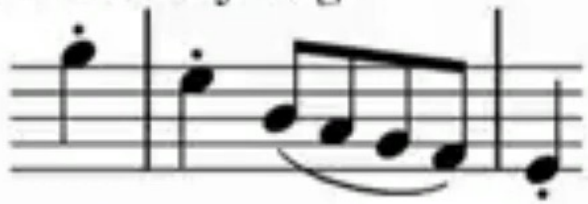
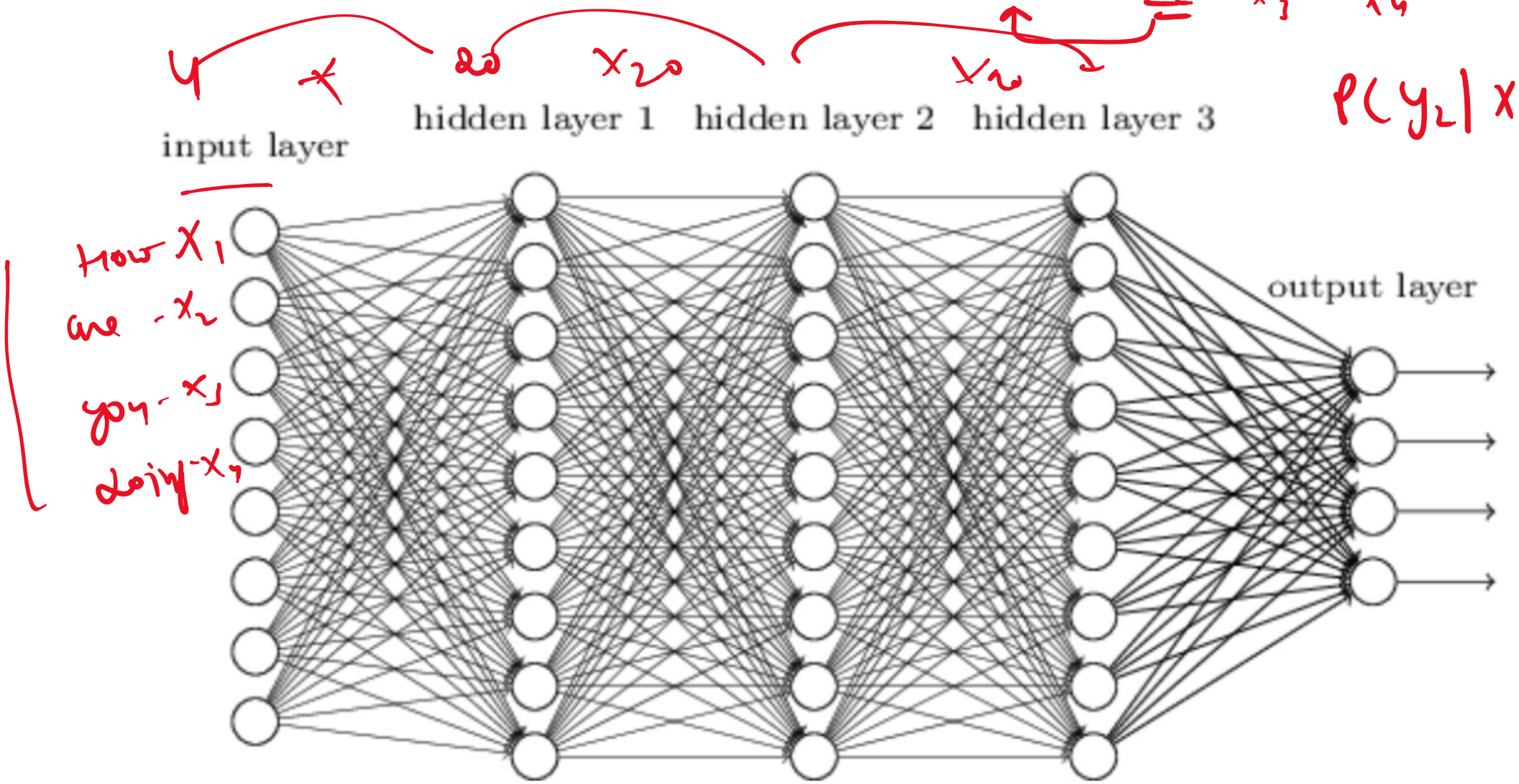
Sequence Models – RNN/LSTM

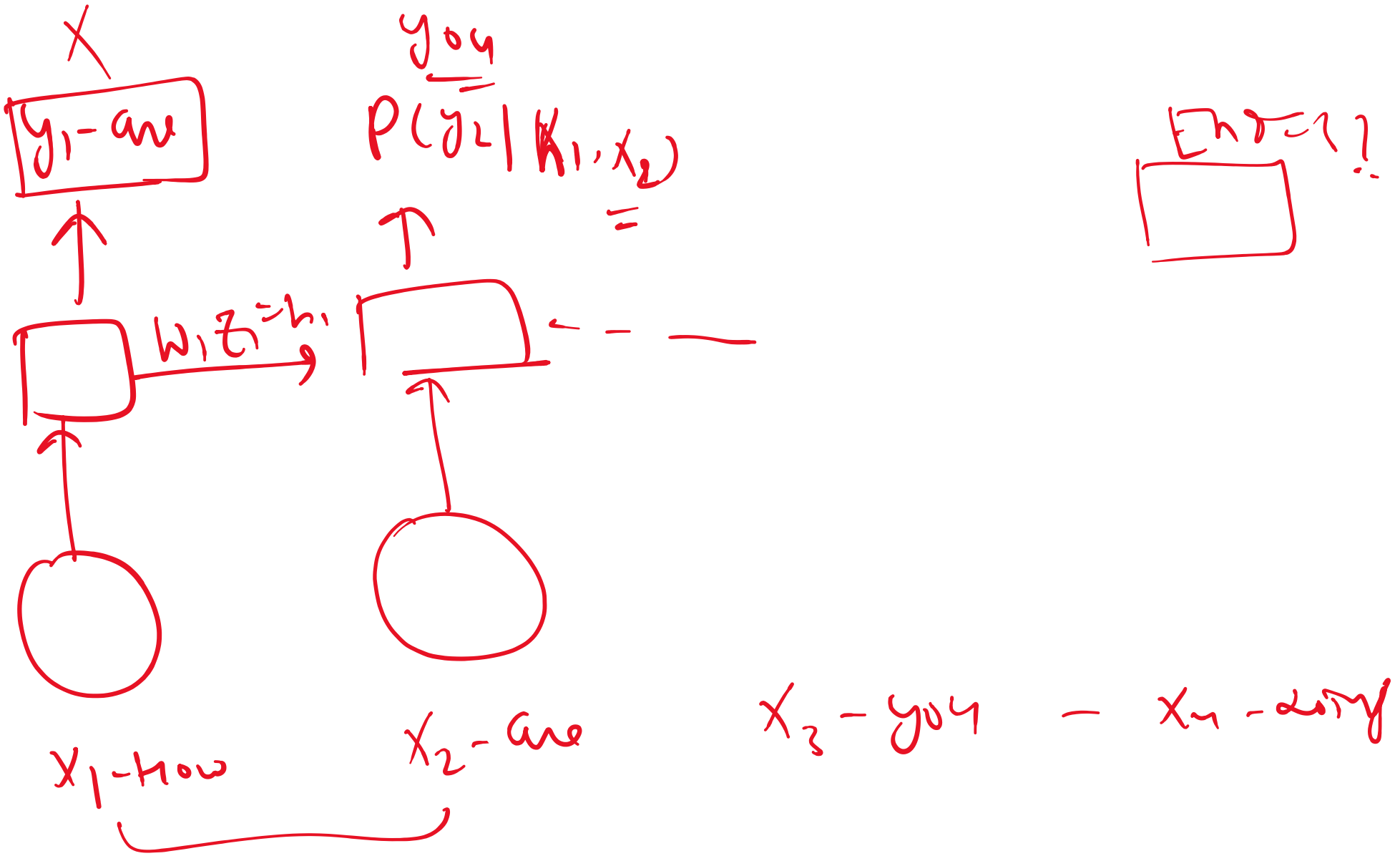
Aniket Chhabra

Examples of sequence data

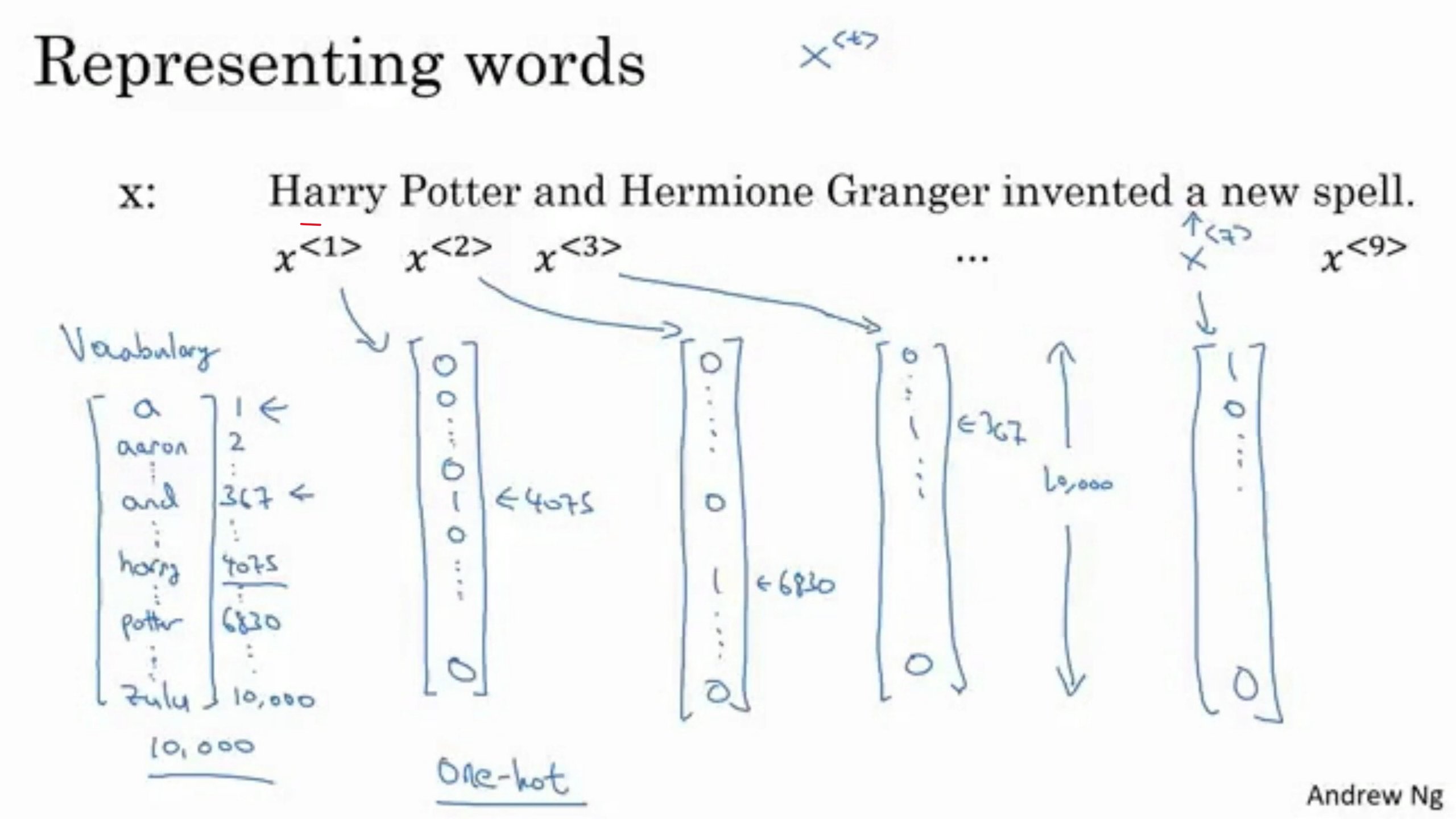
Speech recognition		→	"The quick brown fox jumped over the lazy dog."
Music generation		→	
Sentiment classification	"There is nothing to like in this movie."	→	★☆☆☆☆
DNA sequence analysis	AGCCCCTGTGAGGAACTAG	→	AGCCCCTGTGAGGAACTAG
Machine translation	Voulez-vous chanter avec moi?	→	Do you want to sing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

Neural Networks limitation



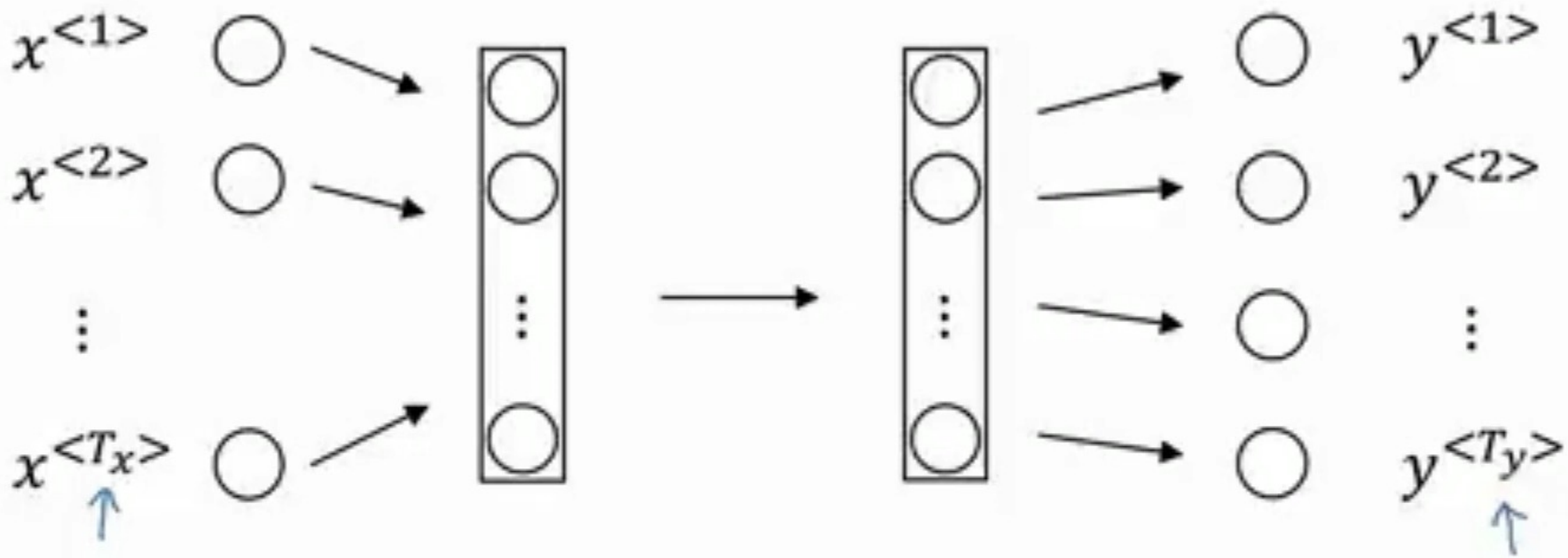

X
y1 - are
you
P(y2 | x1, x2)
=
Enter ?!
W1 t1 = h1
x1 - How
x2 - are
x3 - you
xn - doing

Representing words



Andrew Ng

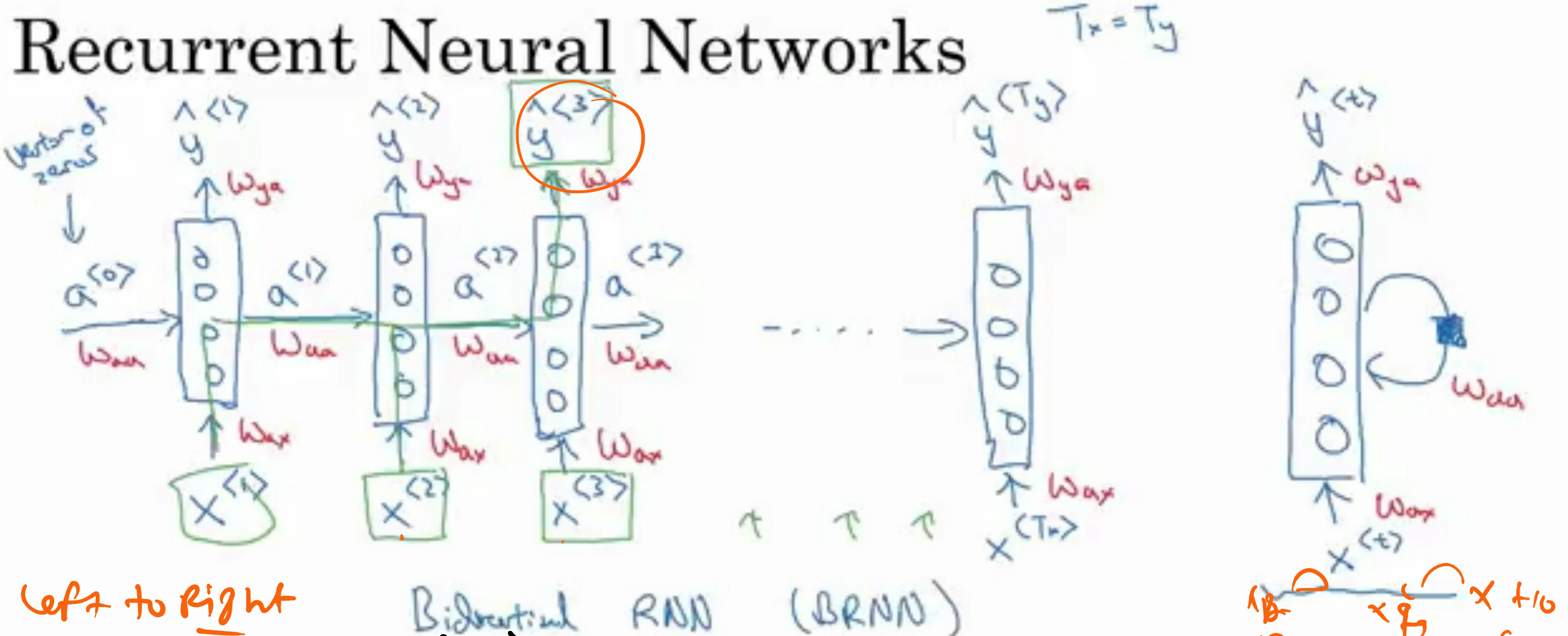
Why not a standard network?



Problems:

- Inputs, outputs can be different lengths in different examples.
- Doesn't share features learned across different positions of text.

Recurrent Neural Networks



Left to Right

Bidirectional RNN (BRNN)

Right to left

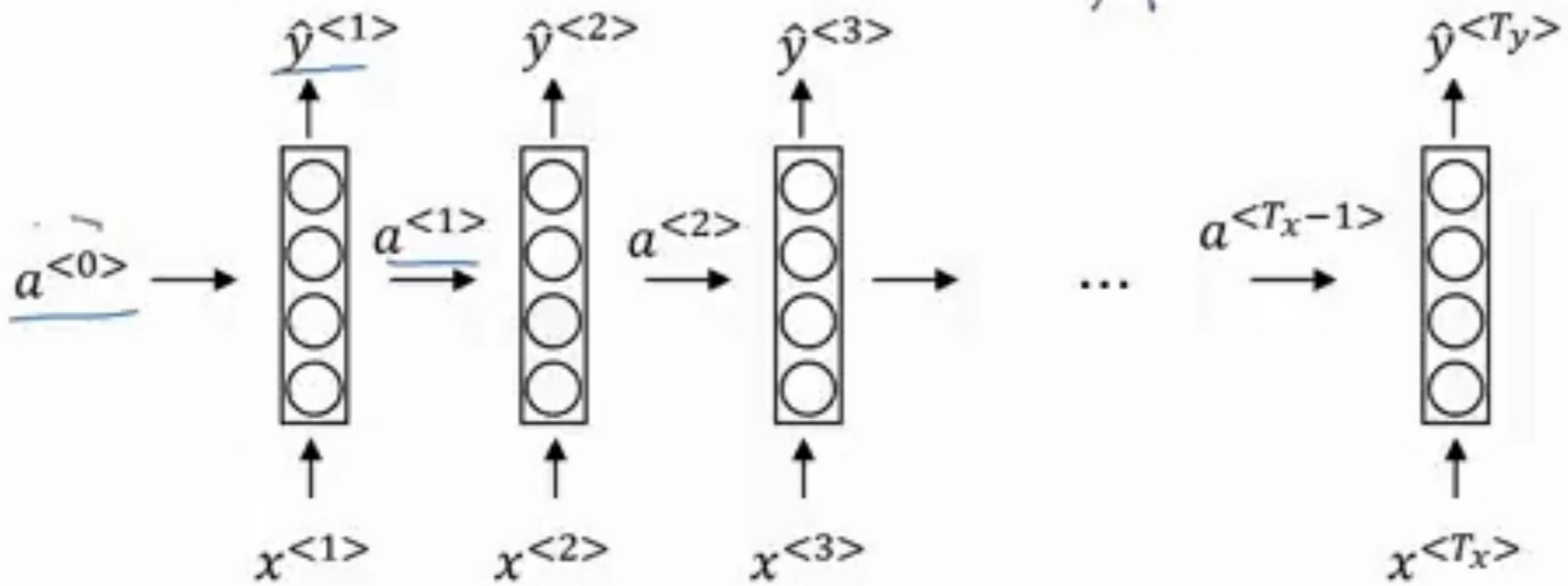
He said, "Teddy Roosevelt was a great President."

