

2024年度暑期强化课程

# 循环神经网络语言模型

**授课人：曹亚男**



中国科学院 信息工程研究所  
INSTITUTE OF INFORMATION ENGINEERING, CAS

## 3. 循环神经网络

---

3.1

RNN语言模型

3.2

RNN激活单元

3.3

双向循环神经网络

3.4

Seq2seq框架

## 3. 循环神经网络

---

3.1

RNN语言模型

3.2

RNN激活单元

3.3

双向循环神经网络

3.4

Seq2seq框架

# 知识回顾：统计语言模型

- 统计语言模型的作用是为一个长度为 $m$ 的字符串确定一个概率分布 $P(w_1, w_2, \dots, w_m)$ ，表示其存在的可能性。其中 $w_1$ 到 $w_m$ 依次表示这段文本中的各个词

$$\begin{aligned} P(w_1, w_2, \dots, w_m) &= P(w_1)P(w_2 | w_1)P(w_3 | w_1, w_2) \\ &\quad \dots P(w_i | w_1, w_2, \dots, w_{i-1}) \dots P(w_m | w_1, w_2, \dots, w_{m-1}) \\ &= \prod_{i=1}^m P(w_i | w_1, w_2, \dots, w_{i-1}) \end{aligned}$$

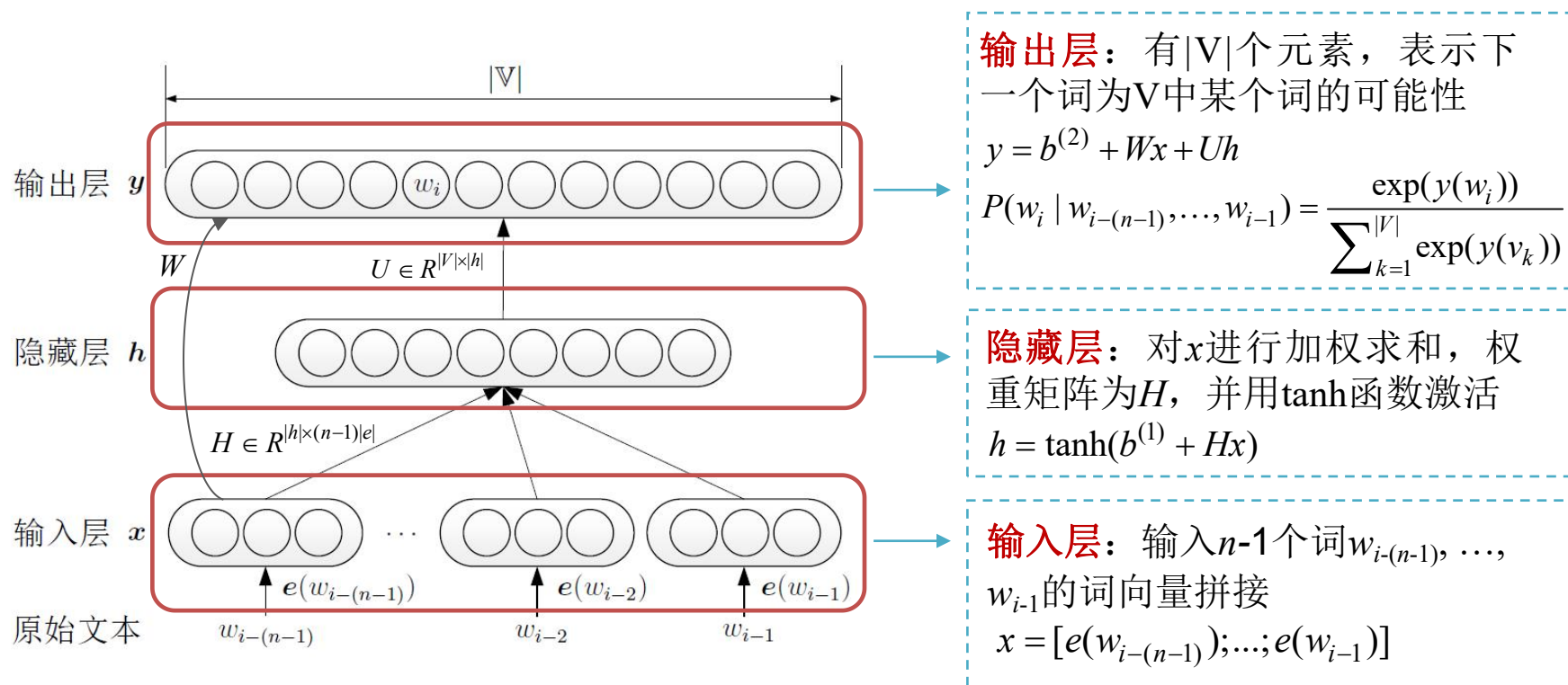
- 由于 $P(w_i | w_1, w_2, \dots, w_{i-1})$ 很难估算，通常使用 $n$ -gram模型来简化

$$P(w_i | w_1, w_2, \dots, w_{i-1}) \approx P(w_i | w_{i-(n-1)}, \dots, w_{i-1})$$

NNLM

# 知识回顾：神经网络语言模型 (NNLM)

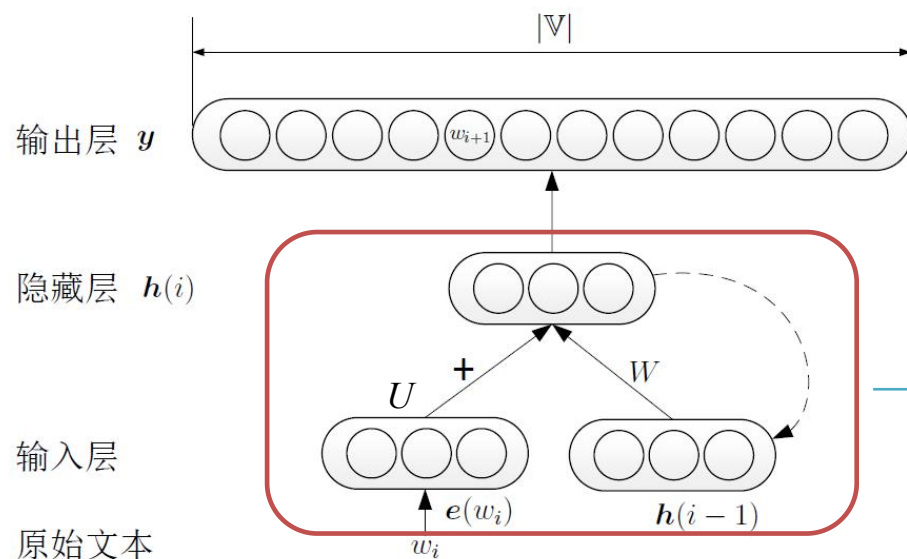
- 采用神经网络结构对 $n$ -gram模型进行建模，估算 $P(w_i | w_{i-(n-1)}, \dots, w_{i-1})$ 的值。输入为条件部分的整个词序列，输出为目标词 $w_i$ 的分布



神经网络语言模型 (NNLM) 结构图

# 循环神经网络语言模型 (RNNLM)

- 不同于NNLM是对n-gram建模，RNNLM直接对 $P(w_i|w_1, w_2, \dots, w_{i-1})$ 进行建模。RNNLM可以利用 $w_i$ 所有上文信息，预测下一个词



循环神经网络语言模型 (RNNLM) 结构图

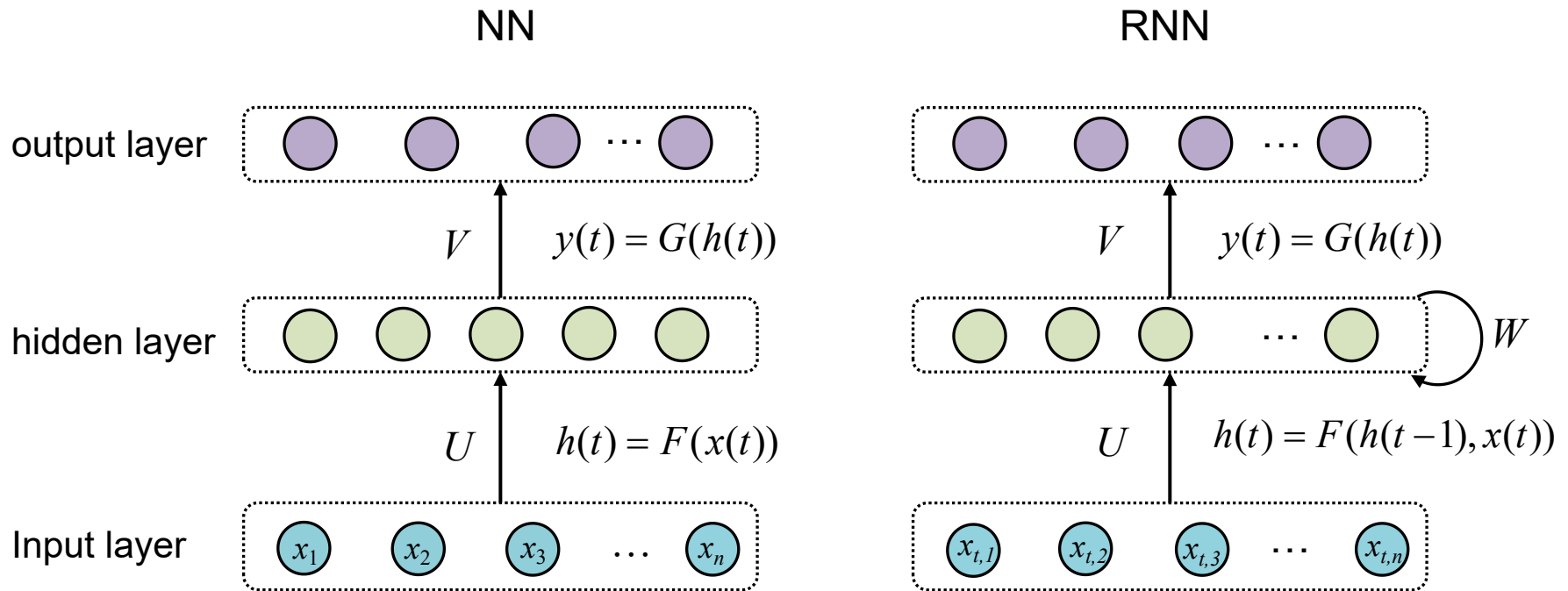
## 独特的隐藏层算法:

$$h(i) = f(Ux(i) + Wh(i-1))$$

- $h(i)$ 表示文本中第 $i$ 个词对应的隐藏层，由当前词的词向量及上一个词对应的隐藏层结合得到
- 隐藏层初始状态为 $h(0)$ ，模型逐个读入语料中的词 $w_1, w_2, \dots$ ，隐藏层不断更新为 $h(1), h(2), \dots$

