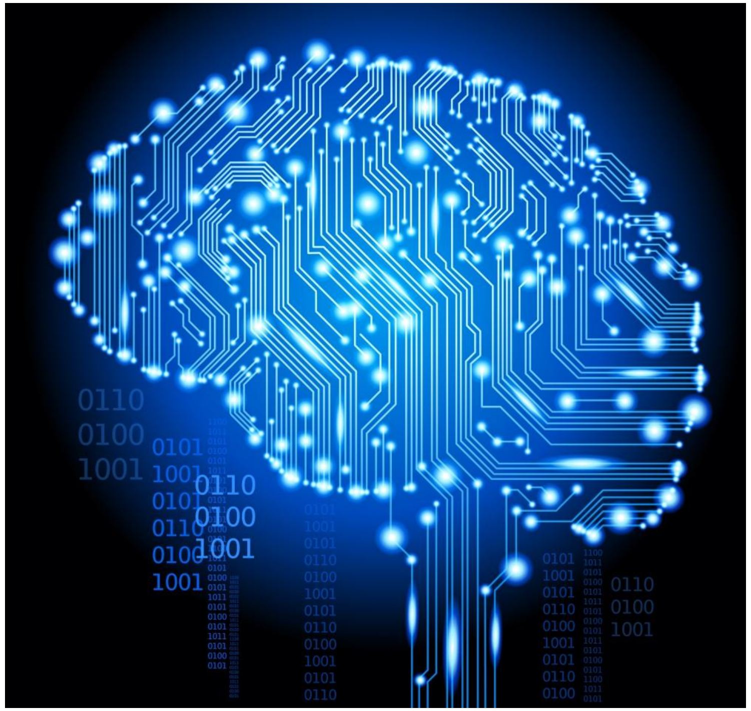
序列数据的深度学习模型

中国科学院自动化研究所

吴高巍

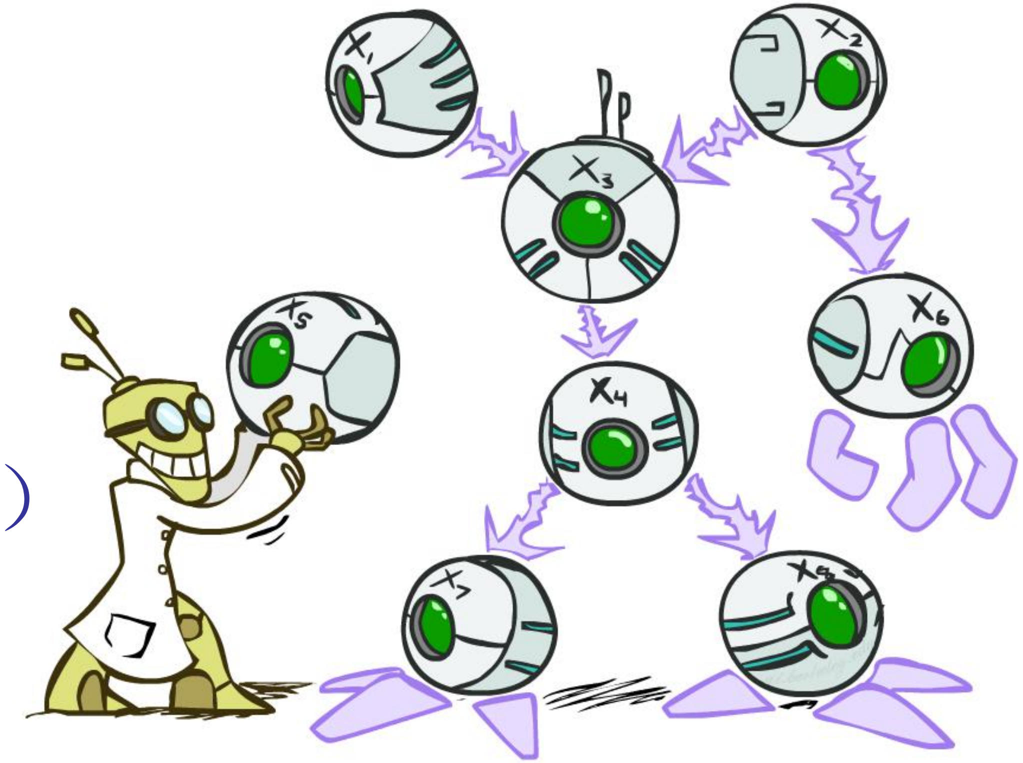
gaowei.wu@ia.ac.cn

2021-10-12



内容

- 循环神经网络
    - Recurrent Neural Networks, RNN
- 长序列的循环神经网络
- 序列模型（sequence to sequence）



循环神经网络

——Recurrent Neural Networks, RNN

序列模型

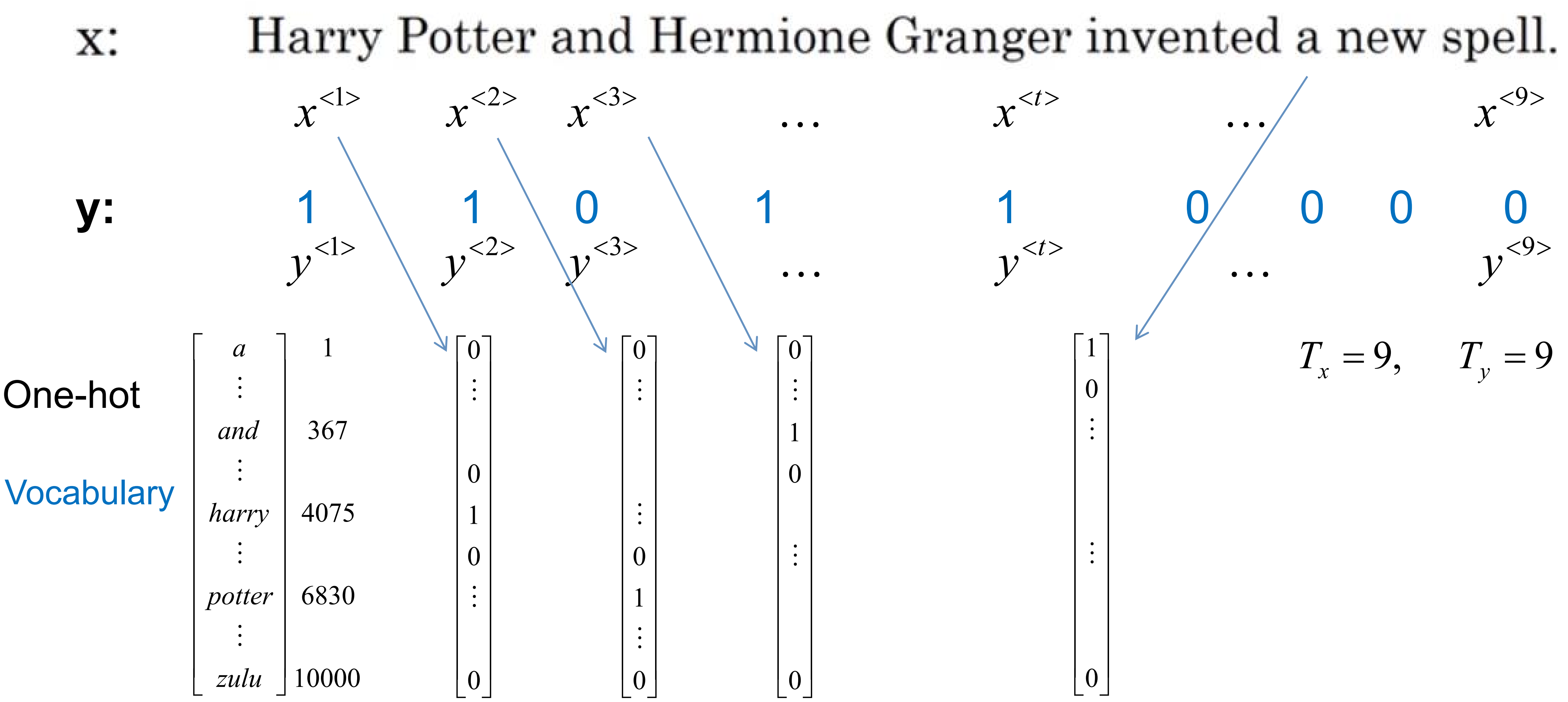
Speech recognition		→	“The quick brown fox jumped over the lazy dog.”
Music generation	Ø	→	
Sentiment classification	“There is nothing to like in this movie.”	→	★☆☆☆☆
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	→	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing with me?
Video activity recognition		→	Running
Name entity recognition	Yesterday, Harry Potter met Hermione Granger.	→	Yesterday, Harry Potter met Hermione Granger.

序列数据建模(modeling sequences)

- 学习序列数据，常需要转换输入序列到不同领域的输出序列
  - eg. turn a sequence of sound pressures into a sequence of word identities.
- 如果没有分离的目标序列，可以通过预测输入序列中的下一项来得到“教师信号”
  - The target output sequence is the input sequence with an advance of 1 step.
  - For temporal sequences there is a natural order for the predictions.
- 预测序列的下一项，模糊了监督学习与非监督学习的差别
  - It uses methods designed for supervised learning, but it doesn’t require a separate teaching signal.

示例

命名实体识别：



x: Harry Potter and Hermione Granger invented a new spell.

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ … $x^{<t>}$ … $x^{<9>}$

y: 1 1 0 1 1 0 0 0 0

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ … $y^{<t>}$ … $y^{<9>}$

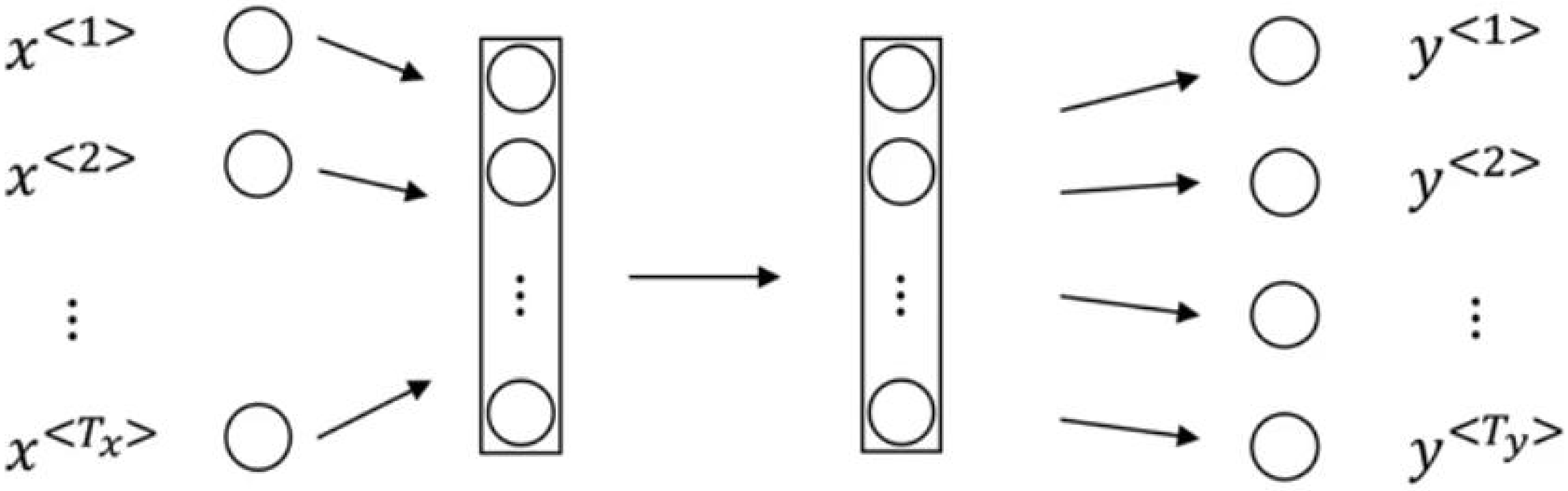
One-hot

Vocabulary

$$\begin{bmatrix} a \\ \vdots \\ and \\ \vdots \\ harry \\ \vdots \\ potter \\ \vdots \\ zulu \end{bmatrix} \begin{array}{c} 1 \\ \\ 367 \\ \\ 4075 \\ \\ 6830 \\ \\ 10000 \end{array} \quad \begin{bmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ \vdots \\ \vdots \\ 0 \\ 1 \\ \vdots \\ 0 \end{bmatrix} \quad \begin{bmatrix} 0 \\ \vdots \\ 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \quad \begin{bmatrix} 1 \\ 0 \\ \vdots \\ \vdots \\ 0 \end{bmatrix}$$

$$T_x = 9, \quad T_y = 9$$

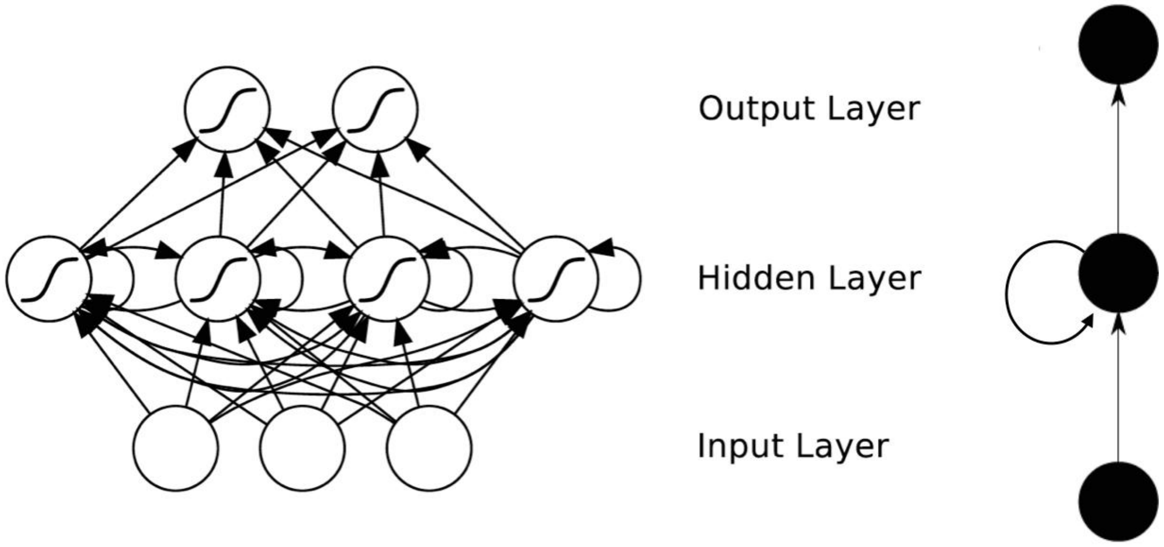
Why not a standard network?



- 问题：
  - 输入和输出数据在不同例子中可以有不同的长度
  - 不共享从文本的不同位置上学到的特征

Recurrent Neural Networks, RNN

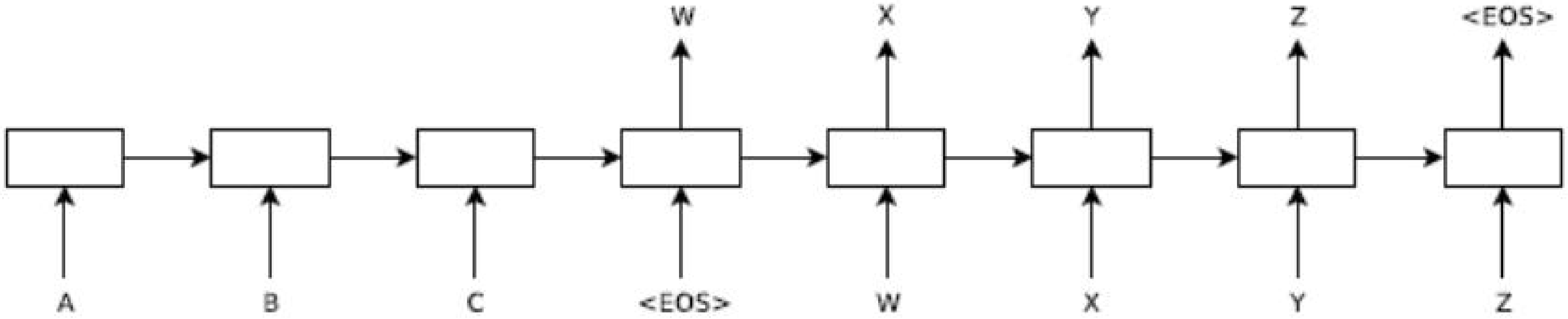
- Hidden Layer会有连向下一时间Hidden Layer的边



- RNN功能强大
  - Distributed hidden state that allows them to store a lot of information about the past efficiently.
  - Non-linear dynamics that allows them to update their hidden state in complicated ways.

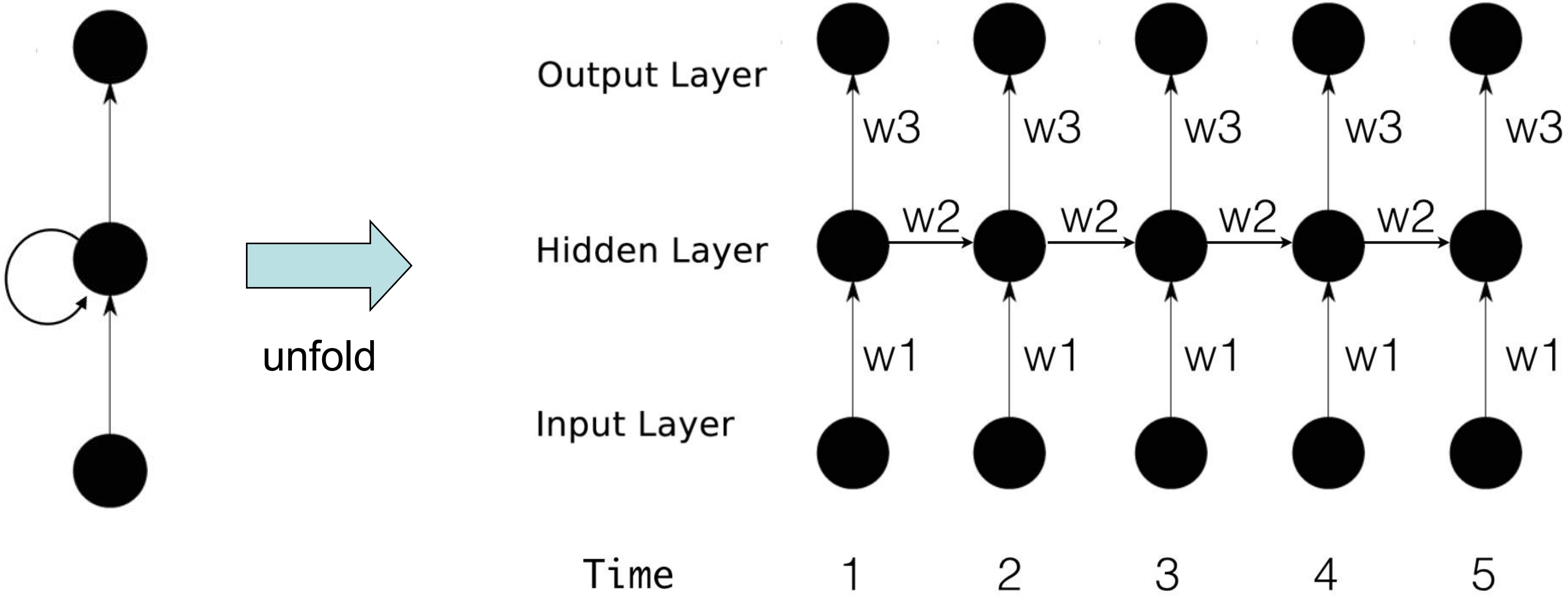
RNN

- 一般来说，RNN每一时间的输入和输出是不一样的
- 例，序列学习
    - 对于序列数据是将序列项依次传入，每个序列项再对应不同的输出

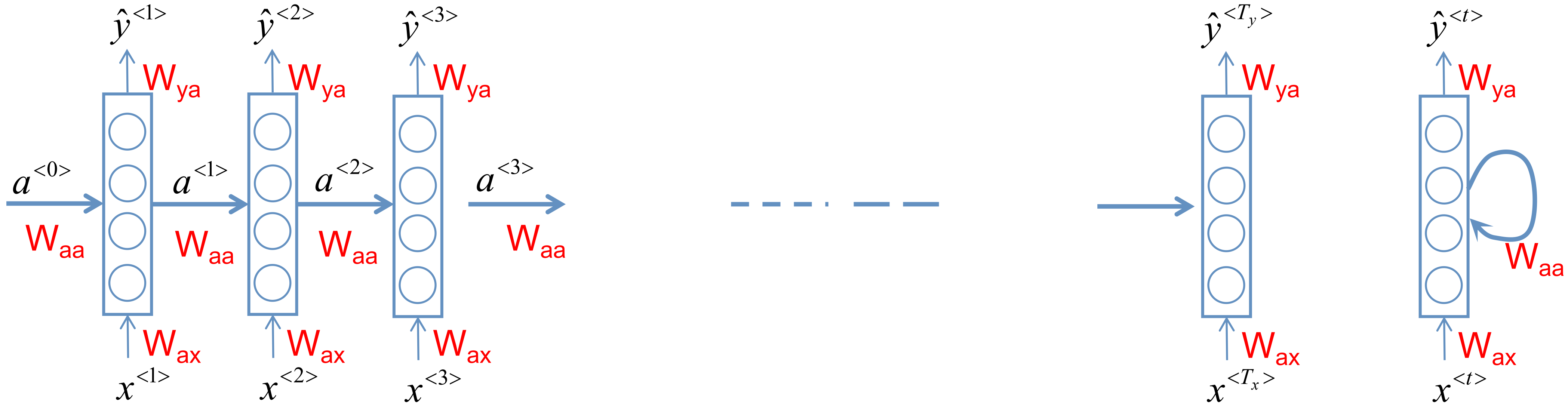


RNN

- 时序展开
  - 在RNN中每一个时间步骤用到的参数都是一样的



Recurrent Neural Networks



x: Harry Potter and Hermione Granger invented a new spell.

