RNN

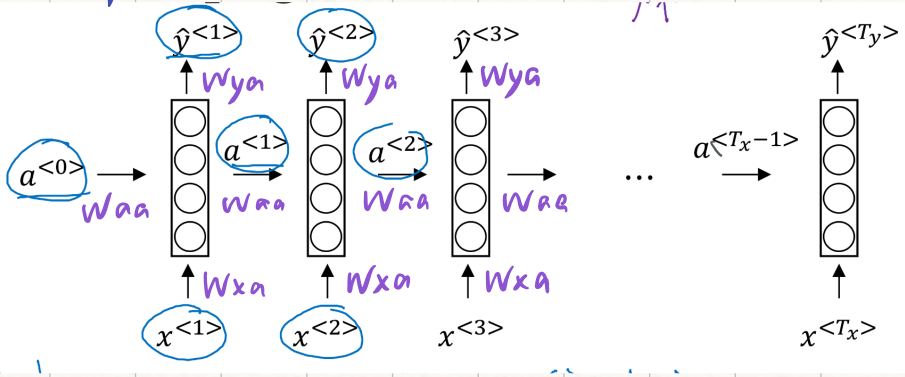
一. 循环序列模型 Sequence model

1. Notation

words: $x^{<1>}\ x^{<2>}\ \cdots\ x^{<t>}$ $T_x = 9$

diuts: $y^{<1>}\ y^{<2>}\ \cdots\ y^{<t>}$ $T_y = 9$

2. 循环神经网络 Recurrent Neural Network



考虑一句话中某个词语的前(后)文.

共用 Waa Wxa Wya

$$a^{<t>} = g(W_{aa}a^{<t-1>} + W_{ax}x^{<t>} + b_a)$$ tanh / relu

(100,100) 100 (100,10,000) 10,000

$$\hat{y}^{<t>} = g(W_{ya}a^{<t>} + b_y)$$ sigmoid

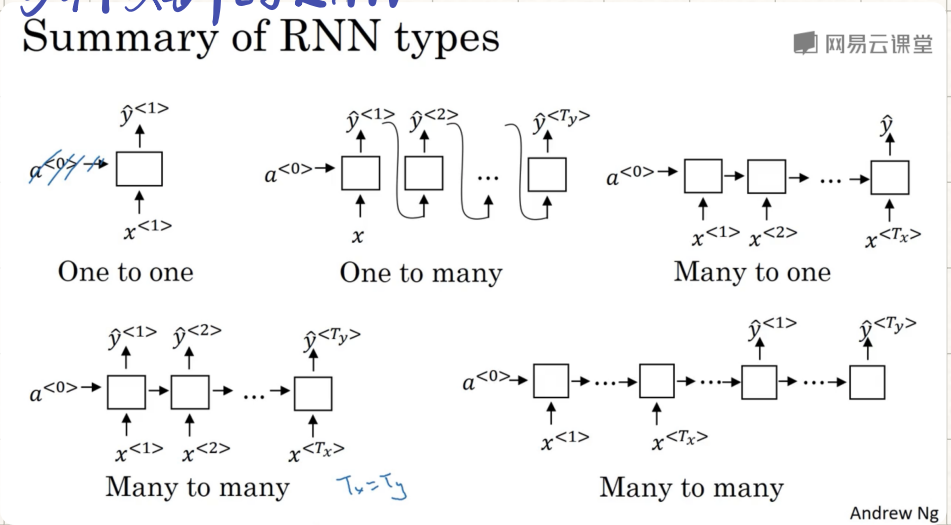
反向传播.

对 $\hat{y}^{<t>}$ 使用交叉熵损失 $(\text{logistic}) \Rightarrow L(t)$

$-\sum y \log(\hat{y})$

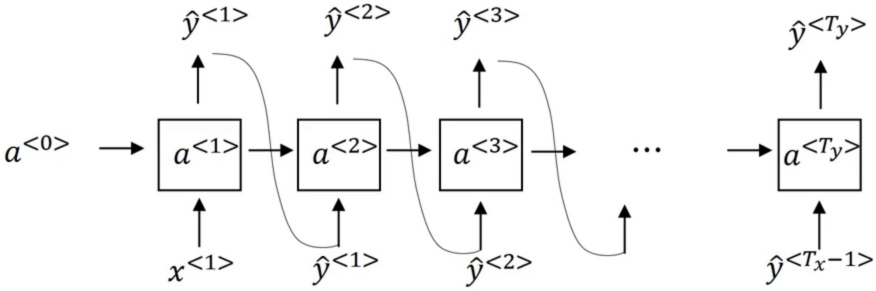
总损失 $L = \sum L(t)$. backprop through time

3. 多种类别的 RNN



4. 语言模型和序列生成

5. 新序列采样



{ 单词库，UNK，EOS.
字符库.