He said, "Teddy bears are on sale!"

bi-directional

Forward Propagation

$a \leftarrow W_{ax} x^{\langle 1 \rangle}$



$a^{\langle 0 \rangle} = \vec{0}$.

$$a^{\langle 1 \rangle} = g_1(W_{aa} a^{\langle 0 \rangle} + W_{ax} x^{\langle 1 \rangle} + b_a) \leftarrow \tanh / \text{ReLU}$$

$$\hat{y}^{\langle 1 \rangle} = g_2(W_{ya} a^{\langle 1 \rangle} + b_y) \leftarrow \text{Sigmoid}$$

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

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

Andrew Ng

Simplified RNN notation

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

W_{aa}: (100, 100) — $a^{<t-1>}$: 100 — W_{ax}: (100, 10,000) — $x^{<t>}$: 10,000

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

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

$\hat{y}$: ↑ — W_y: ↑ — $a^{<t>}$: ↑ — b_y: ↑

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

W_a: 100 × (100 + 10000) = 100 × 10100

$$W_a = \begin{bmatrix} W_{aa} & W_{ax} \end{bmatrix}$$

(100, 100) (100, 10000)

$$[a^{<t-1>}, x^{<t>}] = \begin{bmatrix} a^{<t-1>} \\ x^{<t>} \end{bmatrix}$$

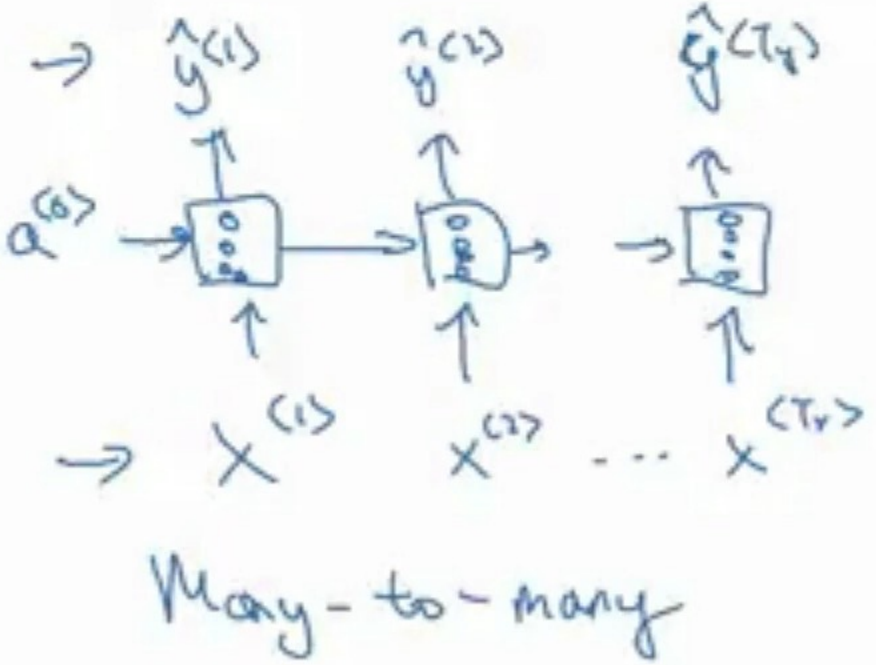
100 + 10000 = 10100

$$\begin{bmatrix} W_{aa} & W_{ax} \end{bmatrix} \begin{bmatrix} a^{<t-1>} \\ x^{<t>} \end{bmatrix} = W_{aa}a^{<t-1>} + W_{ax}x^{<t>}$$

Andrew Ng

Examples of RNN architectures

$T_x = T_y$



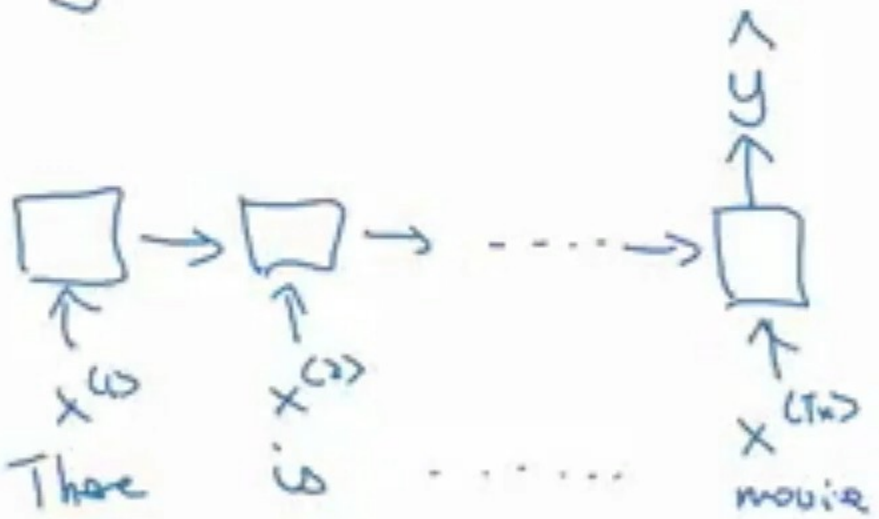
Many-to-many

[Speech Recognition]

Sentiment classification

x = text

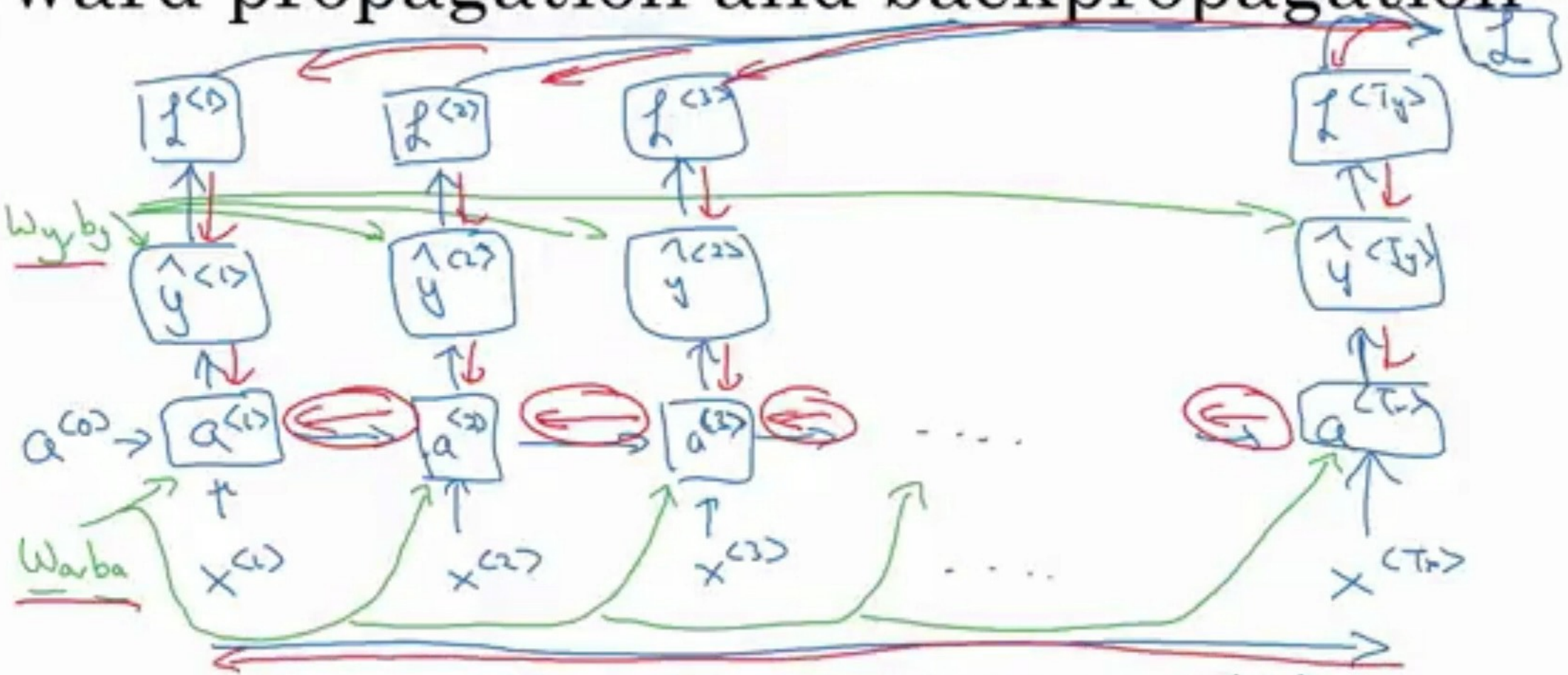
y = 0/1 ... 1...5



Many-to-one

One-to-one

Forward propagation and backpropagation

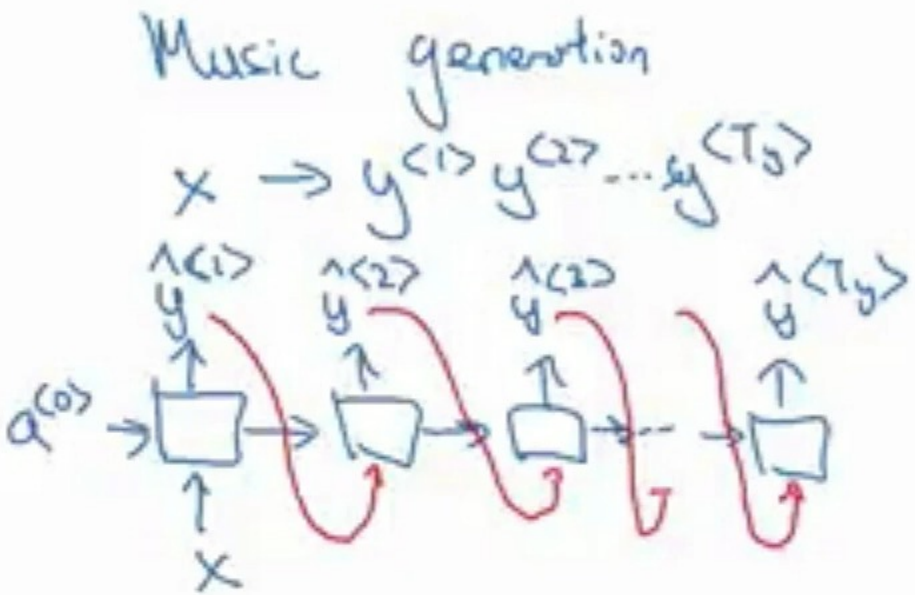


$$\mathcal{L}^{<t>}(\hat{y}^{<t>}, y^{<t>}) = -y^{<t>}\log\hat{y}^{<t>} - (1-y^{<t>})\log(1-\hat{y}^{<t>})$$