Training RNNs with backpropagation

- 权值一致的BP算法
  - BP算法容易实现权值间的线性约束
  - 同样计算梯度，然后改变梯度以满足约束
  - 如果权值开始时满足约束，那就会一直满足约束

$$To\ \ constrain:\quad w_1 = w_2$$

$$we\ \ need:\quad \Delta w_1 = \Delta w_2$$

$$compute:\quad \frac{\partial E}{\partial w_1} \quad and \quad \frac{\partial E}{\partial w_2}$$

$$use\quad \frac{\partial E}{\partial w_1} + \frac{\partial E}{\partial w_2} \quad for\ w_1\ and\ w_2$$

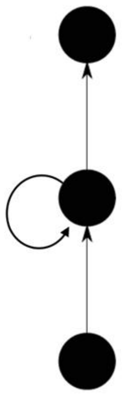
BPTT, Back Propagation Through Time

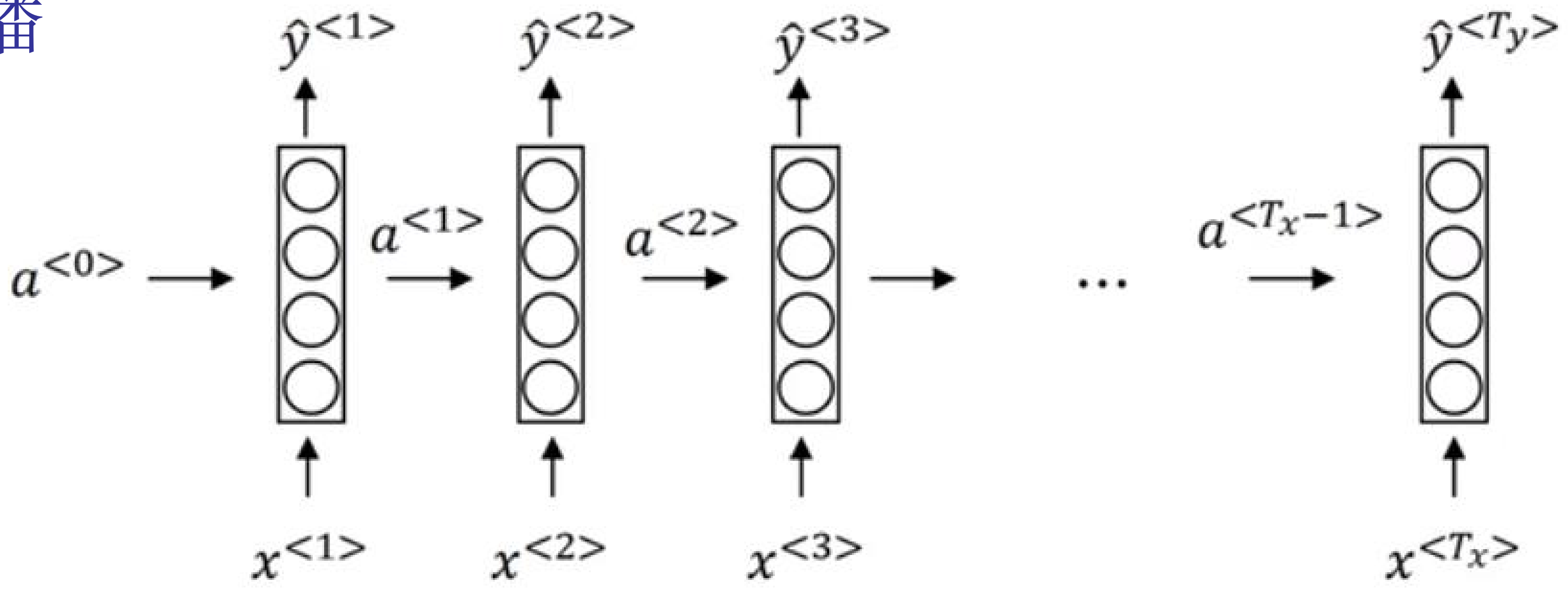
- RNN可看作权值共享的多层、前向网络
  - 训练权值约束的前向网络

- 时间域上的训练算法
  - The forward pass builds up a stack of the activities of all the units at each time step.
  - The backward pass peels activities off the stack to compute the error derivatives at each time step.
  - After the backward pass we add together the derivatives at all the different times for each weight.

BPTT

- 前向传播





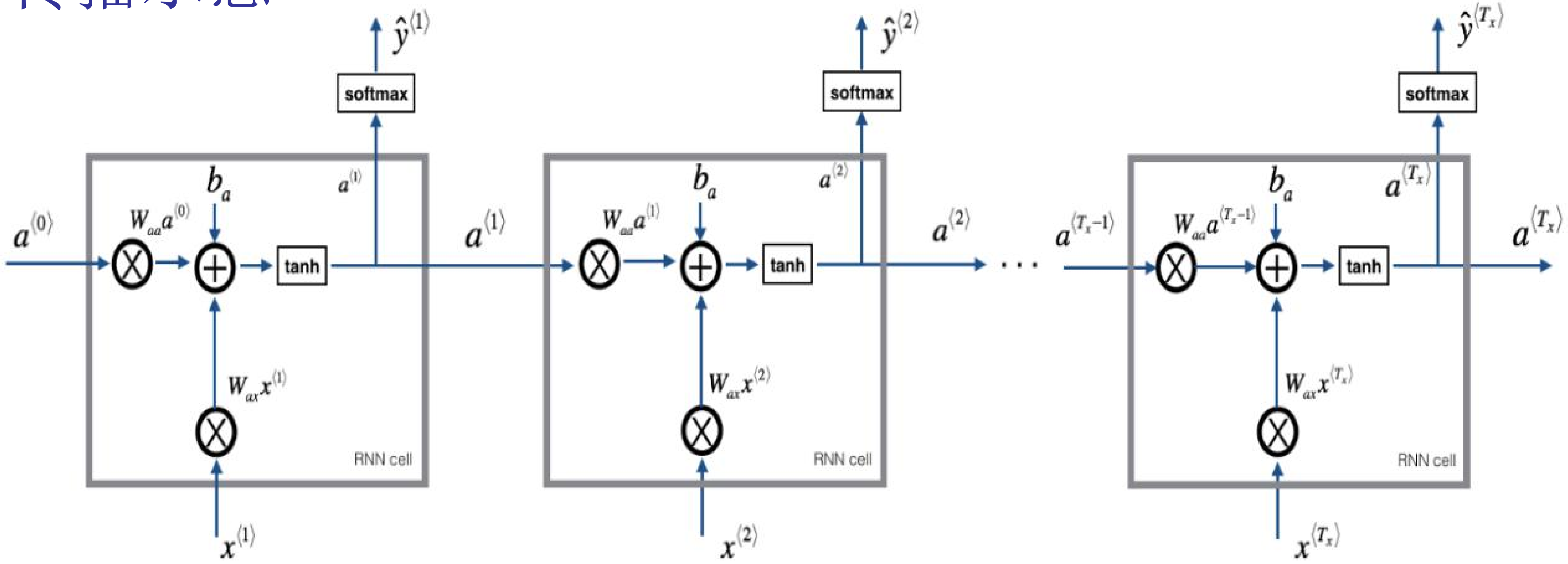
$$a^{<0>} = \vec{0} \qquad a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$$

$$[W_{aa} \vdots W_{ax}] = W_a \qquad \begin{bmatrix} a^{<t-1>} \\ x^{<t>} \end{bmatrix} = [a^{<t-1>}, x^{<t>}]$$

$$\hat{y}^{<t>} = g_2(W_{ya}a^{<t>} + b_y) \qquad a^{<t>} = g(W_a[a^{<t-1>}, x^{<t>}] + b_a)$$

BPTT

- 前向传播示意



$\hat{y}^{\langle 1 \rangle}$
softmax
$\hat{y}^{\langle 2 \rangle}$
softmax
$\hat{y}^{\langle T_x \rangle}$
softmax
$a^{\langle 0 \rangle}$
b_a
$W_{aa}a^{\langle 0 \rangle}$
$a^{\langle 1 \rangle}$
tanh
$W_{ax}x^{\langle 1 \rangle}$
RNN cell
$x^{\langle 1 \rangle}$
$a^{\langle 1 \rangle}$
b_a
$W_{aa}a^{\langle 1 \rangle}$
$a^{\langle 2 \rangle}$
tanh
$W_{ax}x^{\langle 2 \rangle}$
RNN cell
$x^{\langle 2 \rangle}$
$a^{\langle 2 \rangle}$
...
$a^{\langle T_x-1 \rangle}$
b_a
$W_{aa}a^{\langle T_x-1 \rangle}$
$a^{\langle T_x \rangle}$
tanh
$W_{ax}x^{\langle T_x \rangle}$
RNN cell
$x^{\langle T_x \rangle}$
$a^{\langle T_x \rangle}$

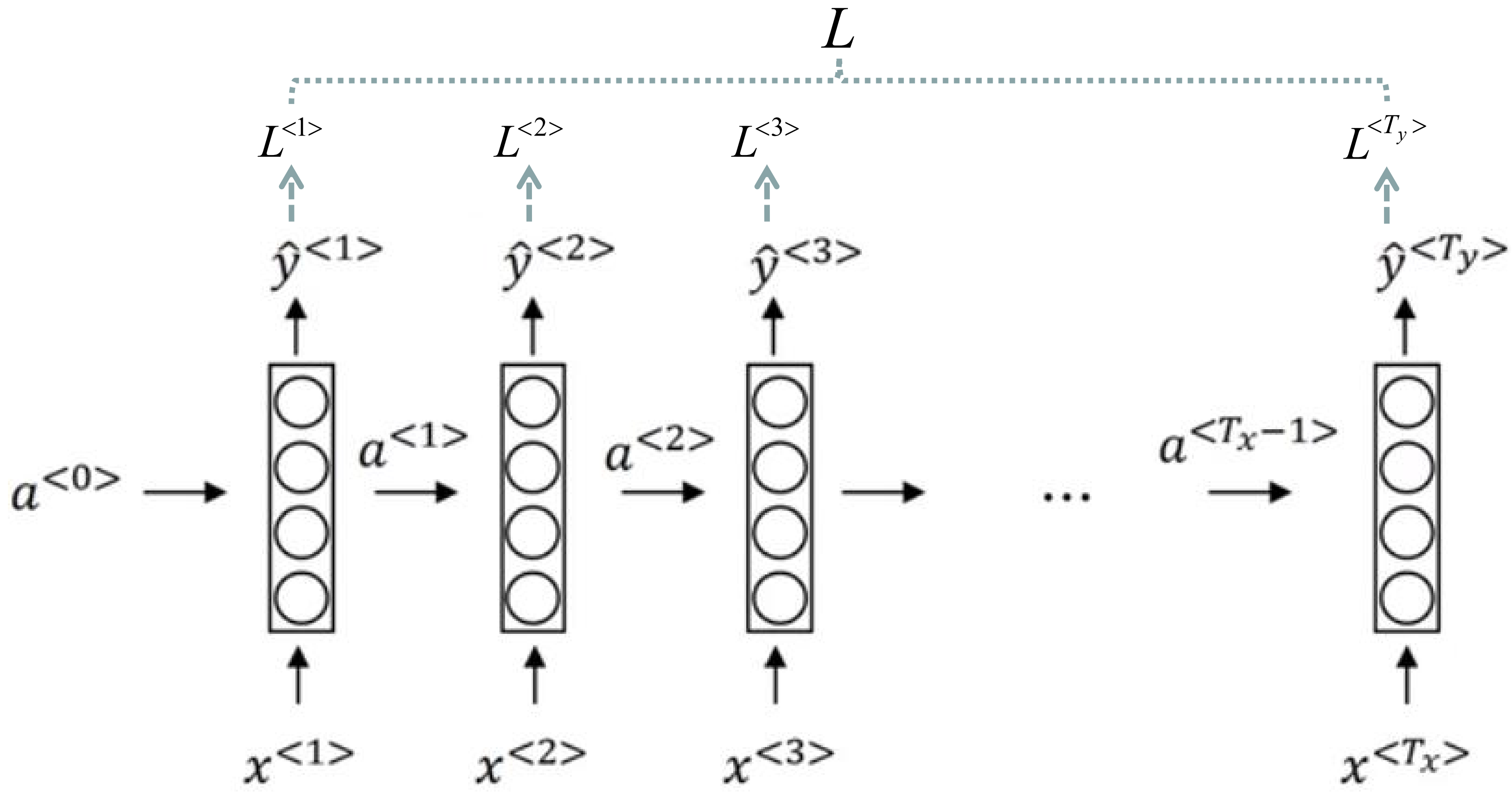
$$a^{<t>} = g_1(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$$ ⟵ tanh | ReLU

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

$$max(0, x)$$

$$\hat{y}^{<t>} = g_2(W_{ya}a^{<t>} + b_y)$$ ⟵ softmax | sigmoid

Forword propagation and backpropagaton



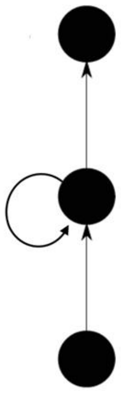
$$L^{<t>}(\hat{y}^{<t>}, y^{<t>}) = -y^{<t>} \log \hat{y}^{<t>} - (1 - y^{<t>}) \log(1 - \hat{y}^{<t>})$$

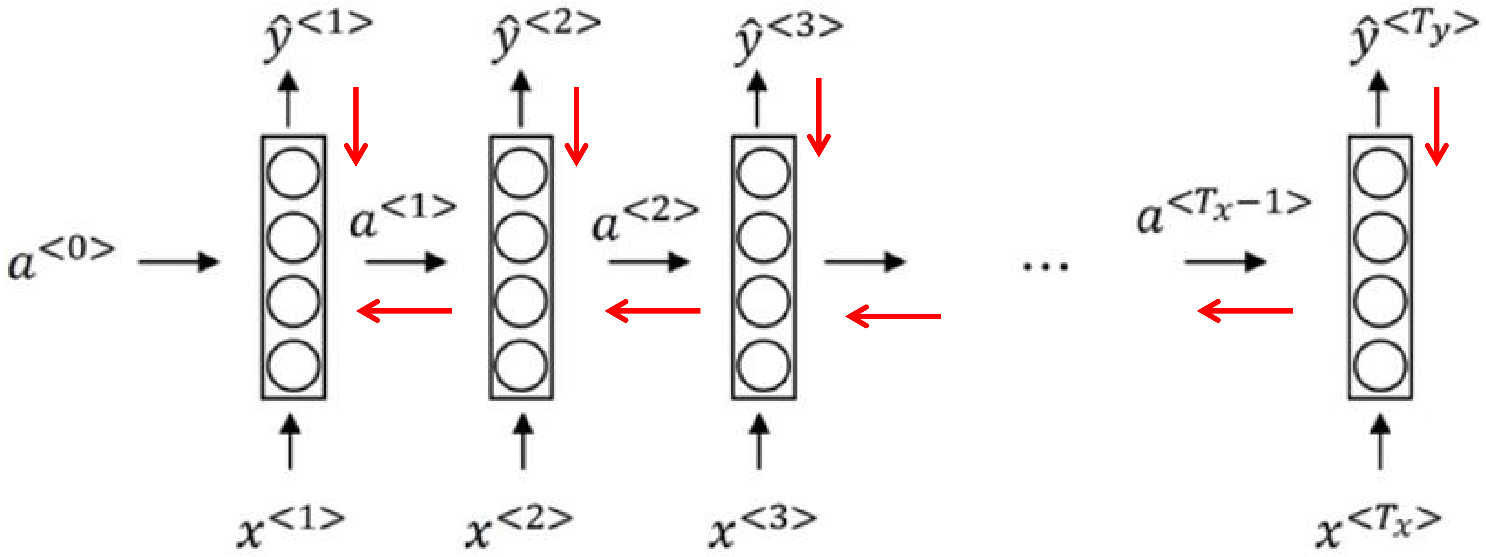
$$L(\hat{y}, y) = \sum_{t=1}^{T_y} L^{<t>}(\hat{y}^{<t>}, y^{<t>})$$

Back Propagation Through Time

BPTT

- 后向传播





$$a^{<t>} = g_1(z_a^{<t>}) = g_1\left(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a\right)$$

$$\hat{y}^{<t>} = g_2(z_y^{<t>}) = g_2\left(W_{ya}a^{<t>} + b_y\right)$$

$$L(\hat{y}, y) = \sum_{t=1}^{T_y} L^{<t>}(\hat{y}^{<t>}, y^{<t>})$$

$$\Delta W = \text{-}\eta\frac{\partial L}{\partial W} = \text{-}\eta\frac{\partial L}{\partial z^{<t>}}\frac{\partial z^{<t>}}{\partial W}$$

$$\delta^{<t>} \hat{=} \frac{\partial L}{\partial z^{<t>}} = \frac{\partial L}{\partial \hat{y}^{<t>}}\frac{\partial \hat{y}^{<t>}}{\partial z^{<t>}}$$

$$\frac{\partial L^{<t>}}{\partial W_{ya}} = \frac{\partial L^{<t>}}{\partial \hat{y}^{<t>}}\frac{\partial \hat{y}^{<t>}}{\partial z_y^{<t>}}\frac{\partial z_y^{<t>}}{\partial W_{ya}} = \frac{\partial L^{<t>}}{\partial \hat{y}^{<t>}} g_2'(z_y^{<t>})a^{<t>} = \delta_y^{<t>}a^{<t>}$$

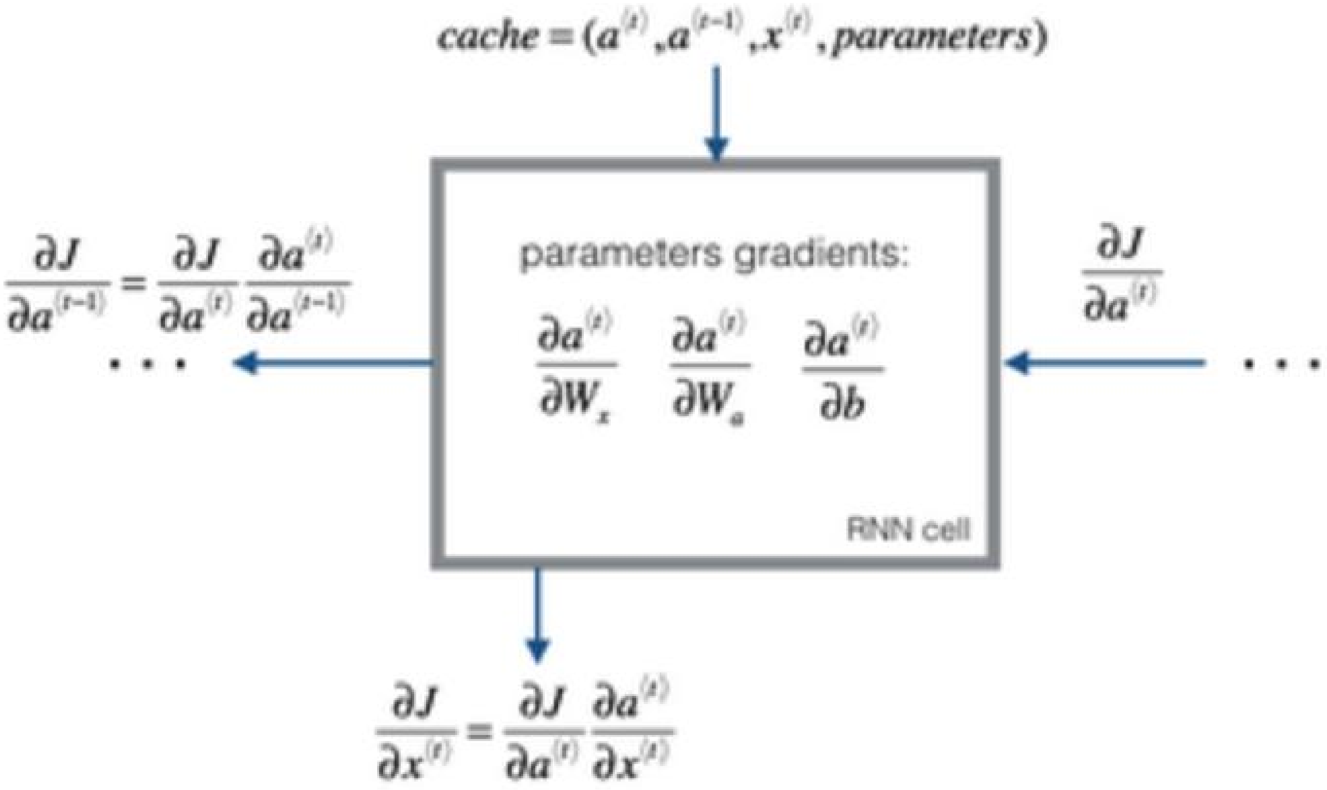
$$\delta_a^{<t>} = \frac{\partial L}{\partial a^{<t>}}\frac{\partial a^{<t>}}{\partial z_a^{<t>}} = g_1'(z_a^{<t>})\left(\sum_y \delta_y^{<t>} w_{ya} + \sum_{a'} \delta_{a'}^{<t+1>} w_{aa'}\right)$$

$$\frac{\partial L}{\partial W_{ya}} = \sum_{t=1}^{T_y}\frac{\partial L^{<t>}}{\partial W_{ya}} = \sum_t \delta_y^{<t>}a^{<t>}$$

$$\frac{\partial L}{\partial W_{aa}} = \sum_{t=1}^{T_x}\frac{\partial L}{\partial z_a^{<t>}}\frac{\partial z_a^{<t>}}{\partial W_{aa}} = \sum_{t=1}^{T_x}\delta_a^{<t>}a^{<t-1>}$$

$$\frac{\partial L}{\partial W_{ax}} = \sum_{t=1}^{T_x}\frac{\partial L}{\partial z_a^{<t>}}\frac{\partial z_a^{<t>}}{\partial W_{ax}} = \sum_{t=1}^{T_x}\delta_a^{<t>}x^{<t>}$$

