Recurrent Neural Network
&
LSTM

2023-2 KUBIG 방학세션
DL

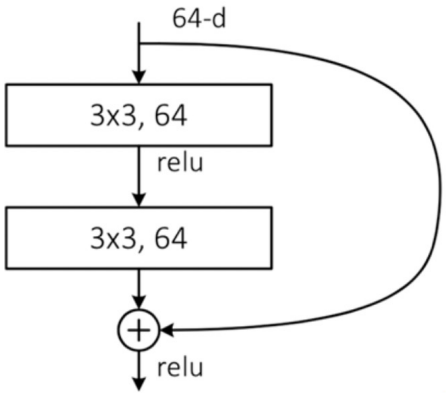
Category

1. Recurrent Neural Network
2. LSTM
3. Image Captioning
4. Q&A

0. Paper Review

0. Paper Review

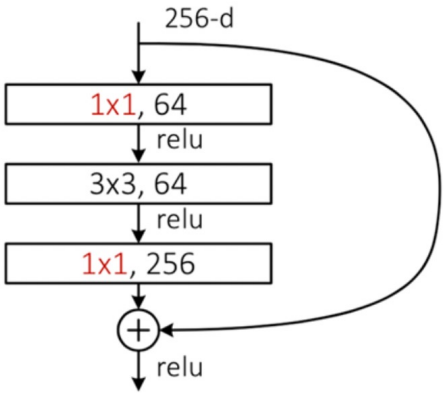
1) ResNet



BasicBlock

(for ResNet-18/34) all-3x3

similar complexity

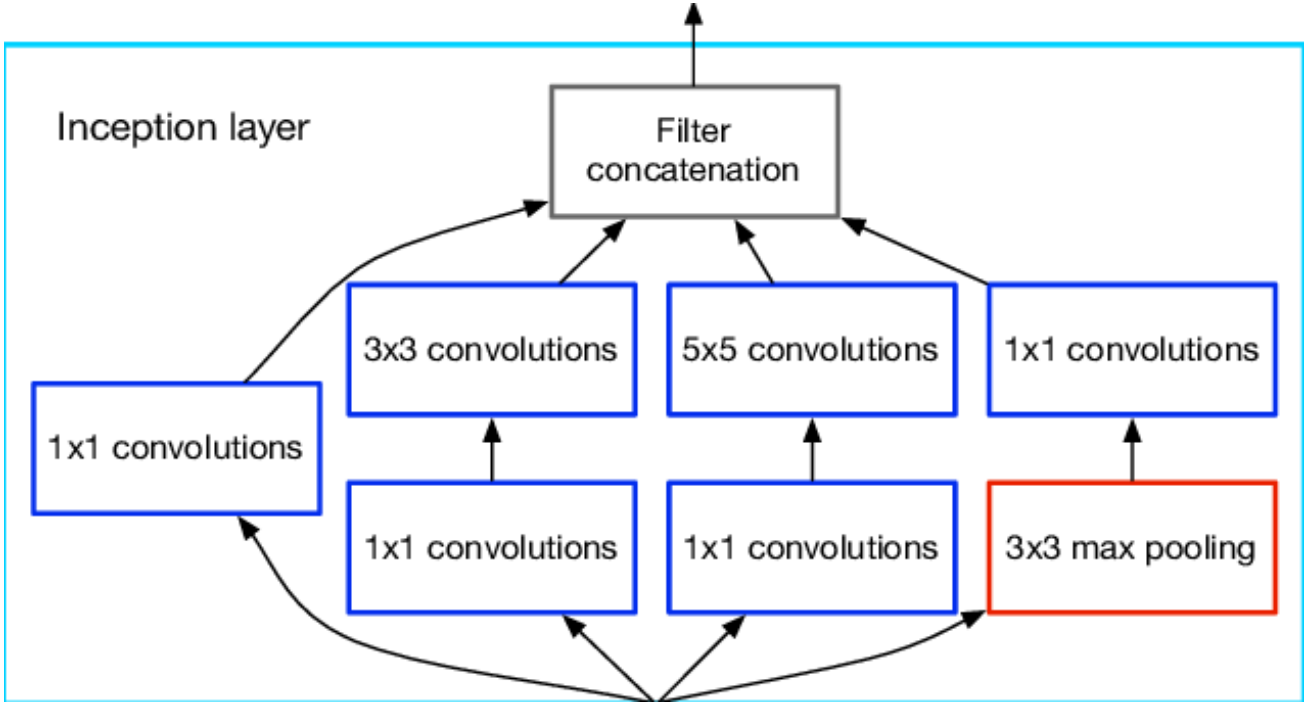


bottleneck

(for ResNet-50/101/152)

