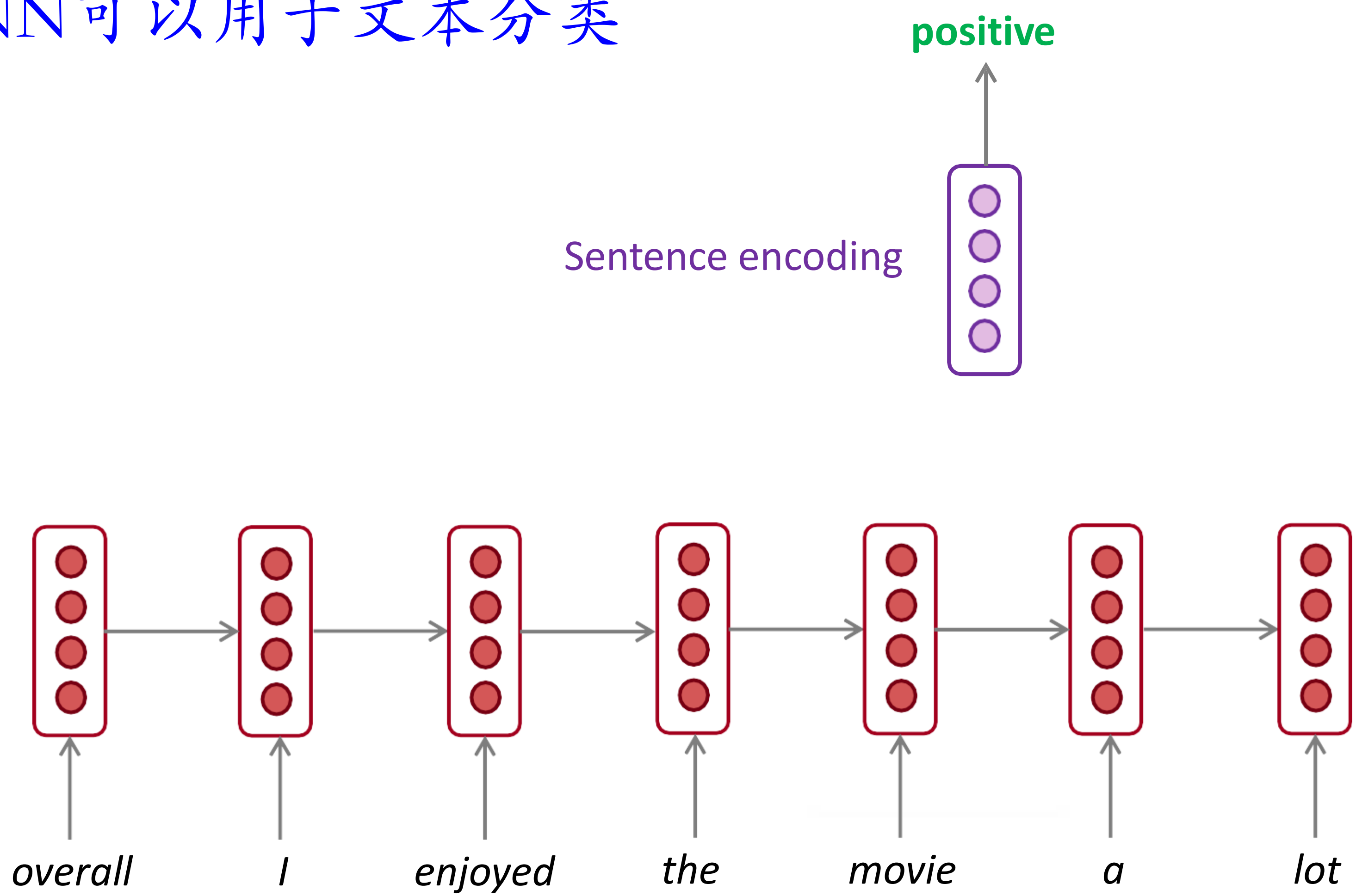
# 注意!