RNN反向传播示意



$cache = (a^{\langle t\rangle}, a^{\langle t-1\rangle}, x^{\langle t\rangle}, parameters)$
parameters gradients:
$\frac{\partial a^{\langle t\rangle}}{\partial W_x}$ $\frac{\partial a^{\langle t\rangle}}{\partial W_a}$ $\frac{\partial a^{\langle t\rangle}}{\partial b}$
RNN cell
$\frac{\partial J}{\partial a^{\langle t\rangle}}$
. . .
$\frac{\partial J}{\partial a^{\langle t-1\rangle}} = \frac{\partial J}{\partial a^{\langle t\rangle}} \frac{\partial a^{\langle t\rangle}}{\partial a^{\langle t-1\rangle}}$
. . .
$\frac{\partial J}{\partial x^{\langle t\rangle}} = \frac{\partial J}{\partial a^{\langle t\rangle}} \frac{\partial a^{\langle t\rangle}}{\partial x^{\langle t\rangle}}$

$$a^{\langle t\rangle} = \tanh(W_{ax}x^{\langle t\rangle} + W_{aa}a^{\langle t-1\rangle} + b)$$

$$\frac{\partial \tanh(x)}{\partial x} = 1 - \tanh(x)^2$$

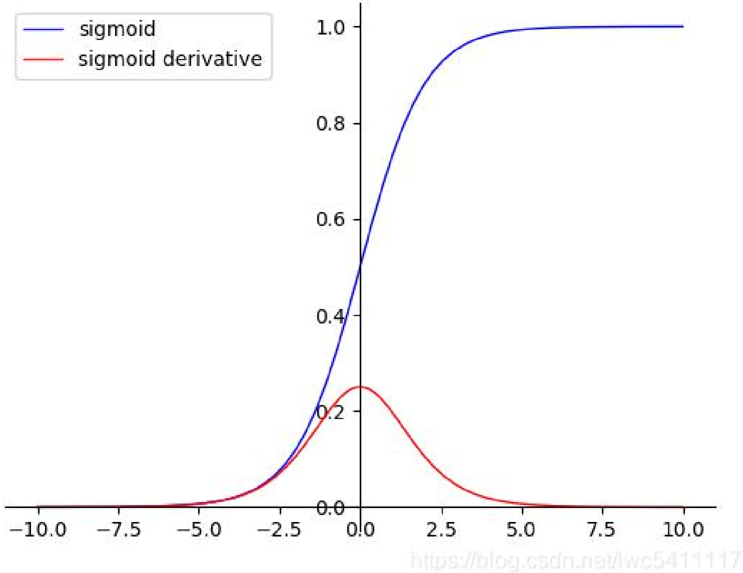
$$\frac{\partial a^{\langle t\rangle}}{\partial W_{ax}} = (1 - \tanh(W_{ax}x^{\langle t\rangle} + W_{aa}a^{\langle t-1\rangle} + b)^2).x^{\langle t\rangle T}$$

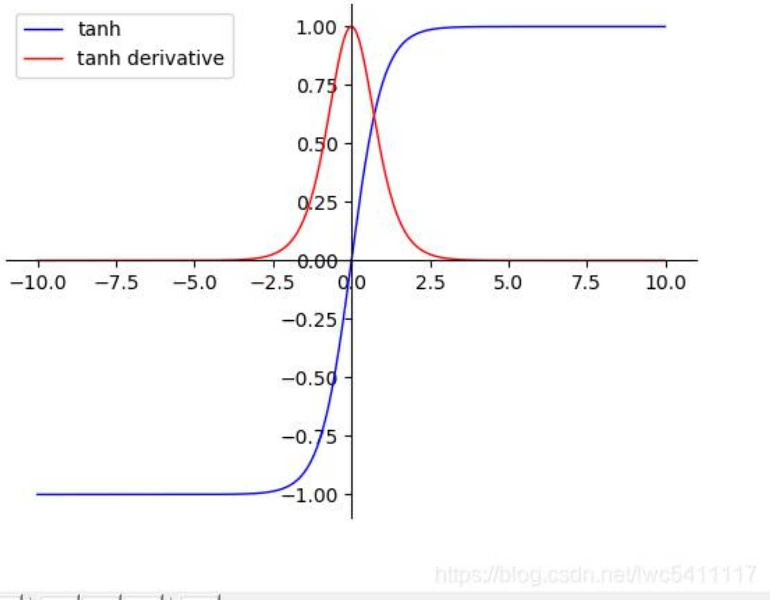
$$\frac{\partial a^{\langle t\rangle}}{\partial W_{aa}} = (1 - \tanh(W_{ax}x^{\langle t\rangle} + W_{aa}a^{\langle t-1\rangle} + b)^2).a^{\langle t-1\rangle T}$$

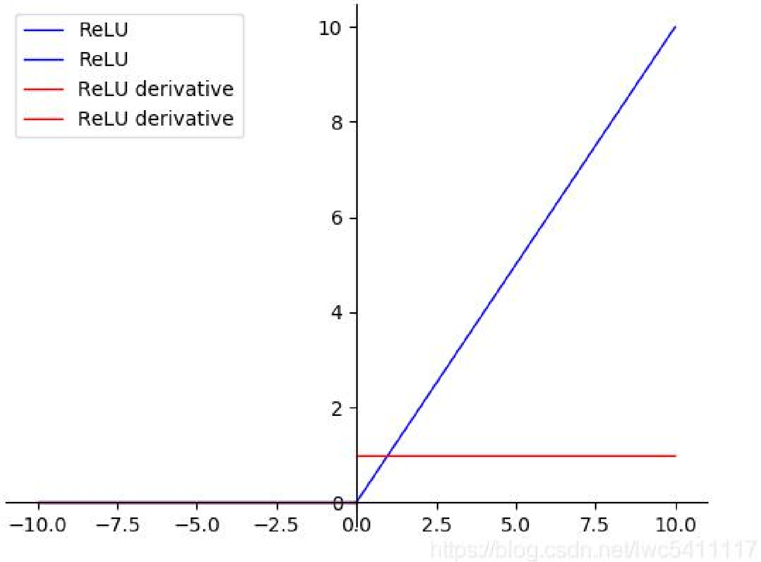
$$\frac{\partial a^{\langle t\rangle}}{\partial b} = \sum_{batch}(1 - \tanh(W_{ax}x^{\langle t\rangle} + W_{aa}a^{\langle t-1\rangle} + b)^2)$$

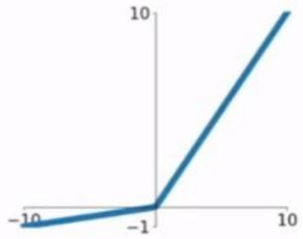
$$\frac{\partial a^{\langle t\rangle}}{\partial x^{\langle t\rangle}} = W_{ax}{}^T.(1 - \tanh(W_{ax}x^{\langle t\rangle} + W_{aa}a^{\langle t-1\rangle} + b)^2)$$

$$\frac{\partial a^{\langle t\rangle}}{\partial a^{\langle t-1\rangle}} = W_{aa}{}^T.(1 - \tanh(W_{ax}x^{\langle t-1\rangle} + W_{aa}a^{\langle t-1\rangle} + b)^2)$$





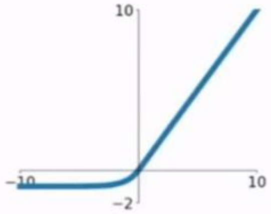




Leaky ReLU

$$f(x) = \max(0.01x, x)$$

Exponential Linear Units (ELU)



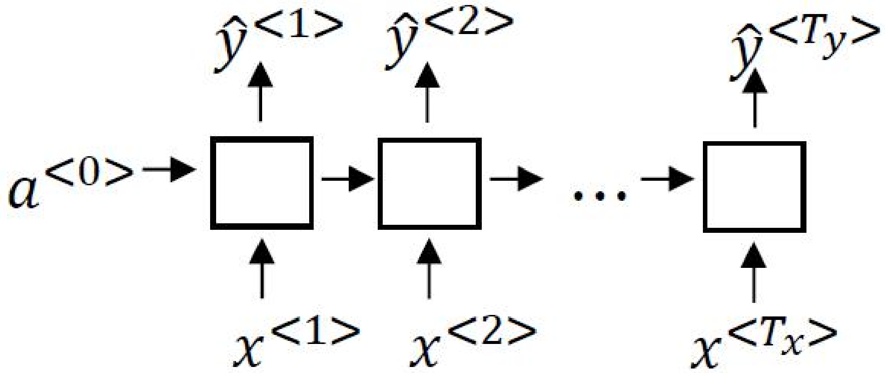
$$f(x) = \begin{cases} x & \text{if } x > 0 \\ \alpha\,(\exp(x) - 1) & \text{if } x \le 0 \end{cases}$$

不同类型的RNN结构

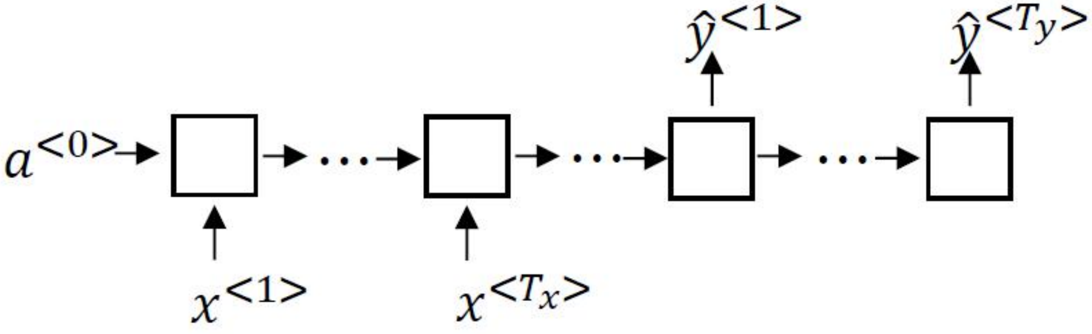
Speech recognition		→	“The quick brown fox jumped over the lazy dog.”
Music generation	Ø	→	
Sentiment classification	“There is nothing to like in this movie.”	→	★☆☆☆☆
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	→	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing with me?
Video activity recognition		→	Running
Name entity recognition	Yesterday, Harry Potter met Hermione Granger.	→	Yesterday, Harry Potter met Hermione Granger.

Andrew Ng

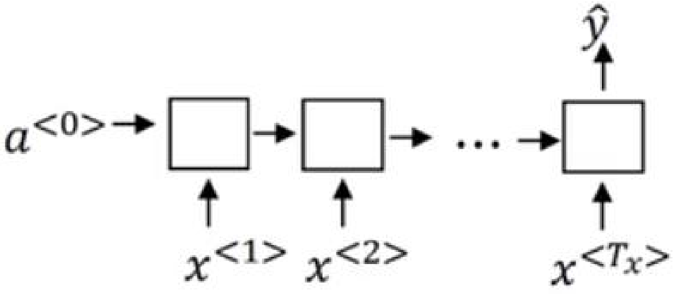
不同类型的RNN结构



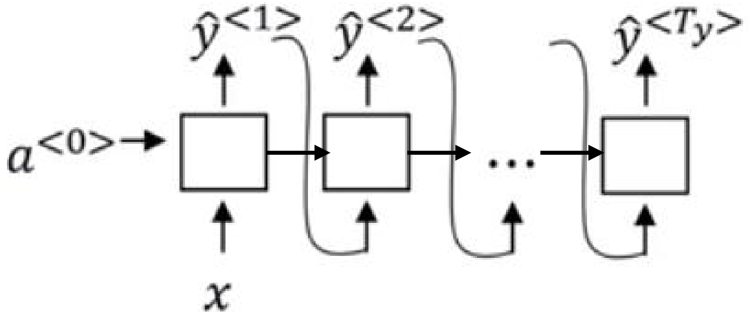
Many to many



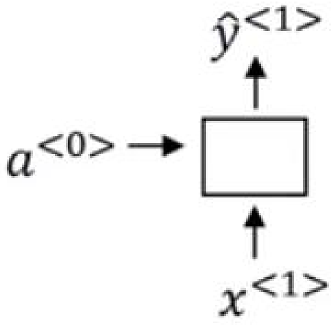
Many to many



Many to one



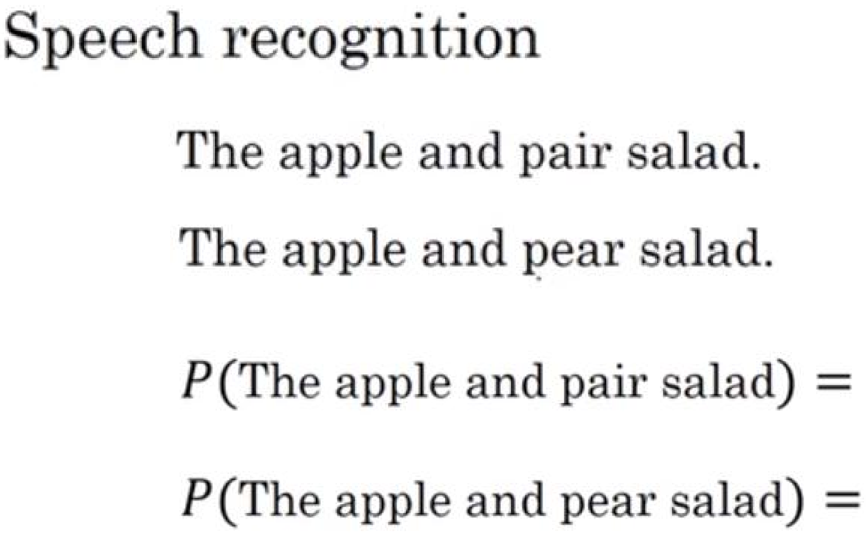
One to many



One to one

RNN示例：语言模型

- 语言模型



Speech recognition

The apple and pair salad.

The apple and pear salad.

$P(\text{The apple and pair salad}) =$ 3.2×10^{-13}

$P(\text{The apple and pear salad}) =$ 6.7×10^{-10}

P(Sentence) = ? $P\left(y^{<1>}, y^{<2>}, \ldots, y^{<T_y>}\right)$

Language modelling with an RNN

Training set: large corpus of english text.

Cats average 15 hours of sleep a day. <EOS>

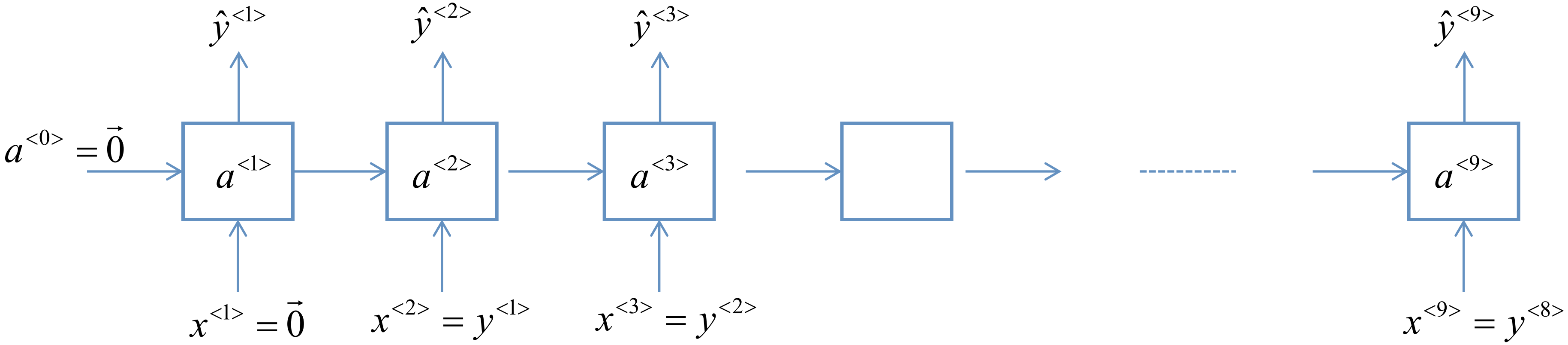
$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ …… $y^{<8>}$ $y^{<9>}$

$x^{<t>} = y^{<t-1>}$

The Egyptian ~~Mau~~ is a bread of cat. <EOS>

<UNK>

RNN model



Cats average 15 hours of sleep a day. <EOS>

$$\mathcal{L}(\hat{y}^{<t>}, y^{<t>}) = -\sum_i y_i^{<t>} \log \hat{y}_i^{<t>}$$

$$\mathcal{L} = \sum_t \mathcal{L}^{<t>}(\hat{y}^{<t>}, y^{<t>})$$

$$y^{<1>} \quad y^{<2>} \quad y^{<3>}$$

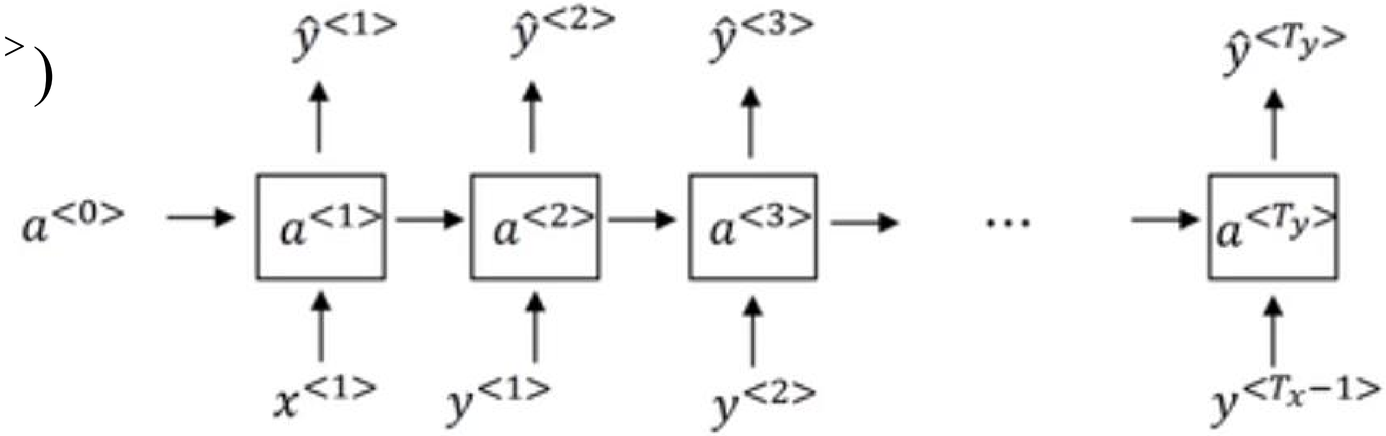
$$P\left(y^{<1>}, y^{<2>}, y^{<3>}\right)$$
$$= P\left(y^{<1>}\right) P\left(y^{<2>} \middle| y^{<1>}\right) P\left(y^{<3>} \middle| y^{<1>}, y^{<2>}\right)$$

新序列采样

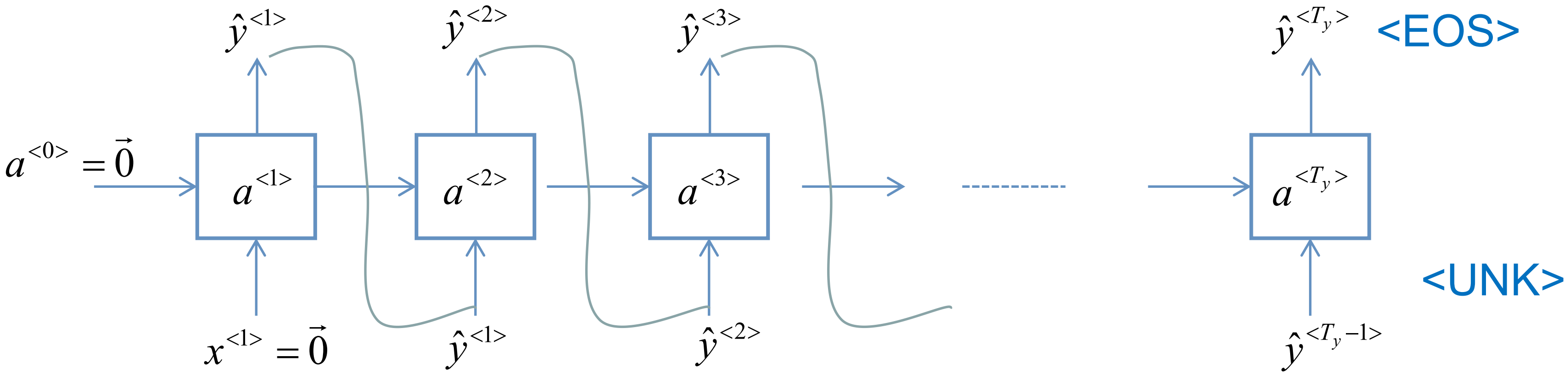
- Sampling a sequence from a trained RNN

$$P(y^{<1>}, y^{<2>}, \ldots, y^{<T_y>})$$

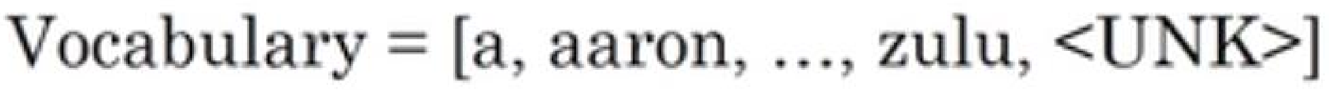
Training:



Sampling:



Character-level language model



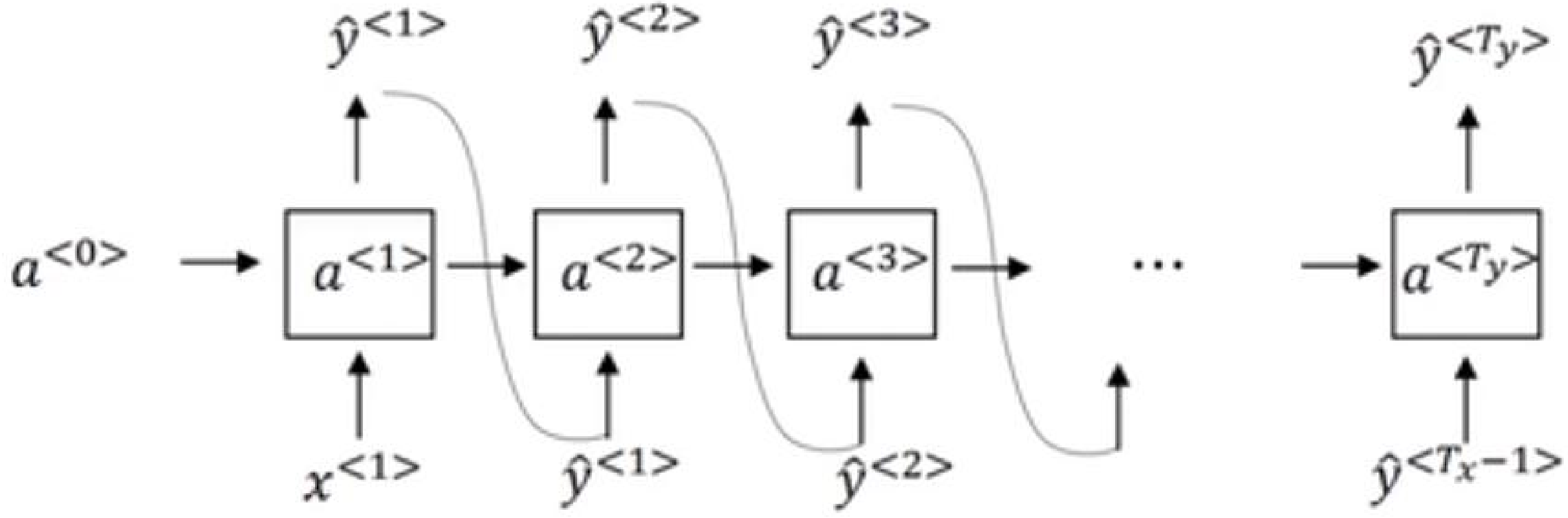
Vocabulary = [a, aaron, ..., zulu, <UNK>]