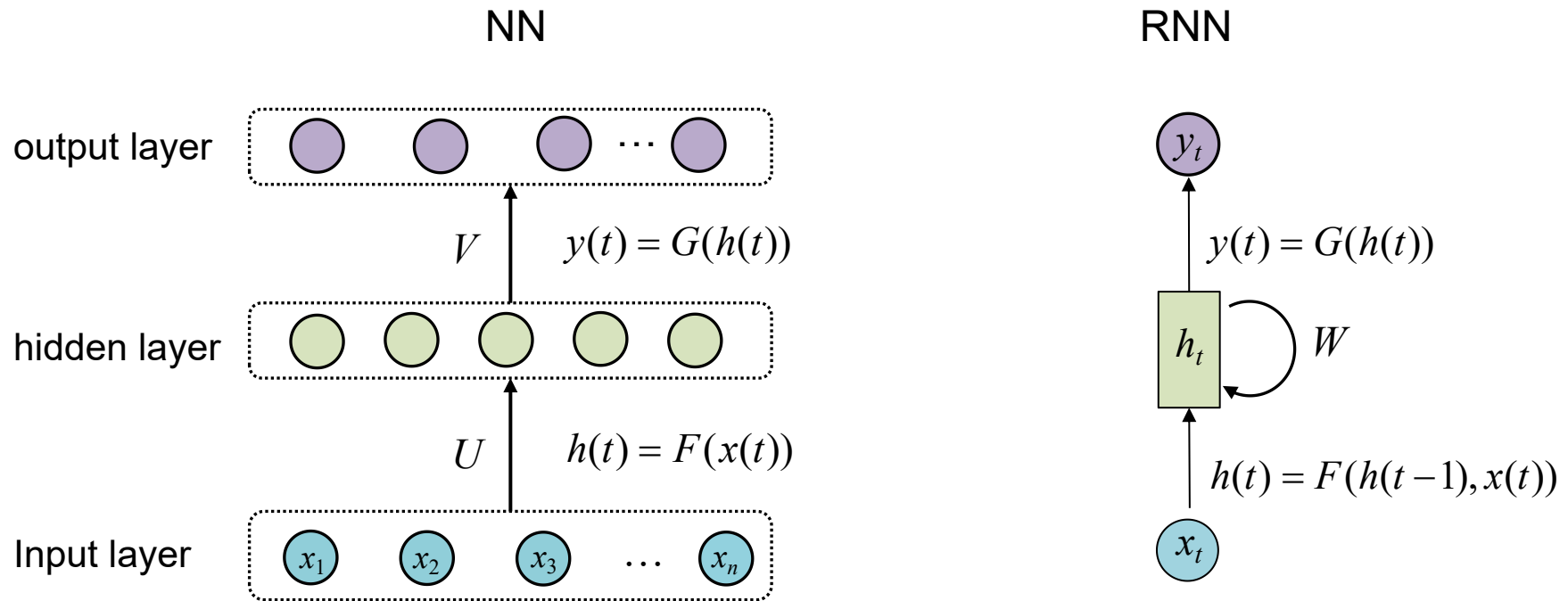
1. Tomas Mikolov, et.al. Statistical language models based on neural networks. 2012
2. Tomas Mikolov, et.al. Recurrent neural network based language model. 2010

# RNNLM vs NNLM



- 对于RNN来说，隐层的更新可通过  $h(t) = f(Ux(t) + Wh(t-1))$  来实现，其中  $f$  可以是非线性激活函数（如sigmoid），也可以是复杂的模型结构（如LSTM）

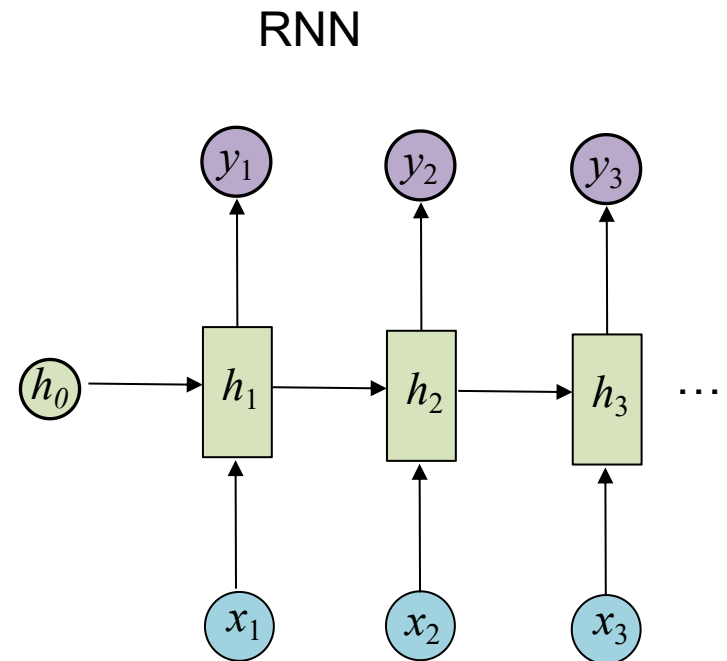
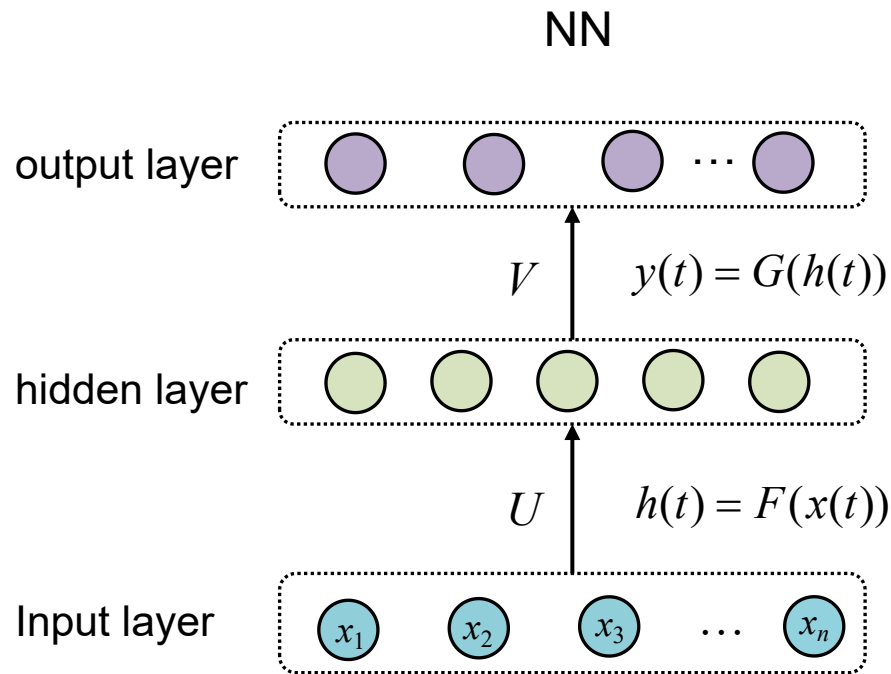
# RNNLM vs NNLM



- 对于RNN来说，隐层的更新可通过  $h(t) = f(Ux(t) + Wh(t-1))$  来实现，其中  $f$  可以是非线性激活函数（如sigmoid），也可以是复杂的模型结构（如LSTM）



# RNNLM vs NNLM



- 对于RNN来说，隐层的更新可通过  $h(t) = f(Ux(t) + Wh(t-1))$  来实现，其中  $f$  可以是非线性激活函数（如sigmoid），也可以是复杂的模型结构（如LSTM）

# RNN应用场景：序列标注任务

- 结构预测任务大多可以看作序列标注任务，例如词性标注、命名实体识别等

## 一个例子

- 采用常用的BIO(Beginning, Inside, Other)实体标注体系

### 1. 以字作为基本输入单元：

Input :	主	席	习	近	平	发	表	重	要	讲	话
	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓
Output:	O	O	B-PER	I-PER	I-PER	O	O	O	O	O	O

### 2. 以词作为基本输入单元：

Input :	主席	习	近平	发表	重要	讲话
	↓	↓	↓	↓	↓	↓
Output:	O	B-PER	I-PER	O	O	O

# RNN for Speech Recognition

Table 1: Performance of models on WSJ DEV set when increasing size of training data.

Model	# words	PPL	WER
KN5 LM	200K	336	16.4
KN5 LM + RNN 90/2	200K	271	15.4
KN5 LM	1M	287	15.1
KN5 LM + RNN 90/2	1M	225	14.0
KN5 LM	6.4M	221	13.5
KN5 LM + RNN 250/5	6.4M	156	11.7

Table 2: Comparison of various configurations of RNN LMs and combinations with backoff models while using 6.4M words in training data (WSJ DEV).