6. 梯度消失.

靠后的梯度很难影响靠前的层.

即句子后段的单词无法长期依赖前段的单词.

{ 梯度爆炸：剪枝
梯度消失：？（GRU）如何在长序列保持有效

6. GRU门控循环单元 gated recurrent unit

c = memory cell.

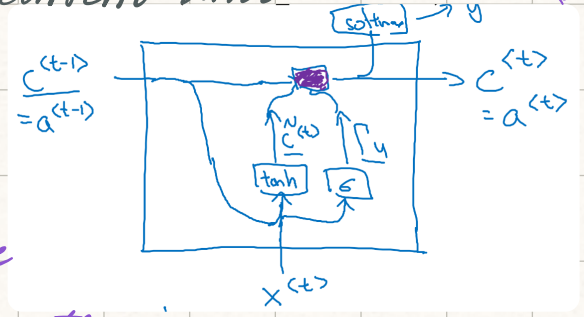
$\tilde{h}$ $\tilde{c}^{<t>} = \tanh(W_c[\Gamma_r * c^{<t-1>}, x^{<t>}] + b_c)$

u $\Gamma_u = \sigma(W_u[c^{<t-1>}, x^{<t>}] + b_u)$ update/optic gate

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

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

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



Γ_u 一般很小，因此 $c^{<t>}$ 可以记忆很久

用 c 代替 a.

7. LSTM 长短时记忆网络 Long Short Term Memory org

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

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

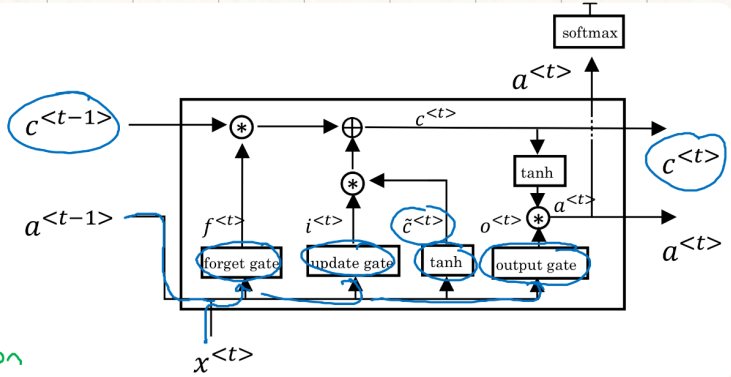
$\Gamma_f = \sigma(W_f[a^{<t-1>}, x^{<t>}] + b_f)$ Forget

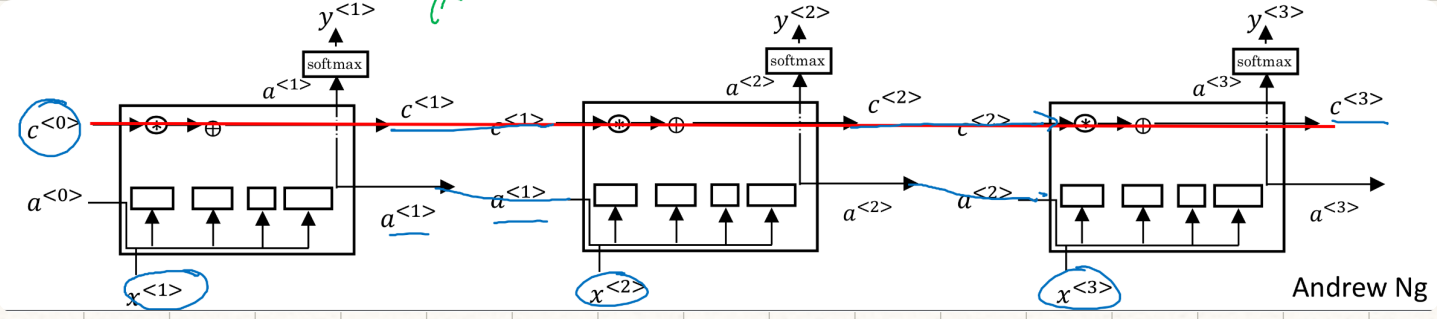
$\Gamma_o = \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o)$ Output