Vocabulary=[a,b,...,z,A,B,...Z, ,0,...,9]

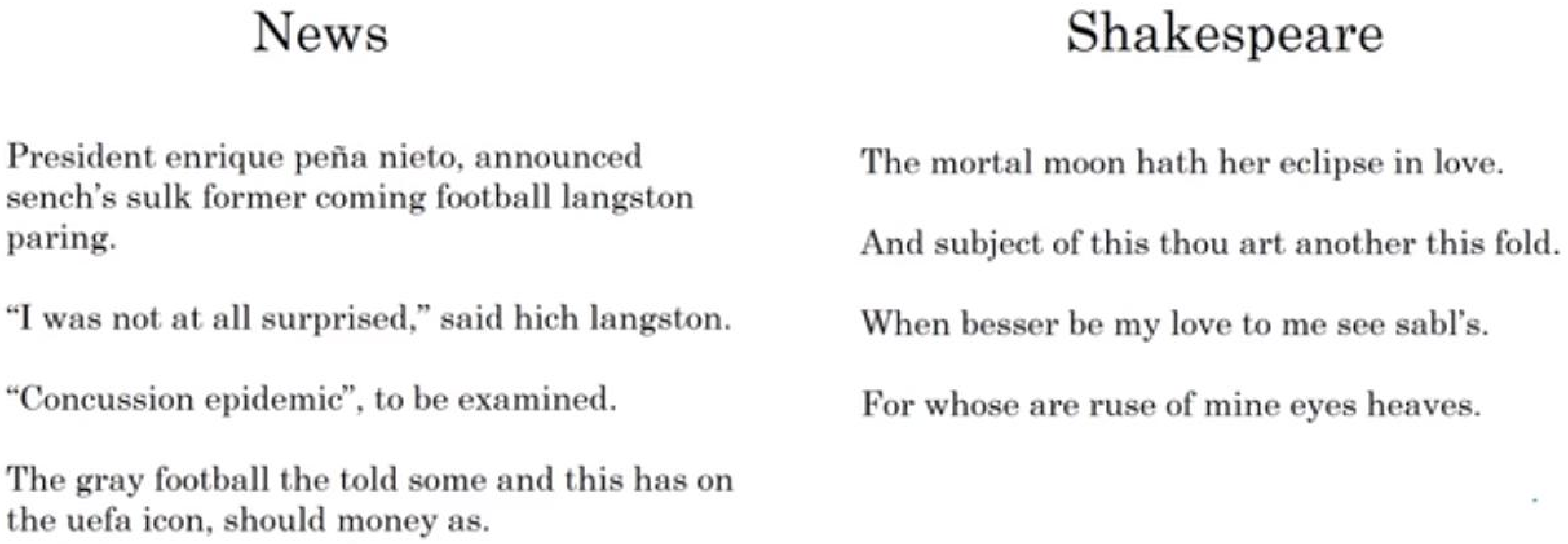
$y^{<t>}$

Cats average ......

$y^{<1>}\ y^{<2>}\ y^{<3>}\ y^{<4>}\ y^{<5>}\ldots\ldots$



Sequence generation



News

President enrique peña nieto, announced sench's sulk former coming football langston paring.

"I was not at all surprised," said hich langston.

"Concussion epidemic", to be examined.

The gray football the told some and this has on the uefa icon, should money as.

Shakespeare

The mortal moon hath her eclipse in love.

And subject of this thou art another this fold.

When besser be my love to me see sabl's.

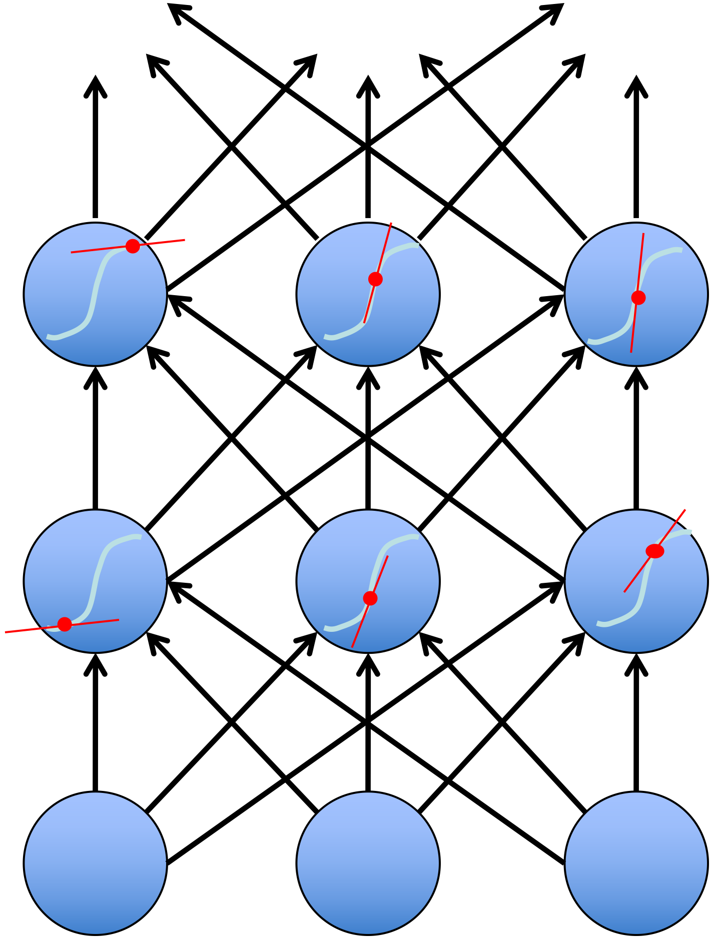
For whose are ruse of mine eyes heaves.

长序列循环神经网络

BP的困难

- The backward pass is linear
  - There is a big difference between the forward and backward passes.
  - In the forward pass we use squashing functions (like the logistic) to prevent the activity vectors from exploding.
  - The backward pass, is completely linear. If you double the error derivatives at the final layer, all the error derivatives will double.
    - The forward pass determines the slope of the linear function used for backpropagating through each neuron.

$$\delta_a^{<t>} = g_1'(z_a^{<t>})\left(\sum_y \delta_y^{<t>} w_{ya} + \sum_{a'} \delta_{a'}^{<t+1>} w_{aa'}\right)$$

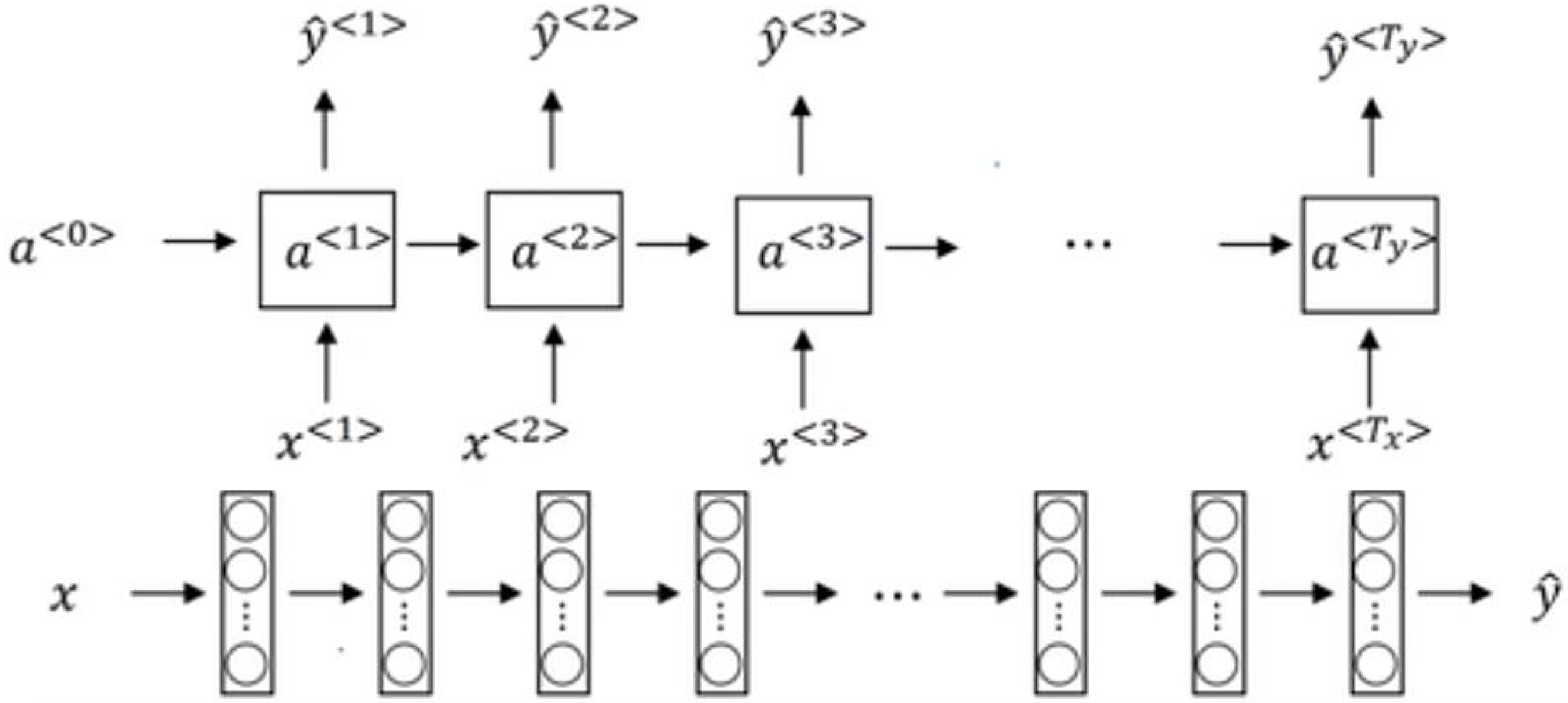


梯度的膨胀或消散

- 如果向后传播很多层，梯度 $\frac{\partial a^{(t)}}{\partial a^{(t-1)}} = {W_{aa}}^{T}.(1-\tanh(W_{ax}x^{(t-1)} + W_{aa}a^{(t-1)} + b)^{2})$
  - If the weights are small, the gradients shrink exponentially.
  - If the weights are big, the gradients grow exponentially.
- 训练长序列 (100 time steps) RNN中，梯度很容易膨胀或消散
  - Avoid this by initializing the weights very carefully.
- 即使好的初始化，也难以检测当前目标输出对很多步之前的输入的依赖关系
  - So RNNs have difficulty dealing with long-range dependencies.

Vanishing gradients with RNNs

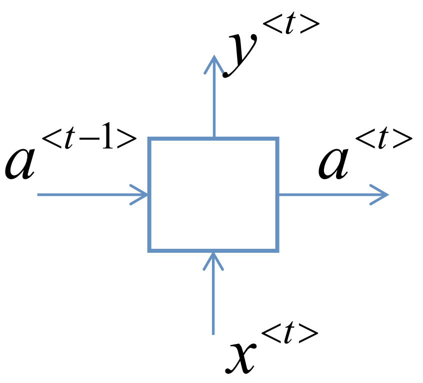
The cat, which already ate ……, was full.
The cats, which ate ……, were full.



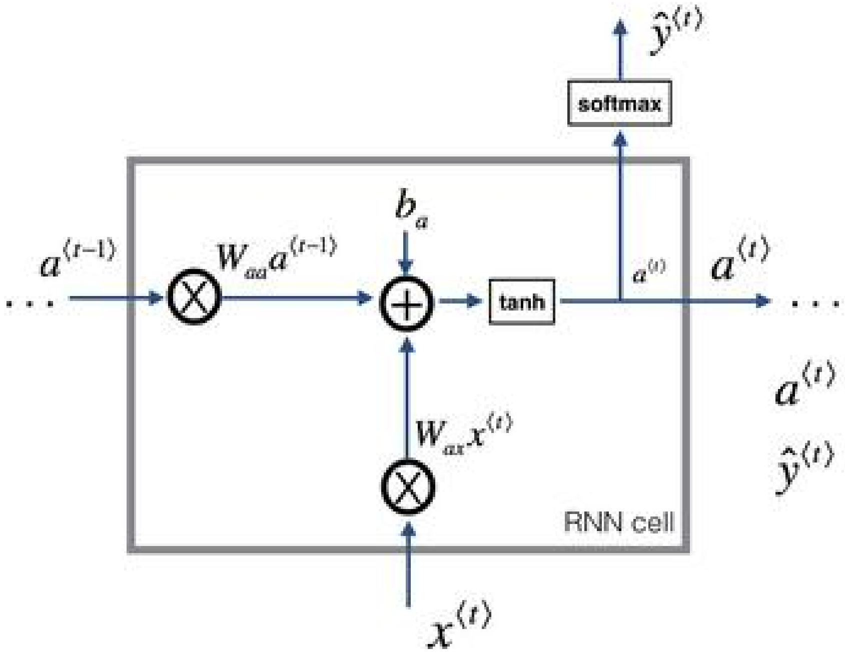
Exploding gradients. ← 梯度修剪 Gradient Clipping

GRU单元（Gated Recurrent Unit（GRU））

- RNN Unit



$$a^{<t>} = g(W_a[a^{<t-1>}, x^{<t>}] + b_a)$$



$$a^{\langle t\rangle} = \tanh(W_{ax}x^{\langle t\rangle} + W_{aa}a^{\langle t-1\rangle} + b_a)$$

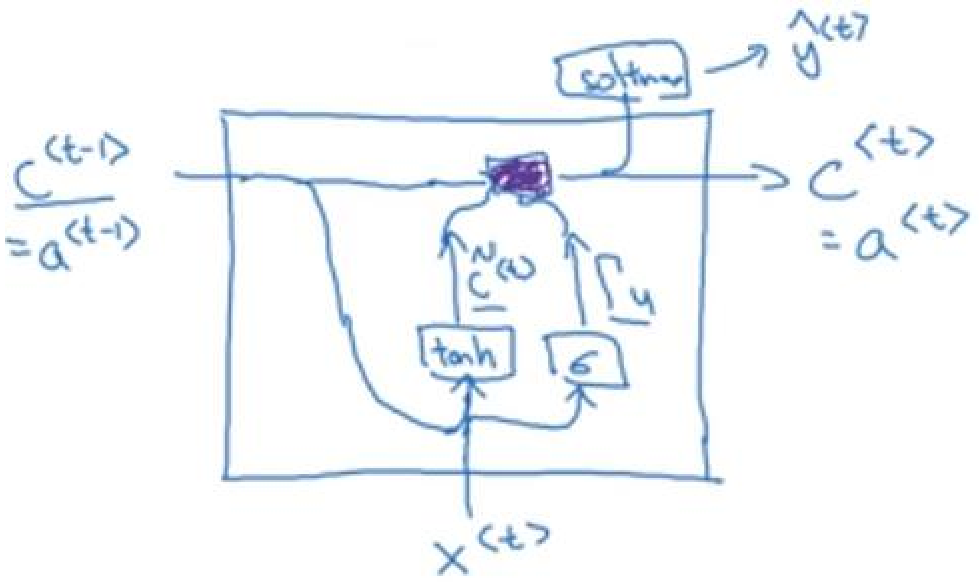
$$\hat{y}^{\langle t\rangle} = soft\max(W_{ya}a^{\langle t\rangle} + b_y)$$

The cat, which already ate ..., was full.

[Cho et al., 2014. On the properties of neural machine translation: Encoder-decoder approaches]
[Chung et al., 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling]

Gated Recurrent Unit（GRU）

- GRU(simplified)



C --- memory cell

$$c^{<t>} = a^{<t>}$$

$$\tilde{c}^{<t>} = \tanh\left(W_c\left[c^{<t-1>}, x^{<t>}\right] + b_c\right)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1-\Gamma_u) * c^{<t-1>}$$

$$\Gamma_u = \sigma\left(W_u\left[c^{<t-1>}, x^{<t>}\right] + b_u\right)$$

Γ_u=1 Γ_u=0 Γ_u=0 …… Γ_u=1

$c^{<t>}$=1 $c^{<t>}$=1 …… ……

The $\boxed{\text{cat}}$, which already ate …, was full.

Gated Recurrent Unit（GRU）

- Full GRU

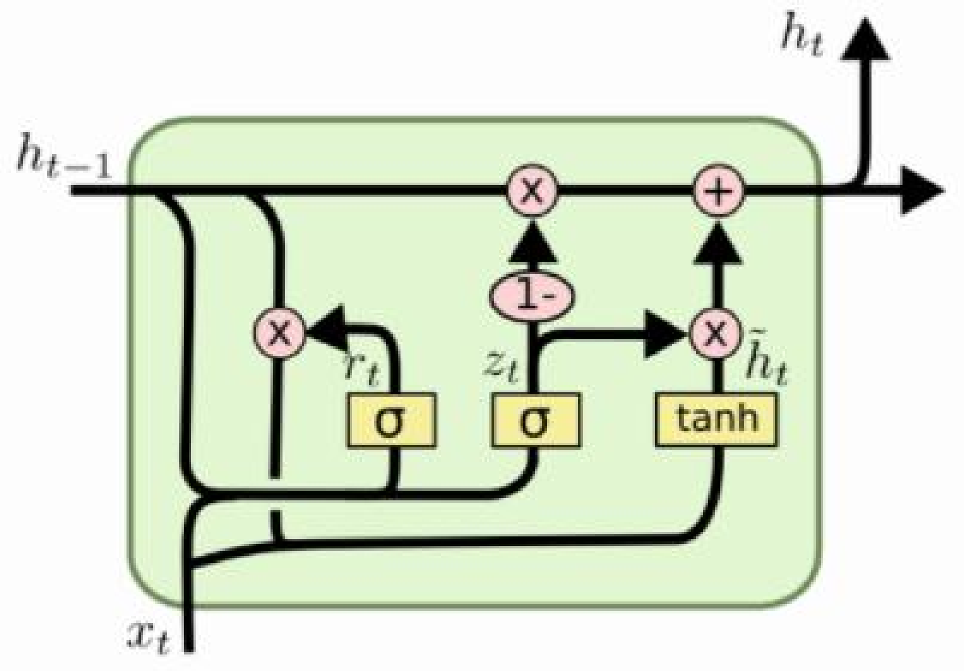
$$\tilde{c}^{<t>} = \tanh\left(W_c\left[\Gamma_r * c^{<t-1>}, x^{<t>}\right] + b_c\right)$$

$$\Gamma_u = \sigma\left(W_u\left[c^{<t-1>}, x^{<t>}\right] + b_u\right)$$

$$\Gamma_r = \sigma\left(W_r\left[c^{<t-1>}, x^{<t>}\right] + b_r\right)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

$$a^{<t>} = c^{<t>}$$



The cat, which already ate ..., was full.

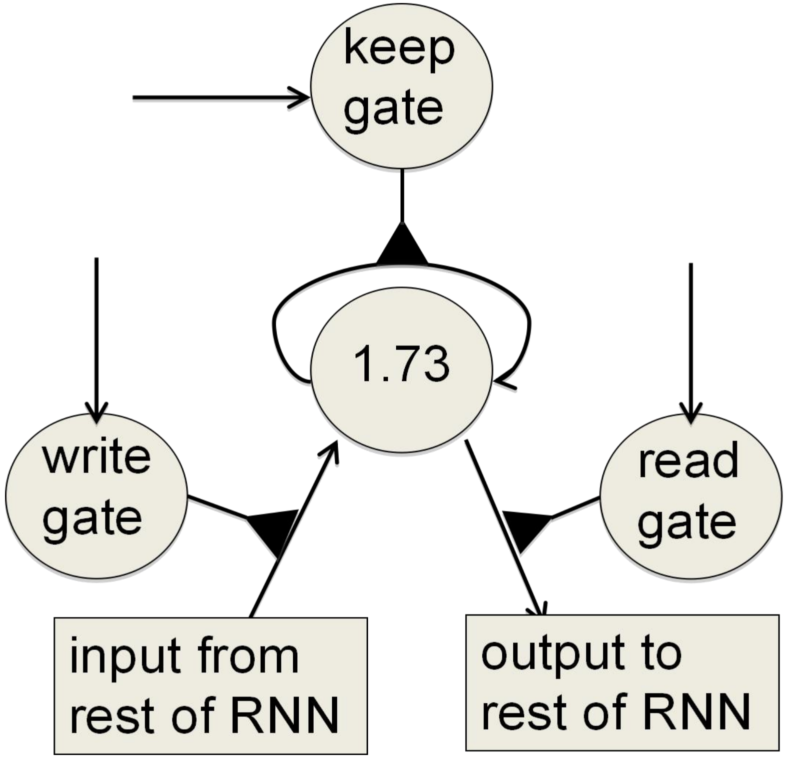
Long Short Term Memory, LSTM

- 解决了RNN长期(like hundreds of time steps)记忆的问题
  - Hochreiter & Schmidhuber (1997)

- LSTM是一个存储单元，使用logistic和linear单元执行乘法运算
  - Information gets into the cell whenever its “write” gate is on.
  - Information stays in the cell so long as its “keep/forget” gate is on.
  - Information can be read from the cell by turning on its “read” gate.