Model	PPL		WER	
	RNN	RNN+KN	RNN	RNN+KN
KN5 - baseline	-	221	-	13.5
RNN 60/20	229	186	13.2	12.6
RNN 90/10	202	173	12.8	12.2
RNN 250/5	173	155	12.3	11.7
RNN 250/2	176	156	12.0	11.9
RNN 400/10	171	152	12.5	12.1
3xRNN static	151	143	11.6	11.3
3xRNN dynamic	128	121	11.3	11.1

## 减小词表

$$P(w_i(t+1) | w(t), s(t-1)) = \begin{cases} \frac{y_{rare}}{C_{rare}} & \text{if } w_i(t+1) \text{ is rare,} \\ y_i(t) & \text{otherwise} \end{cases}$$

- 出现频度低于阈值的词合并为特殊标记
- 词表越大，稀有词合并阈值越大

## 超参数设置

- 激活单元使用sigmoid函数
- 采用不同的隐单元数量
- 权重矩阵初始化为均值为0、方差为0.1的随机分布

## 训练技巧

- 静态训练 vs. 动态：模型在训练阶段不断更新，测试阶段不更新或更新
- 使用标准的SGD算法，学习率设为0.1，学习率提升明显时保持不变，没有明显提升时，学习率减半

# RNN for Speech Recognition

Table 1: Performance of models on WSJ DEV set when increasing size of training data.

Model	# words	PPL	WER
KN5 LM	200K	336	16.4
KN5 LM + RNN 90/2	200K	271	15.4
KN5 LM	1M	287	15.1
KN5 LM + RNN 90/2	1M	225	14.0
KN5 LM	6.4M	221	13.5
KN5 LM + RNN 250/5	6.4M	156	11.7

Table 2: Comparison of RNN and combinations with KN5 LM in training data (WSJ DEV set).

Model	PPL		WER	
	RNN	RNN+KN	RNN	RNN+KN
KN5 - baseline	-	221	-	13.5
RNN 60/20	229	186	13.2	12.6
RNN 90/10	202	173	12.8	12.2
RNN 250/5	173	155	12.3	11.7
RNN 250/2	176	156	12.0	11.9
RNN 400/10	171	152	12.5	12.1
3xRNN static	151	143	11.6	11.3
3xRNN dynamic	128	121	11.3	11.1

困惑度降低  
50%

错误率降低  
18%

## 评价标准

- 错误率：将识别出的词序列转为标准词序列，所需替换、删除、插入词的比例

$$WER = \frac{S + D + I}{N}$$

- 困惑度：困惑度越小、测试集中句子概率越大、语言模型越好

$$PPL = 2^{H(L,q)} \approx 2^{-\frac{1}{n} \log_2 q(x_1^n)} = [q(x_1^n)]^{-\frac{1}{n}}$$

交叉熵

## 实验结论

- RNN比KN5 LM好
- 在一定范围内，隐单元数量越多，效果越好
- 动态训练效果比静态训练效果好

# 3. 循环神经网络

---

3.1

RNN语言模型

3.2

RNN激活单元

3.3

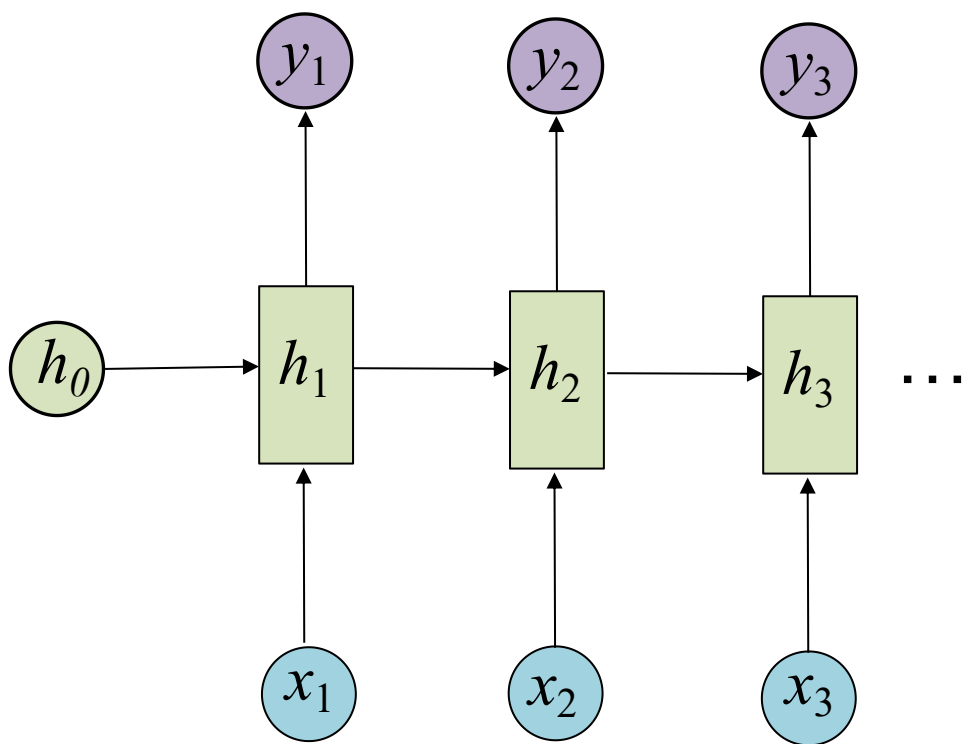
双向循环神经网络

3.4

Seq2seq框架

# 回顾RNN

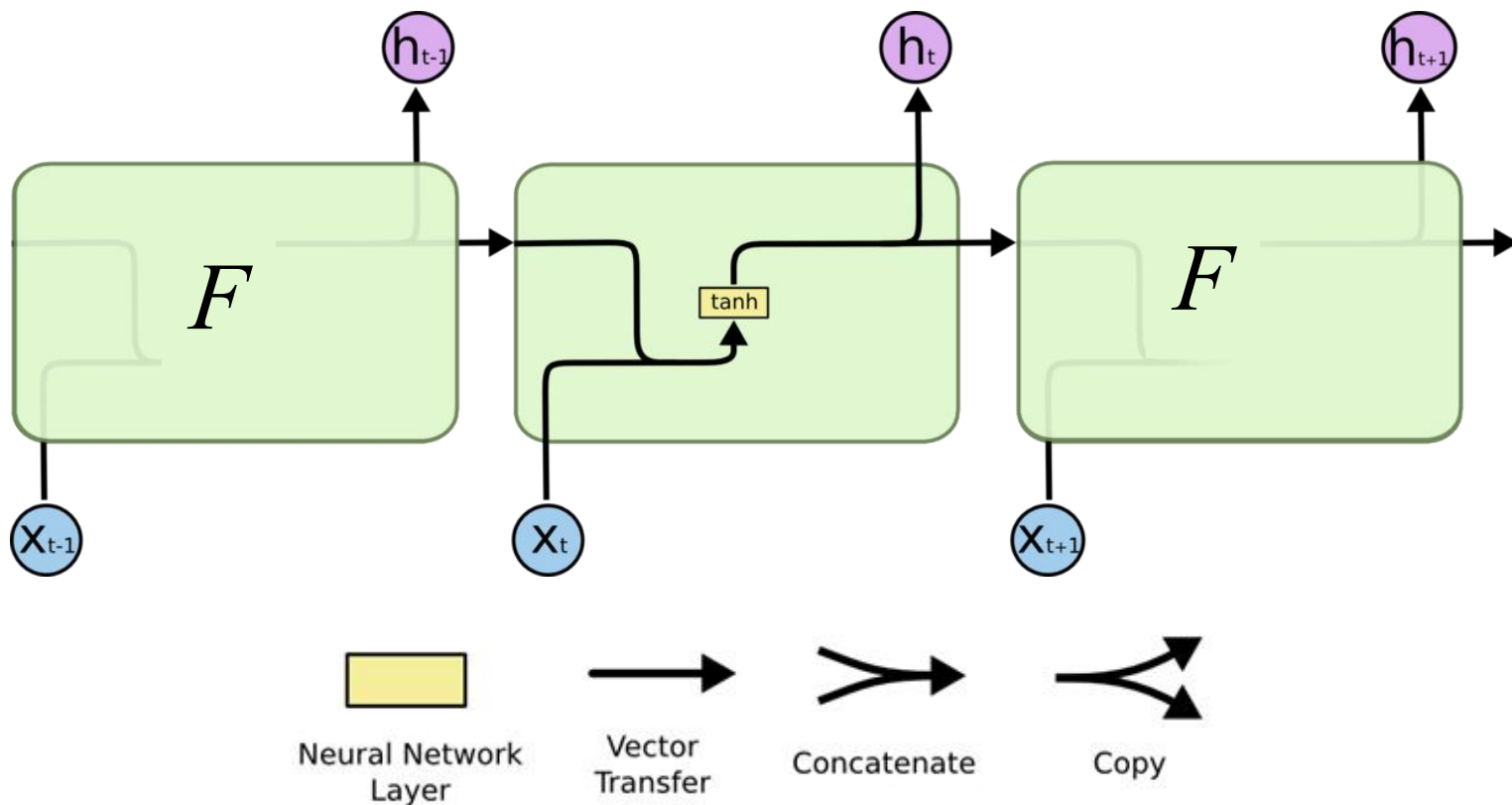
- 对于RNN来说，隐层的前向计算方法是  $h(t) = f(Ux(t) + Wh(t-1))$