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

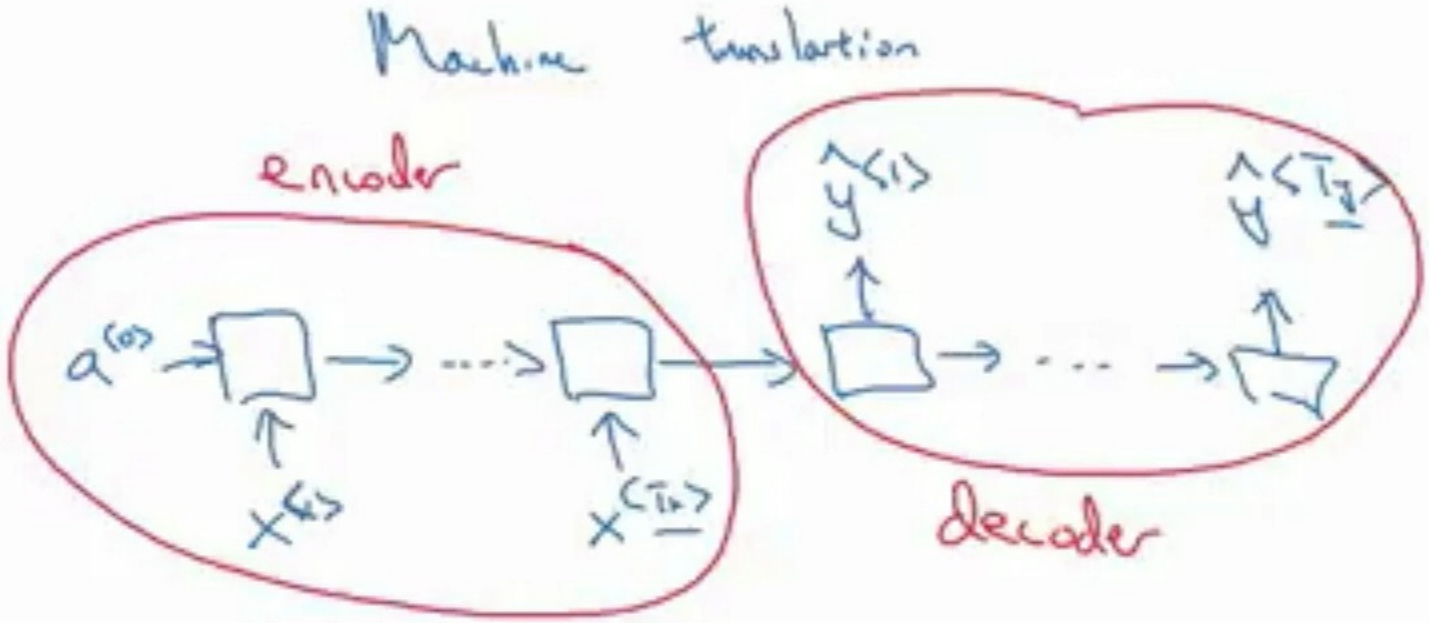
Backpropagation through time

Examples of RNN architectures



One-to-many

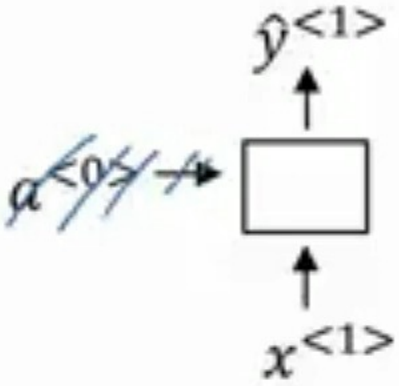
$x = \phi$



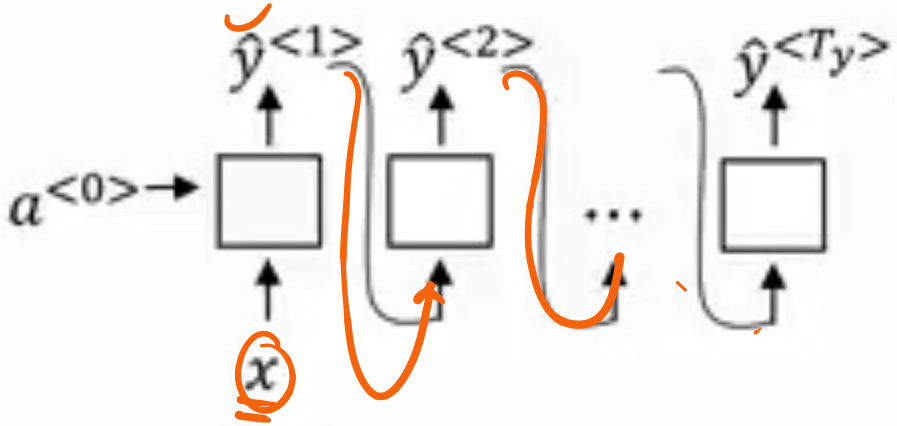
Many-to-many

→ Attention

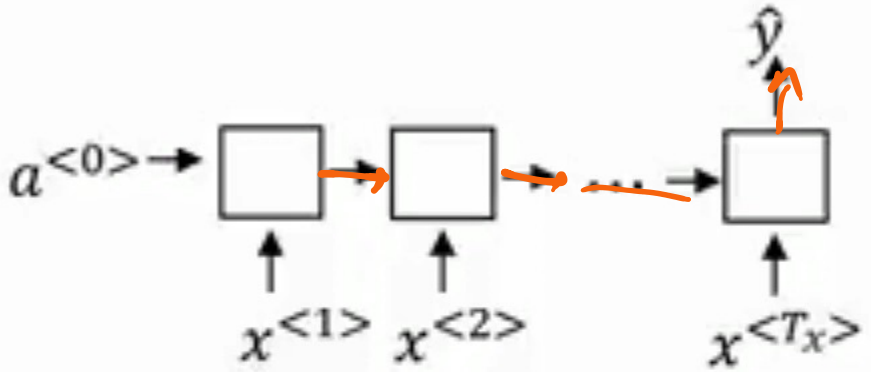
Summary of RNN types



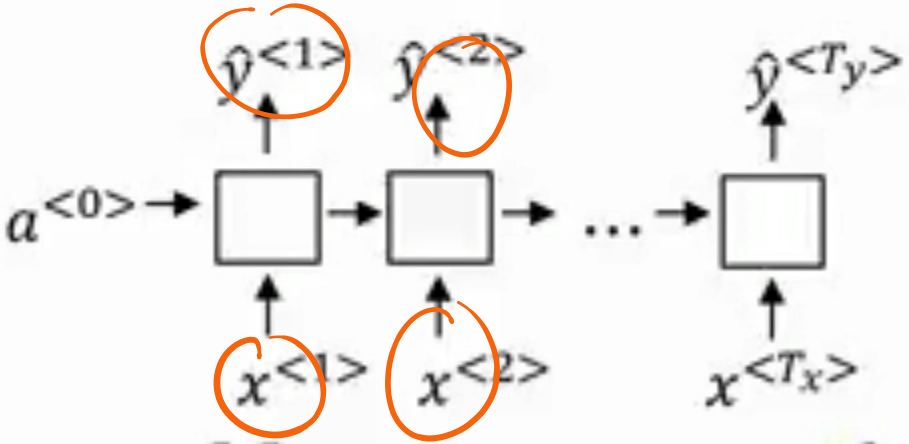
One to one



One to many

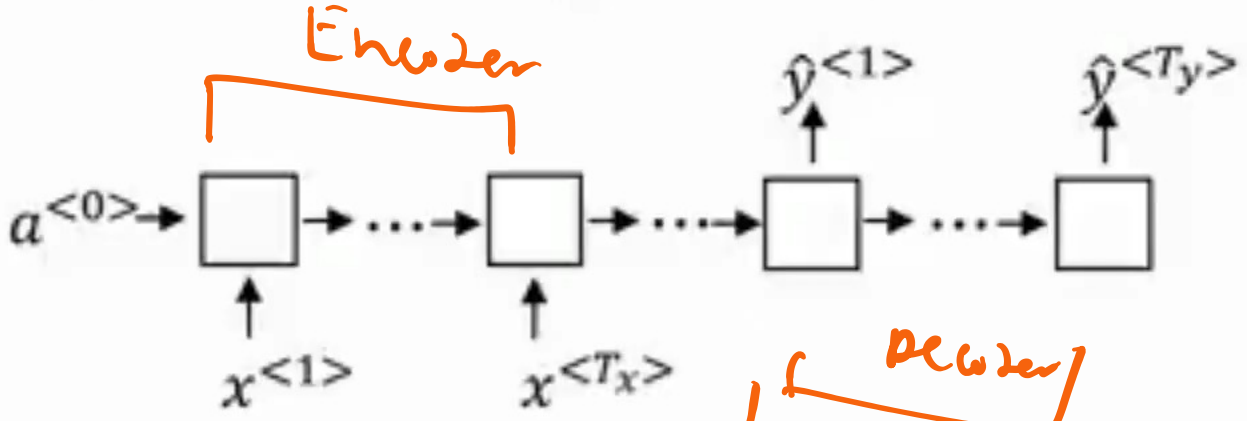


Many to one



Many to many

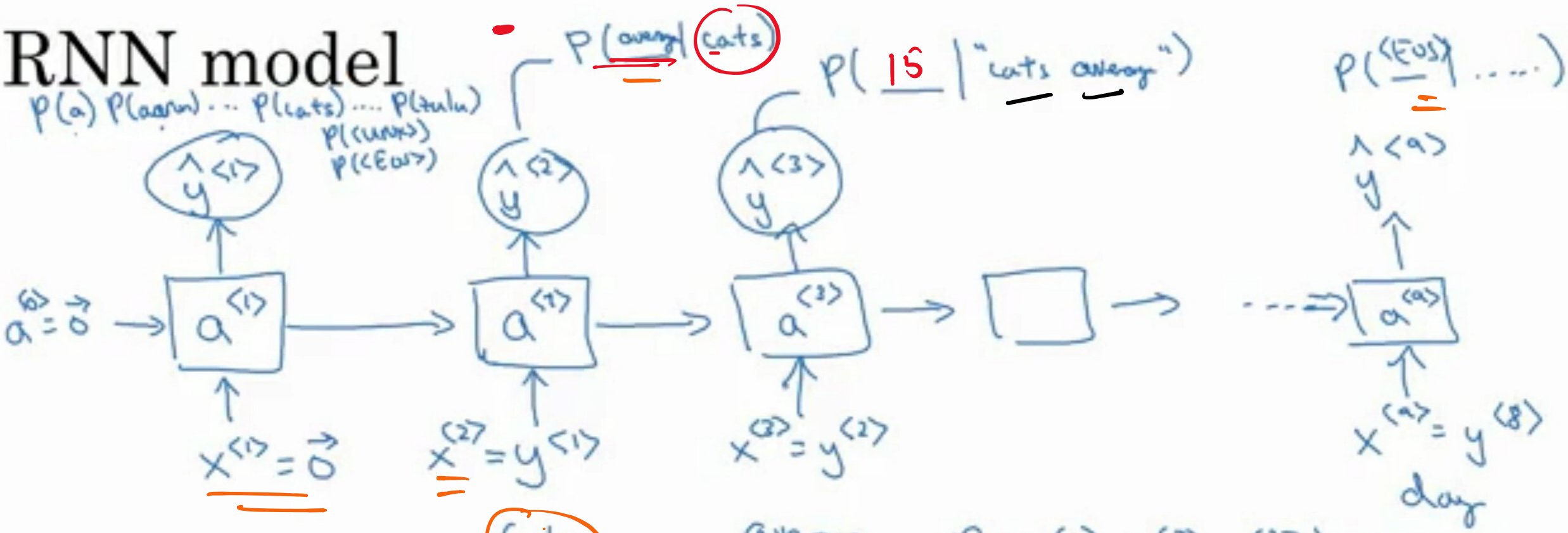
$T_x = T_y$



Many to many

Andrew Ng

RNN model



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

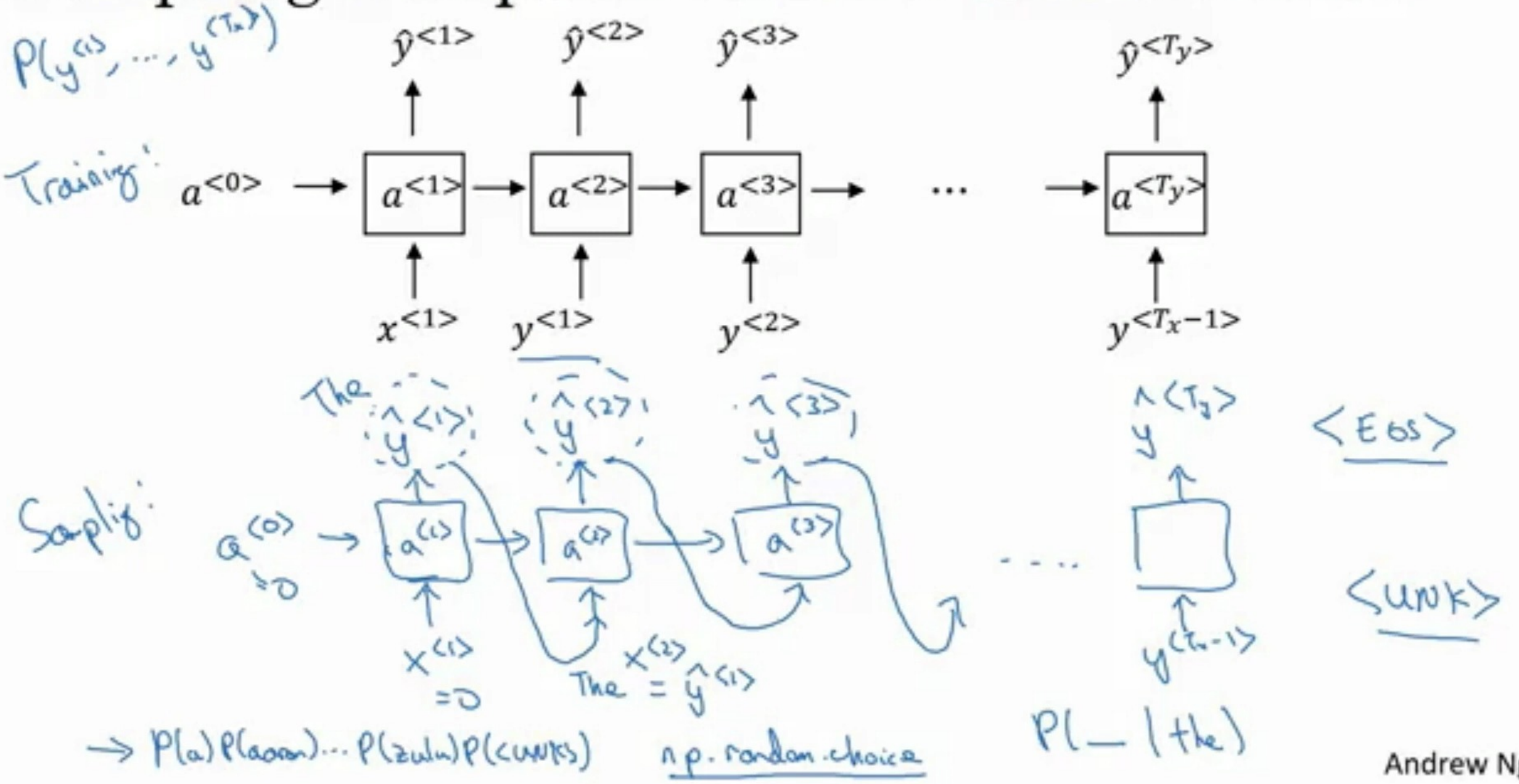
$$\mathcal{L}(\hat{y}^{<t>}, y^{<t>}) = -\sum_i y_i^{<t>} \log \hat{y}_i^{<t>}$$

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

$$P(y^{<1>}, y^{<2>}, y^{<3>}) = P(y^{<1>})\, P(y^{<2>} \mid y^{<1>})\, P(y^{<3>} \mid y^{<1>}, y^{<2>})$$

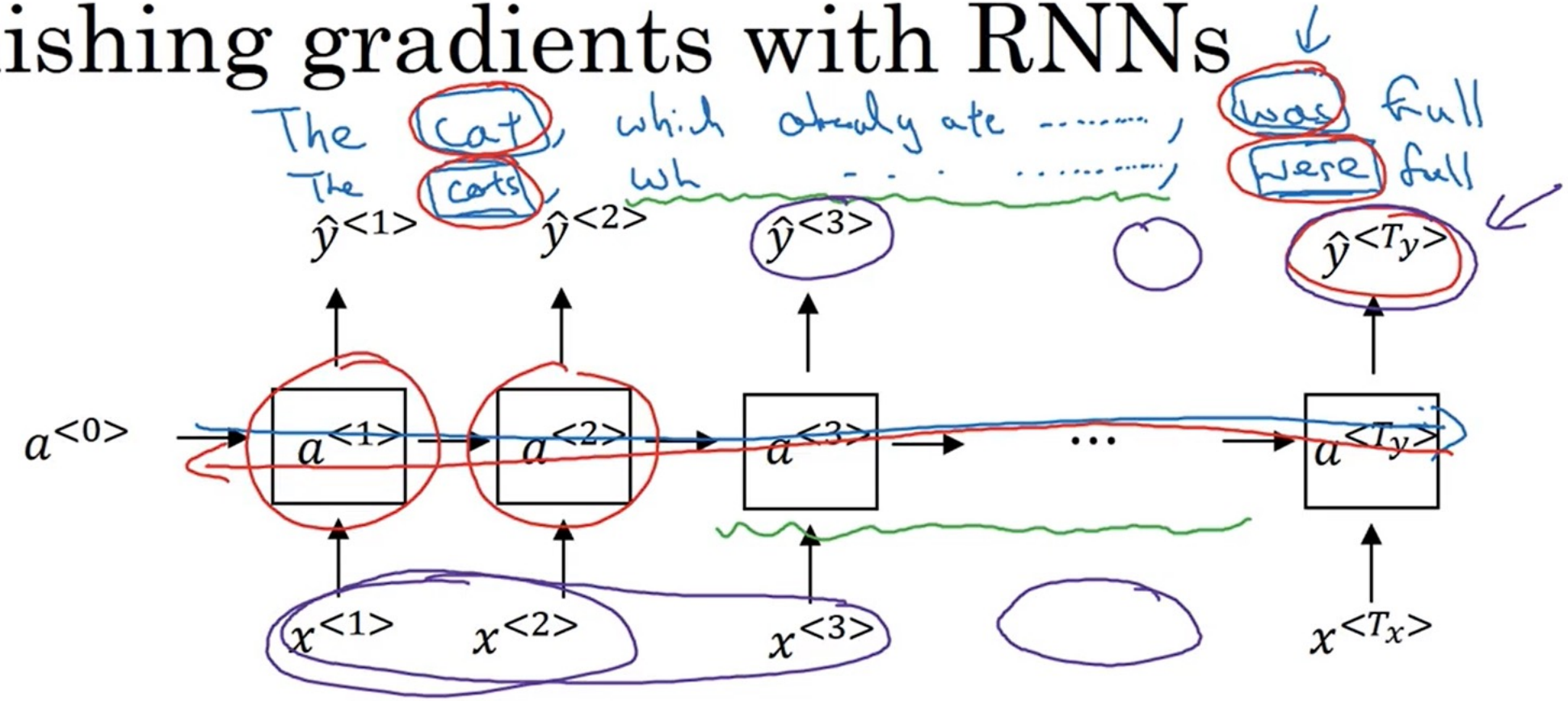
Andrew Ng

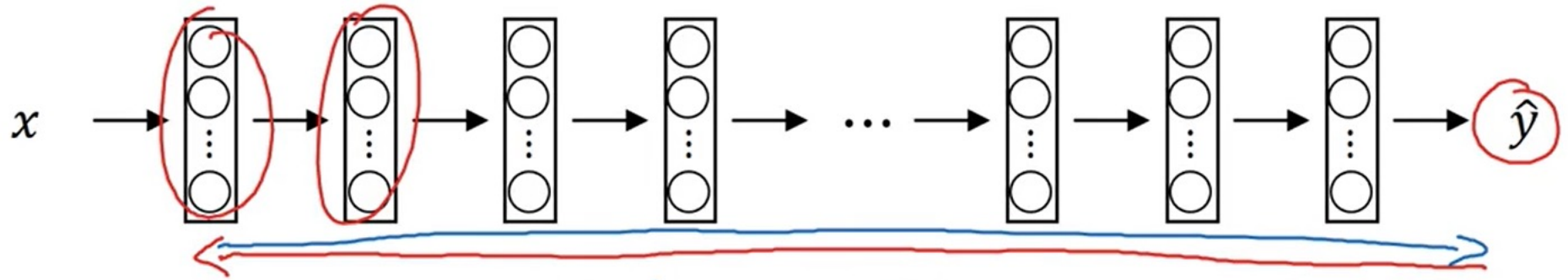
Sampling a sequence from a trained RNN



Andrew Ng

Vanishing gradients with RNNs





Exploding gradients.

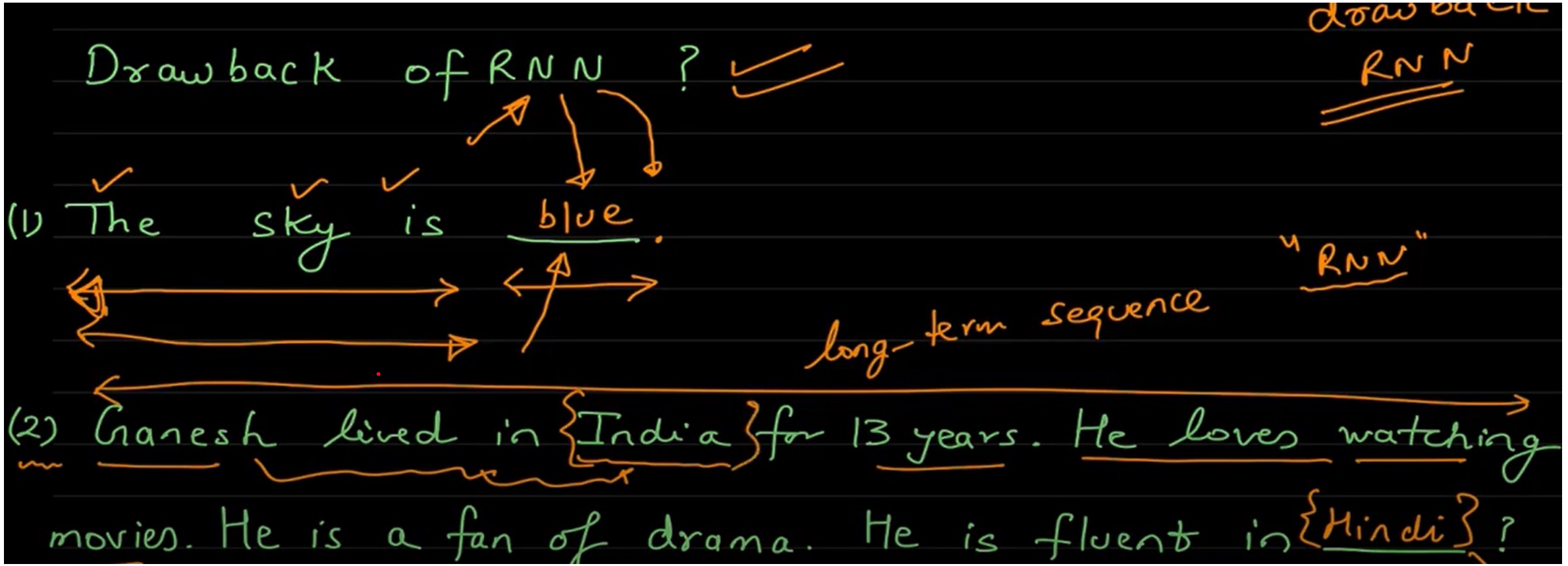
NaN

Gradient clipping

Exponentially Lary

Andrew Ng

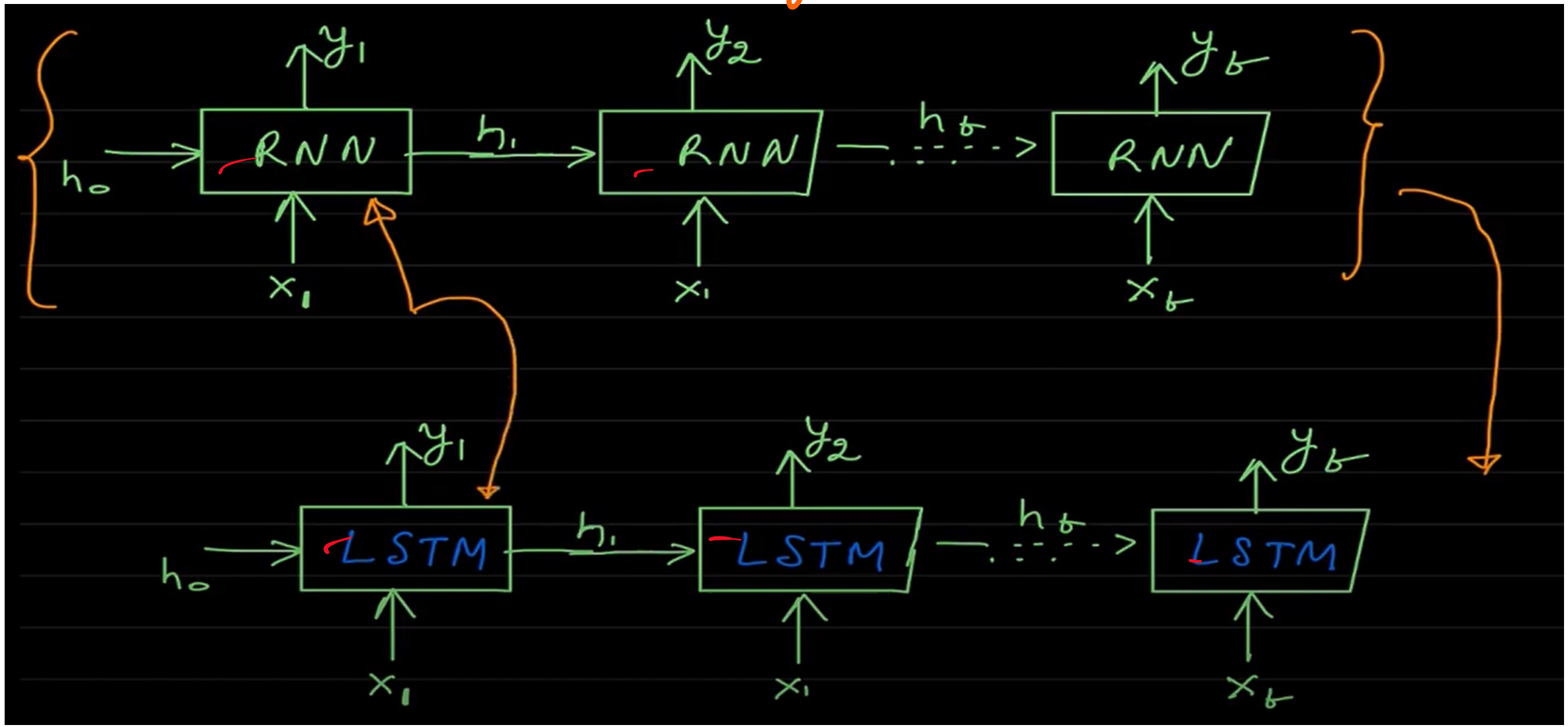
Why LSTM


Drawback of RNN ?
RNN
(1) The sky is blue.
"RNN"
long-term sequence
(2) Ganesh lived in {India} for 13 years. He loves watching movies. He is a fan of drama. He is fluent in {Hindi} ?