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

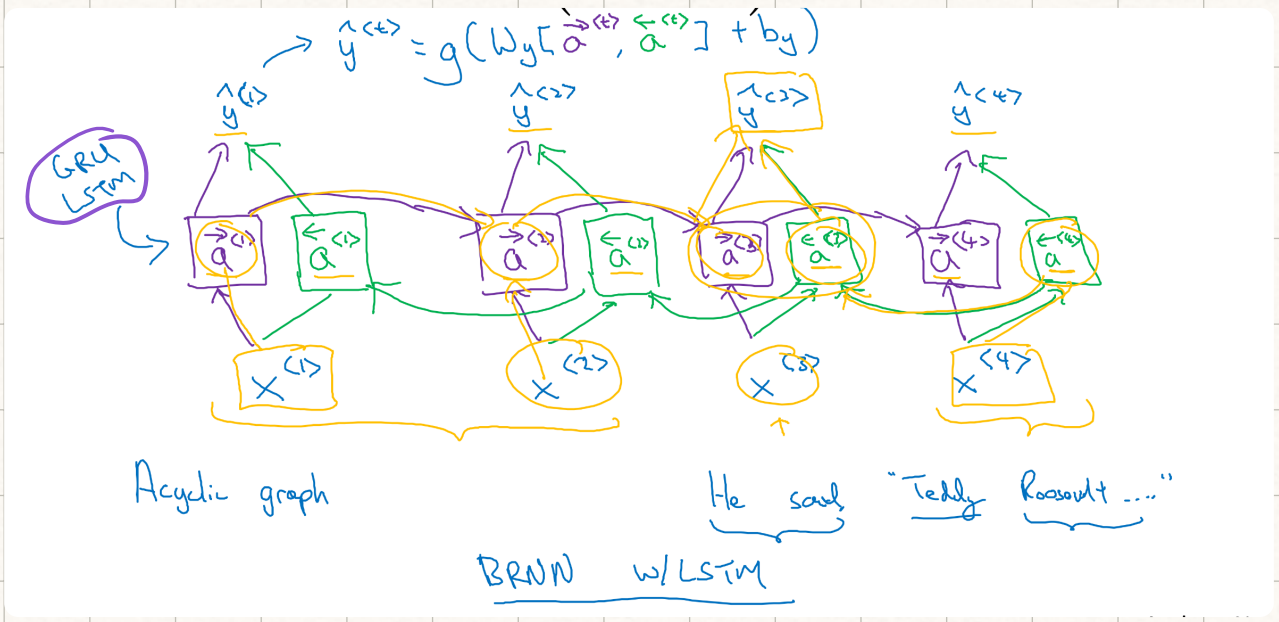
$a^{<t>} = \Gamma_o * c^{<t>}$

$c^{<t-1>}$ peephole connection



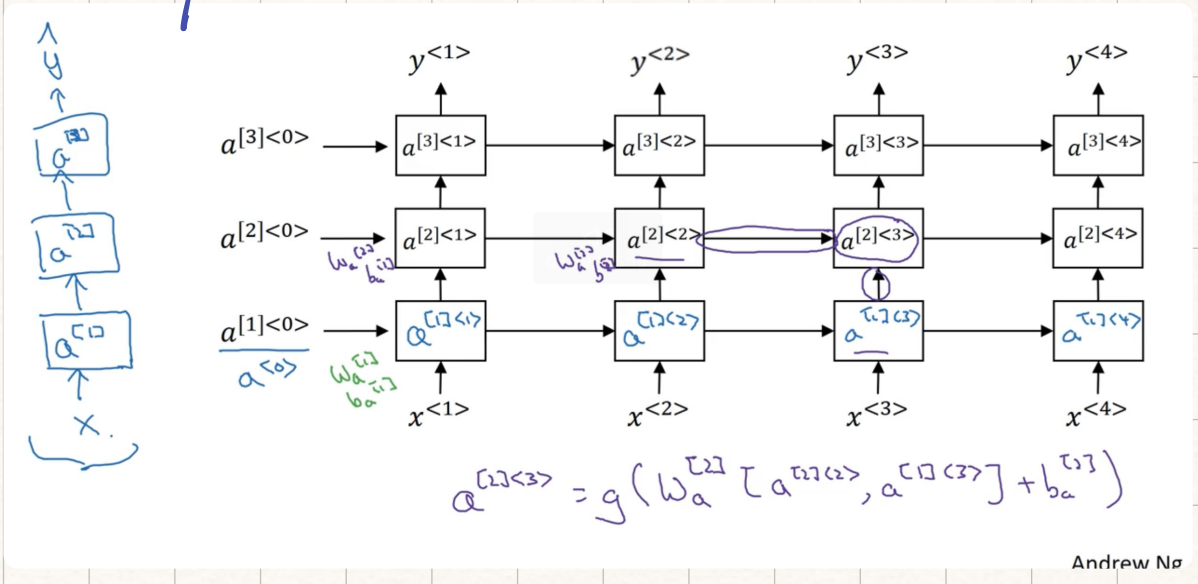


8. BRNN 双向循环网络 Bidirectional



同时考虑句子前后的信息来进行预测. 常用 BRNN + LSTM

9. Deep RNN