记忆单元(Memory Cell)

- To preserve information for a long time in the activities of an RNN
  - A linear unit that has a self-link with a weight of 1 will maintain its state.
  - Information is stored in the cell by activating its write gate.
  - Information is retrieved by activating the read gate.
  - We can backpropagate through this circuit because logistics are have nice derivatives.



$$g(z_a^{<t>}) = \frac{z_a^{<t>}}{w_{aa}}$$

$$\delta_a^{<t>} = g'(z_a^{<t>})\left(\sum_y \delta_y^{<t>} w_{ya} + \sum_{a'} \delta_{a'}^{<t+1>} w_{aa'}\right) \qquad \frac{\partial a^{<t+1>}}{\partial a^{<t>}} = g' w_{aa} = 1.0$$

$$a^{<t+1>} = g(z_a^{<t>}) = g(w_{aa} a^{<t>}) = a^{<t>}$$

GRU and LSTM

GRU

$$\tilde{c}^{<t>} = \tanh\left(W_c\left[\Gamma_r * c^{<t-1>}, x^{<t>}\right] + b_c\right)$$

$$\Gamma_u = \sigma\left(W_u\left[c^{<t-1>}, x^{<t>}\right] + b_u\right)$$

$$\Gamma_r = \sigma\left(W_r\left[c^{<t-1>}, x^{<t>}\right] + b_r\right)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + (1 - \Gamma_u) * c^{<t-1>}$$

$$a^{<t>} = c^{<t>}$$

LSTM

$$\tilde{c}^{<t>} = \tanh\left(W_c\left[a^{<t-1>}, x^{<t>}\right] + b_c\right)$$

$$\Gamma_u = \sigma\left(W_u\left[a^{<t-1>}, x^{<t>}\right] + b_u\right)$$

$$\Gamma_f = \sigma\left(W_f\left[a^{<t-1>}, x^{<t>}\right] + b_f\right)$$

$$\Gamma_o = \sigma\left(W_o\left[a^{<t-1>}, x^{<t>}\right] + b_o\right)$$

$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_o * c^{<t>}$$

LSTM

$$\tilde{c}^{<t>} = \tanh\left(W_c\left[a^{<t-1>}, x^{<t>}\right] + b_c\right)$$

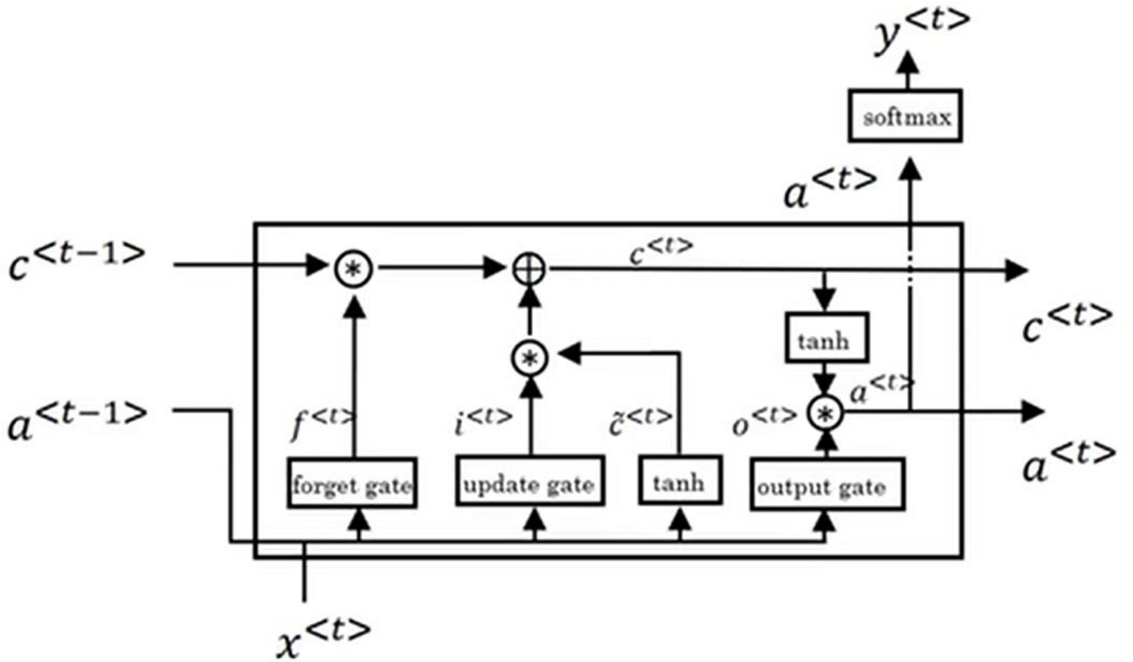
$$\Gamma_u = \sigma\left(W_u\left[a^{<t-1>}, x^{<t>}\right] + b_u\right)$$

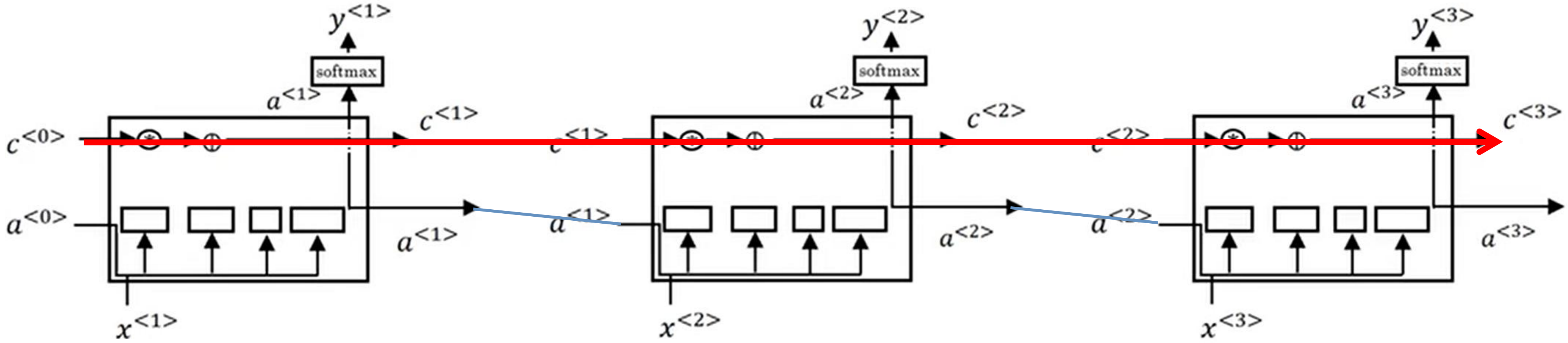
$$\Gamma_f = \sigma\left(W_f\left[a^{<t-1>}, x^{<t>}\right] + b_f\right)$$

$$\Gamma_o = \sigma\left(W_o\left[a^{<t-1>}, x^{<t>}\right] + b_o\right)$$

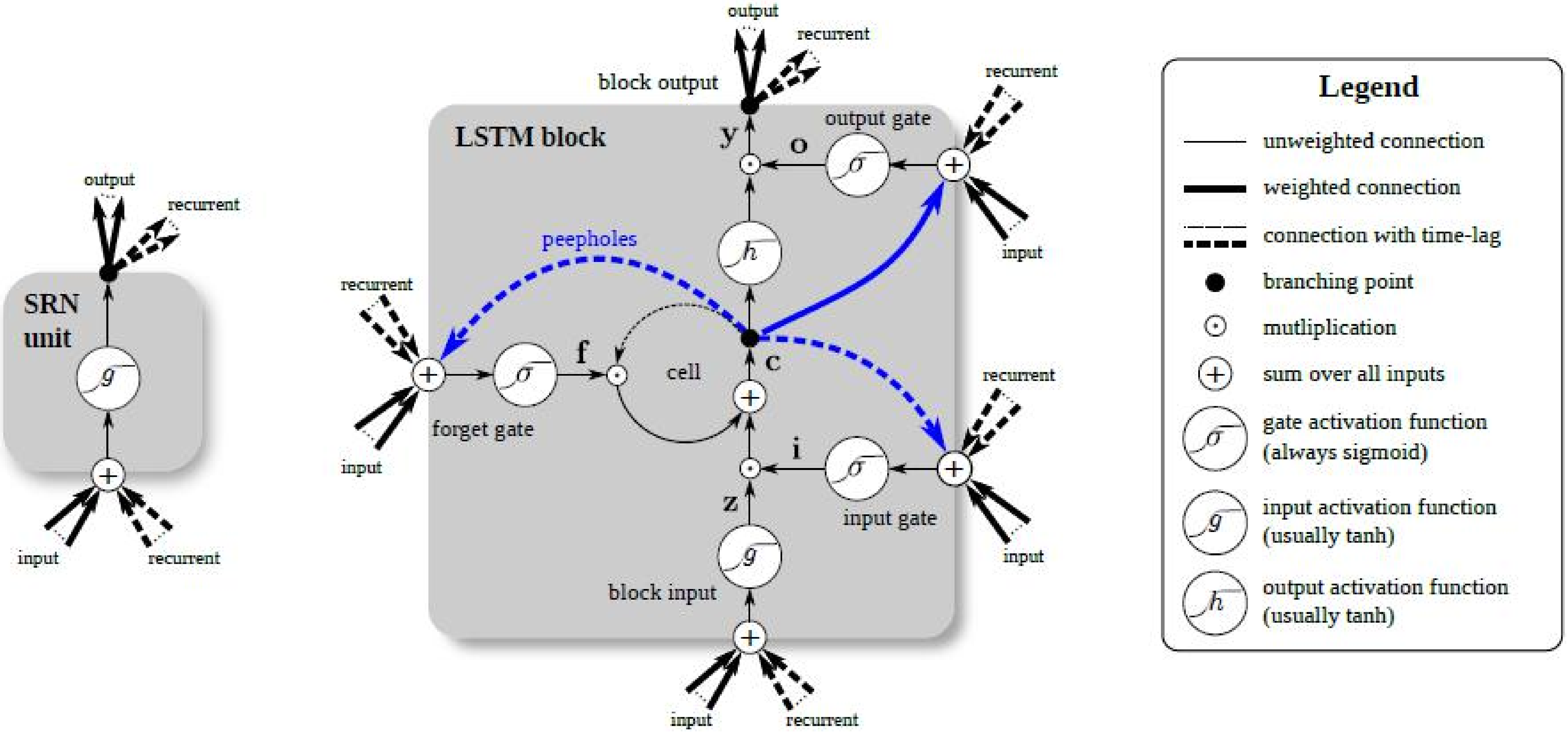
$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

$$a^{<t>} = \Gamma_o * \tanh(c^{<t>})$$

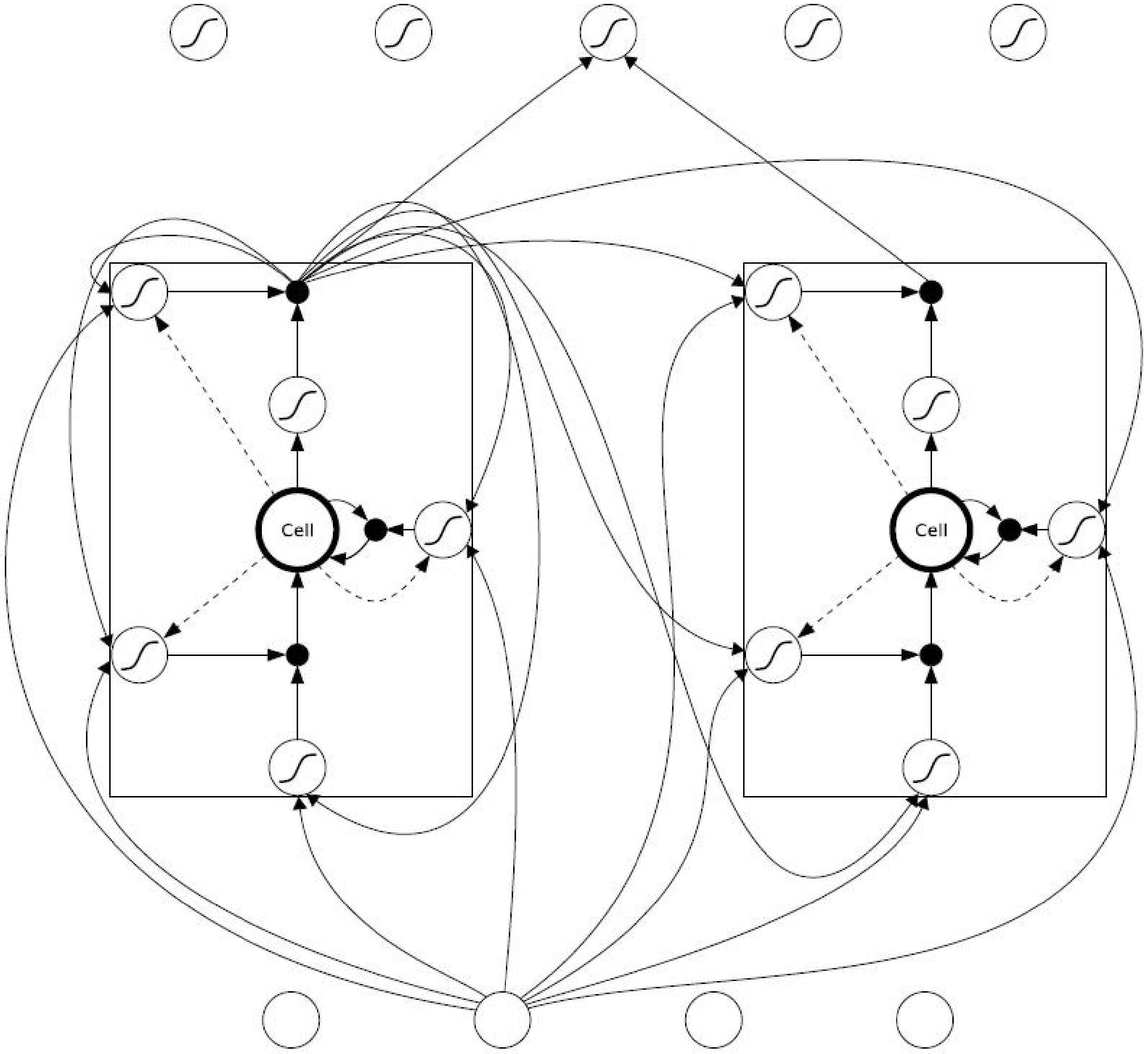




LSTM


output
recurrent
SRN unit
input
recurrent
output
recurrent
block output
recurrent
LSTM block
y
o
output gate
input
peepholes
recurrent
f
cell
c
forget gate
input
i
recurrent
z
input gate
input
block input
input
recurrent
Legend
unweighted connection
weighted connection
connection with time-lag
branching point
mutliplication
sum over all inputs
gate activation function (always sigmoid)
input activation function (usually tanh)
output activation function (usually tanh)

LSTM网络



- The network consists of four input units, a hidden layer of two single-cell LSTM memory blocks and five output units.
- Note that each block has four inputs but only one output.

LSTM

- 前向传播

Input Gates:

$$a_\iota^t = \sum_{i=1}^{I} w_{i\iota} x_i^t + \sum_{h=1}^{H} w_{h\iota} b_h^{t-1} + \sum_{c=1}^{C} w_{c\iota} s_c^{t-1}$$

$$b_\iota^t = f(a_\iota^t)$$

Cells:

$$a_c^t = \sum_{i=1}^{I} w_{ic} x_i^t + \sum_{h=1}^{H} w_{hc} b_h^{t-1}$$

$$s_c^t = b_\phi^t s_c^{t-1} + b_\iota^t g(a_c^t)$$

Cell Outputs: $b_c^t = b_\omega^t h(s_c^t)$

Forget Gates:

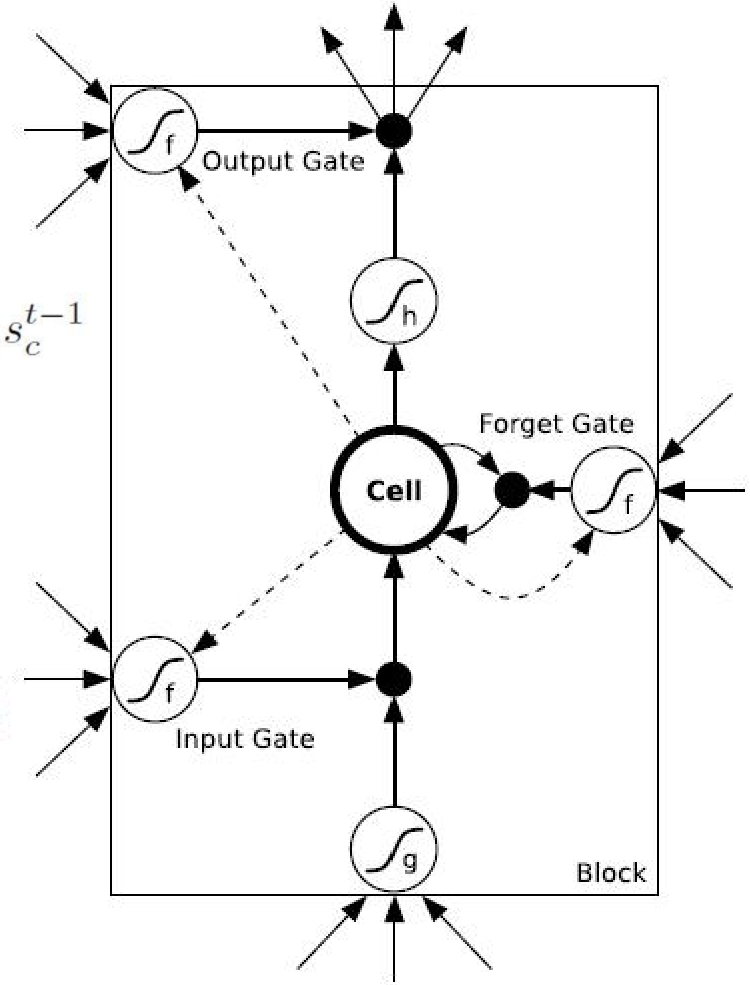
$$a_\phi^t = \sum_{i=1}^{I} w_{i\phi} x_i^t + \sum_{h=1}^{H} w_{h\phi} b_h^{t-1} + \sum_{c=1}^{C} w_{c\phi} s_c^{t-1}$$

$$b_\phi^t = f(a_\phi^t)$$

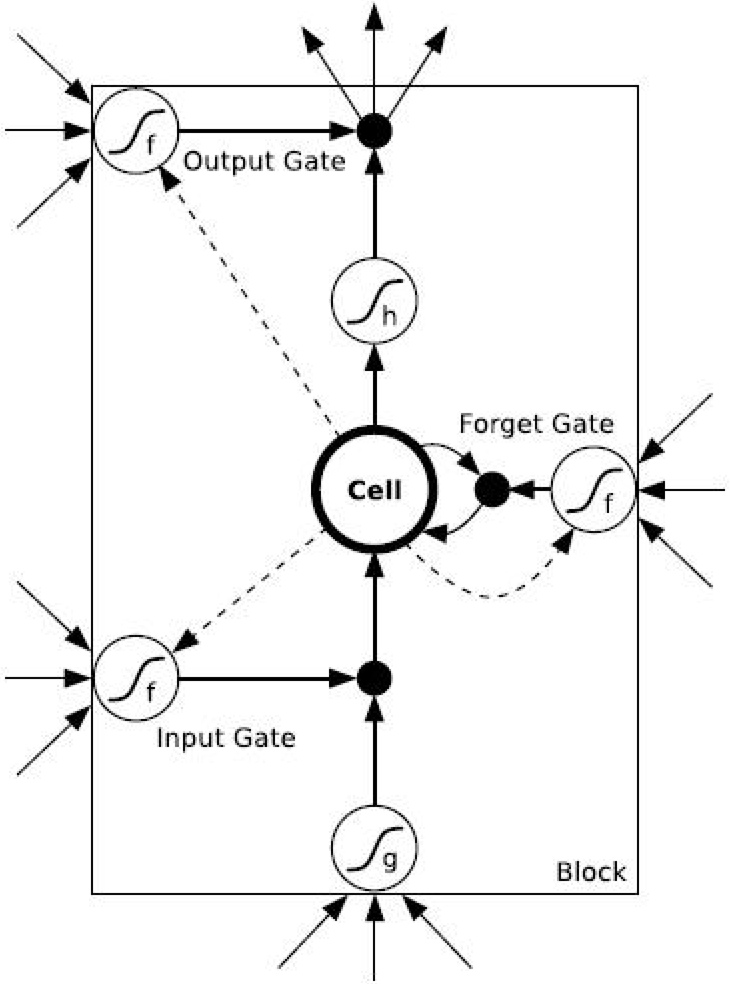
Output Gates:

$$a_\omega^t = \sum_{i=1}^{I} w_{i\omega} x_i^t + \sum_{h=1}^{H} w_{h\omega} b_h^{t-1} + \sum_{c=1}^{C} w_{c\omega} s_c^t$$

$$b_\omega^t = f(a_\omega^t)$$



- 后向传播



$$\epsilon_c^t \overset{\text{def}}{=} \frac{\partial \mathcal{L}}{\partial b_c^t} \qquad \epsilon_s^t \overset{\text{def}}{=} \frac{\partial \mathcal{L}}{\partial s_c^t}$$

Cell Outputs

$$\epsilon_c^t = \sum_{k=1}^{K} w_{ck}\delta_k^t + \sum_{g=1}^{G} w_{cg}\delta_g^{t+1}$$

Output Gates

$$\delta_\omega^t = f'(a_\omega^t)\sum_{c=1}^{C} h(s_c^t)\epsilon_c^t$$

States

$$\epsilon_s^t = b_\omega^t h'(s_c^t)\epsilon_c^t + b_\phi^{t+1}\epsilon_s^{t+1} + w_{c\iota}\delta_\iota^{t+1} + w_{c\phi}\delta_\phi^{t+1} + w_{c\omega}\delta_\omega^t$$