- RNN可以用于做序列标注，如词性标注



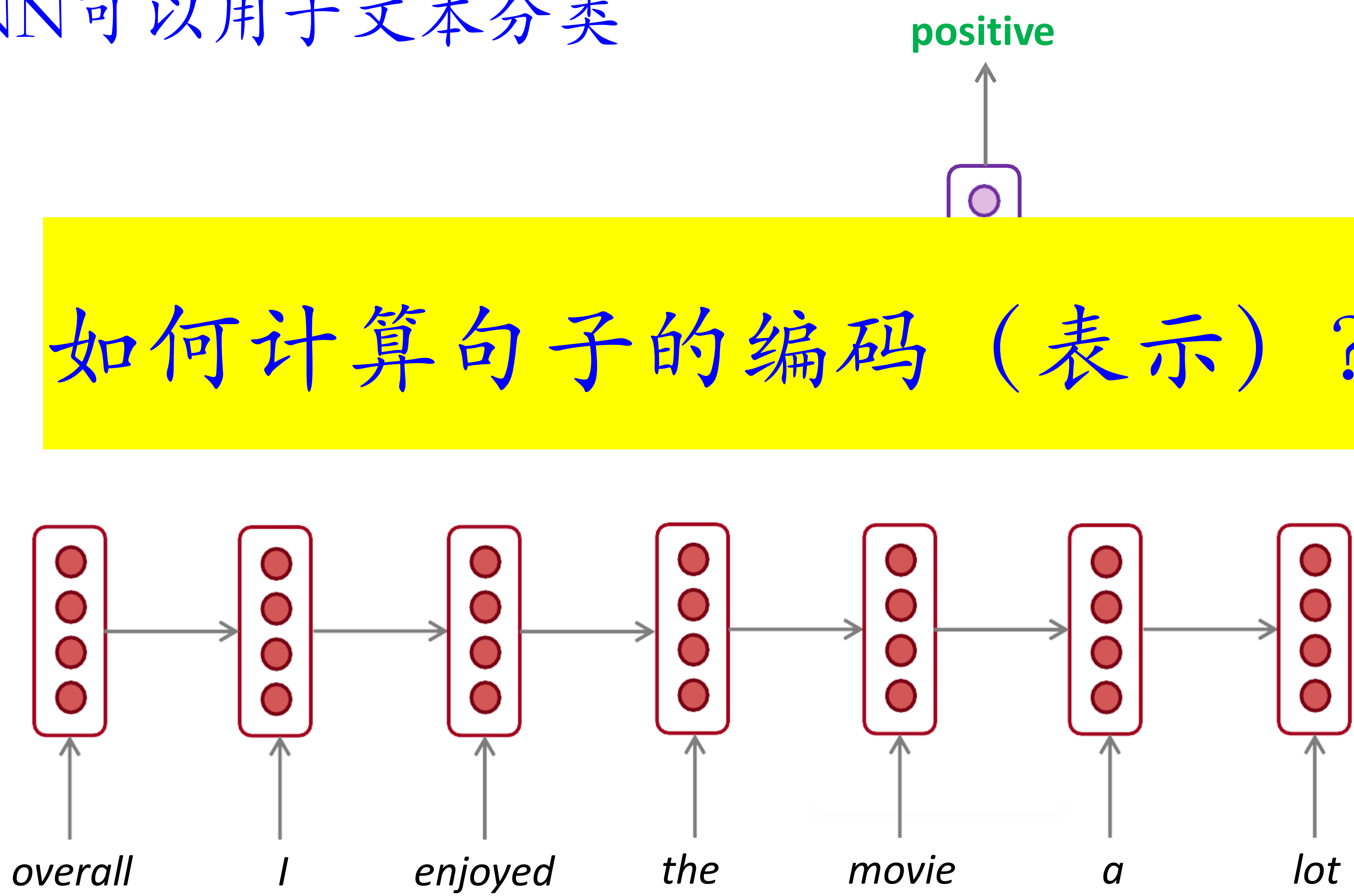
# 注意!