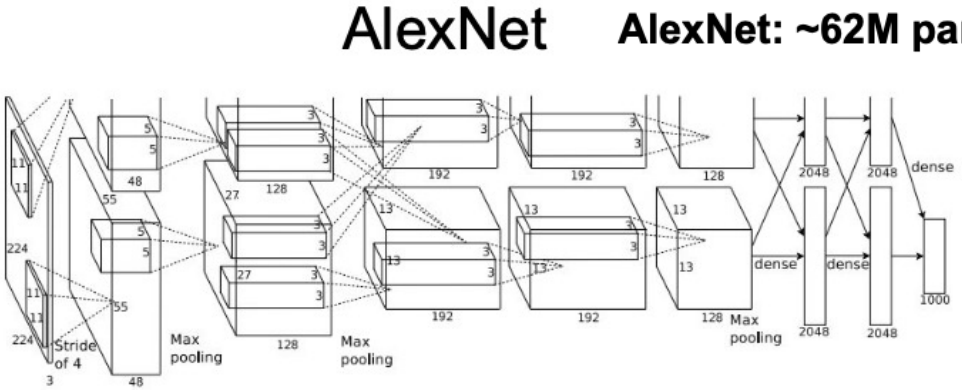
0. Paper Review

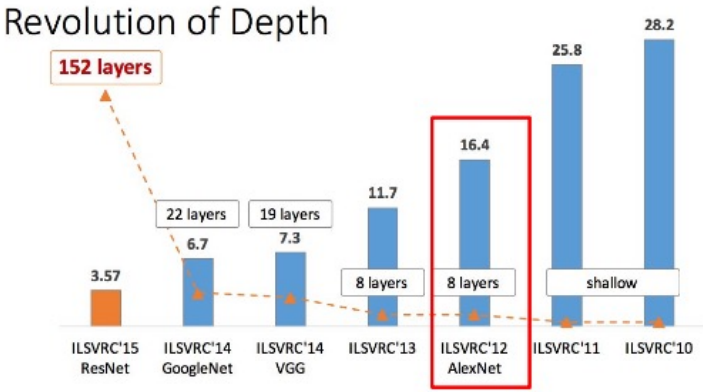
2)GoogLeNet



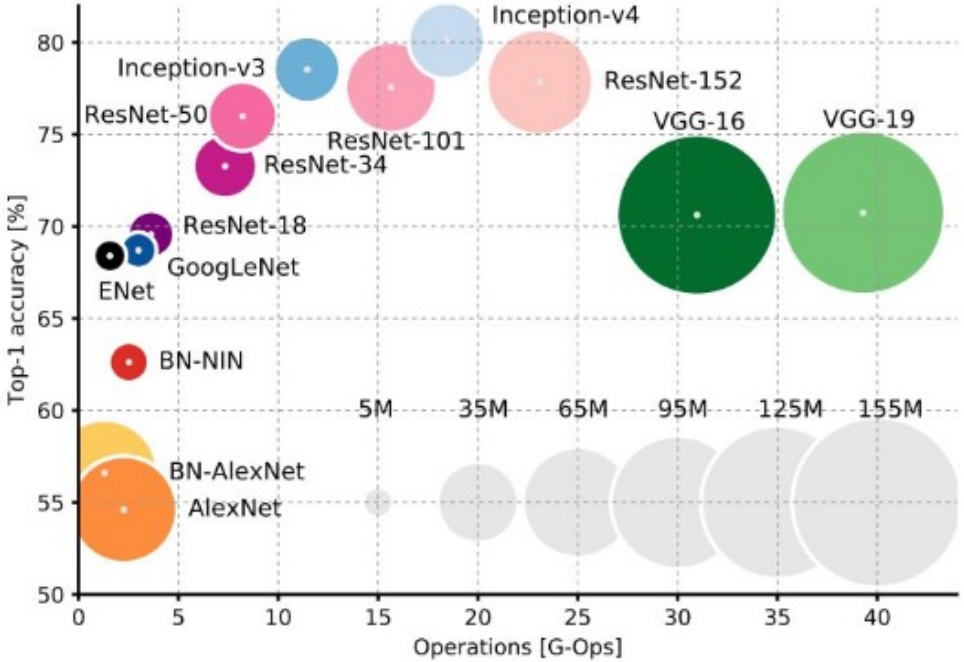
0. Preview

AlexNet & its parameters