Cells

$$\delta_c^t = b_\iota^t g'(a_c^t)\epsilon_s^t$$

Forget Gates

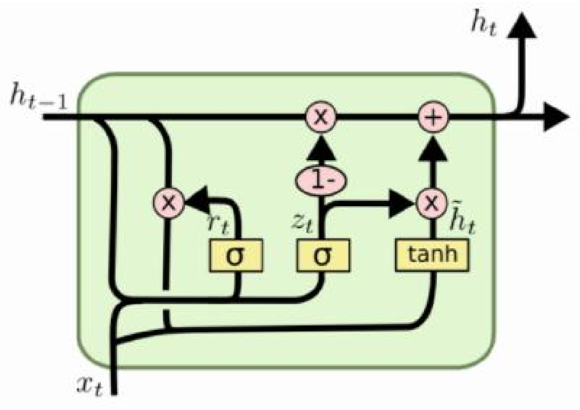
$$\delta_\phi^t = f'(a_\phi^t)\sum_{c=1}^{C} s_c^{t-1}\epsilon_s^t$$

Input Gates

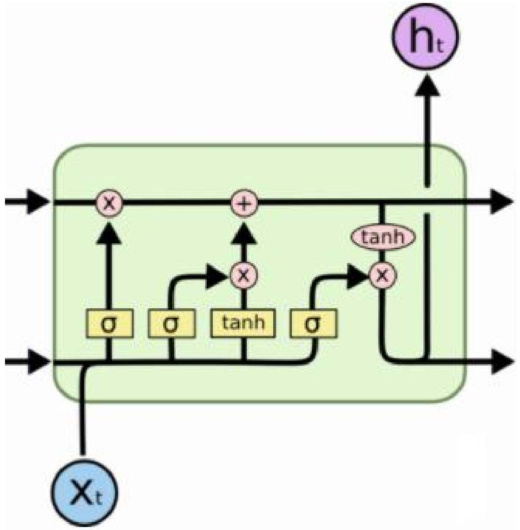
$$\delta_\iota^t = f'(a_\iota^t)\sum_{c=1}^{C} g(a_c^t)\epsilon_s^t$$

LSTM vs GRU

- GRU是更加简单的模型，更容易创建一个更大的网络，而且它只有两个门，在计算性上也运行得更快，可以扩大模型的规模。

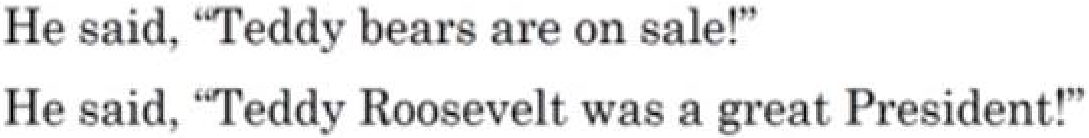


- LSTM更加强大和灵活，有三个门而不是两个。



双向循环神经网络（Bidirectional RNN）

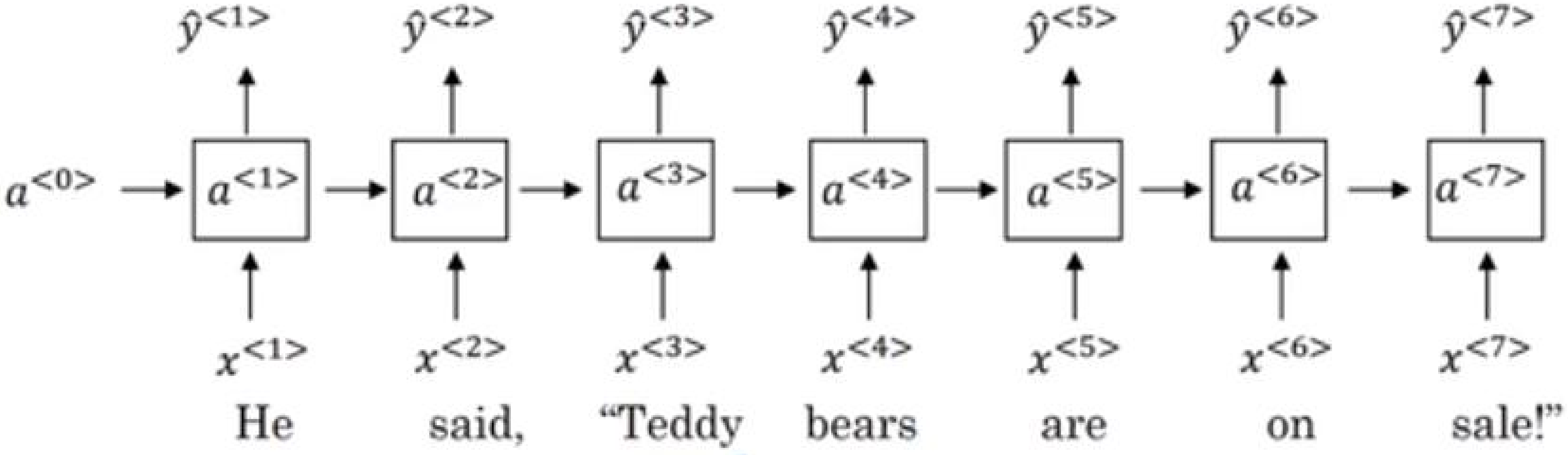
- Name Entity Recognition



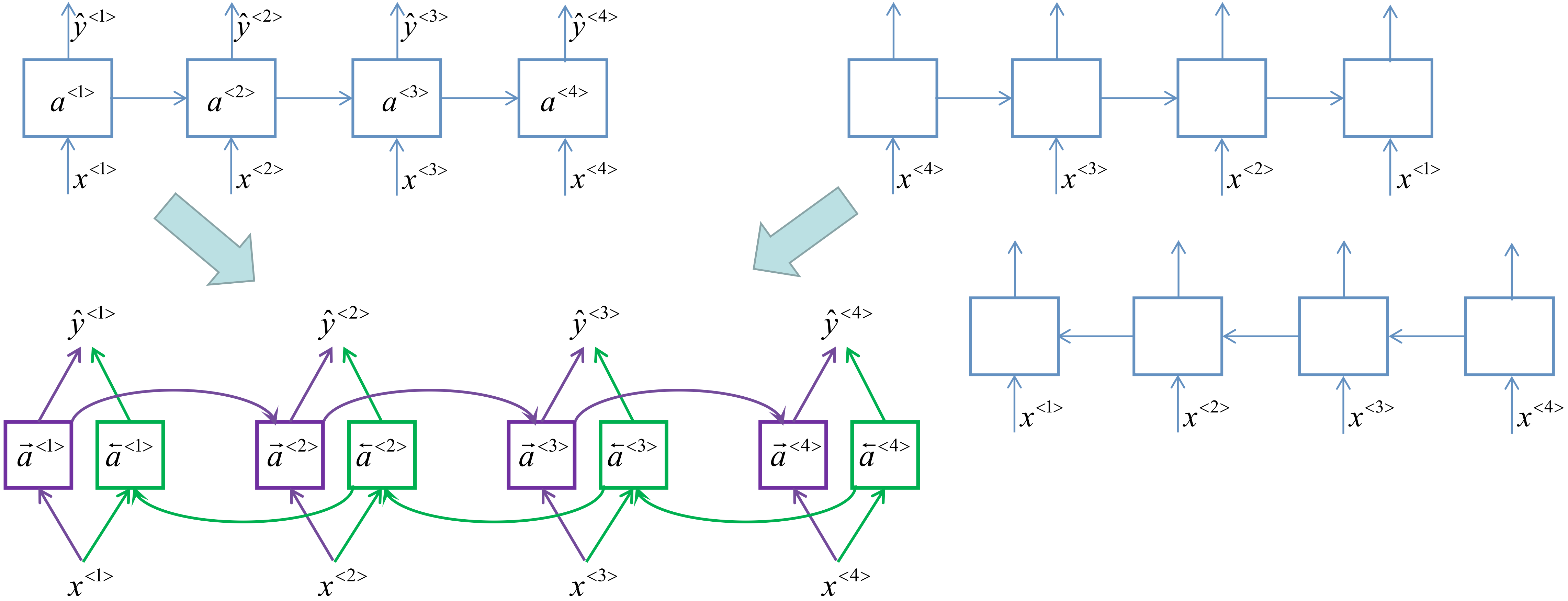
He said, “Teddy bears are on sale!”

He said, “Teddy Roosevelt was a great President!”

- RNN



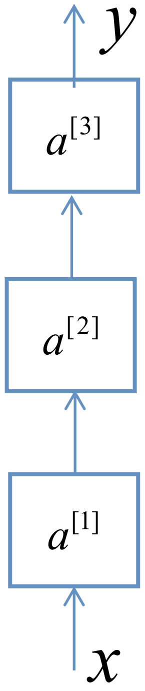
Bidirectional RNN (BRNN)

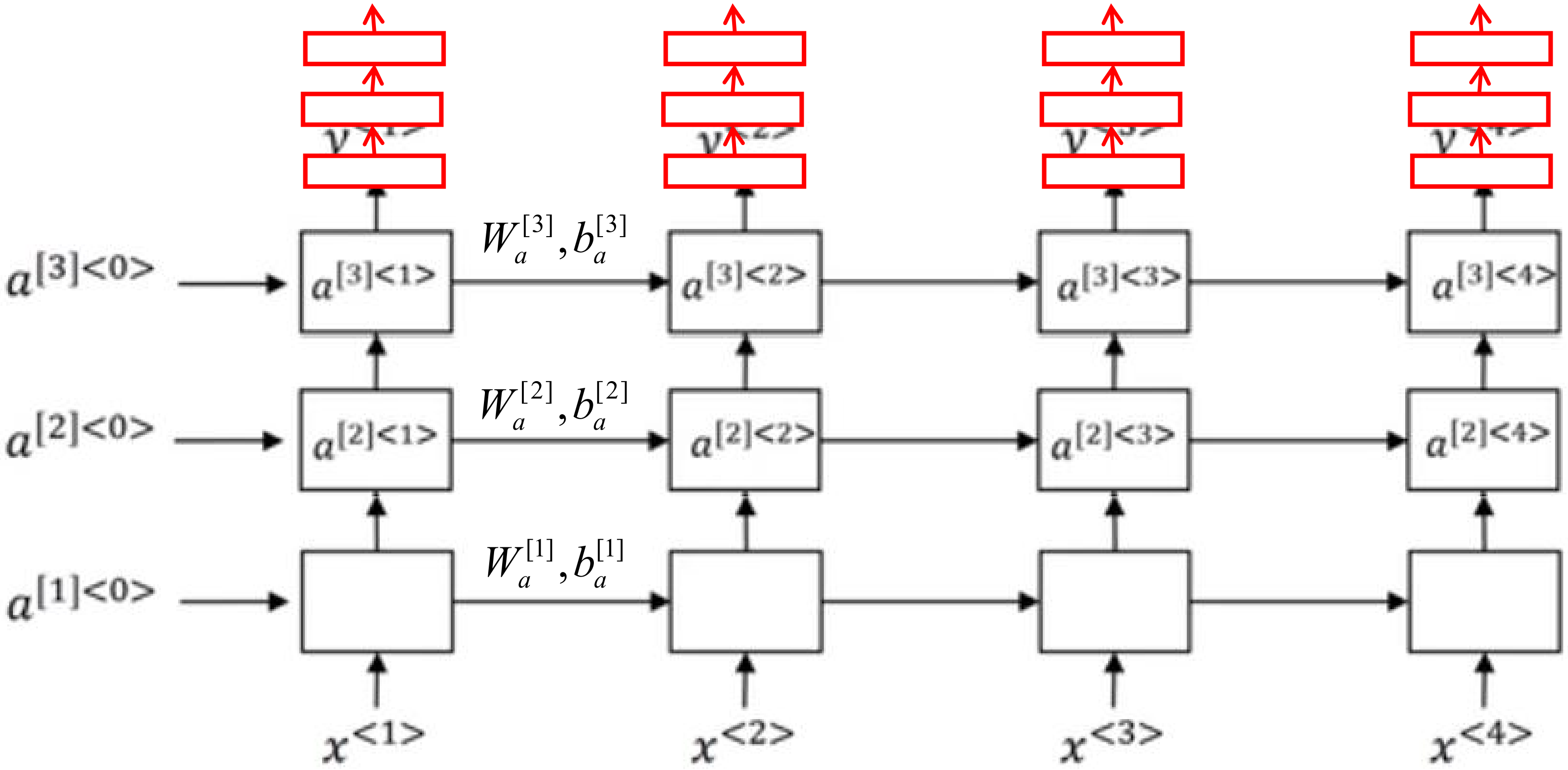


$$\hat{y}^{<t>} = g\left(W_y\left[\vec{a}^{<t>}, \overleftarrow{a}^{<t>}\right] + b_y\right)$$

He said Teddy Roosevelt...

深层循环神经网络（Deep RNNs）





$$a^{[2]<3>} = g(W_a^{[2]} \left[a^{[2]<2>}, a^{[1]<3>}\right] + b_a^{[2]})$$

序列模型

——sequence to sequence

Sequence to sequence model

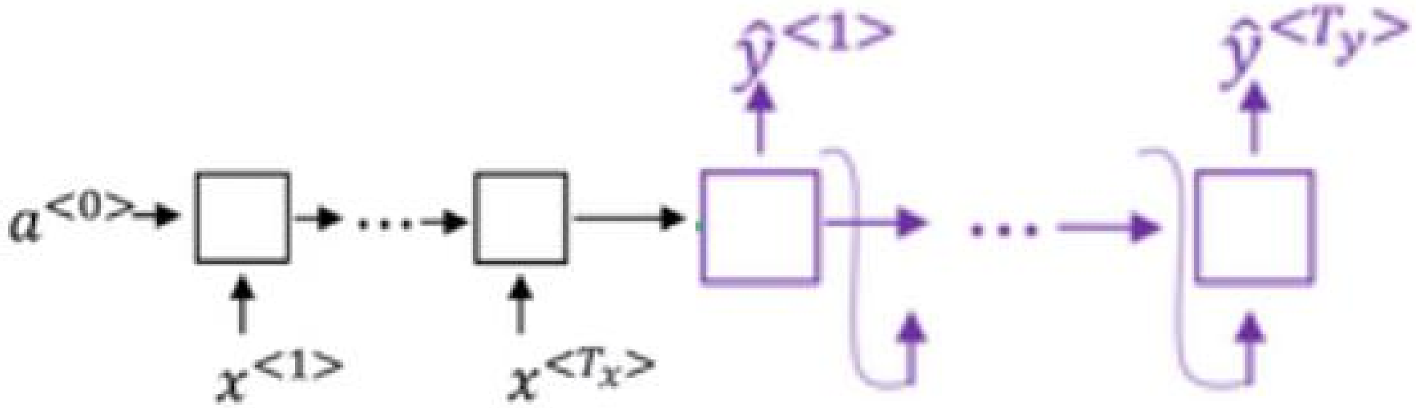
Machine Translation

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ $x^{<4>}$ $x^{<5>}$

Jane visite l'Afrique en septembre

→ Jane is visiting Africa in September.

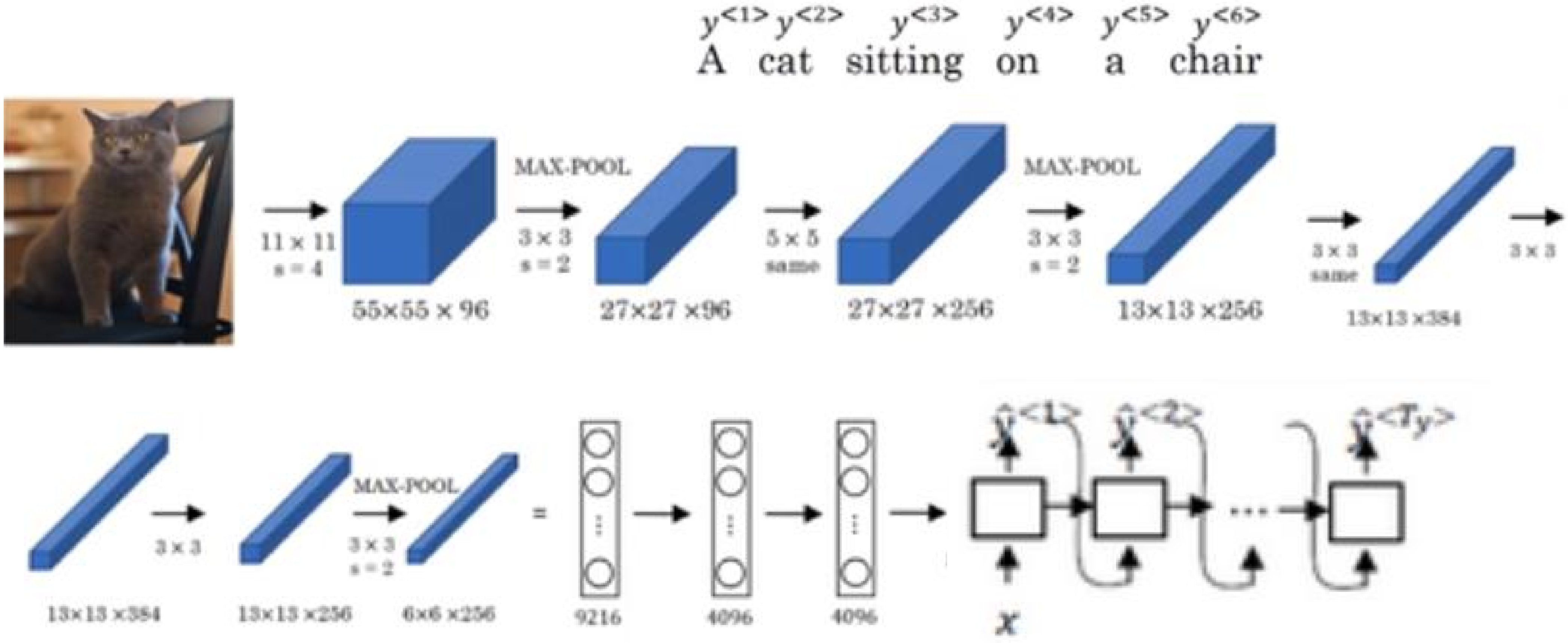
$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ $y^{<4>}$ $y^{<5>}$ $y^{<6>}$



[Sutskever et al., 2014. Sequence to sequence learning with neural networks]

[Cho et al., 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation]

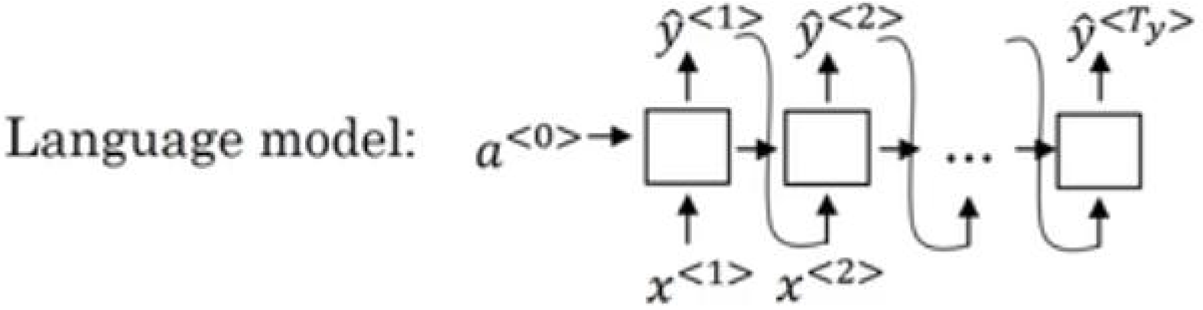
Image captioning



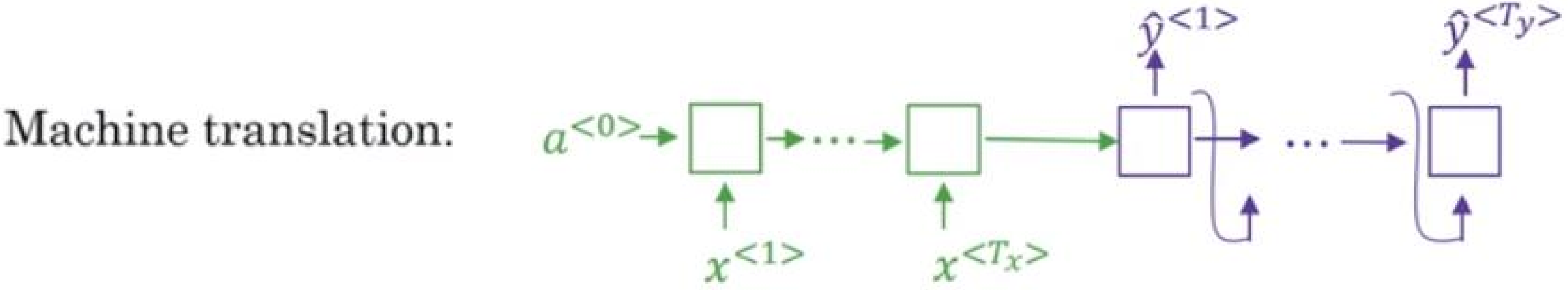
[Mao et. al., 2014. Deep captioning with multimodal recurrent neural networks]
[Vinyals et. al., 2014. Show and tell: Neural image caption generator]
[Karpathy and Li, 2015. Deep visual-semantic alignments for generating image descriptions]

Machine translation

- Machine translation as building a conditional language model



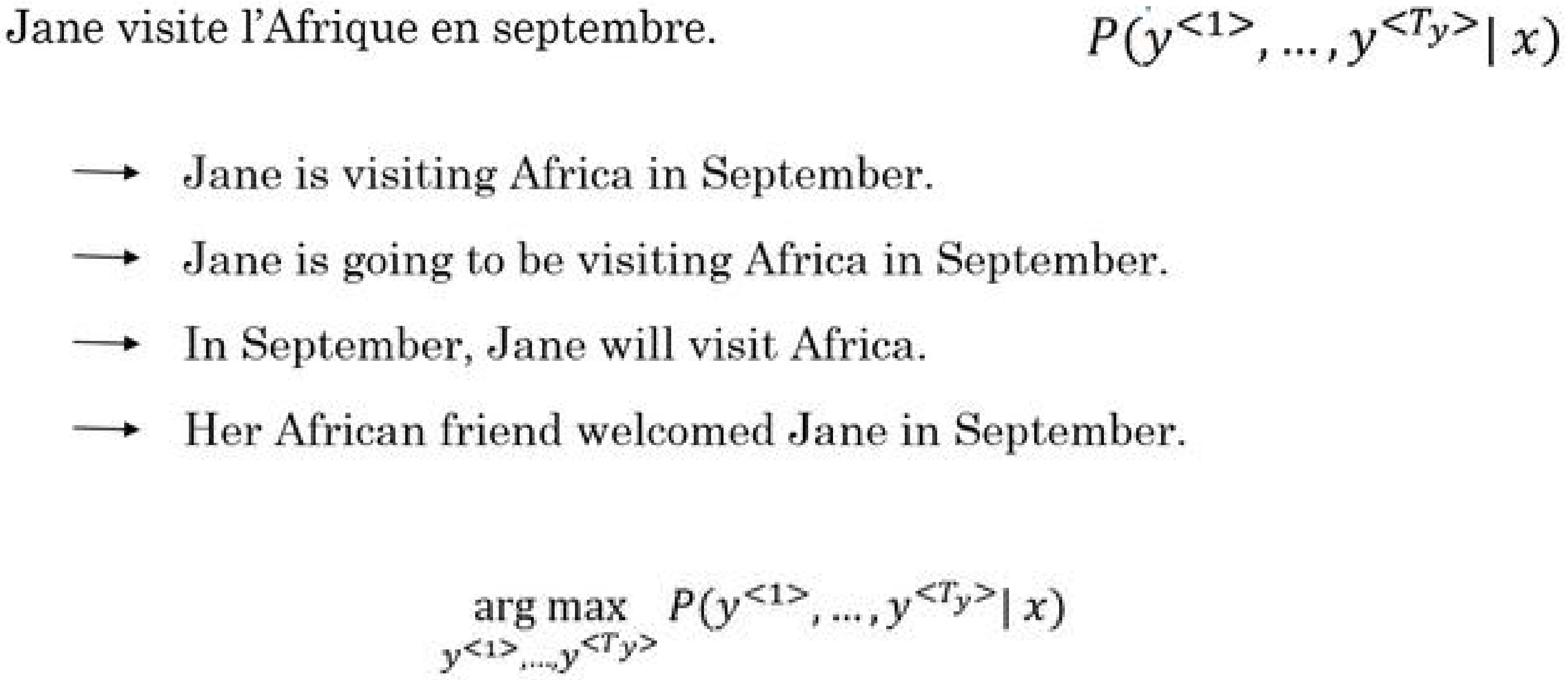
$$P(y^{<1>}, y^{<2>}, \ldots, y^{<T_y>})$$



$$P(y^{<1>}, y^{<2>}, \ldots, y^{<T_y>} \mid x^{<1>}, \ldots x^{<T_x>})$$

Machine translation

- 选择最可能的句子



Jane visite l'Afrique en septembre.