- RNN可以用于文本分类



# 注意！

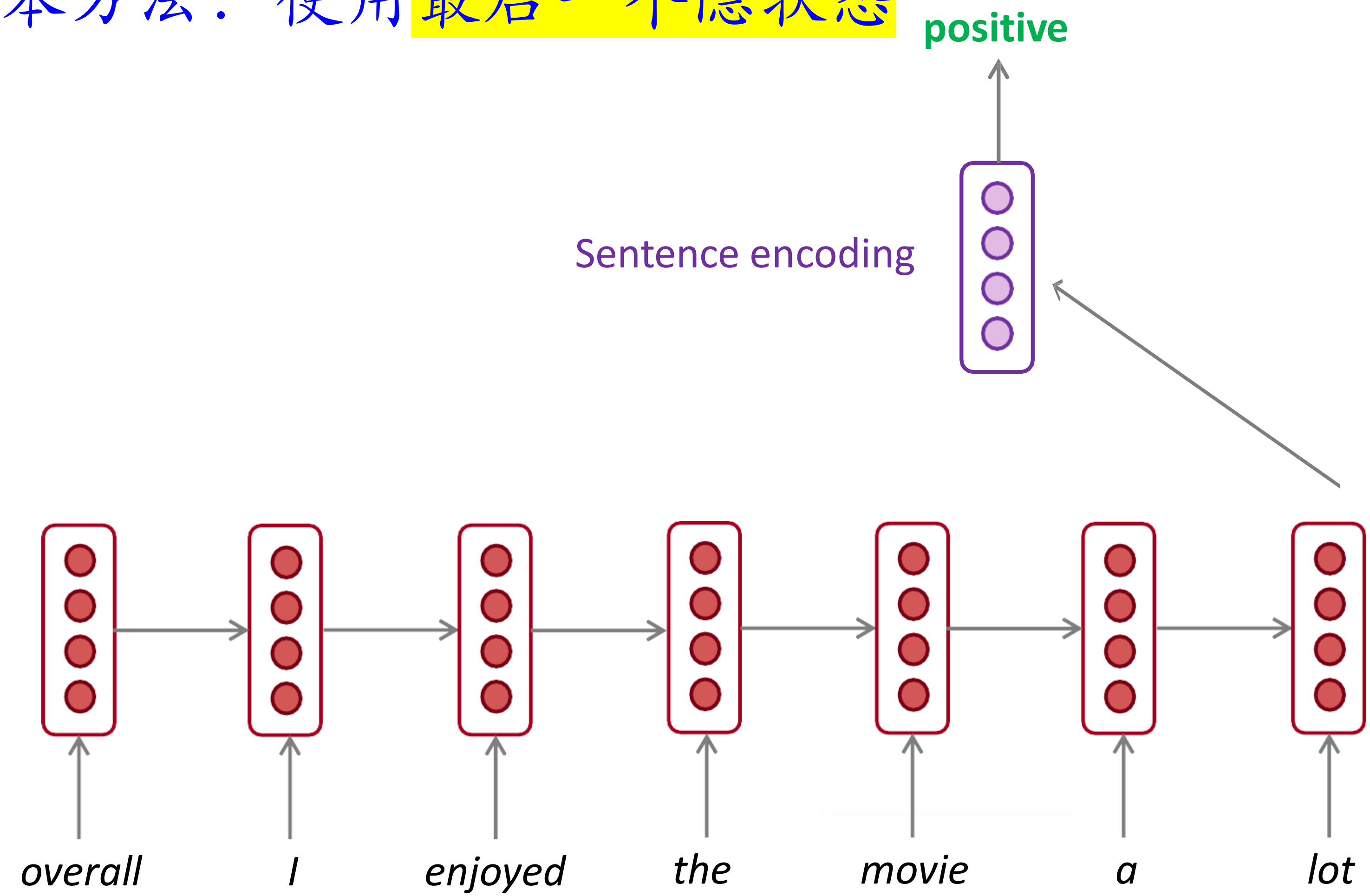