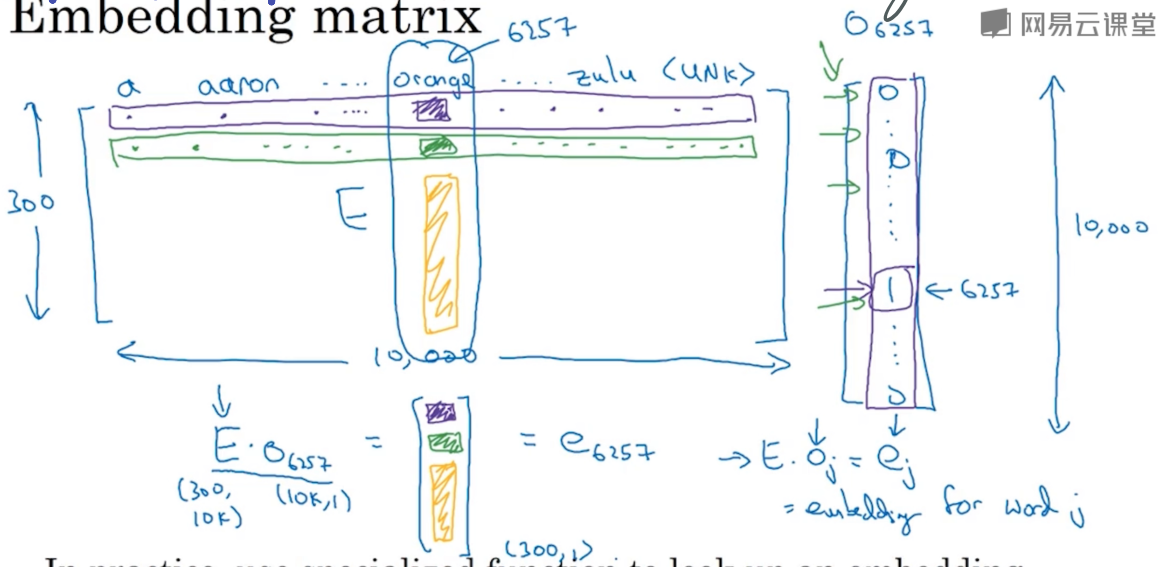


层数不会很深

二. 自然语言处理.

1. 词汇表征和词嵌入. Word Embedding

Embedding matrix



网易云课堂

In practice, use specialized function to look up an embedding.

但当你动手实现时
But if when you're implementing this,

Andrew Ng

三. 序列模型和注意力机制

1. 基础模型. Seq2Seq.