$P(y^{<1>}, \dots, y^{<T_y>} | x)$

→ Jane is visiting Africa in September.

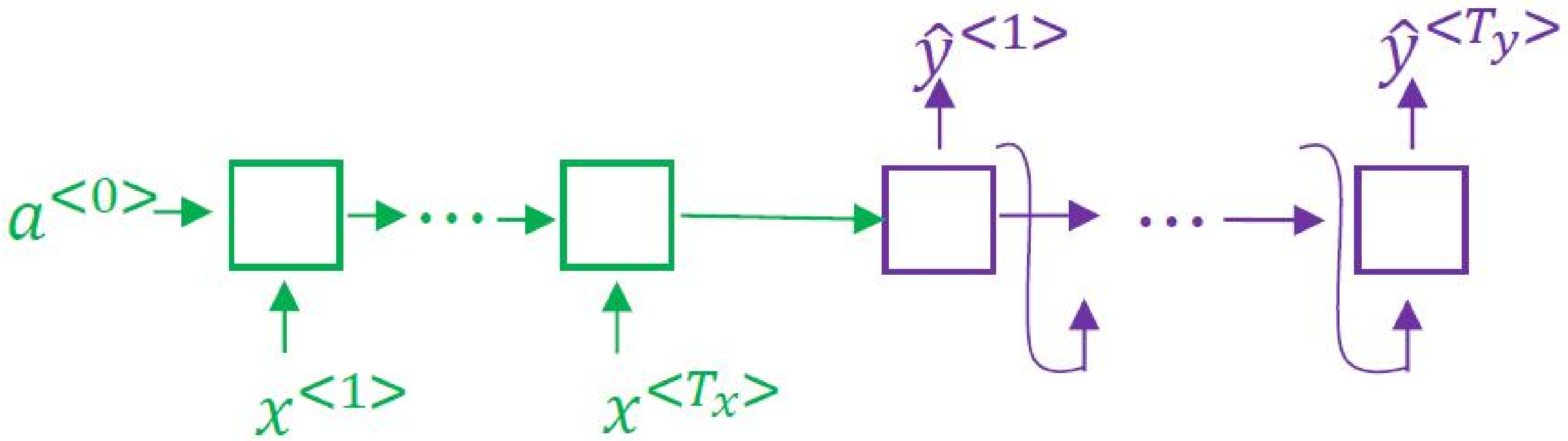
→ Jane is going to be visiting Africa in September.

→ In September, Jane will visit Africa.

→ Her African friend welcomed Jane in September.

$$\underset{y^{<1>}, \dots, y^{<T_y>}}{\arg\max} P(y^{<1>}, \dots, y^{<T_y>} | x)$$

Why not a greedy search?



$$P(y^{<1>}, y^{<2>}, \ldots, y^{<T_y>} \mid x)$$

$$P(y^{<1>}, y^{<2>} \mid x)$$
$$= P(y^{<1>} \mid x) P(y^{<2>} \mid x, y^{<1>})$$

Jane visite l'Afrique en septembre.

→ Jane is visiting Africa in September.

→ Jane is going to be visiting Africa in September.

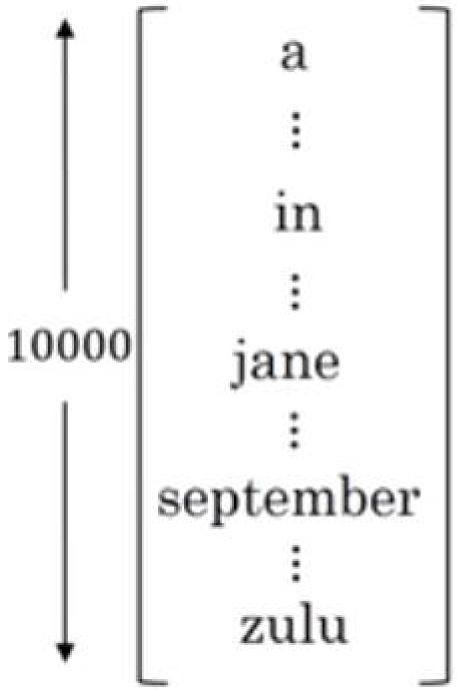
$$P(\text{Jane is going} \mid x) > P(\text{Jane is visiting} \mid x)$$

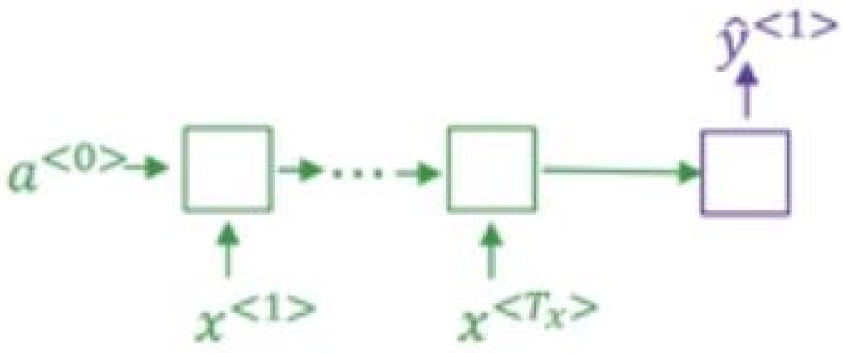
集束搜索（Beam search algorithm）

Jane visite l'Afrique en septembre

→ Jane is visiting Africa in September.

Step 1





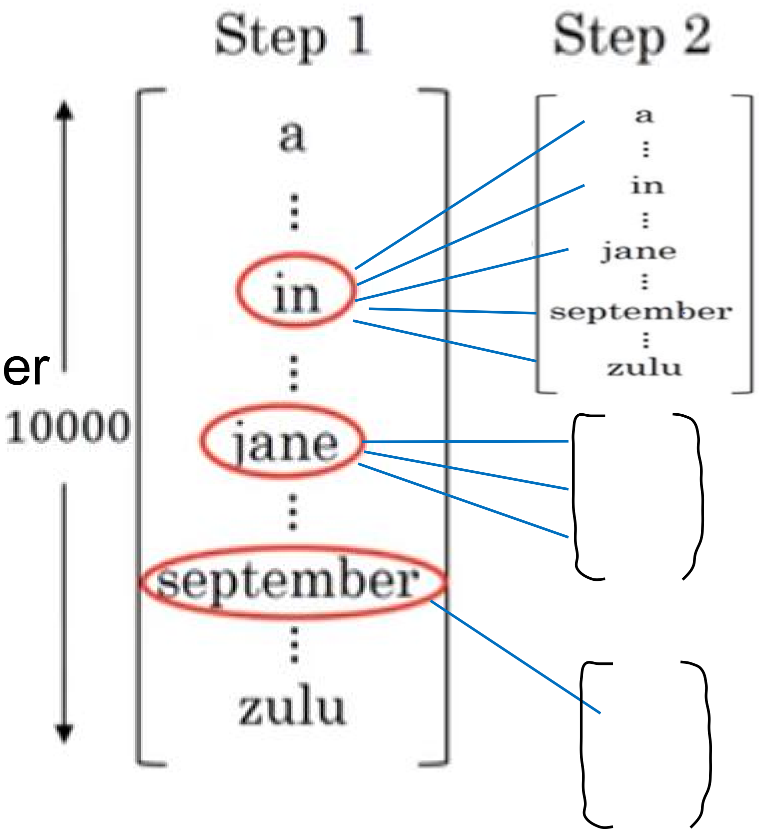
$P(y^{<1>} \mid x)$

B=3

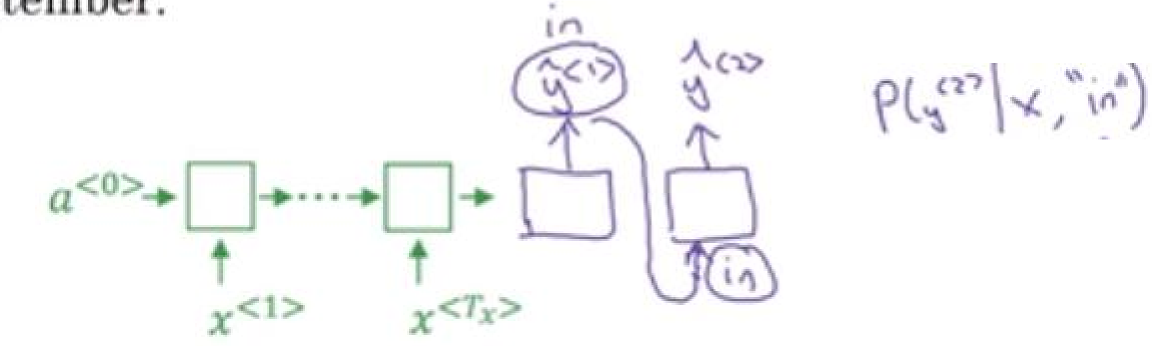
集束搜索（Beam search algorithm）

Jane visite l'Afrique en septembre

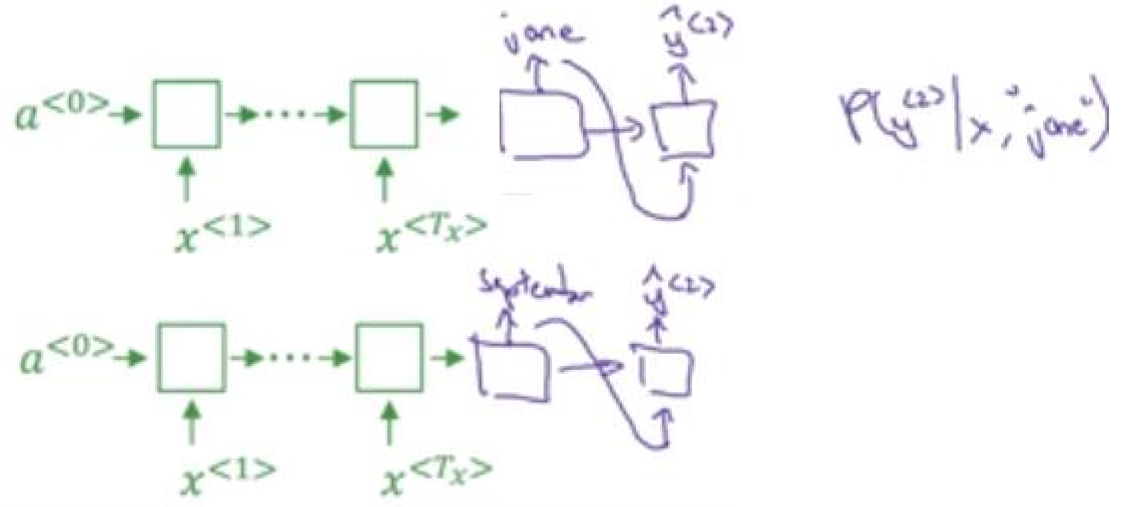
→ Jane is visiting Africa in September.



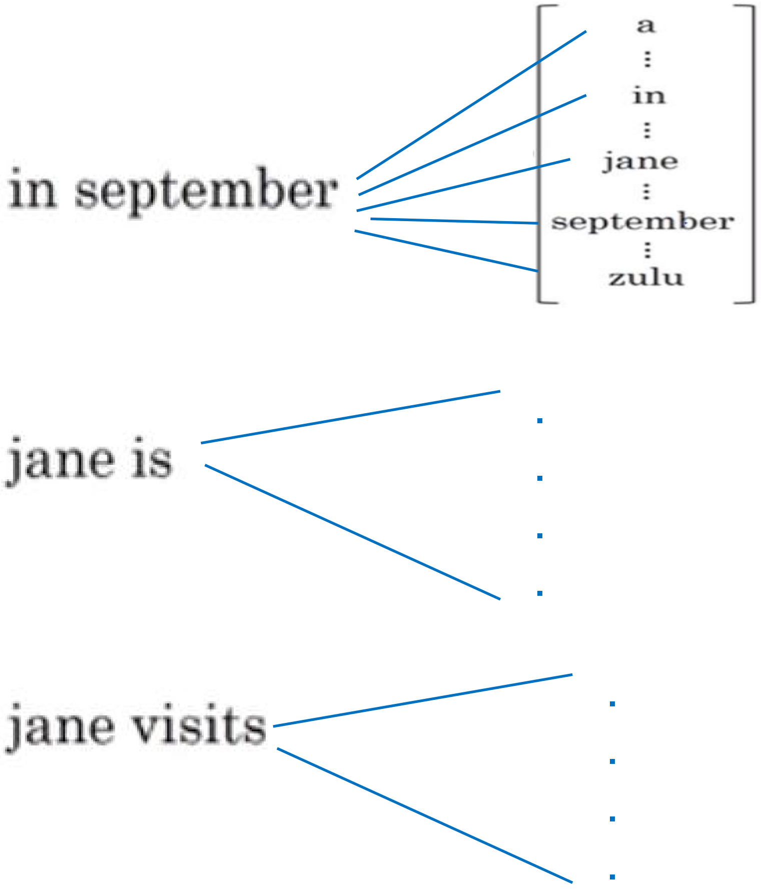
in september
jane is
jane visits



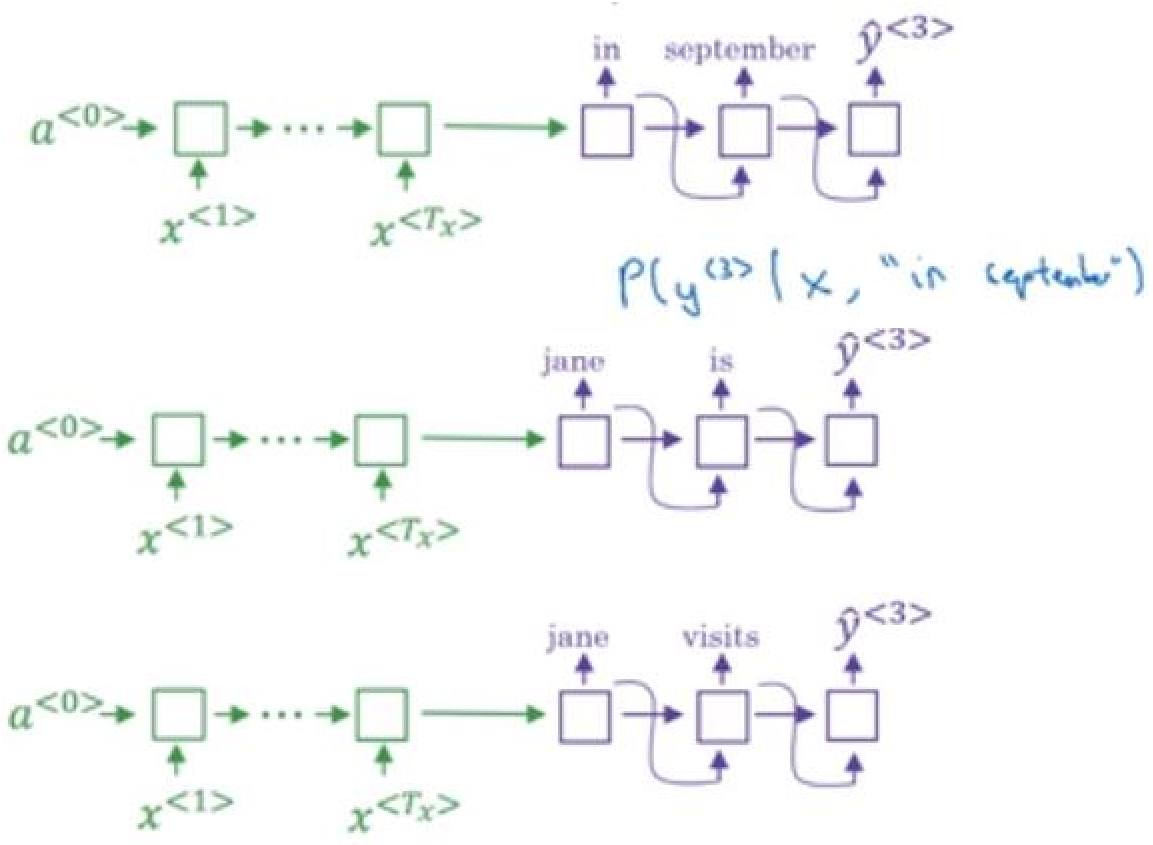
$$P(y^{<1>}, y^{<2>}|x) = P(y^{<1>}|x)P(y^{<2>}|x, y^{<1>})$$



Beam search (B=3)



$P(y^{<1>}, y^{<2>} | x)$



$P(y^{<3>} | x, \text{"in september"})$

jane visits africa in september. <EOS>

改进集束搜索（Refinements to Beam Search）

- Length normalization

$$\arg\max_{y} \prod_{t=1}^{T_y} P(y^{<t>} \mid x, y^{<1>}, \dots, y^{<t-1>})$$

$$\arg\max_{y} \sum_{y=1}^{T_y} \log P(y^{<t>} \mid x, y^{<1>}, \dots, y^{<t-1>})$$

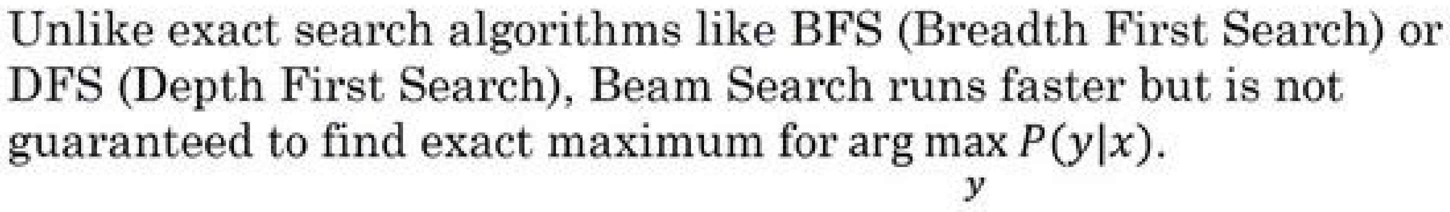
$$\frac{1}{T_y^{\alpha}} \sum_{t=1}^{T_y} \log P(y^{<t>} \mid x, y^{<1>}, \dots, y^{<t-1>})$$

归一化的对数似然目标函数（a normalized log likelihood objective）

Beam search

- 集束宽度

Beam width B?



Unlike exact search algorithms like BFS (Breadth First Search) or DFS (Depth First Search), Beam Search runs faster but is not guaranteed to find exact maximum for $\arg\max_{y} P(y|x)$.

集束搜索的误差分析

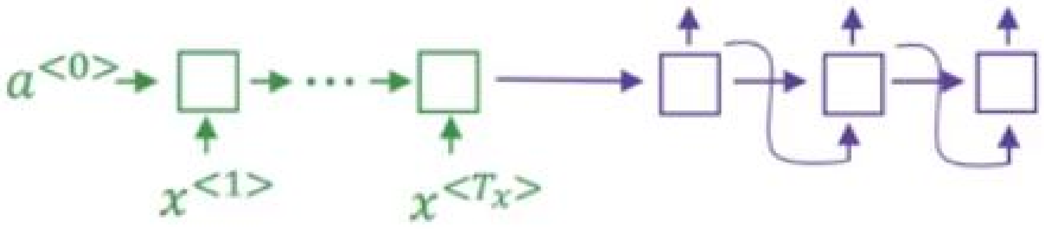
- Example

Jane visite l'Afrique en septembre.