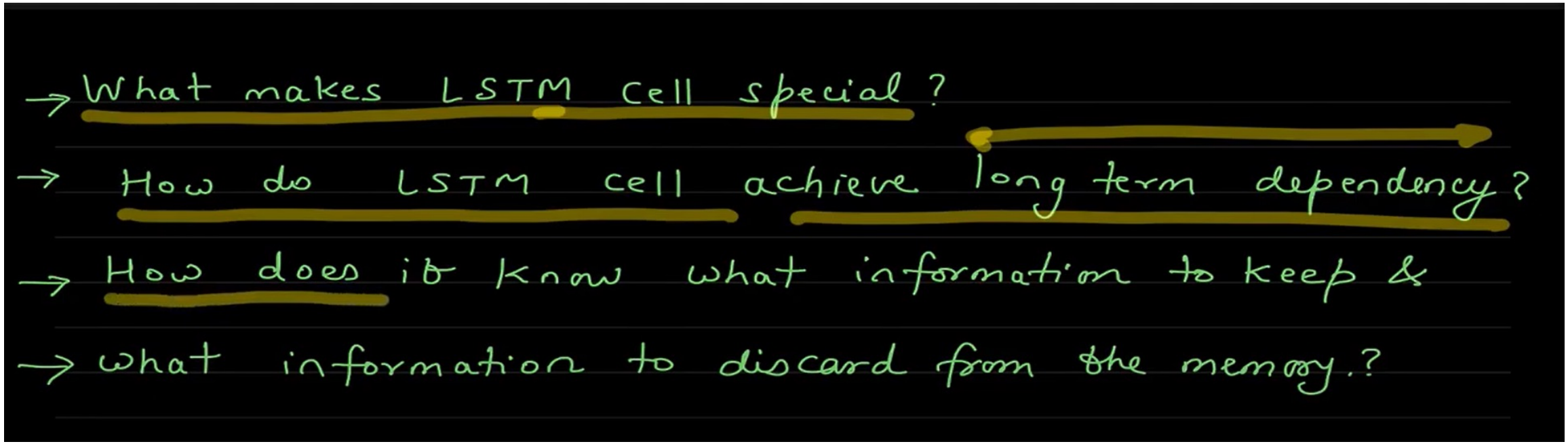
LSTM v/s RNN Architecture

Long short term memory

↳ Short Memory

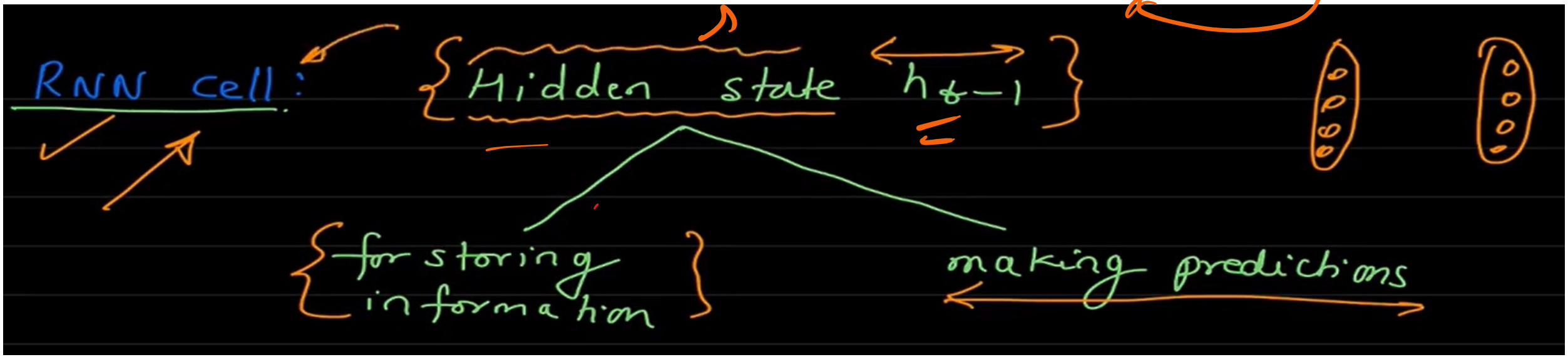


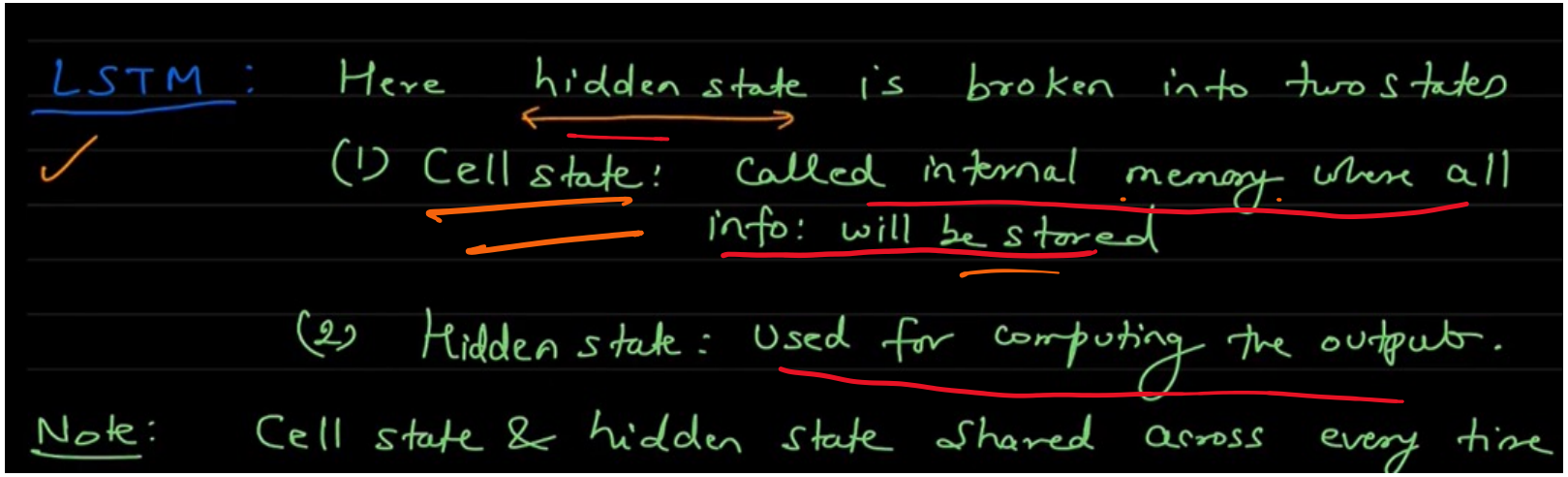
Questions we try to answer in LSTM

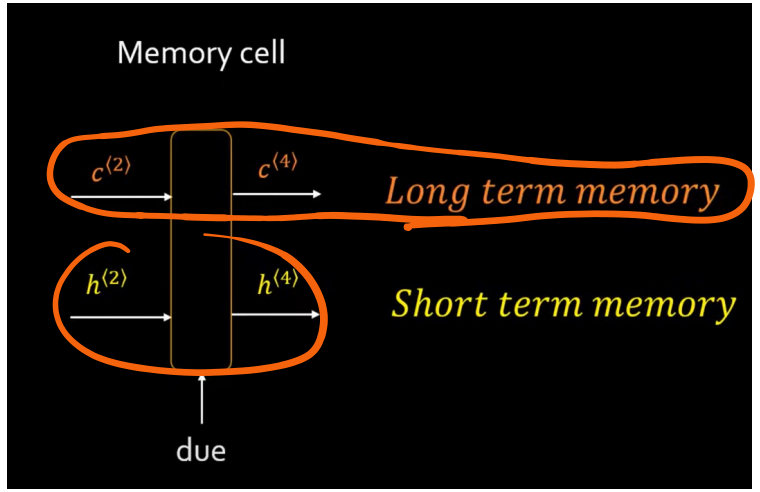


*All these questions can be answered using gates

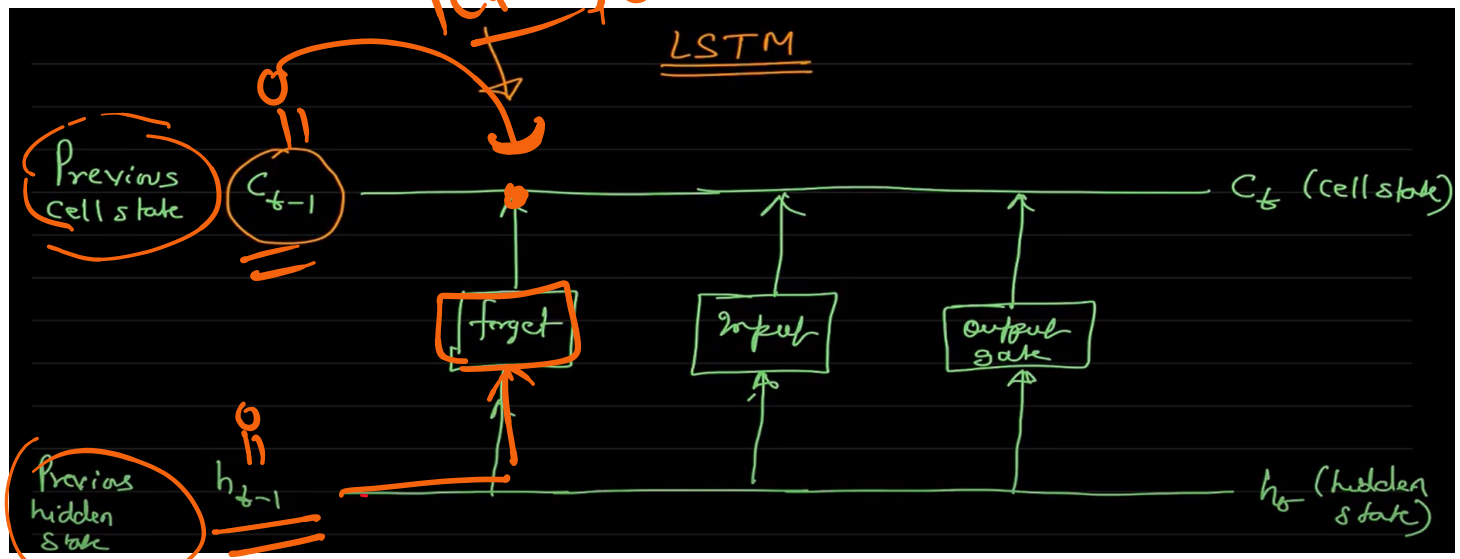
RNN Cell v/s LSTM Cell







LSTM - Forget gate



Forget gate

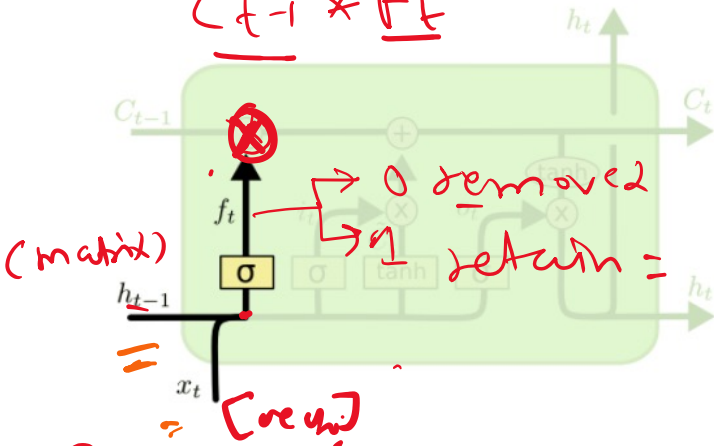
It is responsible for deciding what information should be removed from the cell state.

Consider the following sequence

Mark is a good singer. He lives in California.

Jacob is also a good singer.

Mark is forgotten by forget gate



LSTM – Input gate and Candidate gate

(-1, +1)

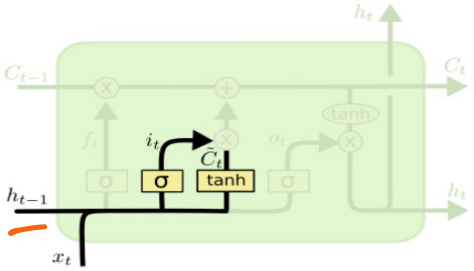
Input Gate:

Input gate is responsible for deciding what information should be stored in the cell state.

Mark is a good singer. He lives in California.

Jacob is also a good singer.

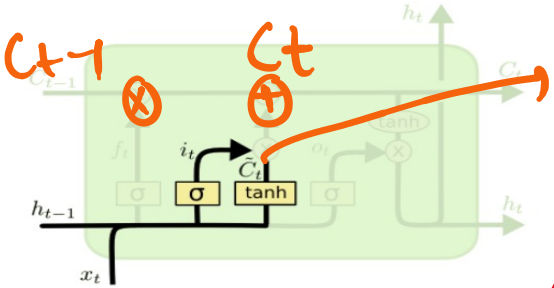
→ Add this into C_{t-1}



0 - info will not be stored and vice versa

$i_t = \sigma(W_i * h_{t-1} + W_i x_t + b_i)$

Updating the cell state: (Adding new information)



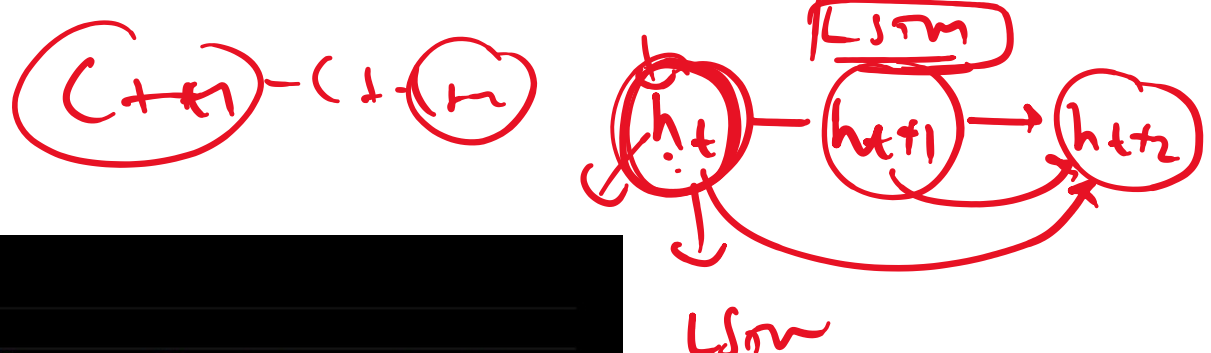
$\tilde{C}_t = \tanh(W_c(h_{t-1} + [C_{-}$
$= W_c(x_t + b_c)$
= (hold state)

This hold state is controlled by input stage

If $i_t * \tilde{C}_t$ = 1 - add new info to ce

$$C_t = f_t * (C_{t-1}) + i_t * (\tilde{C}_t) \rightarrow$$

LSTM – Output gate

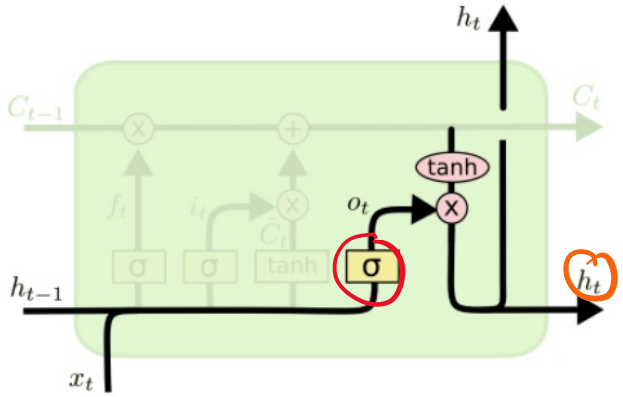


{ Output Gate : }

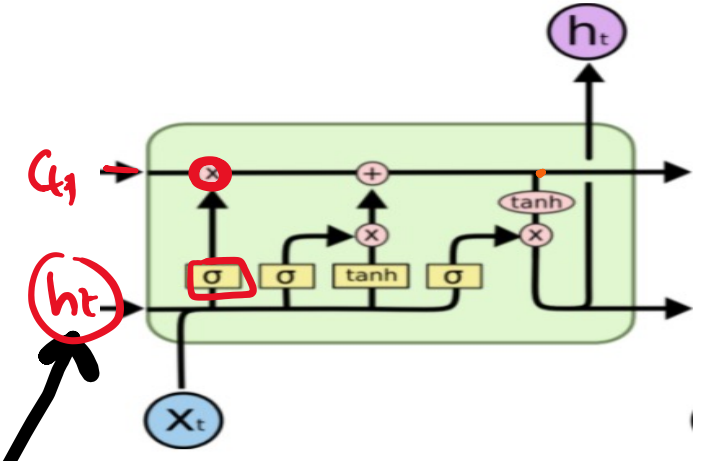
A lot of information is in cell state (memory). The output gate is responsible for deciding what information should be taken from the cell to give as an output.

Consider the following sentence:

Jacob debut album was a huge success. Congratz



h_t output gate $= O_t * \tanh(C_t)$

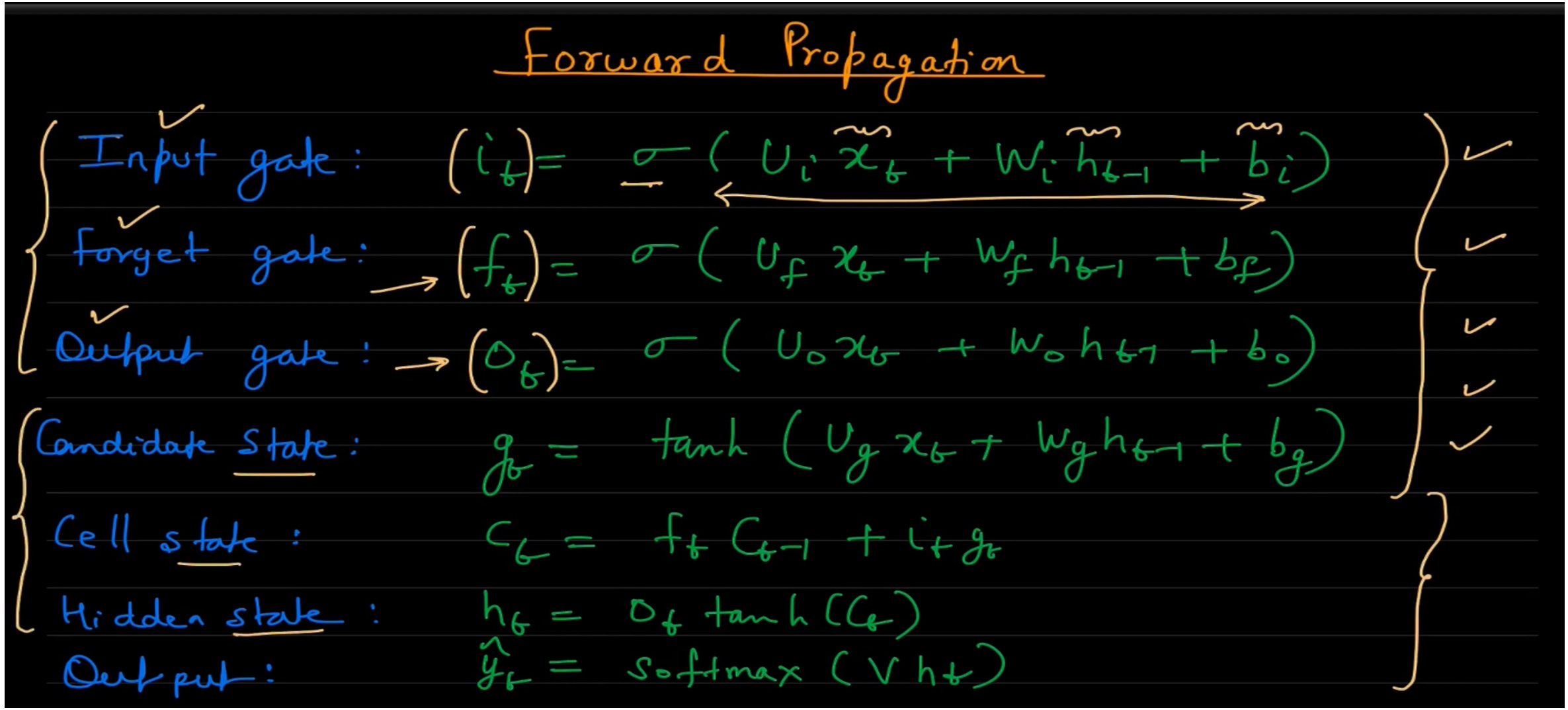


$y_t = \text{Softmax}(h_t)$ — final o/p

$O_t = \sigma(W_o h_{t-1} + W_o x_t + b_o)$

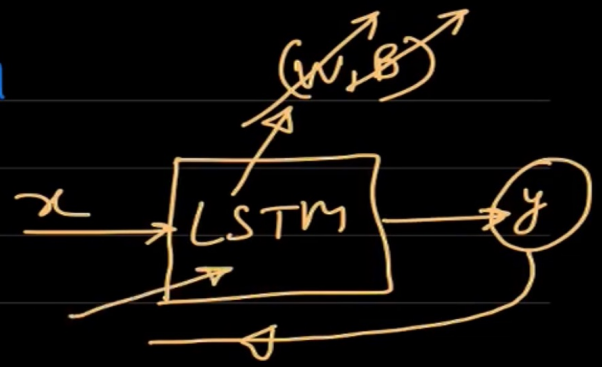
↳ 0 information will not be passed to o/p

LSTM – Forward Propagation



LSTM – Backward Propagation

Backpropagation in LSTM



Loss: $L_t = -y_t \log \hat{y}_t$

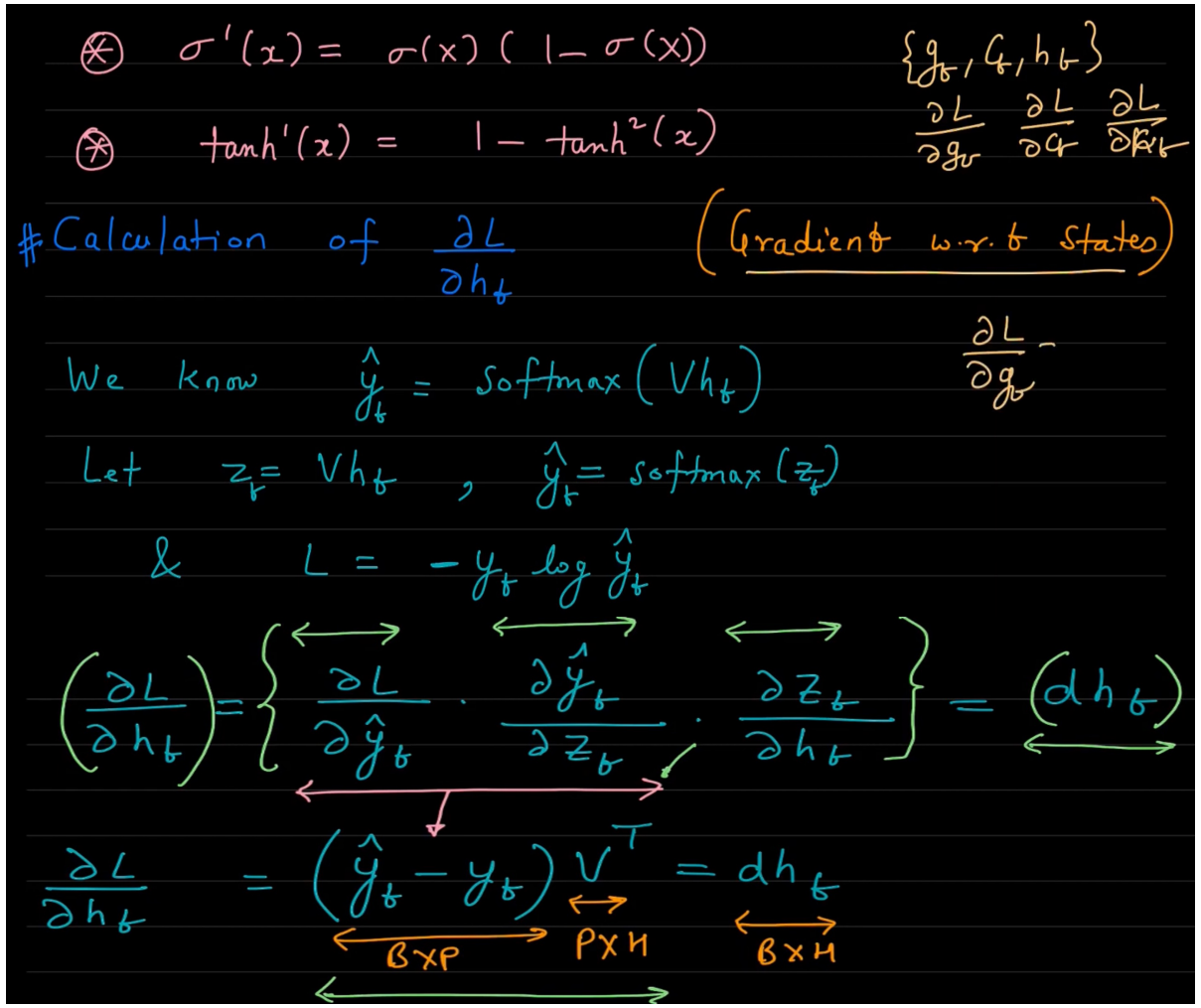
$$L = \sum_{j=0}^{T} L_j$$

$$\left[\frac{\partial L}{\partial U_i}, \frac{\partial L}{\partial U_f}, \frac{\partial L}{\partial U_o}, \frac{\partial L}{\partial U_g}\right], \left[\frac{\partial L}{\partial W_i}, \frac{\partial L}{\partial W_f}, \frac{\partial L}{\partial W_o}, \frac{\partial L}{\partial W_g}\right], \frac{\partial L}{\partial V}$$