- RNN可以用于文本分类




如何计算句子的编码（表示）？

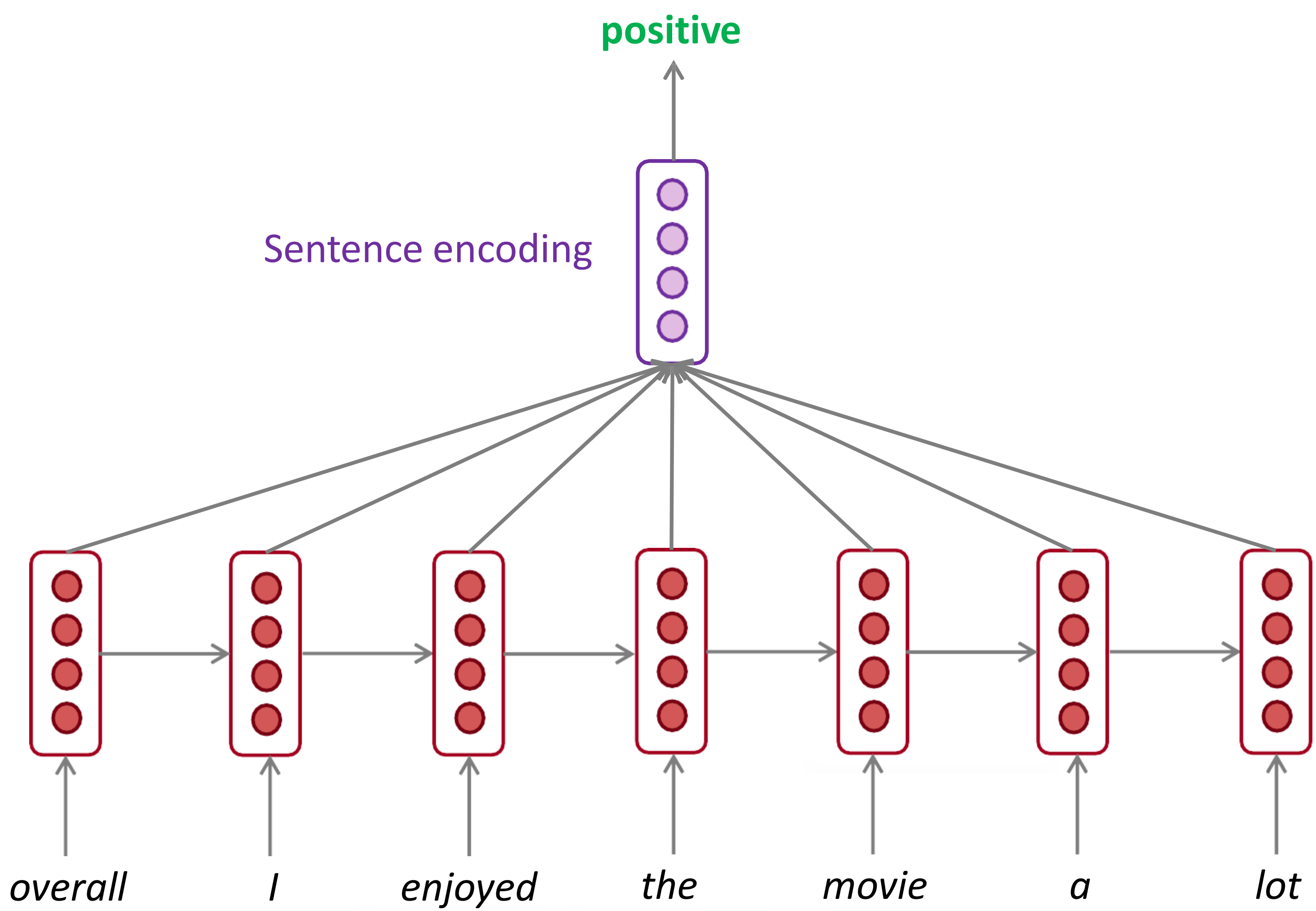
# 注意!