# 非线性激活函数

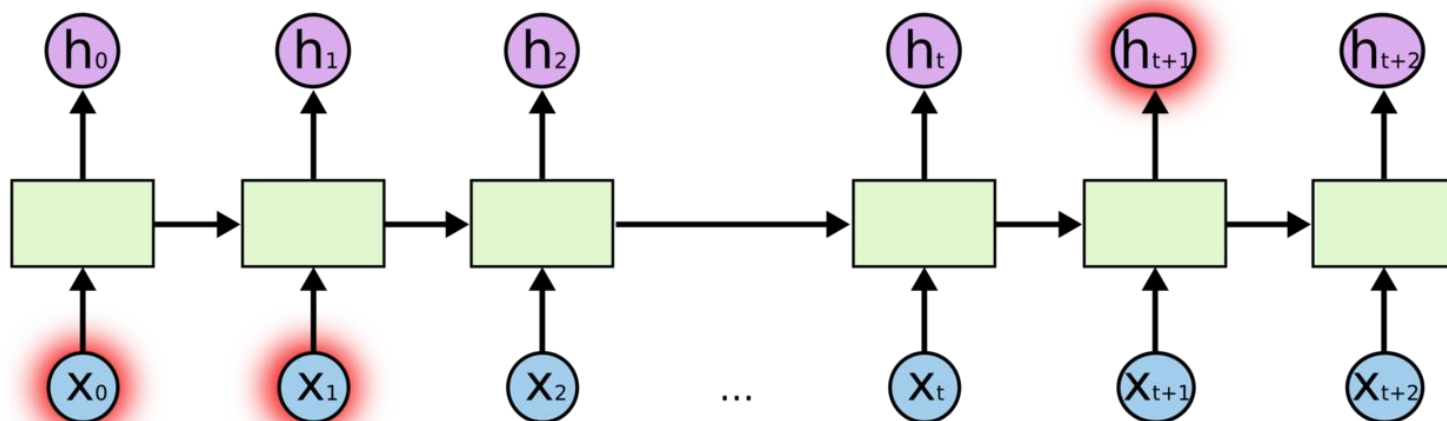
- RNN通常采用sigmoid、tanh函数作为激活函数，容易导致梯度消失

$$h(t) = \tanh(Ux(t) + Wh(t-1))$$



# 长期依赖 (Long-Term Dependency)

- 很久以前的输入，对当前时刻的网络影响较小；反向传播的梯度，也很难影响很久以前的输入
- 例如：
  - The cat, which already ate a bunch of food, (was) full.
  - The cats, which already ate a bunch of food, (were) full.

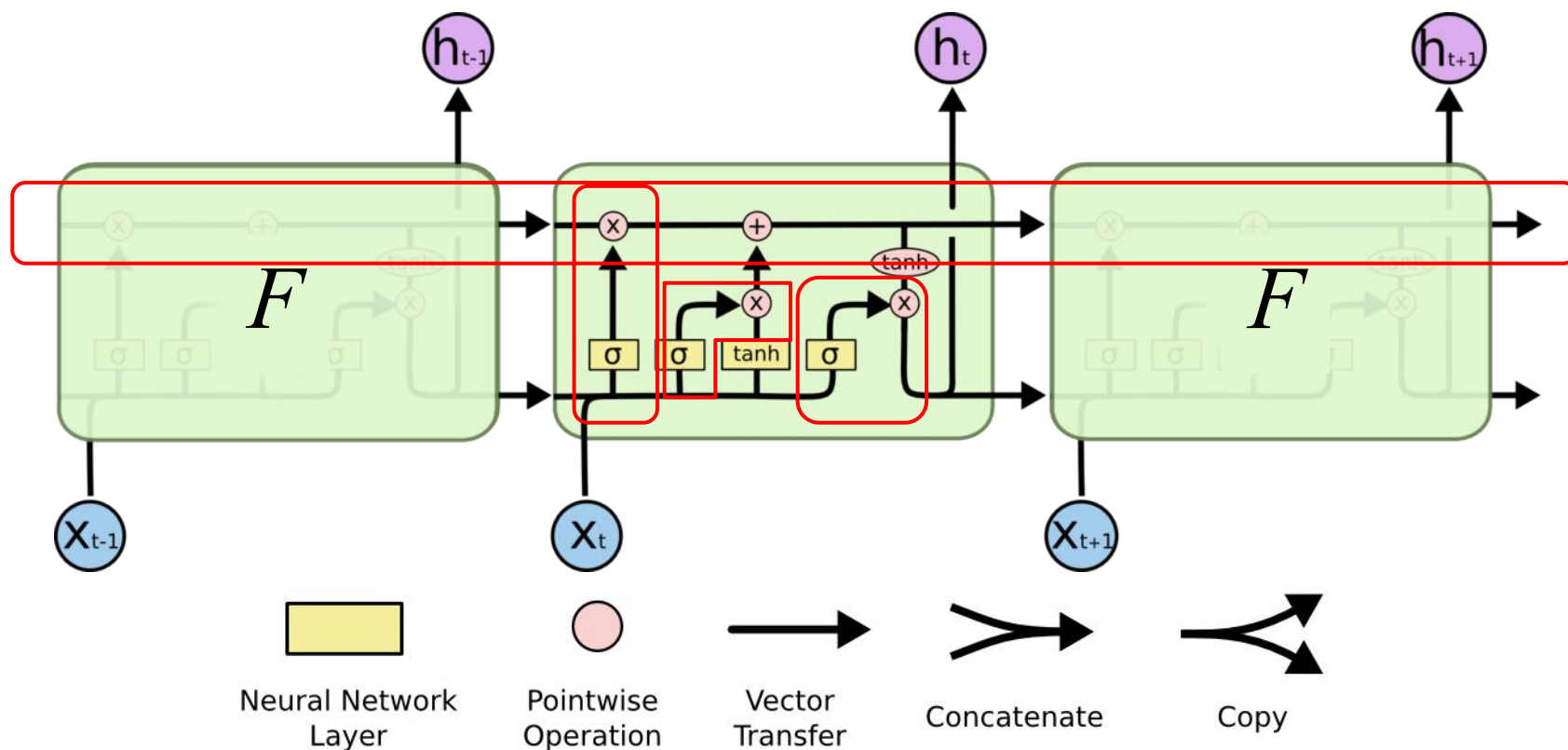


- 解决思路：采用ReLU函数，或采用其他模型来代替非线性激活函数



# LSTM模型

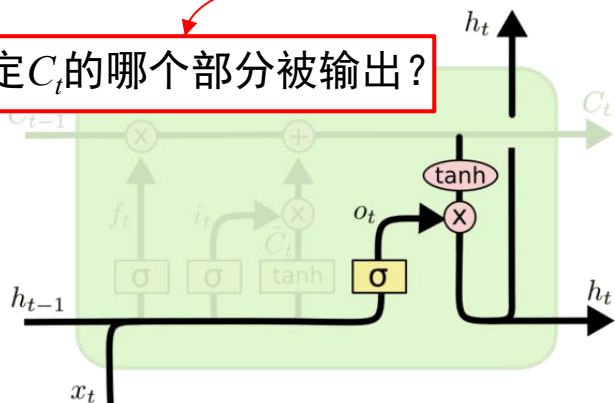
- LSTM单元不仅接受 $x_t$ 和 $h_{t-1}$ ，还维持一个细胞状态 $C_t$ ，保证信息在长期传播过程中不会被丢失
- 通过设计“门”结构保留重要特征，丢弃不重要的特征；每个门结构由一个sigmoid层和一个piecewise操作构成



# LSTM模型

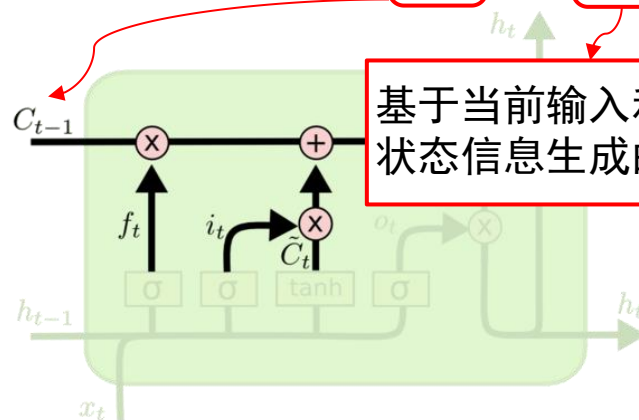
- 输出:  $h_t = o_t * \tanh(C_t)$
- 输出门:  $o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$

决定 $C_t$ 的哪个部分被输出?



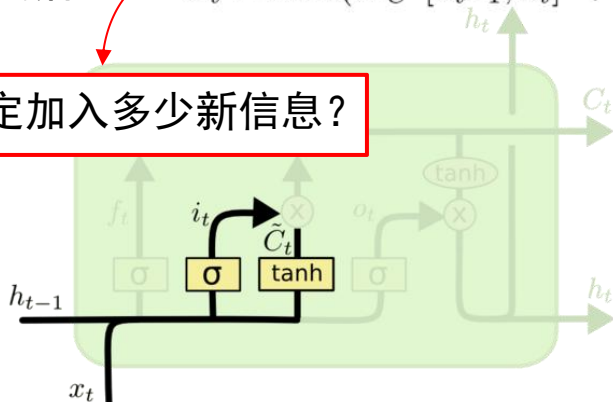
- 细胞状态:  $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$

基于当前输入和上个隐状态信息生成的新信息



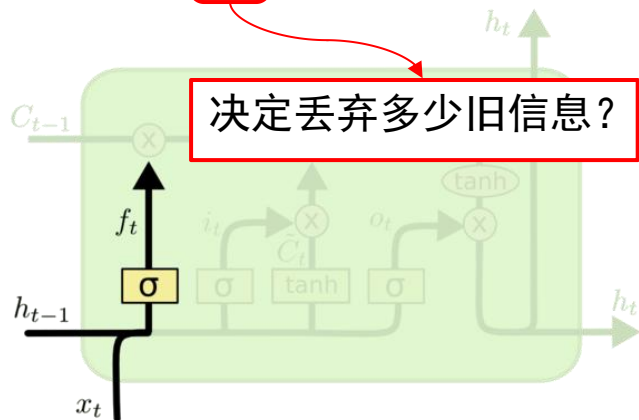
- 输入门:  $i_t = \sigma(W_i [h_{t-1}, x_t] + b_i)$
- 新信息:  $\tilde{C}_t = \tanh(W_C [h_{t-1}, x_t] + b_C)$

决定加入多少新信息?