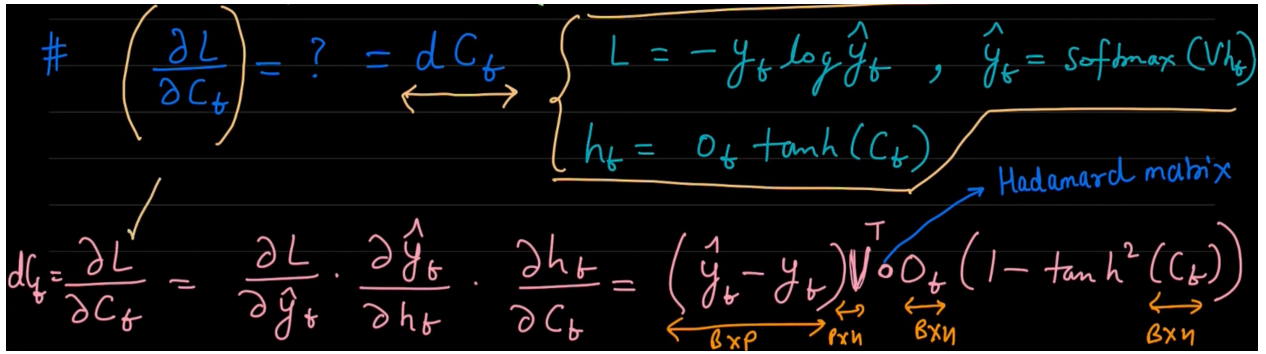
→ Calculating gradients w.r.t weights involve calculation of gradients w.r.t gates & states

LSTM – Backward Propagation Derivatives wrt to states

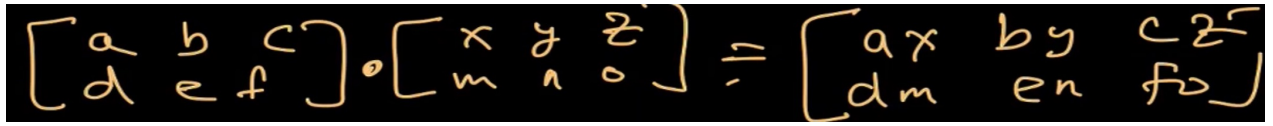
Derivative of loss function wrt h_t



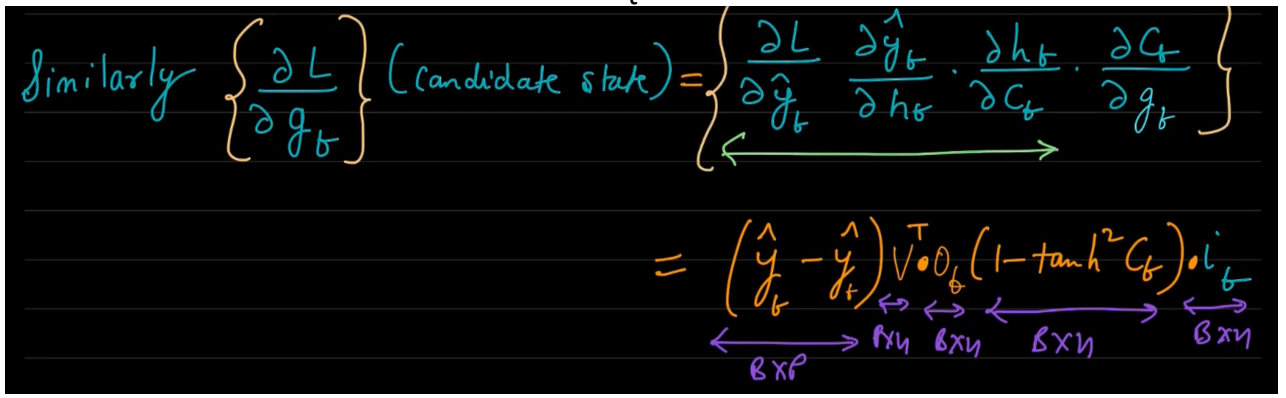
Derivative of loss function wrt C_t



Hadamard Matrix

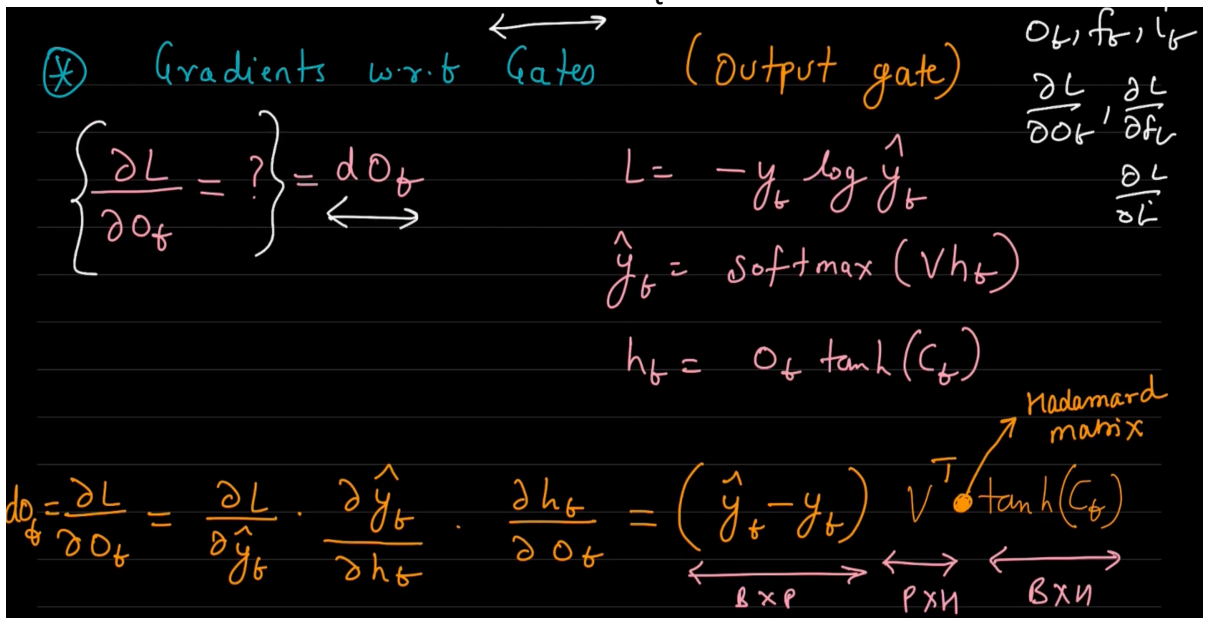


Derivative of loss function wrt G_t

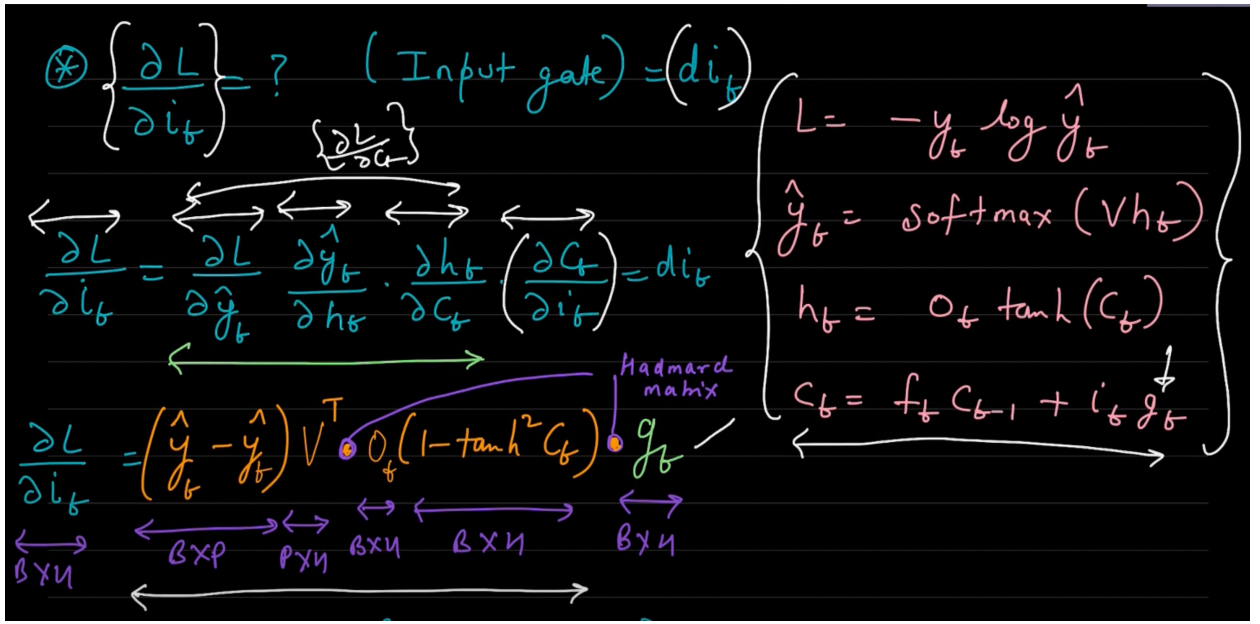


LSTM – Backward Propagation Derivatives wrt to Gates

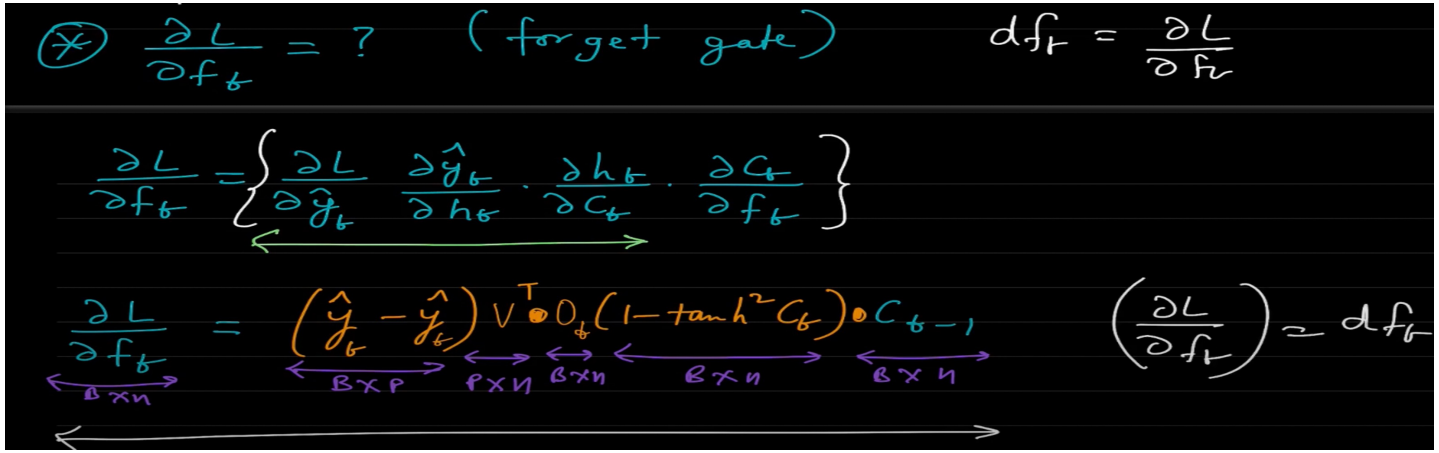
Derivative of loss function wrt O_t



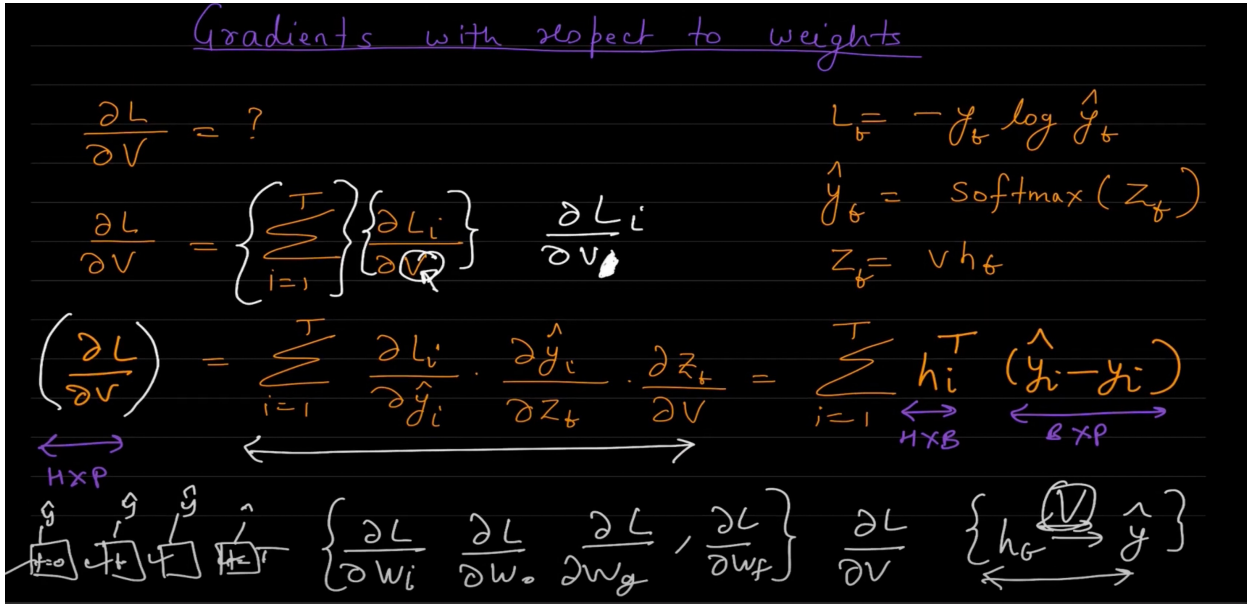
Derivative of loss function wrt I_t

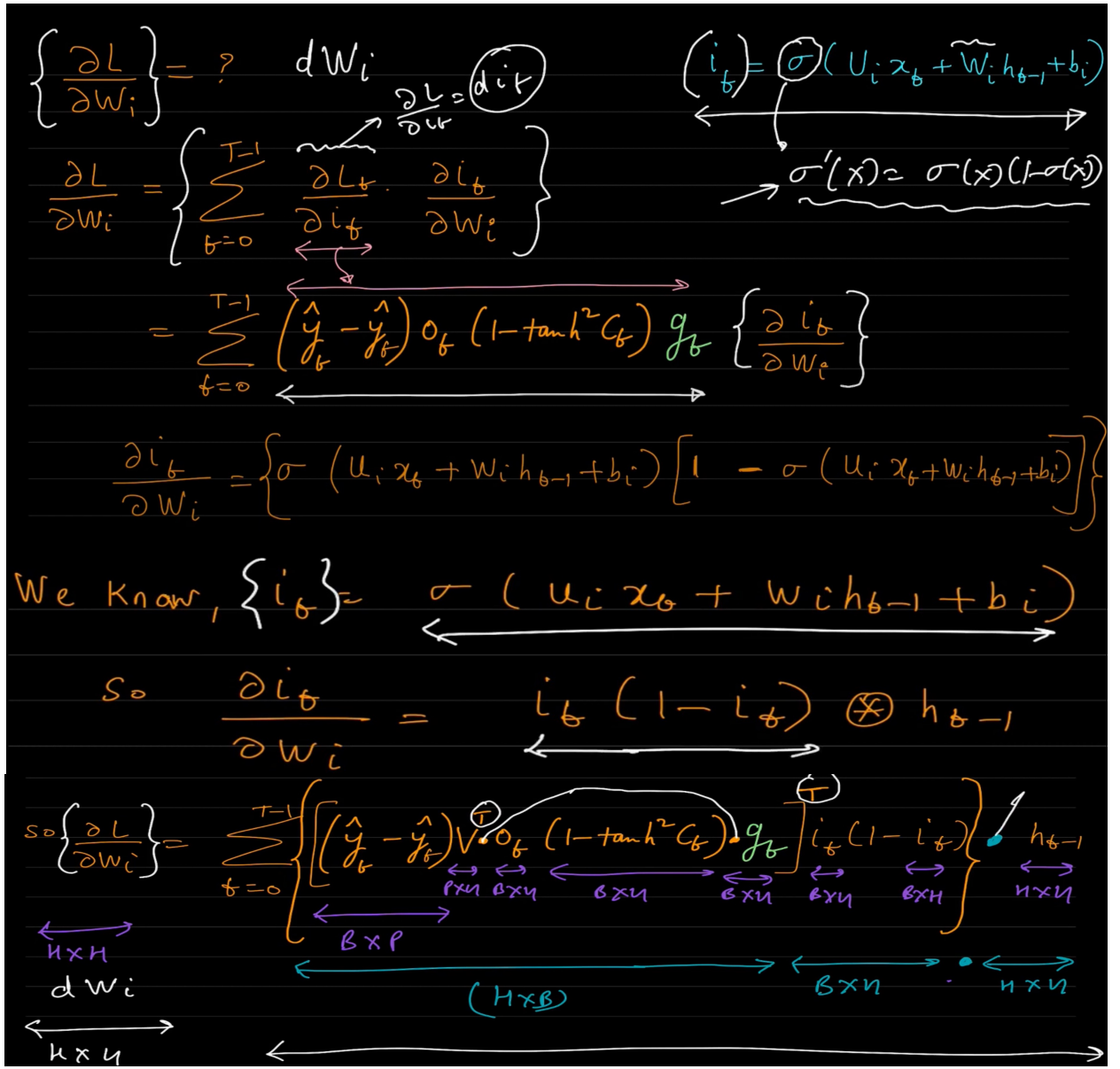


Derivative of loss function wrt f_t

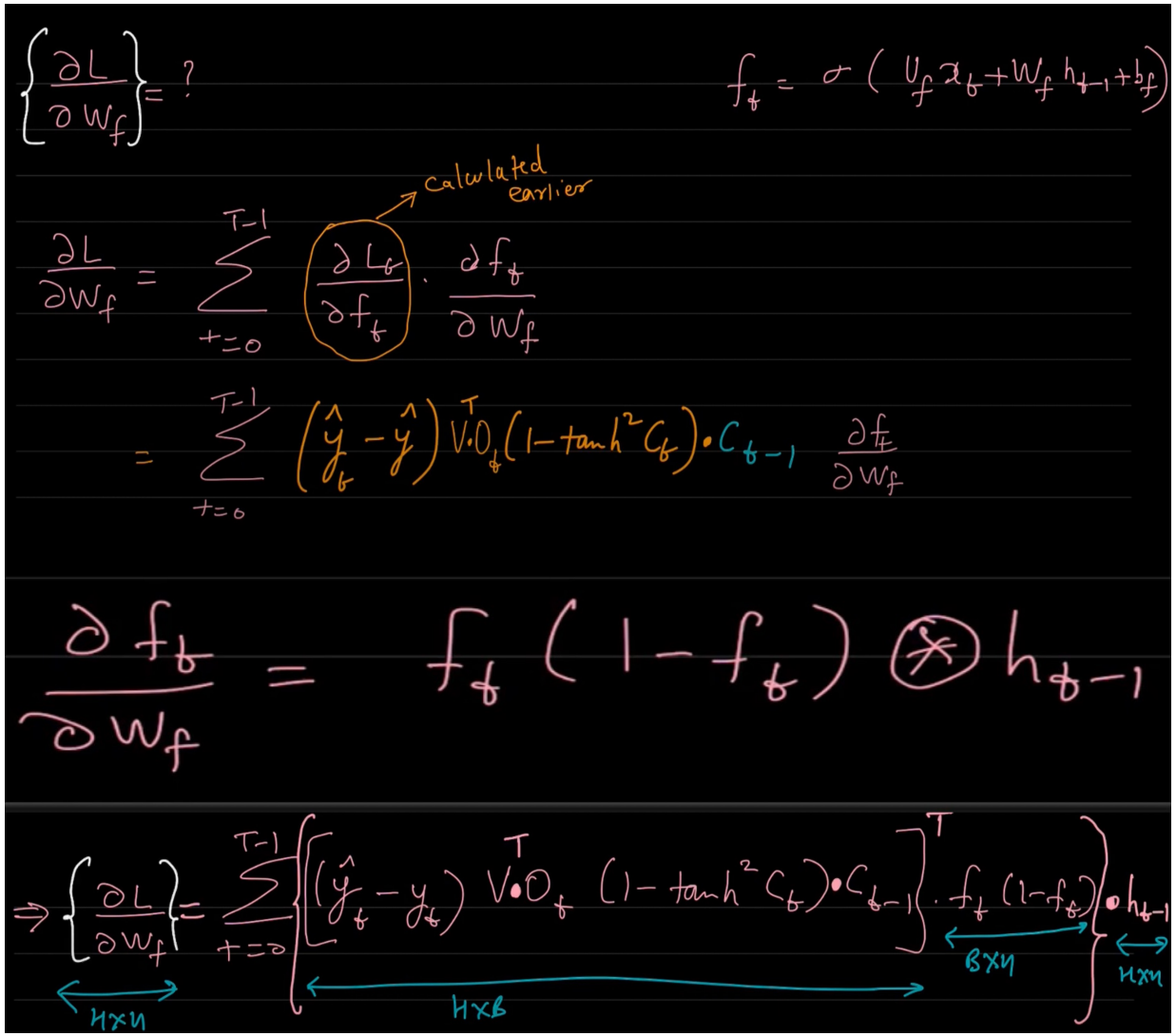


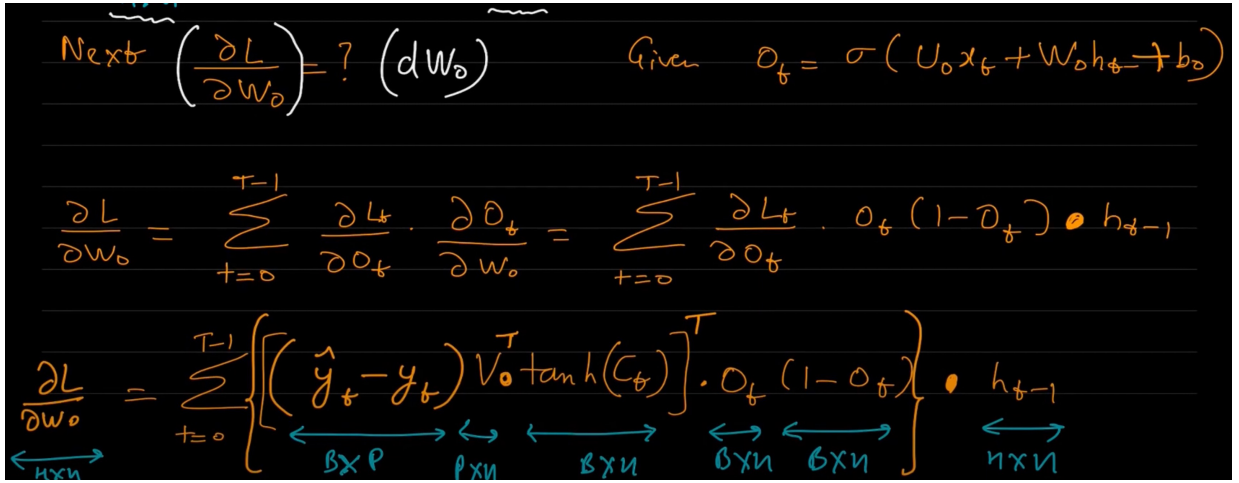
LSTM – Backward Propagation Derivatives wrt to weights