Human: Jane visits Africa in September. y^*

Algorithm: Jane visited Africa last September. $\hat{y}$

Jane visits Africa 。。。。。。

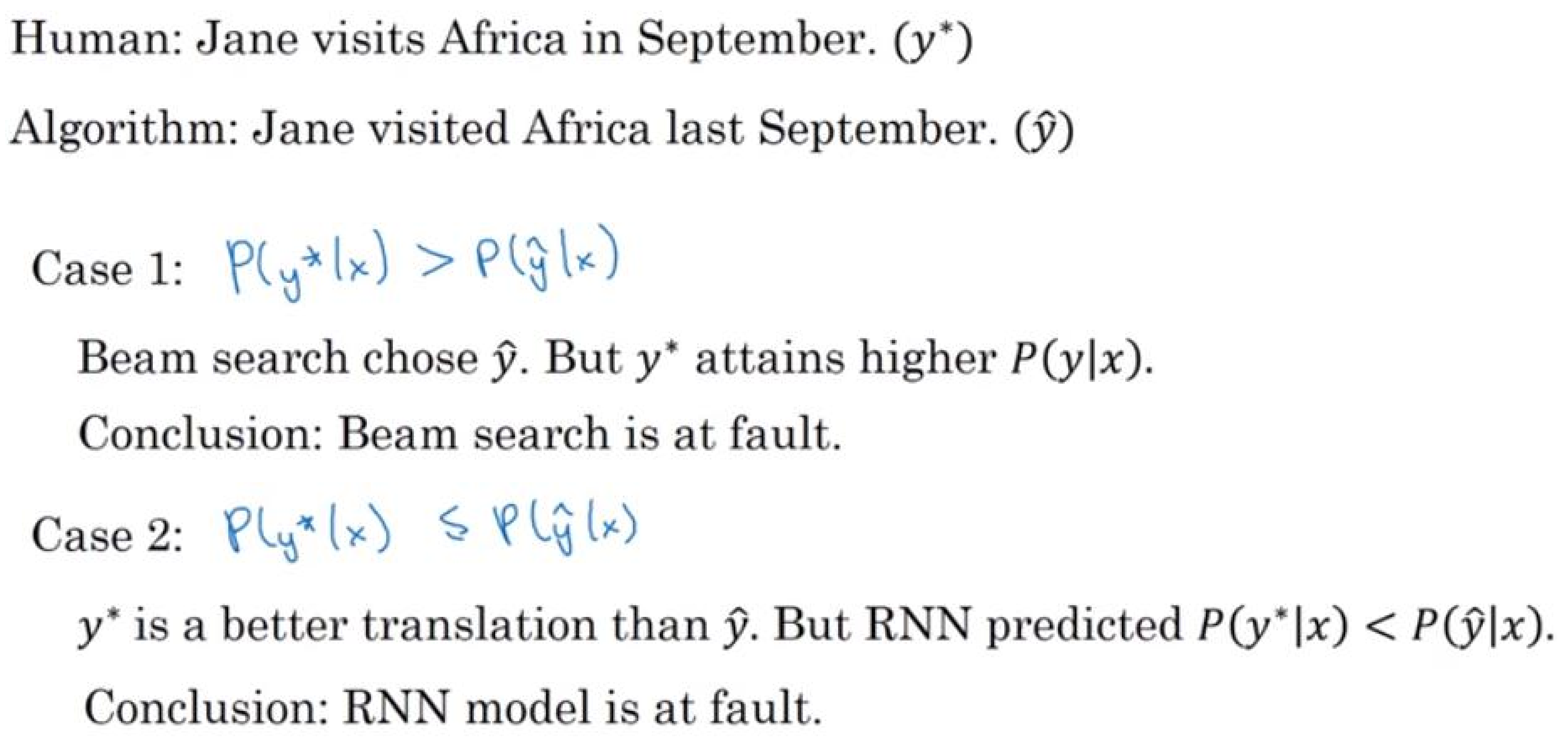


$P(y^*|x)$

$P(y|x)$

$P(\hat{y}|x)$

集束搜索的误差分析



Human: Jane visits Africa in September. (y^*)

Algorithm: Jane visited Africa last September. ($\hat{y}$)

Case 1: $P(y^*|x) > P(\hat{y}|x)$

Beam search chose $\hat{y}$. But y^* attains higher $P(y|x)$.

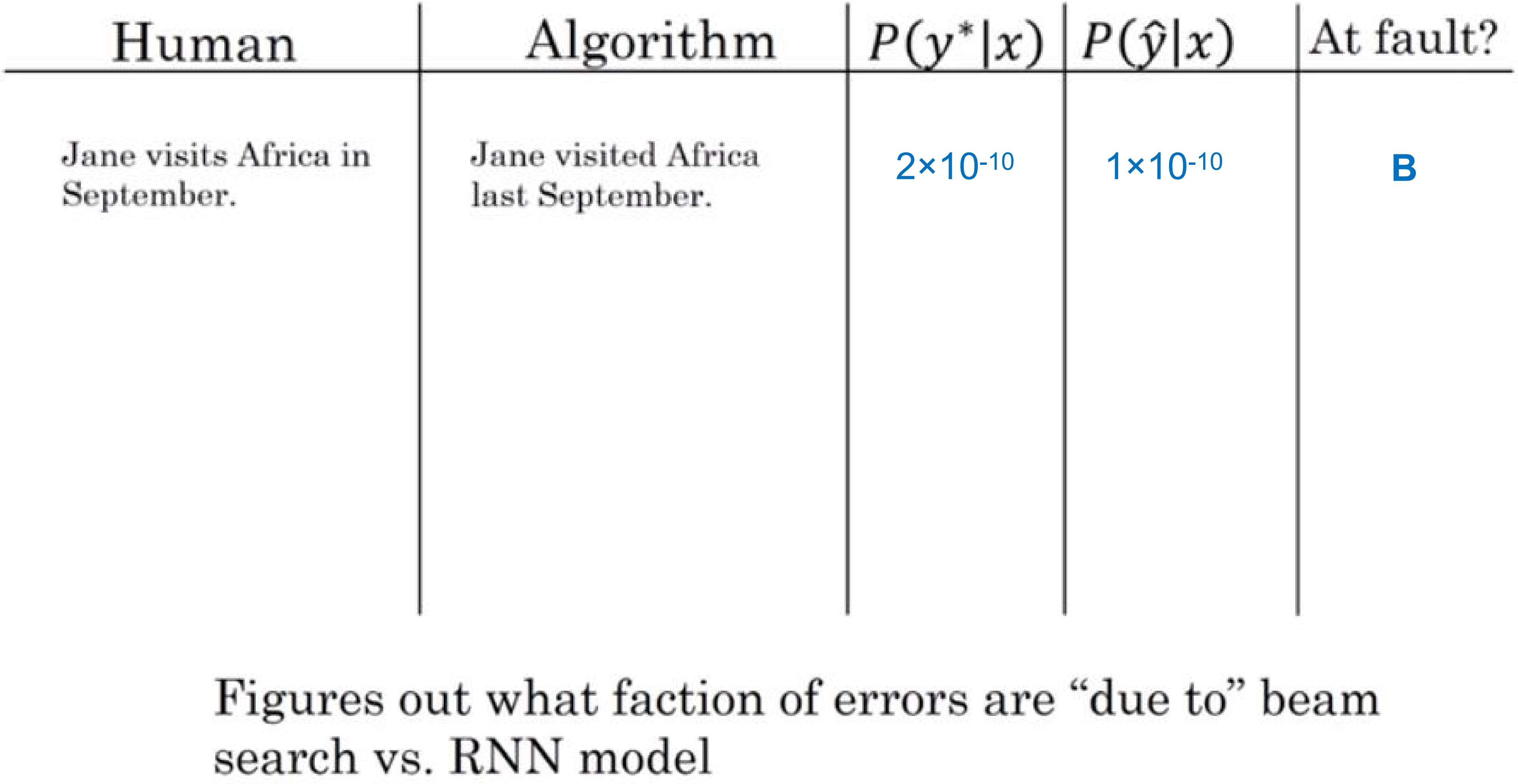
Conclusion: Beam search is at fault.

Case 2: $P(y^*|x) \leq P(\hat{y}|x)$

y^* is a better translation than $\hat{y}$. But RNN predicted $P(y^*|x) < P(\hat{y}|x)$.

Conclusion: RNN model is at fault.

误差分析

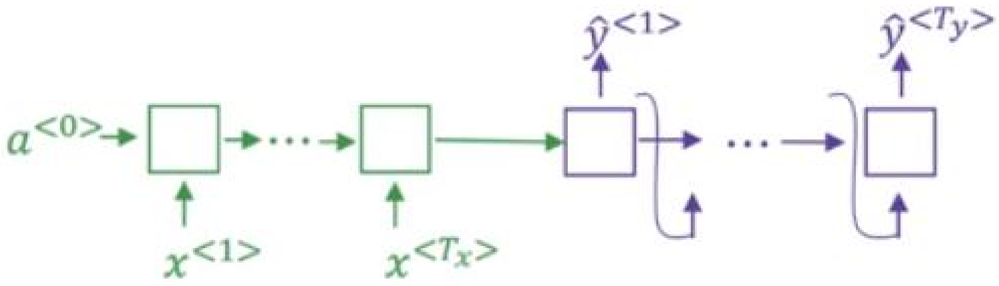


Human	Algorithm	$P(y^*\|x)$	$P(\hat{y}\|x)$	At fault?
Jane visits Africa in September.	Jane visited Africa last September.	2×10^{-10}	1×10^{-10}	B

Figures out what faction of errors are “due to” beam search vs. RNN model

注意力模型

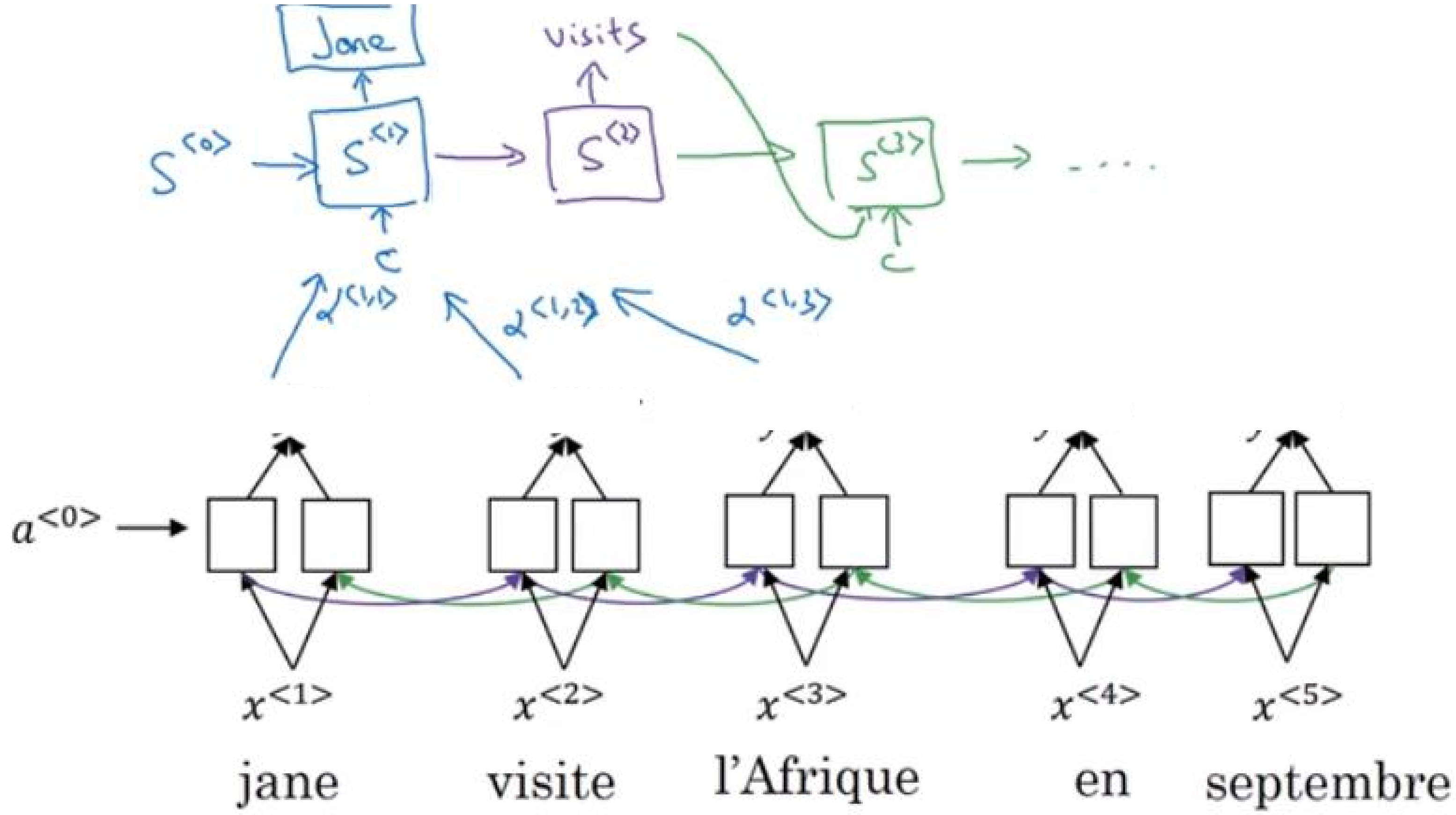
- The problem of long sequence



Jane s'est rendue en Afrique en septembre dernier, a apprécié la culture et a rencontré beaucoup de gens merveilleux; elle est revenue en parlant comment son voyage était merveilleux, et elle me tente d'y aller aussi.

Jane went to Africa last September, and enjoyed the culture and met many wonderful people; she came back raving about how wonderful her trip was, and is tempting me to go too.

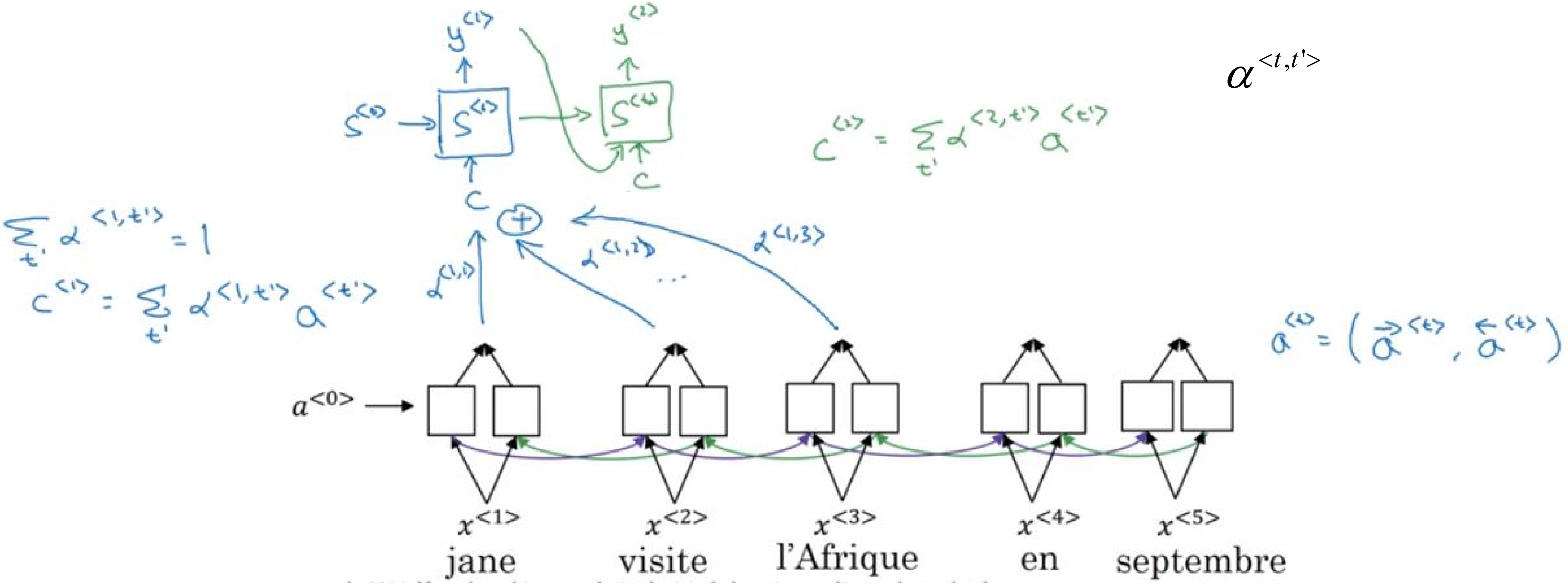
Attention model intuition



[Bahdanau et. al., 2014. Neural machine translation by jointly learning to align and translate]

Attention model

$\alpha^{<t,t'>}$



$$\sum_{t'} \alpha^{<1,t'>} = 1$$

$$c^{<1>} = \sum_{t'} \alpha^{<1,t'>} a^{<t'>}$$

$$c^{<2>} = \sum_{t'} \alpha^{<2,t'>} a^{<t'>}$$

$$a^{<t>} = (\overrightarrow{a}^{<t>}, \overleftarrow{a}^{<t>})$$

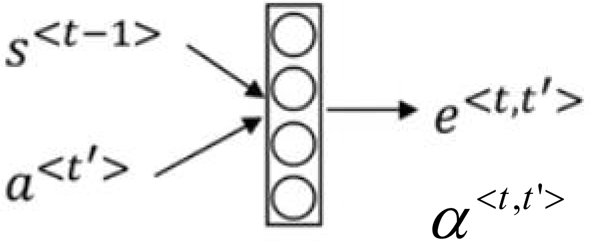
[Bahdanau et. al., 2014. Neural machine translation by jointly learning to align and translate]

Attention model

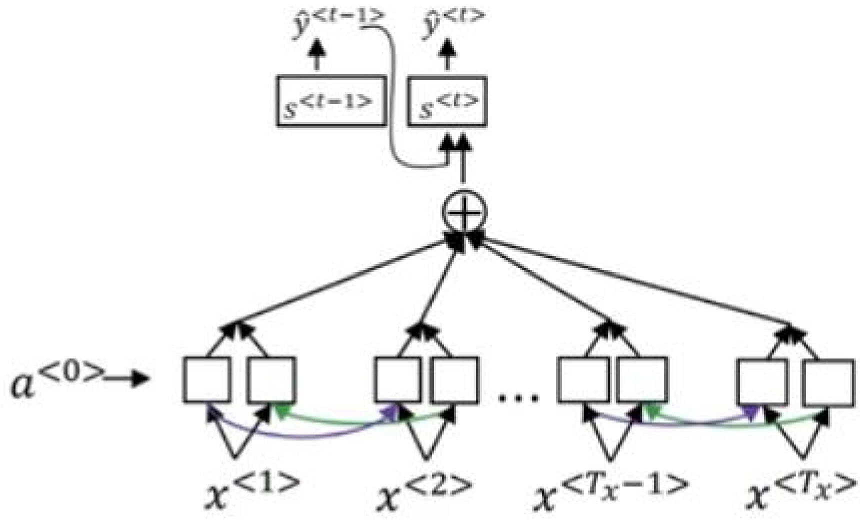
- Computing attention $\alpha^{<t,t'>}$

$$\alpha^{<t,t'>} = \text{amount of attention } y^{<t>} \text{ should pay to } a^{<t'>}$$

$$\alpha^{<t,t'>} = \frac{\exp(e^{<t,t'>})}{\sum_{t'=1}^{T_x} \exp(e^{<t,t'>})}$$

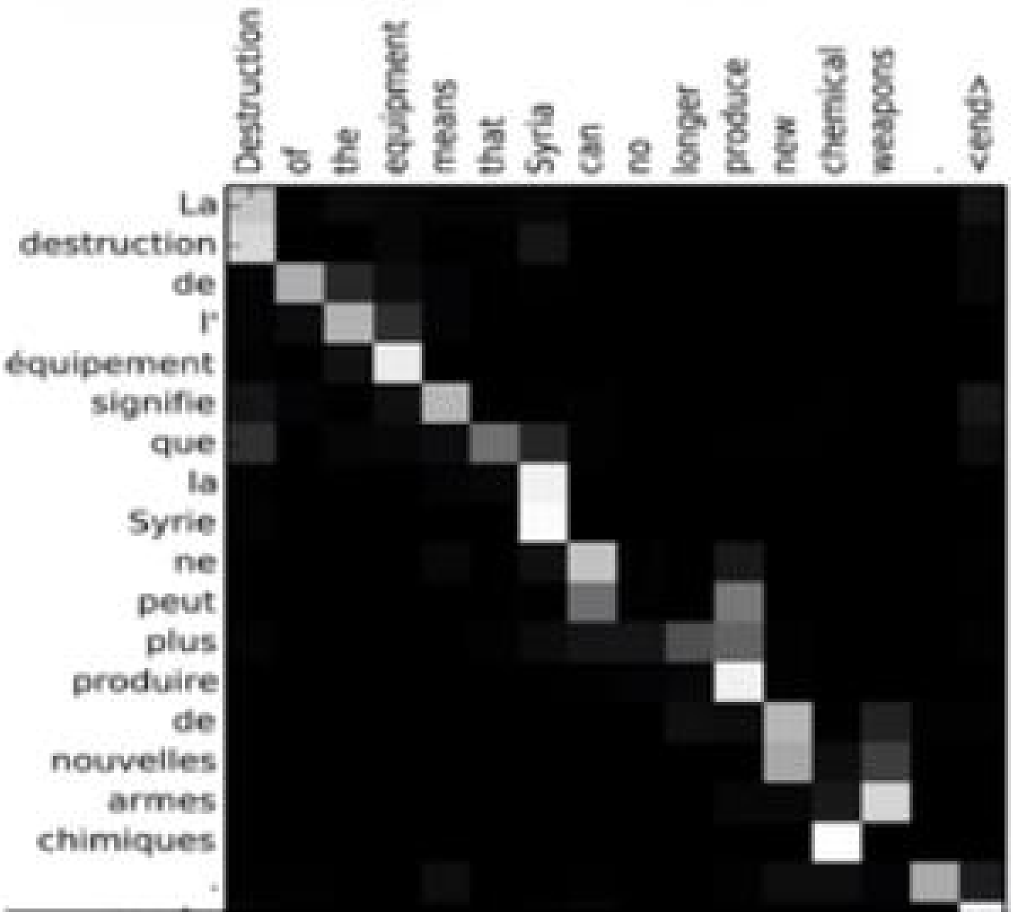


$\alpha^{<t,t'>}$



Attention examples

Visualization of $\alpha^{<t,t'>}$:



[Bahdanau et. al., 2014. Neural machine translation by jointly learning to align and translate]

小结

- 循环序列模型
  - Recurrent Neural Networks, RNN
  - GRU
  - LSTM
  - BRNN
  - Deep RNN
- 序列模型（sequence to sequence）

Thanks !