Sequence to sequence model

$x^{<1>}$ $x^{<2>}$ $x^{<3>}$ $x^{<4>}$ $x^{<5>}$
Jane visite l'Afrique en septembre

→ Jane is visiting Africa in September.
$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ $y^{<4>}$ $y^{<5>}$ $y^{<6>}$

机器翻译

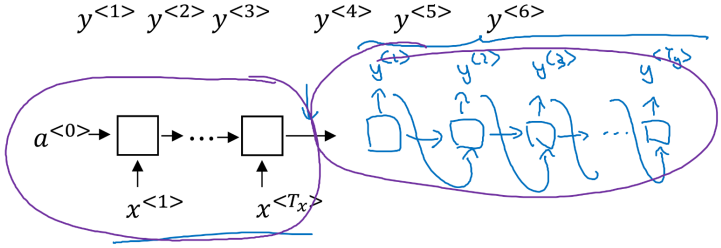
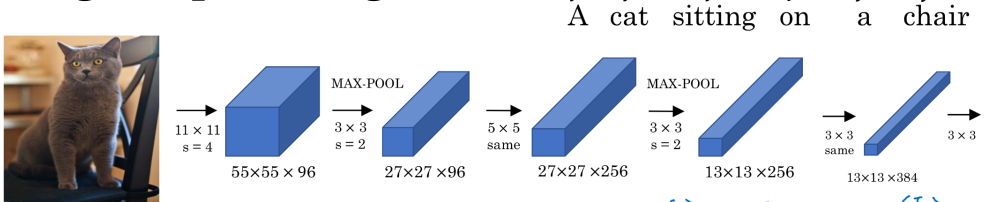
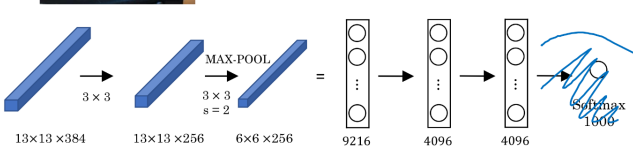
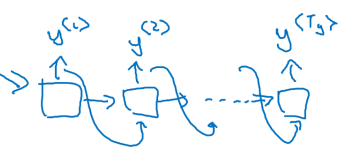


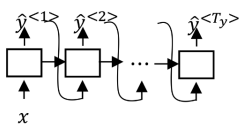
Image captioning

$y^{<1>}$ $y^{<2>}$ $y^{<3>}$ $y^{<4>}$ $y^{<5>}$ $y^{<6>}$
A cat sitting on a chair





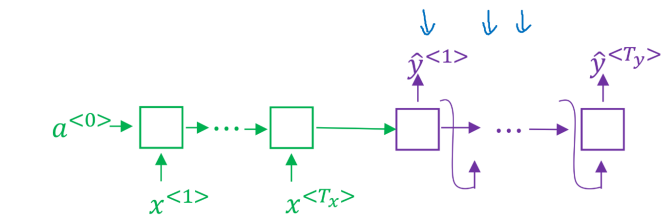




2. 选择最可能的句子.

Why not a greedy search?

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



10,000
10
$10{,}000^{10}$

$\arg\max_y P(\hat{y}^{<1>}, \hat{y}^{<2>}, \ldots, \hat{y}^{<T_y>}|x)$

$P(y|x)$

→ Jane is visiting Africa in September.

→ Jane is going to be visiting Africa in September.

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

Andrew Ng

贪心算法会导致高数词汇次以及句子长度过长.

采用集束算法. 以3为例. 1) 每次保留概率最高的三组词句.

3. 改进定向搜索算法

Length normalization

$P(y^{<1>}\ldots y^{<T_y>}|x) = P(y^{<1>}|x)\,P(y^{<2>}|x, y^{<1>})\cdots P(y^{<T_y>}|x, y^{<1>}, \ldots, y^{<T_y-1>})$

$$\arg\max_y \prod_{t=1}^{T_y} P(y^{<t>}|x, y^{<1>}, \ldots, y^{<t-1>})$$

log

$\log P(y|x) \leftarrow$

$P(y|x) \leftarrow$

$$\arg\max_y \sum_{t=1}^{T_y} \log P(y^{<t>}|x, y^{<1>}, \ldots, y^{<t-1>})$$

$T_y = 1, 2, 3, \ldots, 30.$

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

$\alpha = 0.7$ $\alpha = 1$ $\alpha = 0$

Andrew Ng

减少惩罚长句子 (再乘数倍向更短的句子).