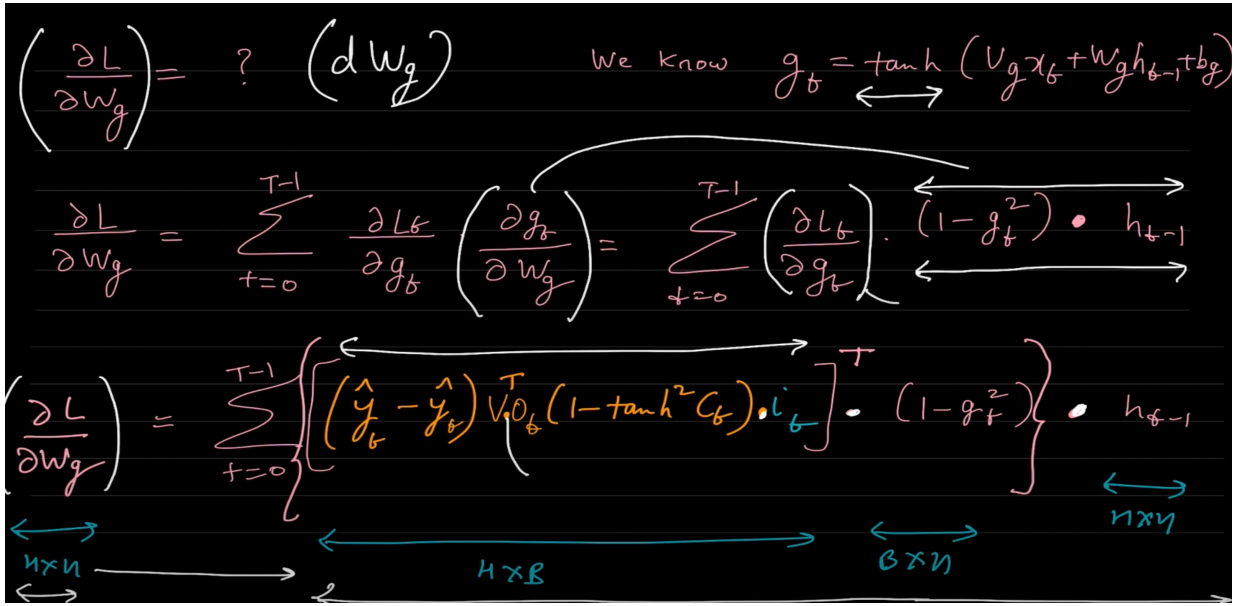




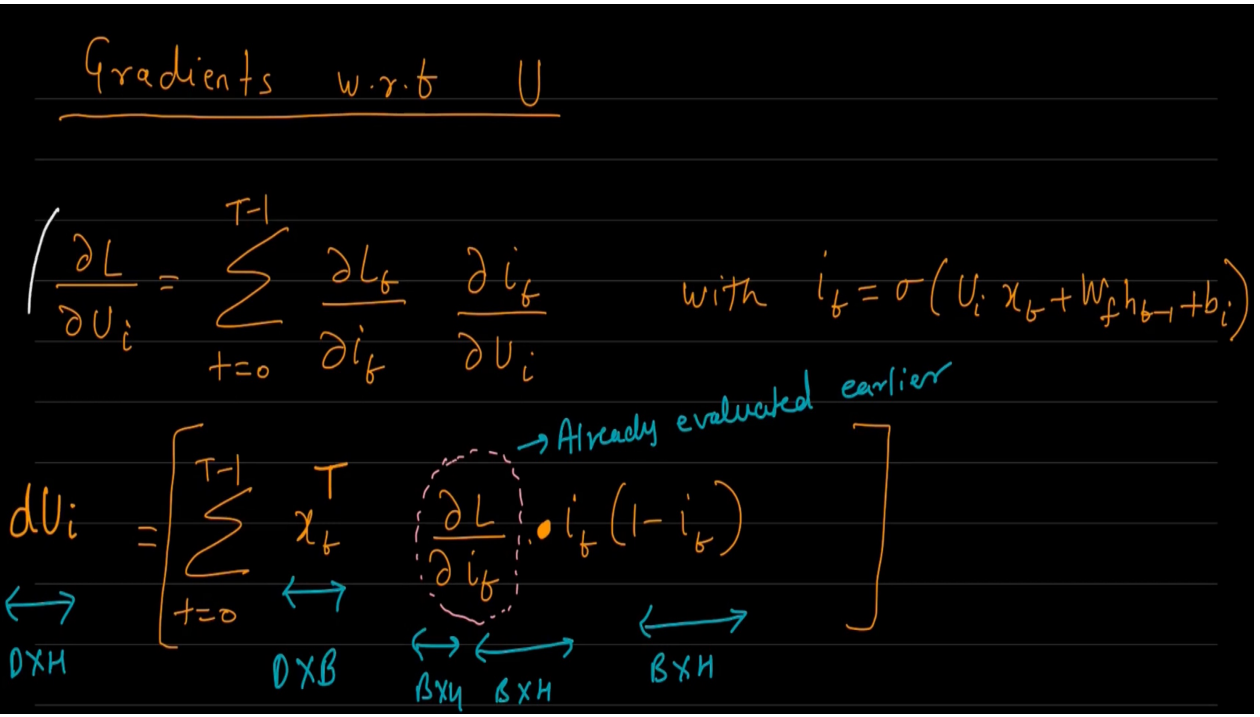
LSTM – Backward Propagation Derivatives wrt to weights contd…

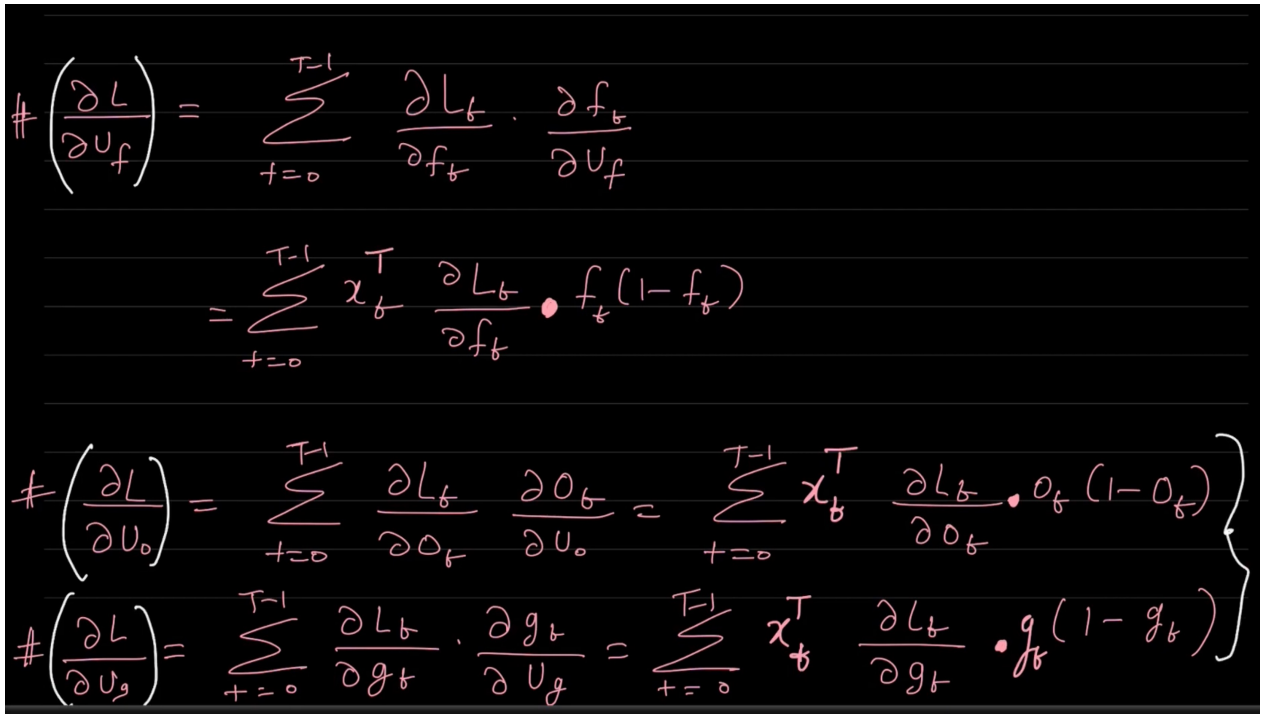




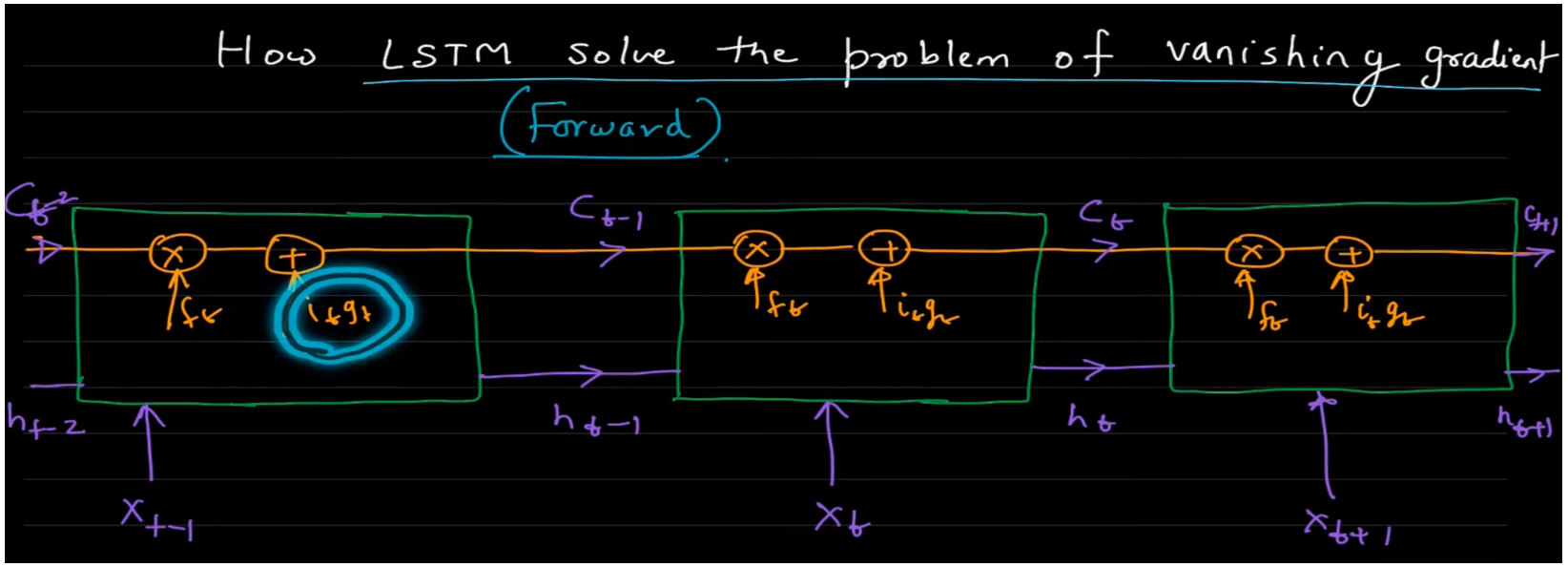


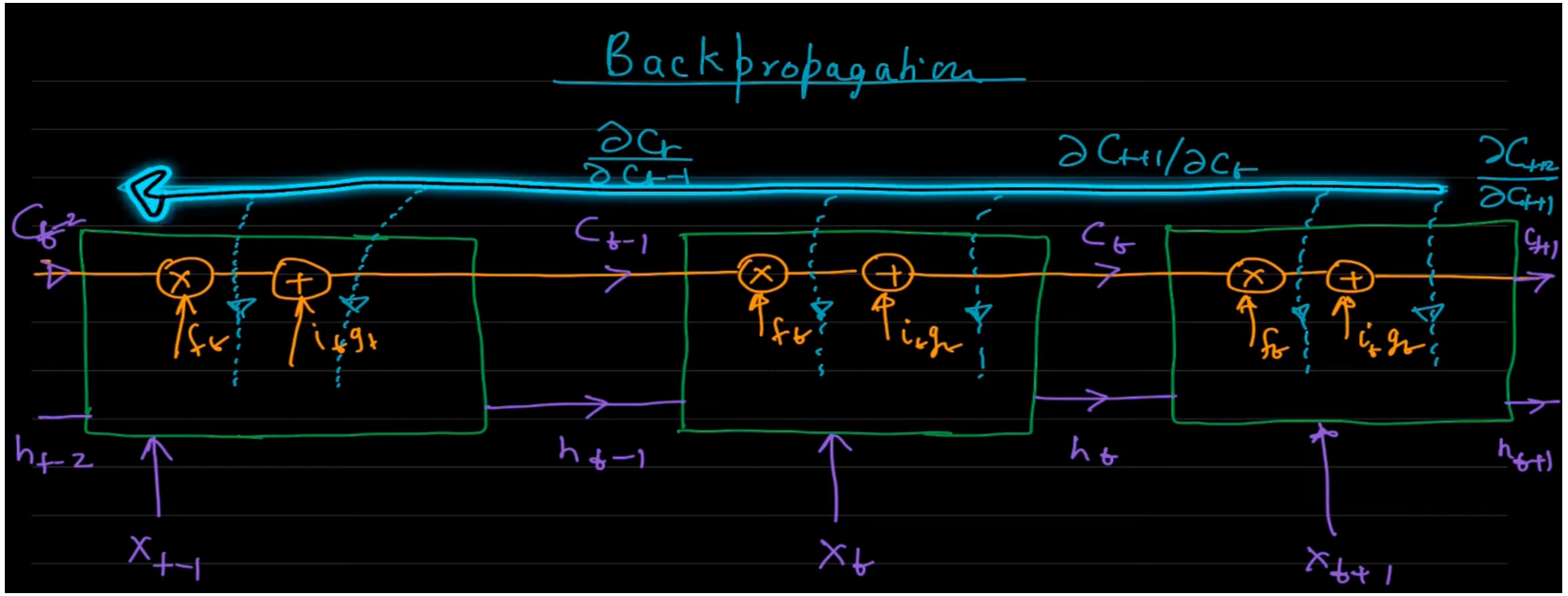
LSTM – Backward Propagation Derivatives wrt to weights contd…



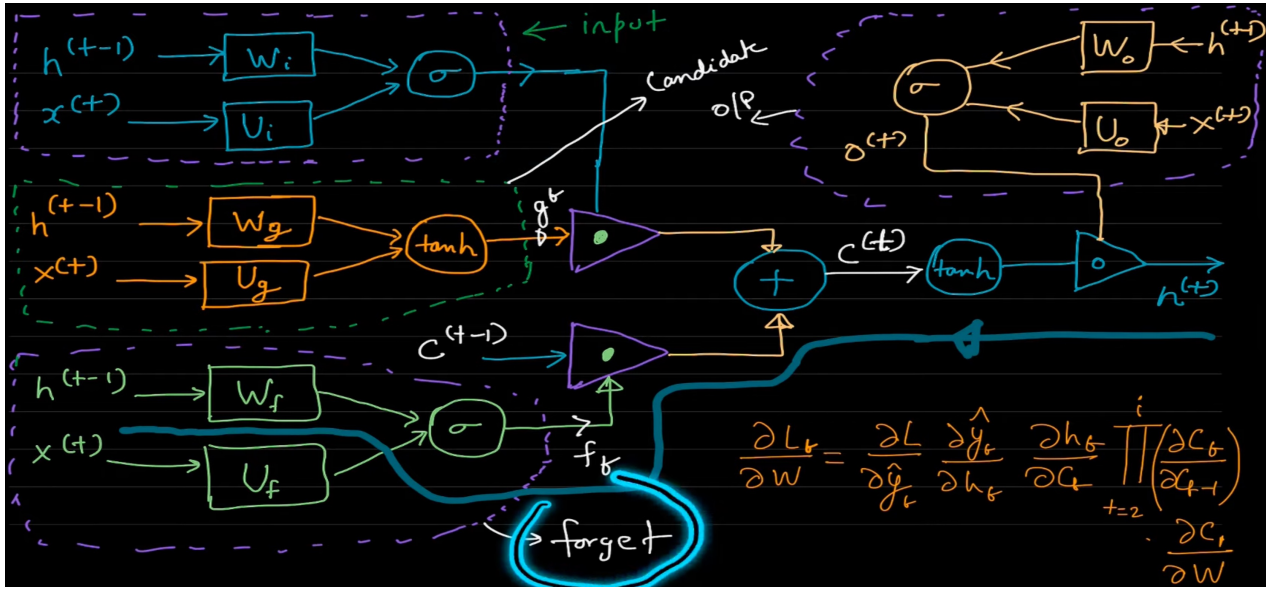


Vanishing Gradient Problem in LSTM





LSTM - Gradients Flow through all Gates



$$\frac{\partial L_t}{\partial w} = \frac{\partial L}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial h_t}\frac{\partial h_t}{\partial C_t}\prod_{t=2}^{i}\left(\frac{\partial C_t}{\partial C_{t-1}}\right)\cdot\frac{\partial C_1}{\partial w}$$

$$\frac{\partial L_t}{\partial w} = \frac{\partial L}{\partial \hat{y}_t}\frac{\partial \hat{y}_t}{\partial h_t}\frac{\partial h_t}{\partial C_t}\left[\prod_{t=2}^{i}\frac{\partial C_t}{\partial C_{t-1}}\right]\frac{\partial C_1}{\partial w}$$

(Motivation) (from RNN)

$$\frac{\partial L}{\partial w} = \sum_{i=1}^{T}\sum_{k=1}^{i}\frac{\partial L_i}{\partial y_i}\frac{\partial y_i}{\partial h_i}\left(\prod_{m=k+1}^{i}\frac{\partial h_m}{\partial h_{m-1}}\right)\frac{\partial^+ h_k}{\partial w}$$

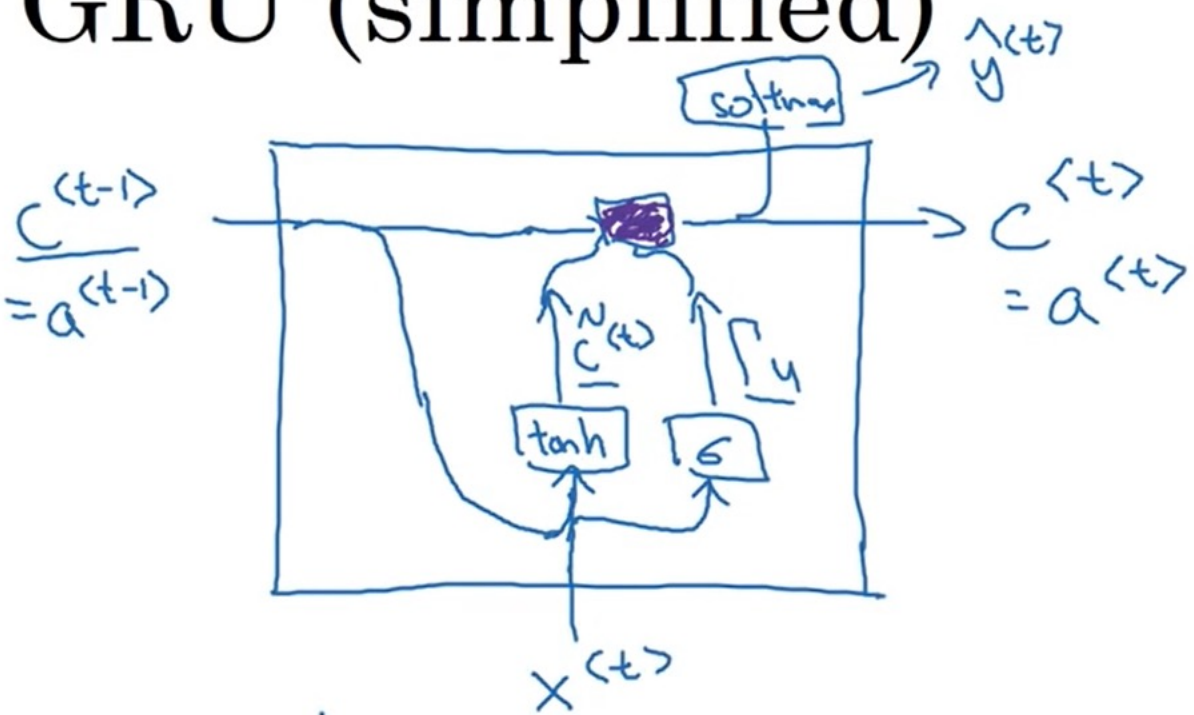
→ exploding
→ vanishing.