- 遗忘门:  $f_t = \sigma(W_f [h_{t-1}, x_t] + b_f)$

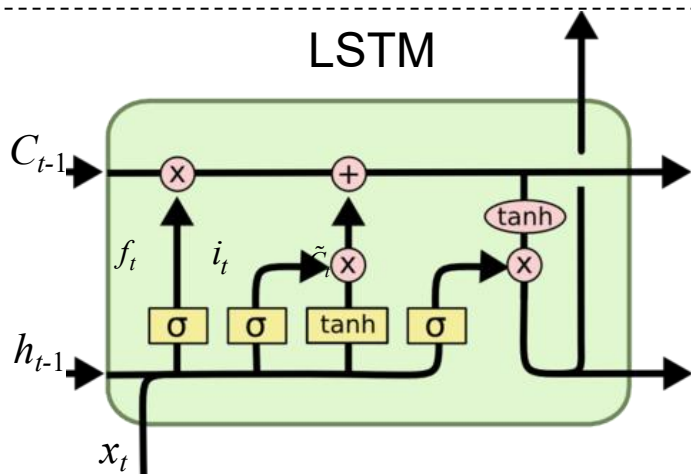
决定丢弃多少旧信息?



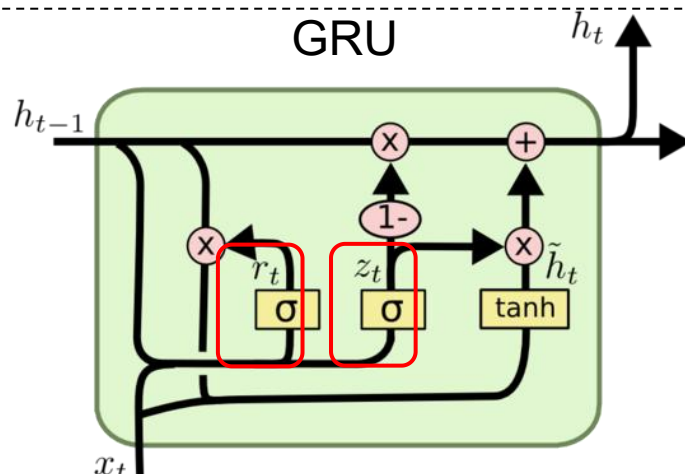
# GRU

- 有单独的细胞状态
- 用输入门和遗忘门决定保留或放弃
- 新信息  $\tilde{C}_t$  来源于  $h_{t-1}$  和  $x_t$
- 输出门控制细胞状态的输出

- 没有单独的细胞状态
- 用更新门决定保留或放弃
- $\tilde{h}_t$  由重置门决定来自  $h_{t-1}$  的信息
- 直接输出隐状态



- 遗忘门  $f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f)$
- 输入门  $i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$
- 新信息  $\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$
- 细胞状态  $C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$
- 输出门  $o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o)$
- 隐状态  $h_t = o_t * \tanh(C_t)$



- 更新门  $z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$
- 重置门  $r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$
- 新信息  $\tilde{h}_t = \tanh(W \cdot [r_t * h_{t-1}, x_t])$
- 隐状态  $h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$

保留哪些旧状态

接收哪些新状态

# Tanh vs. LSTM vs. GRU

			tanh	GRU	LSTM
Music Datasets	Nottingham	train	3.22	2.79	3.08
		test	<b>3.13</b>	3.23	3.20
	JSB Chorales	train	8.82	6.94	8.15
		test	9.10	<b>8.54</b>	8.67
	MuseData	train	5.64	5.06	5.18
		test	6.23	<b>5.99</b>	6.23
	Piano-midi	train	5.64	4.93	6.49
		test	9.03	<b>8.82</b>	9.03
Ubisoft Datasets	Ubisoft dataset A	train	6.29	2.31	1.44
		test	6.44	3.59	<b>2.70</b>
	Ubisoft dataset B	train	7.61	0.38	0.80
		test	7.62	<b>0.88</b>	1.26

Table 2: The average negative log-probabilities of the training and test sets.

## 任务

$$\max_{\theta} \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^{T_n} \log p(x_t^n | x_1^n, \dots, x_{t-1}^n; \theta)$$

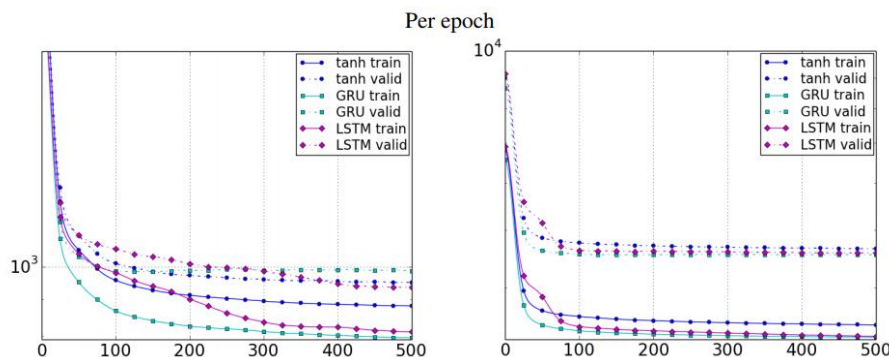
- 最大化语言模型的log似然

## 参数设置

- 三种模型使用相近的参数数量，以避免过拟合造成不同的影响

## 结论

- GRU比LSTM稍好，比tanh好的多；具体需要根据任务数据进行选择
- 收敛速度：GRU>LSTM>Tanh



# 3. 循环神经网络

---

3.1

RNN语言模型

3.2

RNN激活单元

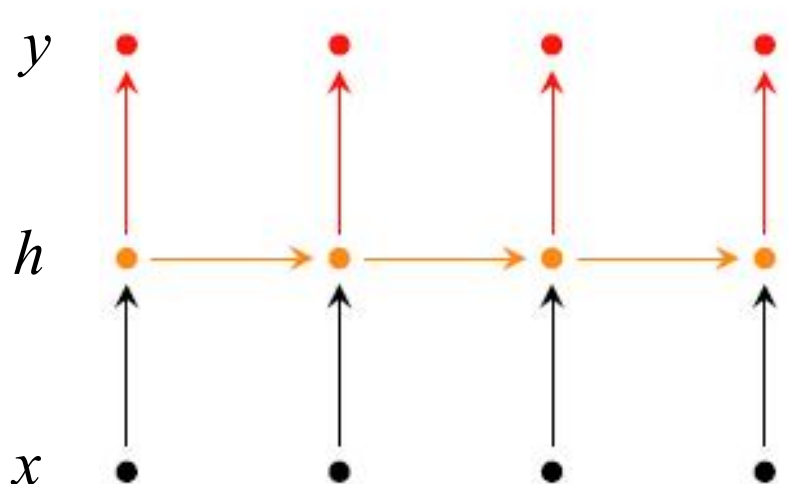
3.3

双向循环神经网络

3.4

Seq2seq框架

# RNN




$$h_t = f(Wx_t + Vh_{t-1} + b)$$

$$y_t = g(Uh_t + c)$$

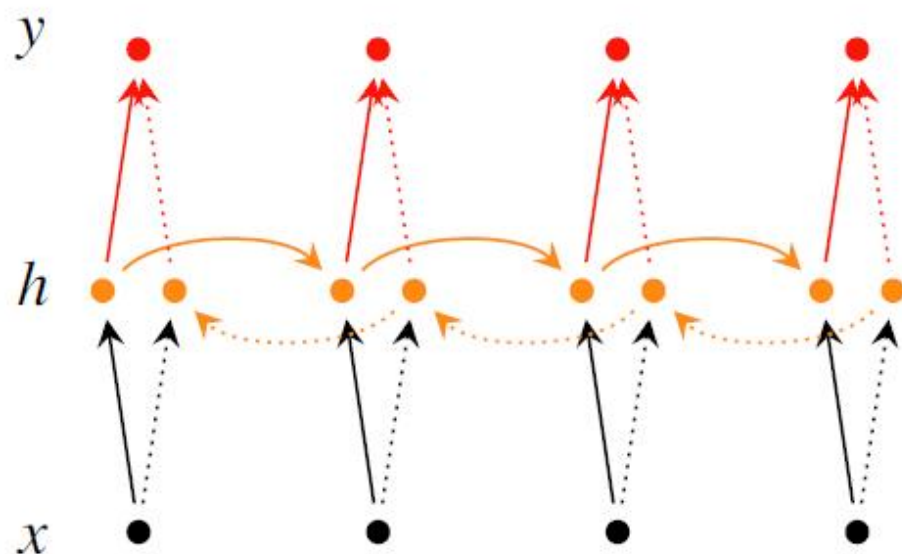
- $x$  表示输入序列中的一个词的词向量， $y$  表示该词对应的输出标签
- $h$  表示隐层，通过当前词和上一个隐层的信息进行计算，存储了截止目前状态输入的句子信息

# Problems

---

- 问题：RNN不能解决上文环境相同，但标注不同的情况
    - I did not go to the rodeo.
    - I [did not accept]<sub>DSE</sub> his suggestion.
  - 解决方案
    - 设置一个固定的窗口，对一个词的上下文进行建模
    - 在模型中考虑过去的状态（上文），也考虑未来的状态（下文）
-  Bidirectional RNN (Schuster and Paliwal, 1997)