4 Bleu Bilingual Evaluation Understudy

Bleu score on unigrams

Example: Reference 1: The cat is on the mat.

Reference 2: There is a cat on the mat.

MT output: The cat the cat on the mat. ($\hat{y}$)

$p_1 = \frac{\sum_{unigram \in \hat{y}} count_{clip}(unigram)}{\sum_{unigram \in \hat{y}} count(unigram)}$

$p_n = \frac{\sum_{ngram \in \hat{y}} count_{clip}(ngram)}{\sum_{ngram \in \hat{y}} count(ngram)}$

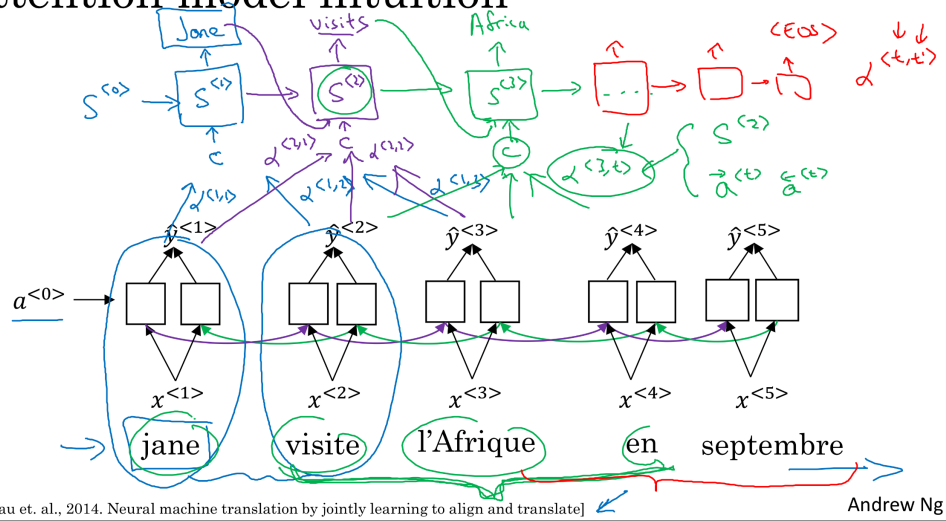
$p_1, p_2, \ldots = 1.0$

BP: $\exp\left(\frac{1}{n}\sum_{n=1} p_n\right)$

5. 注意力模型

长句中. Bleu会下降. 因此需要考虑全局.

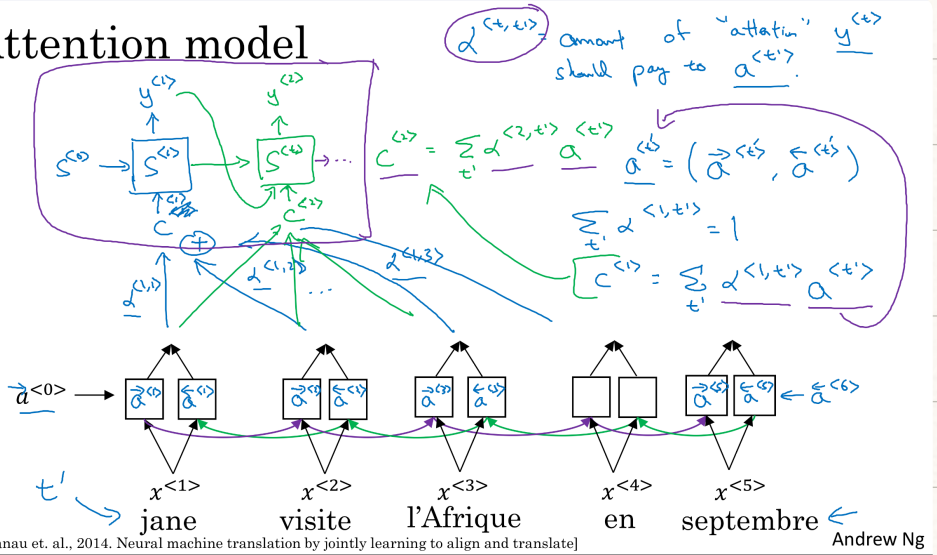
Attention model intuition



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

Andrew Ng

Attention model



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

Andrew Ng

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

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

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