- 基本方法：使用最后一个隐状态



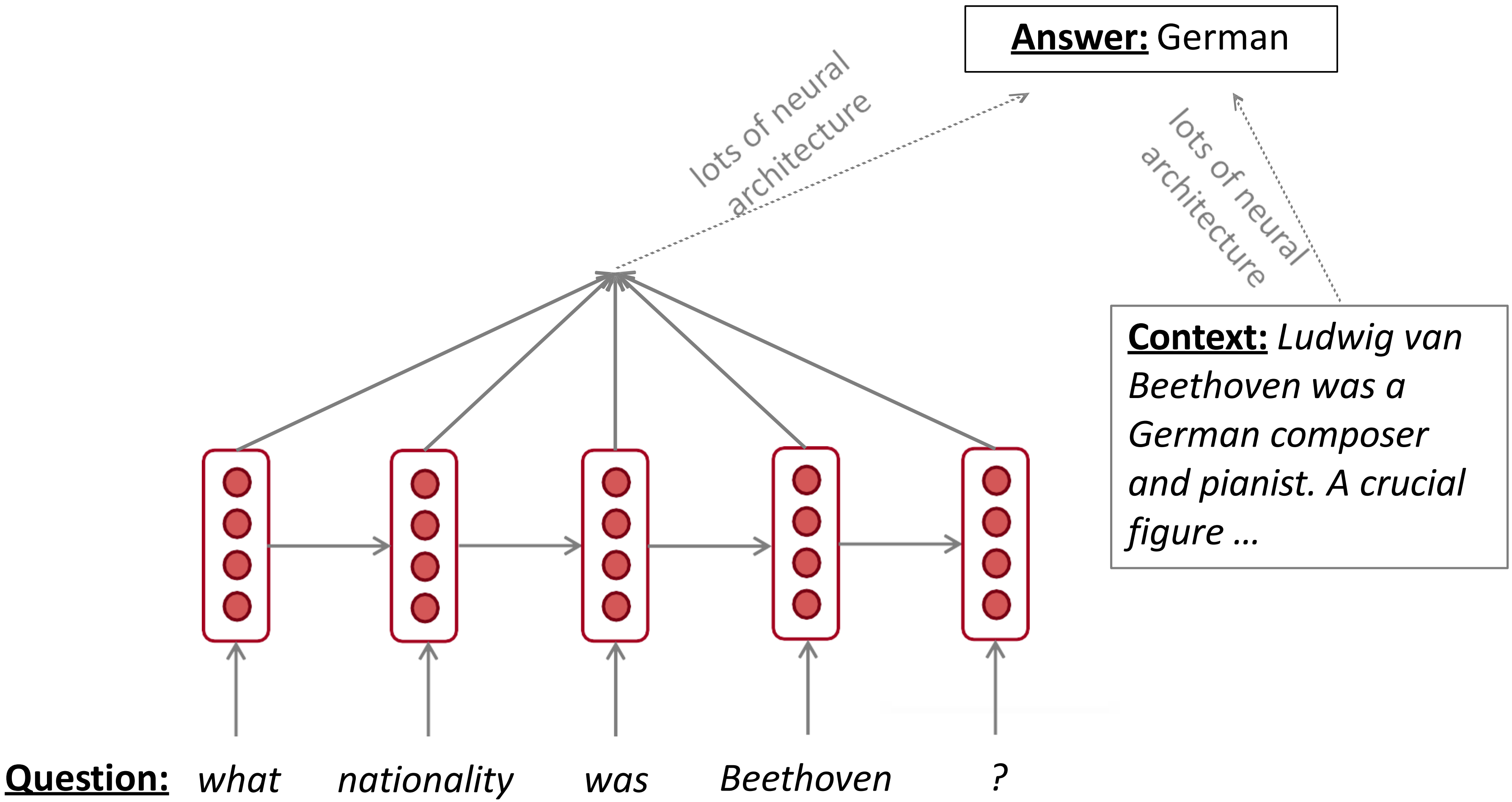
# 注意！

- 更好方法：取所有隐状态的最大值或者平均值



# 注意！

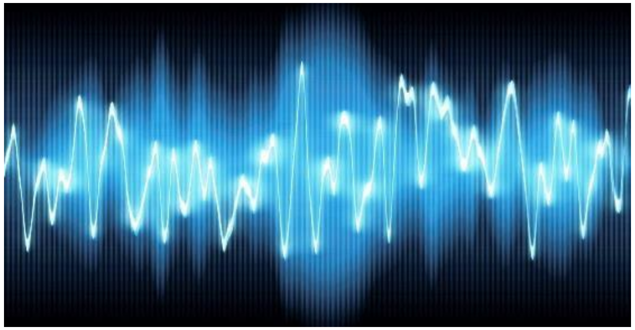
- RNN还可以用于问答系统和机器翻译等任务



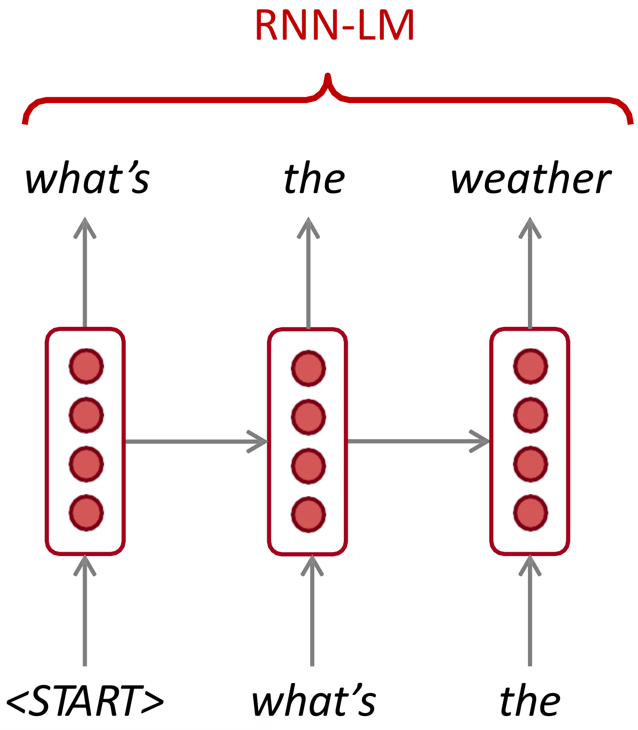
# 注意!

- RNN还可以用于语音识别

Input (audio)



conditioning



# 注意!

- 常见的RNN模型
  - LSTM
  - GRU
  - Bidirectional
  - Multi-layer

# Thank you!

权小军 中山大学数据科学与计算机学院