Solution: if we make ratio of $\frac{\partial h_m}{\partial h_{m-1}} = 1$, then no problem because $(1)^{b-k} = 1$. This is achieved in case of LSTM where the architecture ensures

$$\frac{\partial C_t}{\partial C_{t-1}} = 1$$

Because this gradient is exactly one, it ensures that information stored in memory cell not suffer from vanishing gradient.

GRU (simplified)



c = memory cell

$\rightarrow c^{\langle t \rangle} = a^{\langle t \rangle}$

$\rightarrow \tilde{c}^{\langle t \rangle} = \tanh(W_c [c^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_c)$

$\rightarrow \Gamma_u = \sigma(W_u [c^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_u)$

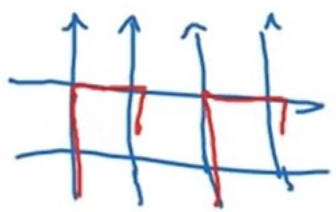
← "update"

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

$\Gamma_u = 1$ $\Gamma_u = 0$ $\Gamma_u = 0$ $\Gamma_u = 0$ ..... $= 1$

$c^{\langle t \rangle} = 1$

$\Gamma_u = 0.00001$



Gate

→ 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] ←

Full GRU

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

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

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

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

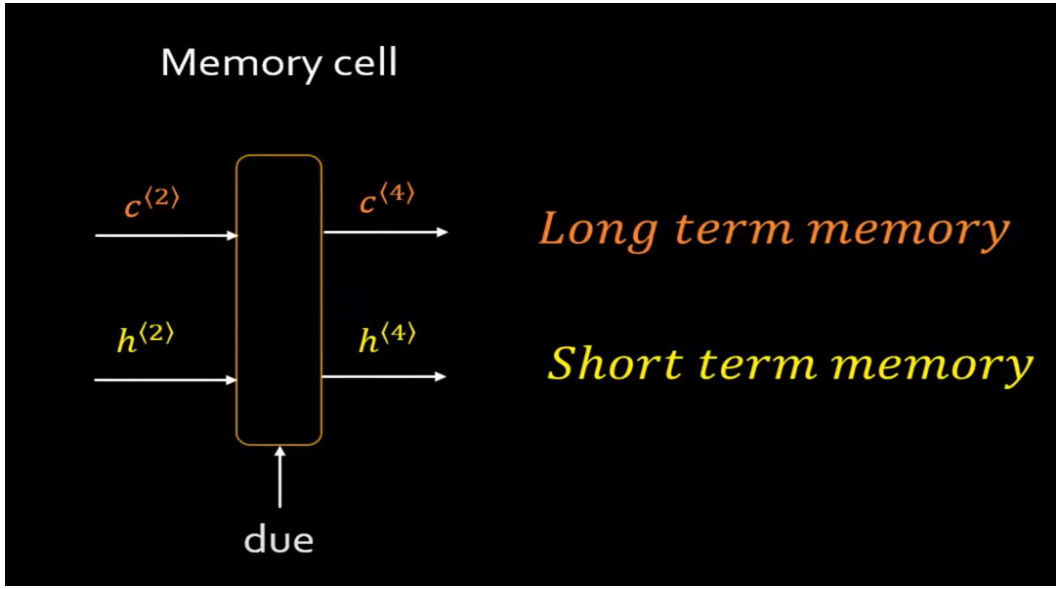
LSTM

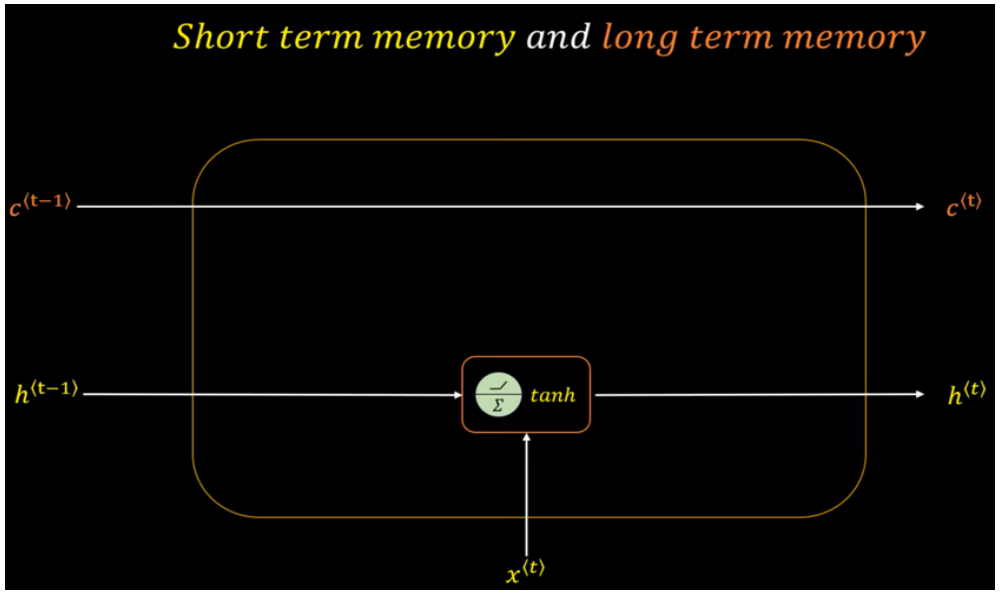
The cat, which ate already, was full.

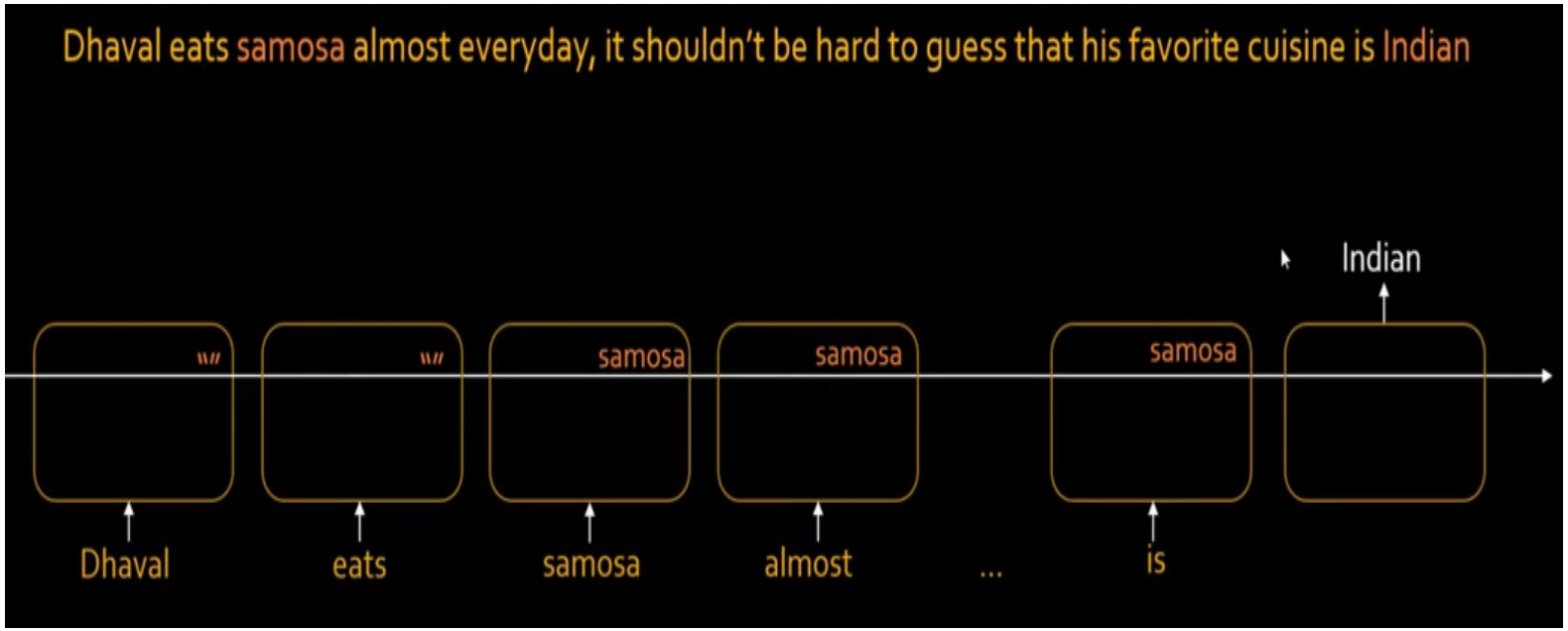
Why tanh is used in LSTM

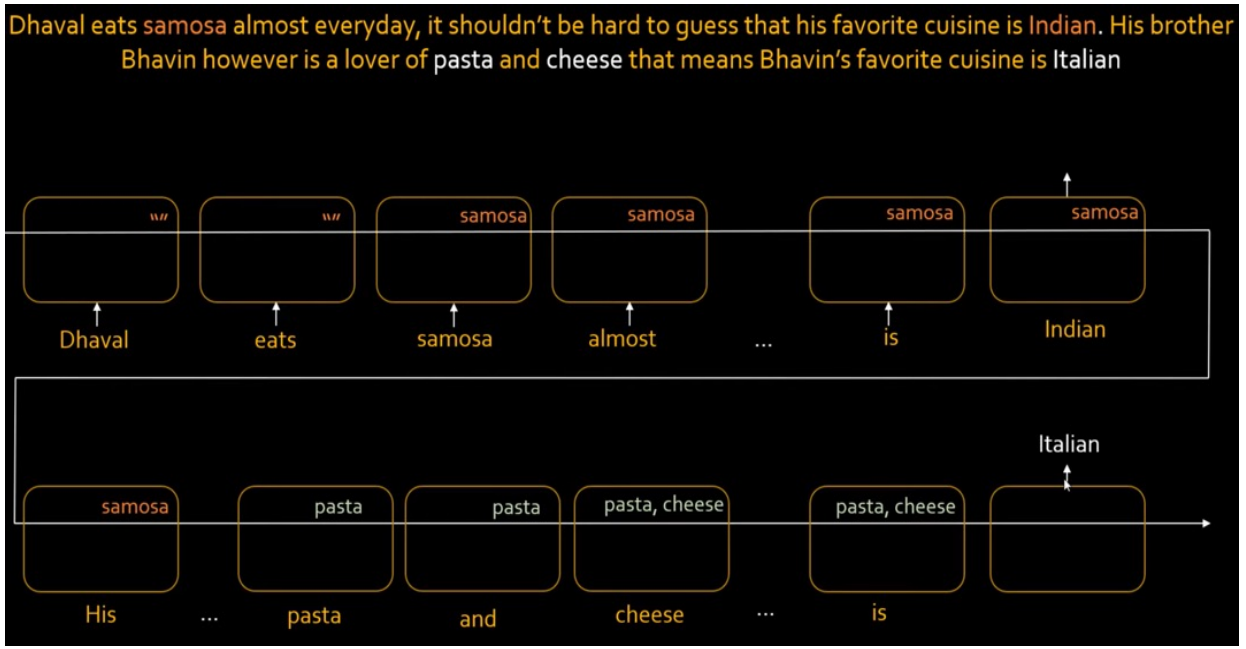
- https://stackoverflow.com/questions/40761185/what-is-the-intuition-of-using-tanh-in-lstm
- https://www.reddit.com/r/MachineLearning/comments/9elxs8/d_why_do_you_use_tanh_in_a_rnn/

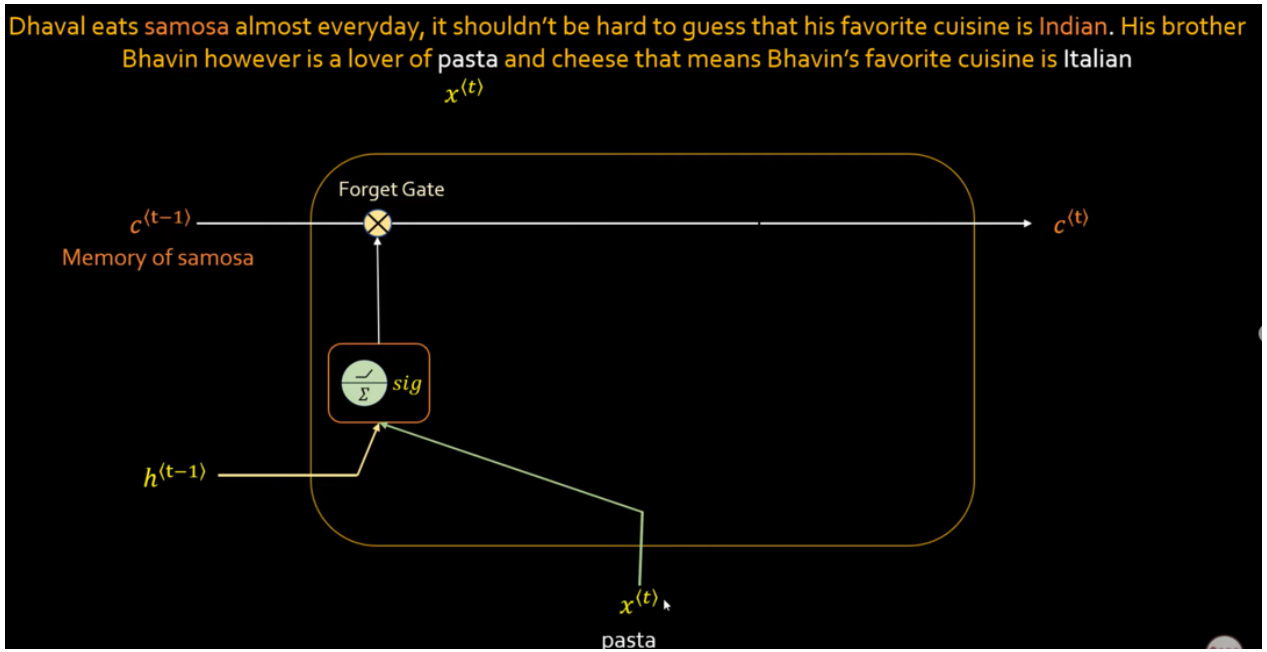
LSTM Example:

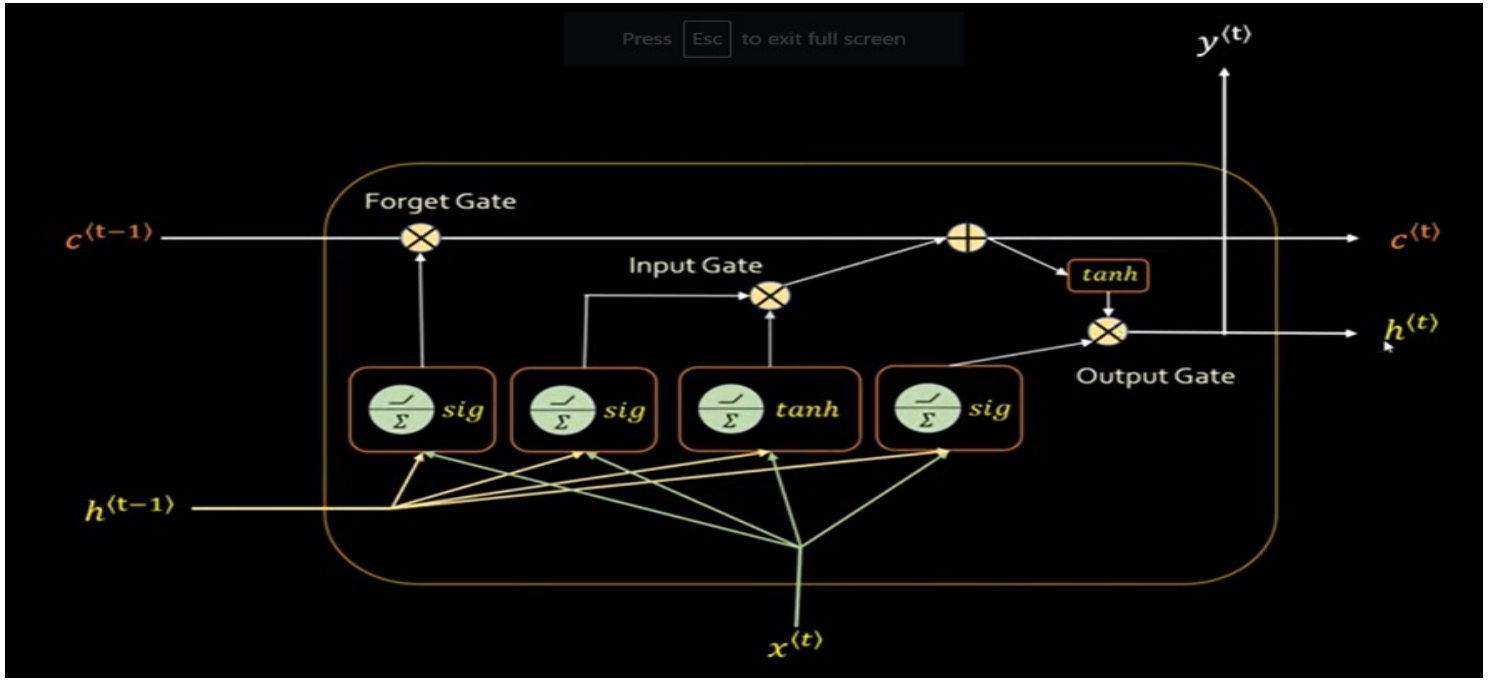







Dhaval eats samosa almost everyday, it shouldn't be hard to guess that his favorite cuisine is Indian. His brother Bhavin however is a lover of pasta and cheese that means Bhavin's favorite cuisine is Italian
Dhaval
eats
samosa
almost
...
is
Indian
His
...
pasta
and
cheese
...
is
Italian


Dhaval eats samosa almost everyday, it shouldn't be hard to guess that his favorite cuisine is Indian. His brother Bhavin however is a lover of pasta and cheese that means Bhavin's favorite cuisine is Italian
$x^{\langle t \rangle}$
Forget Gate
$c^{\langle t-1 \rangle}$
Memory of samosa
$c^{\langle t \rangle}$
sig
$h^{\langle t-1 \rangle}$
$x^{\langle t \rangle}$
pasta


Press Esc to exit full screen
$y^{\langle t \rangle}$
Forget Gate
$c^{\langle t-1 \rangle}$
Input Gate
tanh
$c^{\langle t \rangle}$
$h^{\langle t \rangle}$
Output Gate
sig
sig
tanh
sig
$h^{\langle t-1 \rangle}$
$x^{\langle t \rangle}$