# Bidirectional RNNs



$$\vec{h}_t = f(\vec{W}x_t + \vec{V}\vec{h}_{t-1} + \vec{b})$$

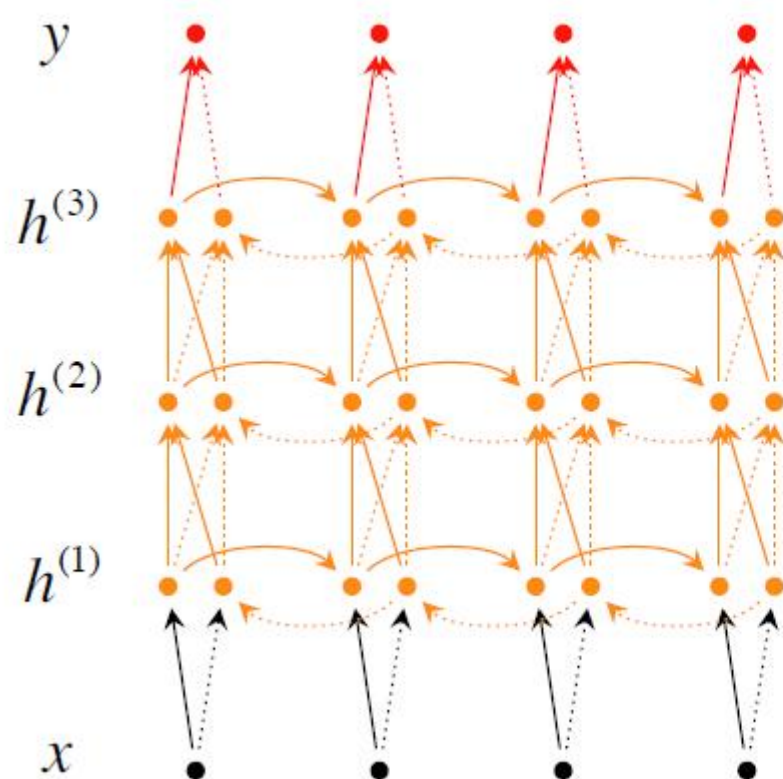
$$\overleftarrow{h}_t = f(\overleftarrow{W}x_t + \overleftarrow{V}\overleftarrow{h}_{t+1} + \overleftarrow{b})$$

$$y_t = g(U[\vec{h}_t; \overleftarrow{h}_t] + c)$$

- 针对每个时刻的输入，都有一个正向隐层、一个反向隐层  $h = [\vec{h}; \overleftarrow{h}]$
- 两个隐层分别表示一个词的上下文信息（即当前时刻之前和未来时刻的信息）



# Deep Bidirectional RNNs



$$\vec{h}_t^{(i)} = f(\vec{W}^{(i)} h_t^{(i-1)} + \vec{V}^{(i)} \vec{h}_{t-1}^{(i)} + \vec{b}^{(i)})$$

$$\overleftarrow{h}_t^{(i)} = f(\overleftarrow{W}^{(i)} h_t^{(i-1)} + \overleftarrow{V}^{(i)} \overleftarrow{h}_{t+1}^{(i)} + \overleftarrow{b}^{(i)})$$

$$y_t = g(U[\vec{h}_t^{(L)}; \overleftarrow{h}_t^{(L)}] + c)$$

- 深度双向RNN采用多个隐层，每个隐层向后一层传递序列信息h

# Elmo

- Elmo使用双向的LSTM语言模型，由一个前向和一个后向语言模型构成，目标函数是取这两个方向语言模型的最大似然
- ELMo首先训练一个完整的语言模型，再用语言模型去处理训练语料，生成相应的词向量，可对复杂特征(如句法和语义)和多义词进行建模



$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_1, t_2, \dots, t_{k-1})$$

$$p(t_1, t_2, \dots, t_N) = \prod_{k=1}^N p(t_k | t_{k+1}, t_{k+2}, \dots, t_N)$$

$$\sum_{k=1}^N (\log p(t_k | t_1, t_2, \dots, t_{k-1}) + \log p(t_k | t_{k+1}, t_{k+2}, \dots, t_N))$$

# RNN for Opinion Mining

- 识别输入序列中表达观点的文本片段，候选类别包括DSE和ESE
  - DSE：显式表达情感、情绪的文本片段（词或短语）
  - ESE：隐式表达情感、情绪的文本片段（词或短语）

The committee, [as usual]<sub>ESE</sub>, [has refused to make any statements]<sub>DSE</sub>.

BIO标注策略（B、I、O表示观点表达式的开始、内部和外部）

The    committee    ,    as    usual    ,    has    refused    to  
O                    O                    O    B\_ESE    I\_ESE                    O    B\_DSE                    I\_DSE                    I\_DSE  
make   any   statements   .  
I\_DSE    I\_DSE    I\_DSE                    O

# RNN for Opinion Mining: Experiments

Layers	$ h $	Precision		Recall		F1	
		Prop.	Bin.	Prop.	Bin.	Prop.	Bin.
Shallow	36	62.24	65.90	65.63*	73.89*	63.83	69.62
Deep 2	29	63.85*	67.23*	65.70*	<b>74.23*</b>	64.70*	70.52*
Deep 3	25	63.53*	67.67*	65.95*	73.87*	64.57*	70.55*
Deep 4	22	<b>64.19*</b>	<b>68.05*</b>	<b>66.01*</b>	73.76*	<b>64.96*</b>	<b>70.69*</b>
Deep 5	21	60.65	61.67	56.83	69.01	58.60	65.06
Shallow	200	62.78	66.28	65.66*	74.00*	64.09	69.85
Deep 2	125	62.92*	66.71*	66.45*	<b>74.70*</b>	64.47	70.36
Deep 3	100	<b>65.56*</b>	<b>69.12*</b>	<b>66.73*</b>	74.69*	<b>66.01*</b>	<b>71.72*</b>
Deep 4	86	61.76	65.64	63.52	72.88*	62.56	69.01
Deep 5	77	61.64	64.90	62.37	72.10	61.93	68.25

Evaluation of RNNs for DSE extraction

## 超参数设置

- 激活函数：输出层softmax，隐层ReLU
- 词向量：word2vec
- 正则：针对较大网络，采用dropout来避免过拟合
- 损失函数：交叉熵
- **SGD**：学习率0.005

	Model	Precision		Recall		F1	
		Prop.	Bin.	Prop.	Bin.	Prop.	Bin.
DSE	CRF	74.96*	82.28*	46.98	52.99	57.74	64.45
	semiCRF	61.67	69.41	<b>67.22*</b>	73.08*	64.27	71.15*
	CRF +vec	<b>74.97*</b>	<b>82.43*</b>	49.47	55.67	59.59	66.44
	semiCRF +vec	66.00	71.98	60.96	68.13	63.30	69.91
	Deep RNN 3 100	65.56	69.12	66.73*	<b>74.69*</b>	<b>66.01*</b>	<b>71.72*</b>
ESE	CRF	56.08	68.36	42.26	51.84	48.10	58.85
	semiCRF	45.64	69.06	58.05	64.15	50.95	66.37*
	CRF +vec	<b>57.15*</b>	69.84*	44.67	54.38	50.01	61.01
	semiCRF +vec	53.76	<b>70.82*</b>	52.72	61.59	53.10	65.73
	Deep RNN 3 100	52.04	60.50	<b>61.71*</b>	<b>76.02*</b>	<b>56.26*</b>	<b>67.18*</b>

Comparison of Deep RNNs to CRF

## 结论

- 双向 vs. 单向：双向浅层RNN（36个隐单元）比单向浅层RNN（65个隐单元）好
- 网络层次：Deep RNN层数不是越多越好，平均看来3层RNN效果最好
- **DeepRNN vs. CRF**：DeepRNN在DSE和ESE上效果均最好

# BI-LSTM-CRF for Sequence Tagging

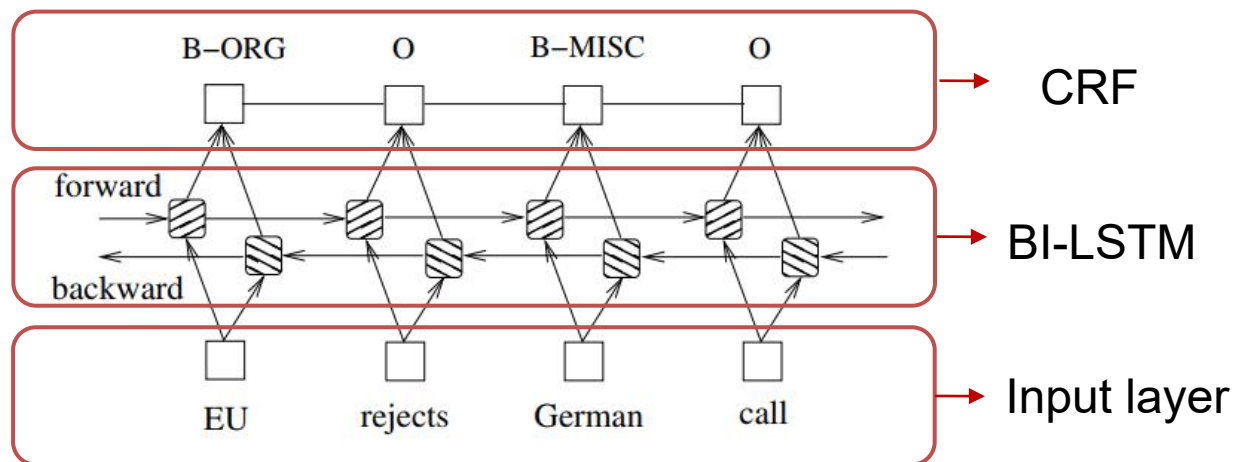


Figure 7: A BI-LSTM-CRF model.

Table 2: Comparison of tagging performance on POS, chunking and NER tasks for various models.