LSTM in pictures

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

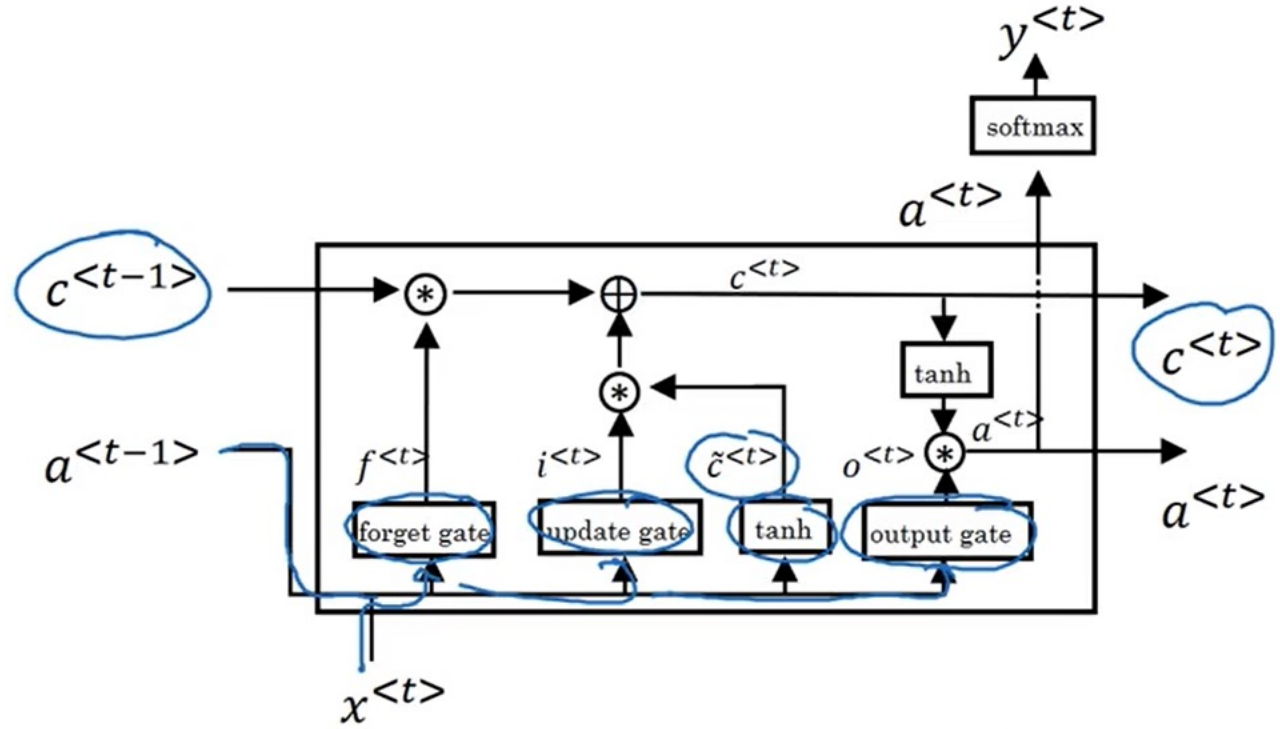
$$\Gamma_u = \sigma(W_u[a^{<t-1>}, x^{<t>}] + b_u)$$

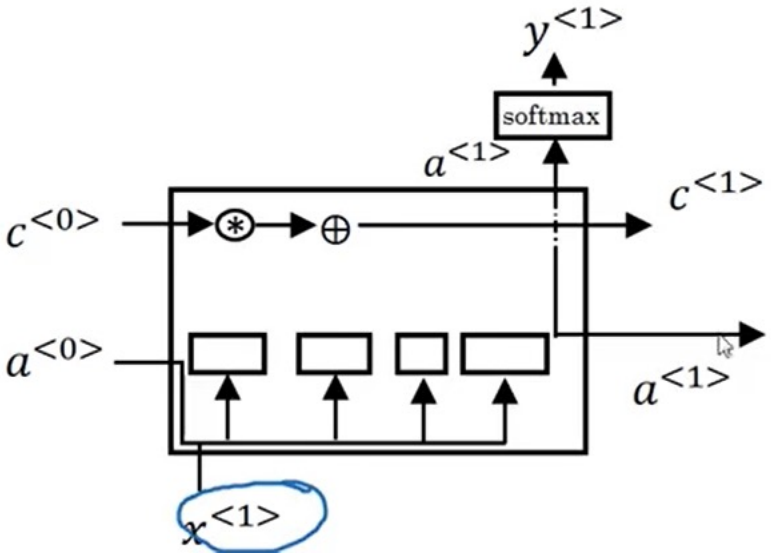
$$\Gamma_f = \sigma(W_f[a^{<t-1>}, x^{<t>}] + b_f)$$

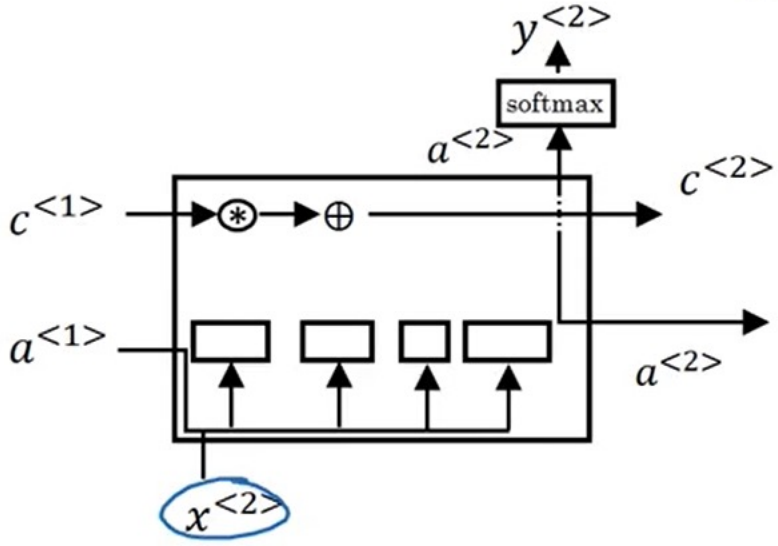
$$\Gamma_o = \sigma(W_o[a^{<t-1>}, x^{<t>}] + b_o)$$

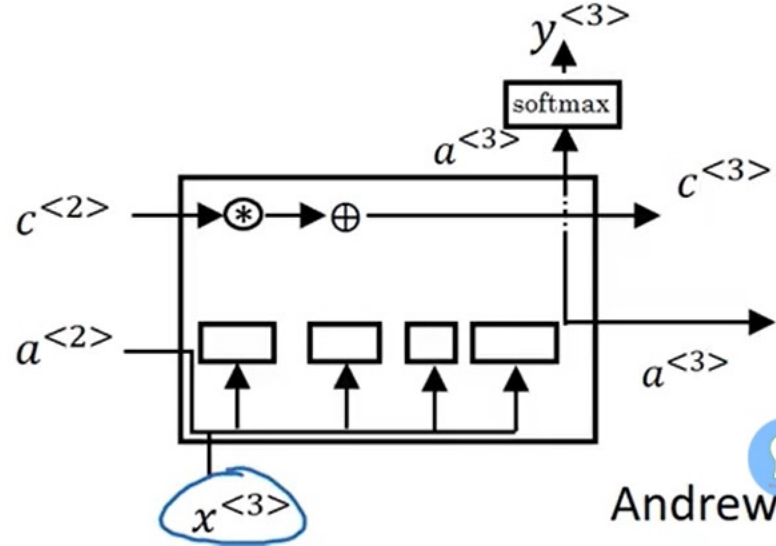
$$c^{<t>} = \Gamma_u * \tilde{c}^{<t>} + \Gamma_f * c^{<t-1>}$$

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



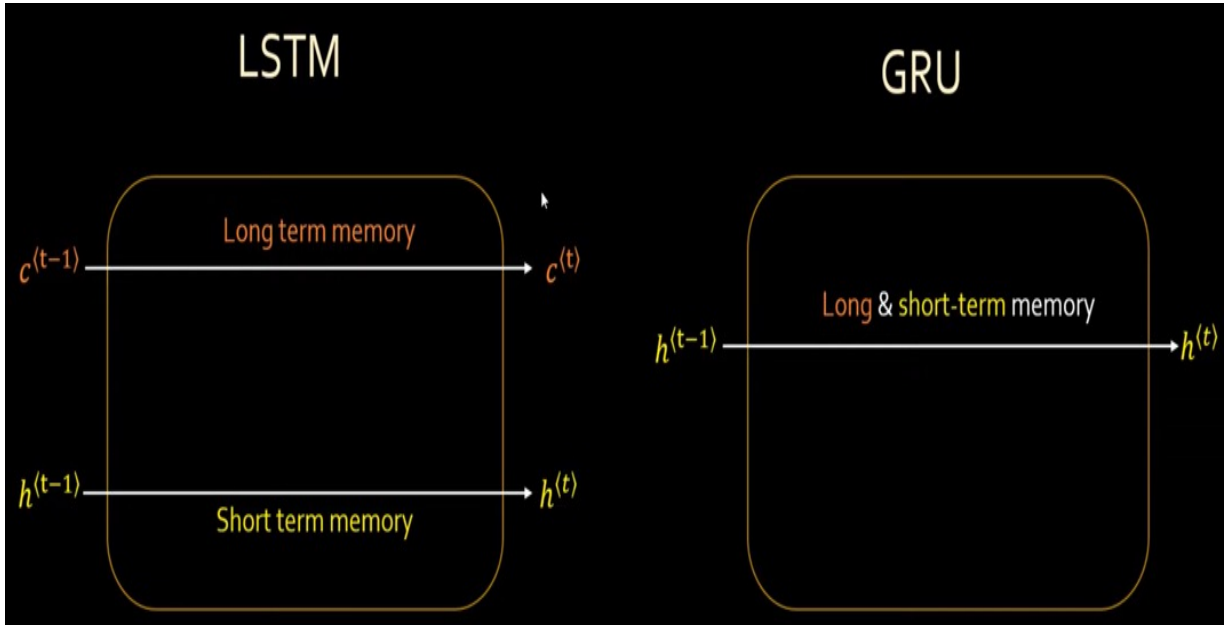


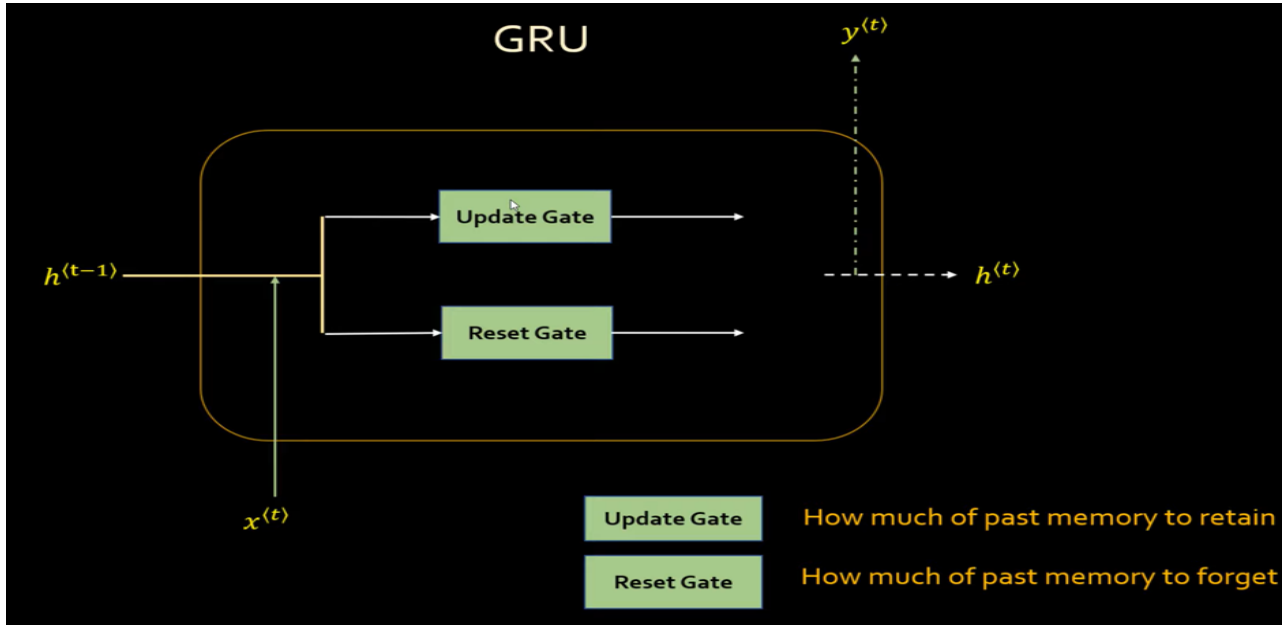


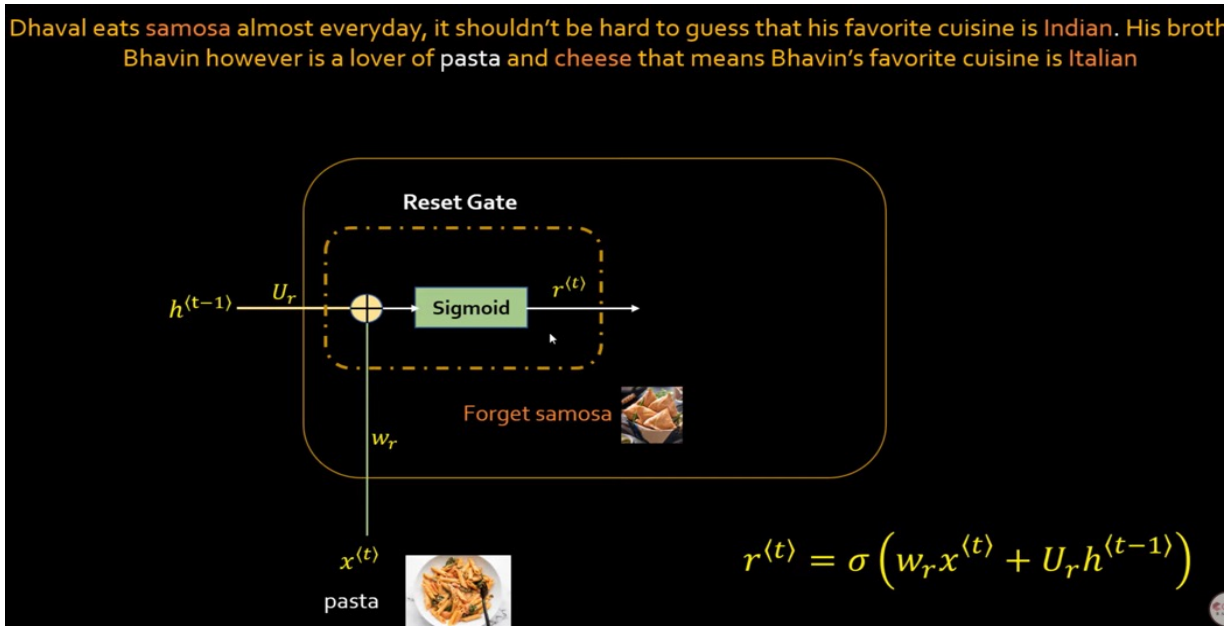


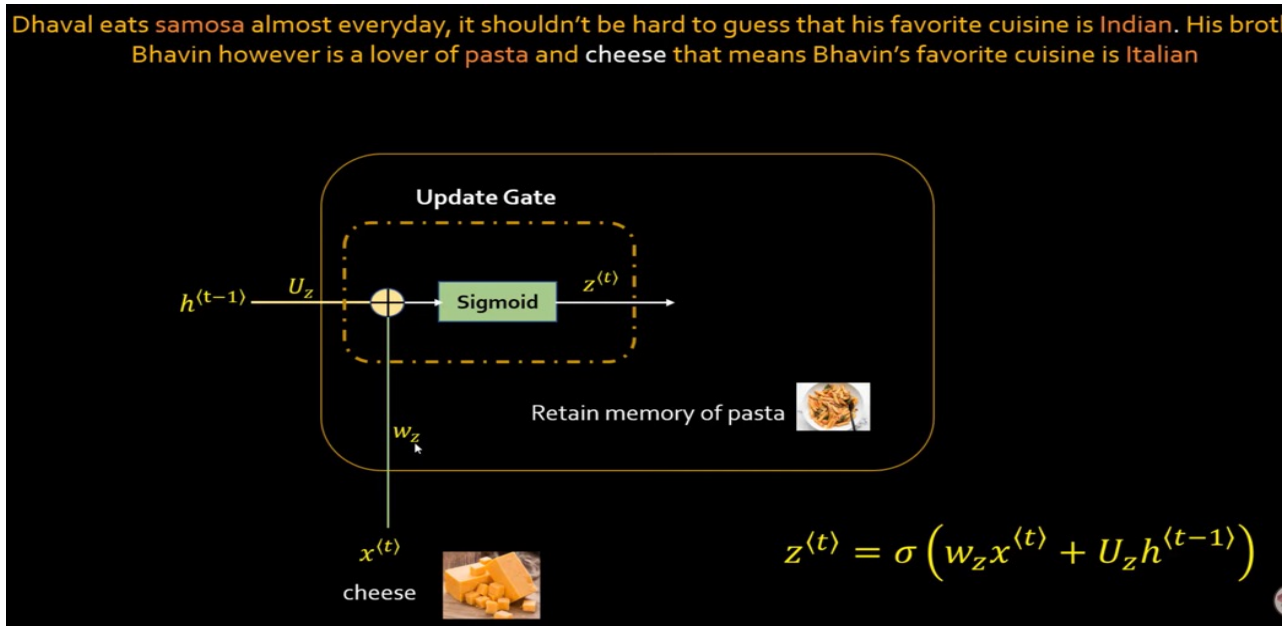
Andrew Ng

RNN Example

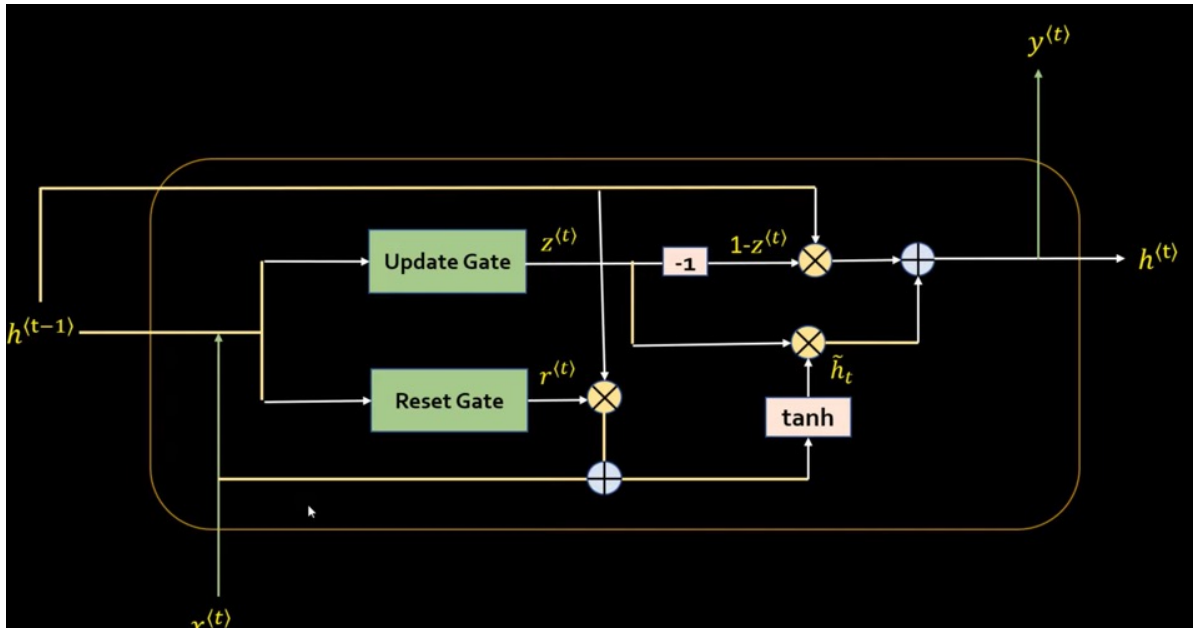

LSTM
GRU
Long term memory
$c^{\langle t-1 \rangle}$
$c^{\langle t \rangle}$
$h^{\langle t-1 \rangle}$
$h^{\langle t \rangle}$
Short term memory
Long & short-term memory
$h^{\langle t-1 \rangle}$
$h^{\langle t \rangle}$


GRU
$y^{\langle t \rangle}$
Update Gate
Reset Gate
$h^{\langle t-1 \rangle}$
$h^{\langle t \rangle}$
$x^{\langle t \rangle}$
Update Gate
How much of past memory to retain
Reset Gate
How much of past memory to forget


Dhaval eats samosa almost everyday, it shouldn't be hard to guess that his favorite cuisine is Indian. His broth
Bhavin however is a lover of pasta and cheese that means Bhavin's favorite cuisine is Italian
Reset Gate
$h^{\langle t-1 \rangle}$
U_r
Sigmoid
$r^{\langle t \rangle}$
Forget samosa
w_r
$x^{\langle t \rangle}$
pasta
$r^{\langle t \rangle} = \sigma\left(w_r x^{\langle t \rangle} + U_r h^{\langle t-1 \rangle}\right)$


Dhaval eats samosa almost everyday, it shouldn't be hard to guess that his favorite cuisine is Indian. His brot
Bhavin however is a lover of pasta and cheese that means Bhavin's favorite cuisine is Italian
Update Gate
$h^{\langle t-1 \rangle}$
U_z
Sigmoid
$z^{\langle t \rangle}$
Retain memory of pasta
w_z
$x^{\langle t \rangle}$
cheese
$z^{\langle t \rangle} = \sigma\left(w_z x^{\langle t \rangle} + U_z h^{\langle t-1 \rangle}\right)$

RNN Example Contd…



LSTM	GRU
3 Gates: Input, output, forget	2 Gates: reset, update
More accurate on longer sequence, less efficient	More efficient computation wise. Getting more popular
Invented: 1995 - 1997	Invented: 2014