		POS	CoNLL2000	CoNLL2003
Random	Conv-CRF (Collobert et al., 2011)	96.37	90.33	81.47
	LSTM	97.10	92.88	79.82
	BI-LSTM	97.30	93.64	81.11
	CRF	97.30	93.69	83.02
	LSTM-CRF	<b>97.45</b>	93.80	84.10
	BI-LSTM-CRF	97.43	<b>94.13</b>	<b>84.26</b>
Senna	Conv-CRF (Collobert et al., 2011)	97.29	94.32	88.67 (89.59)
	LSTM	97.29	92.99	83.74
	BI-LSTM	97.40	93.92	85.17
	CRF	97.45	93.83	86.13
	LSTM-CRF	97.54	94.27	88.36
	BI-LSTM-CRF	<b>97.55</b>	<b>94.46</b>	<b>88.83 (90.10)</b>

- CRF很强大，其效果好于只用LSTM或BI-LSTM
- BI-LSTM与CRF的结合几乎将序列标注任务的效果推向顶峰

# 3. 循环神经网络

---

3.1

RNN语言模型

3.2

RNN激活单元

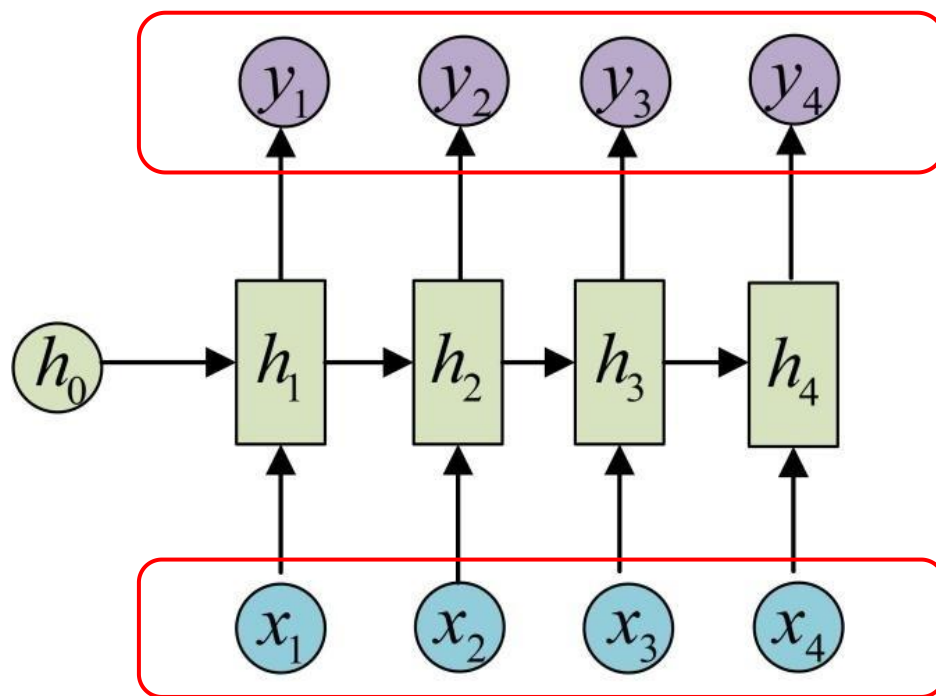
3.3

双向循环神经网络

3.4

Seq2seq框架

# RNN基本结构



$$h(t) = f(Ux(t) + Wh(t-1))$$

$$y(t) = G(h(t))$$

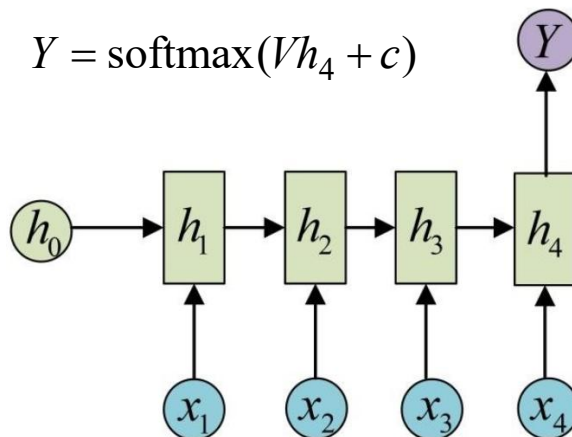
输入序列和输出序列等长，不等长怎么办？



# RNN变体

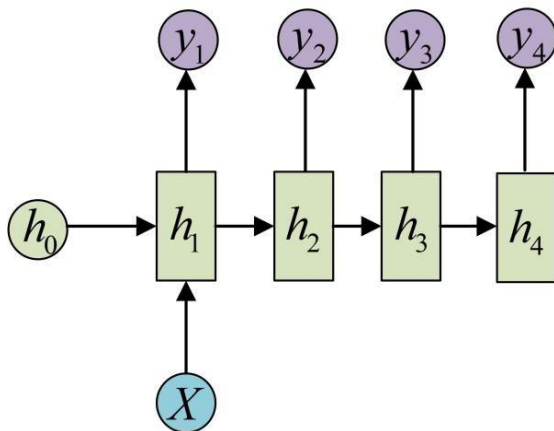
Many to one

$$Y = \text{softmax}(Vh_4 + c)$$



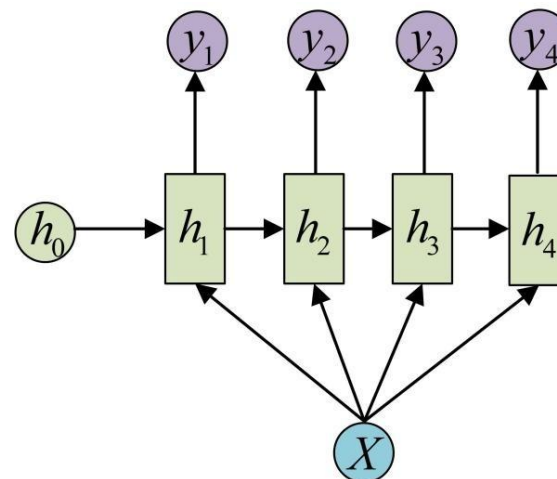
one to many

$$h(t) = \begin{cases} f(Wh(t-1)) & , t > 1 \\ f(UX + Wh(t-1)) & , t = 1 \\ 0 & , t = 0 \end{cases}$$



one to many

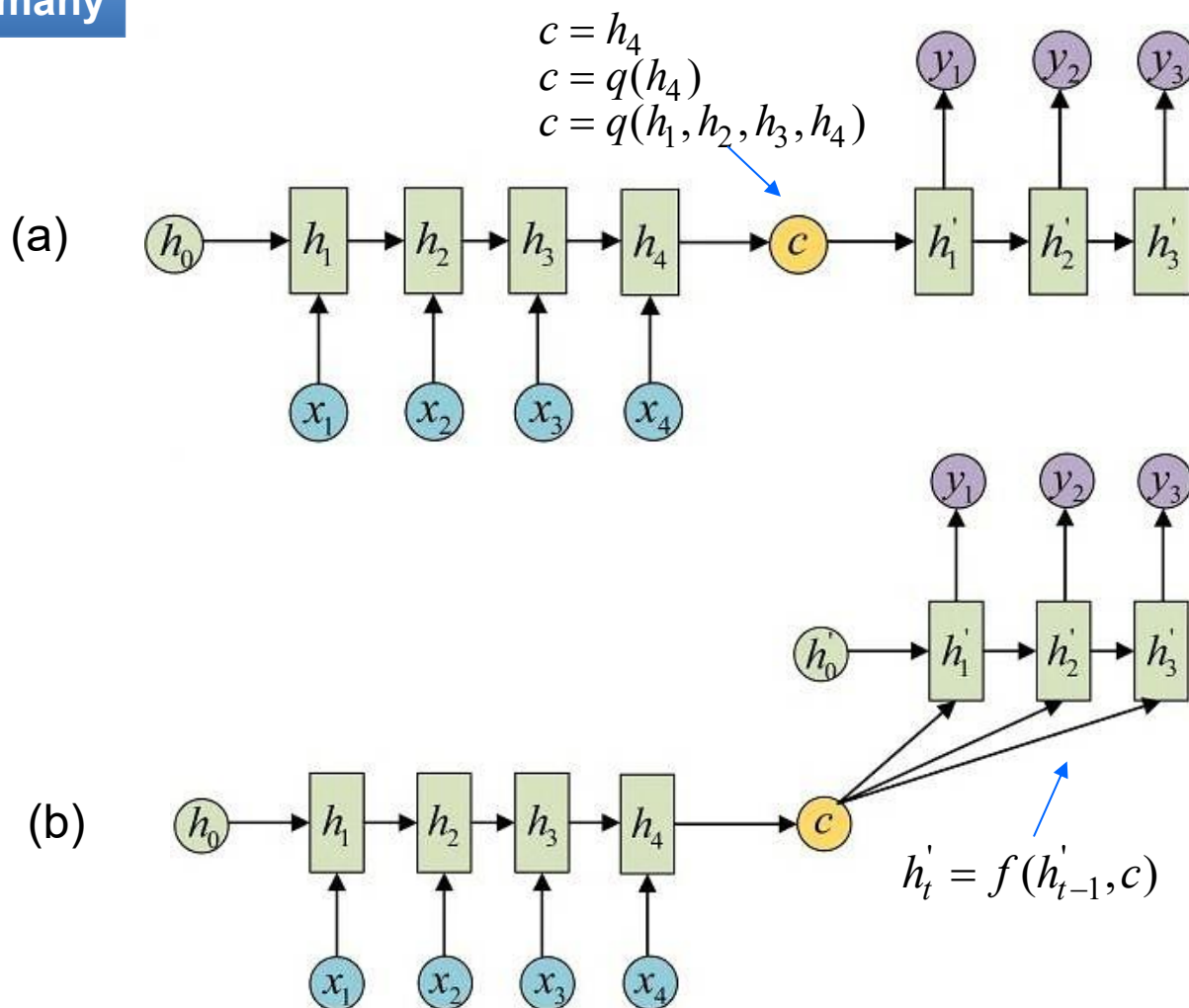
$$h(t) = f(UX + Wh(t-1))$$



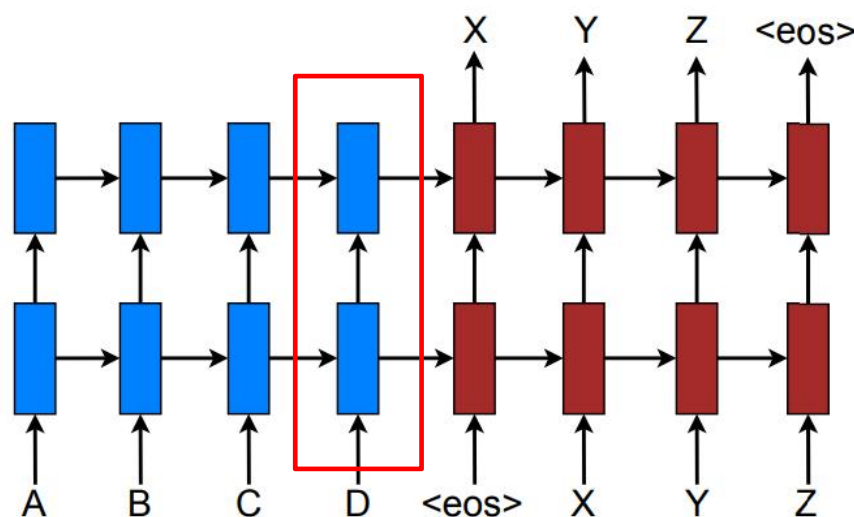
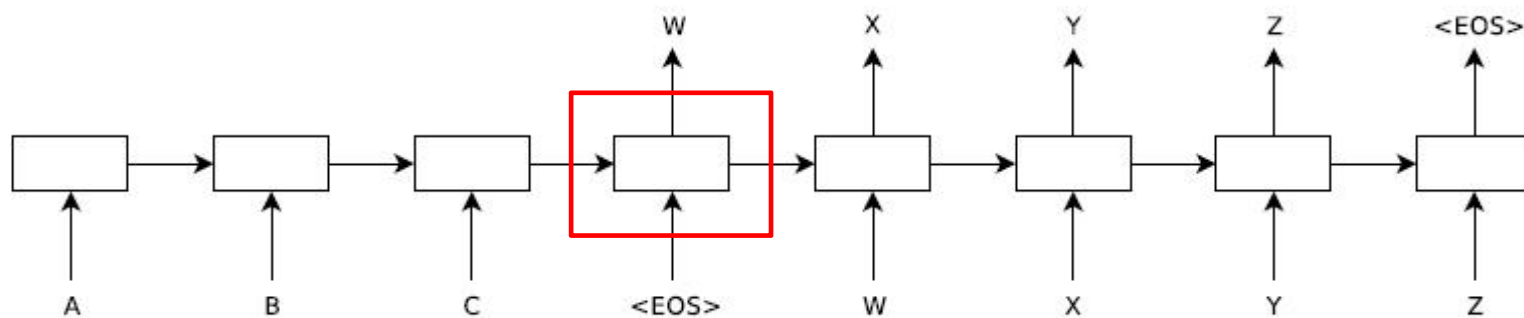


# RNN变体: Encoder-Decoder

many to many

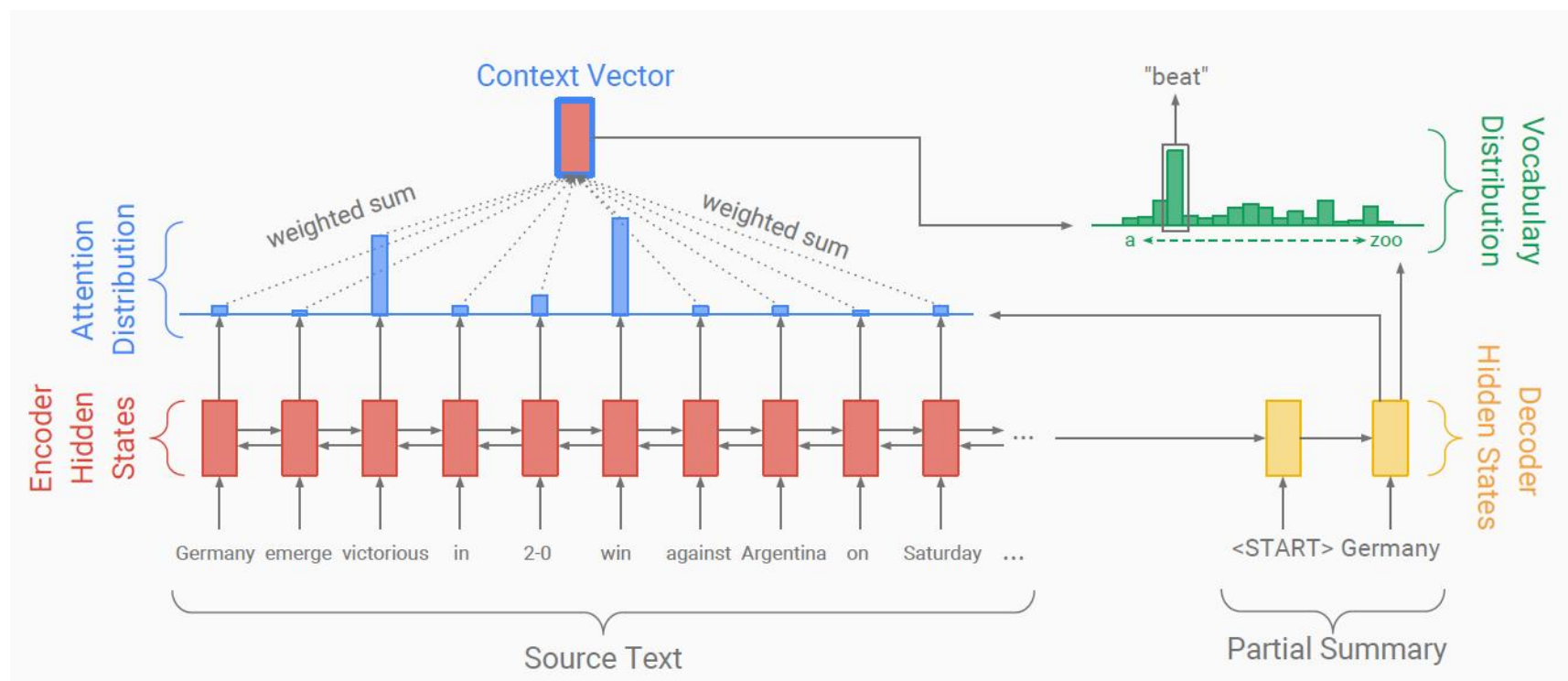


# Seq2Seq Model for Machine Translation



- 深层的编解码器效果更好

# Seq2Seq Model for Text Summarization

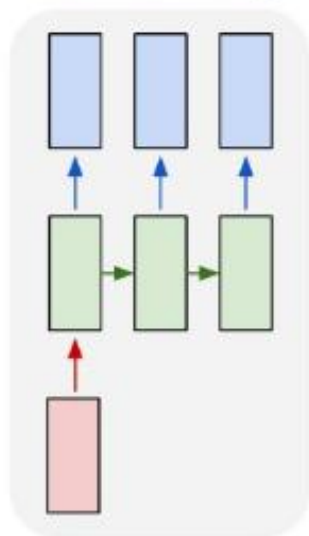


Attention Based Seq2seq Model

Alexander M Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. arXiv preprint arXiv:1509.00685, 2015.

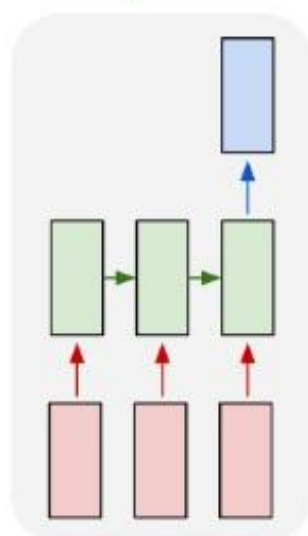
# The capability of RNN

one to many



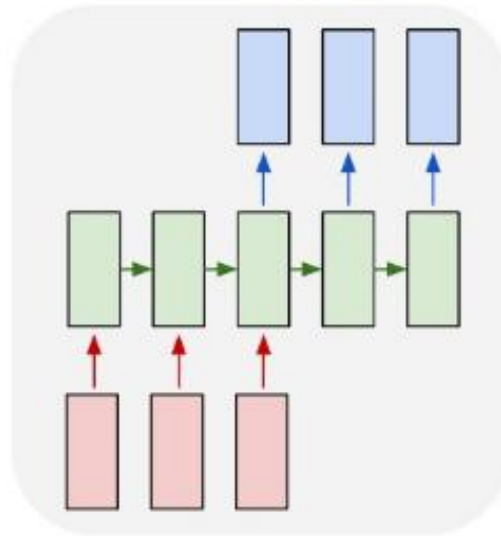
图像生成文字

many to one



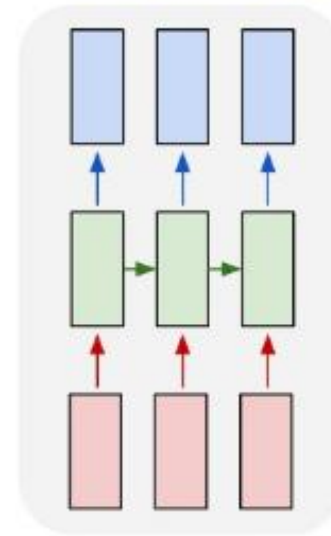
文本分类

many to many



自动文摘、机器翻译等

many to many



序列标注

# 欢迎加入DL4NLP!



中国科学院 信息工程研究所  
INSTITUTE OF INFORMATION ENGINEERING, CAS