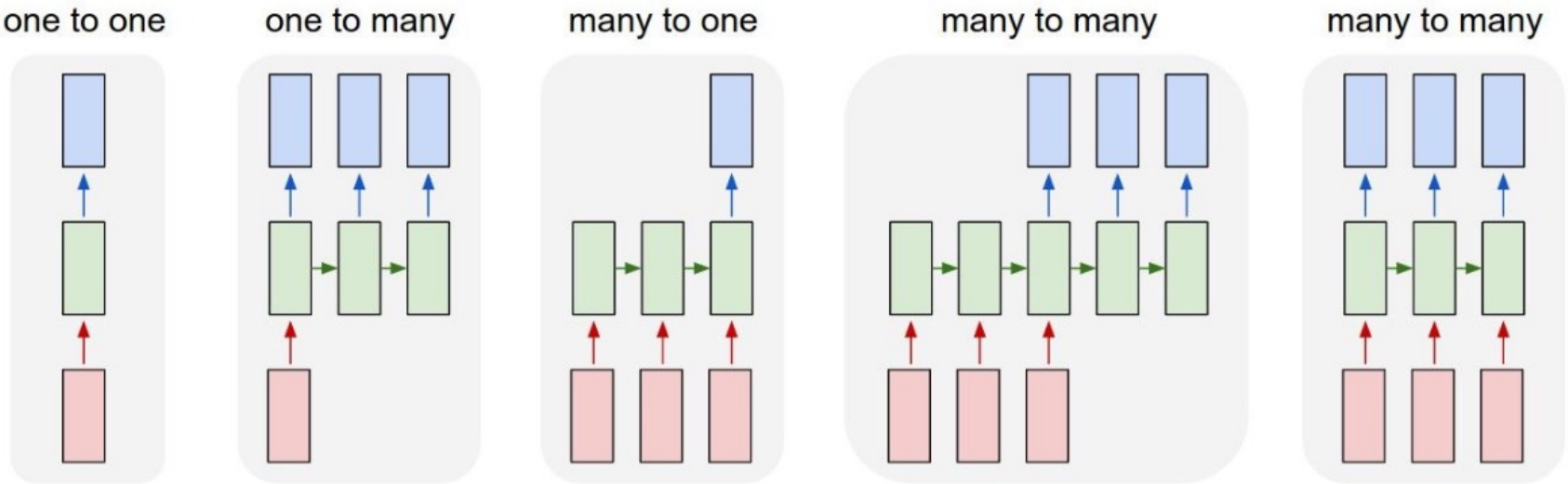
0. Preview


Inception-v4
Inception-v3
ResNet-152
ResNet-50
VGG-16
VGG-19
ResNet-101
ResNet-34
ResNet-18
GoogLeNet
ENet
BN-NIN
5M
35M
65M
95M
125M
155M
BN-AlexNet
AlexNet
Top-1 accuracy [%]
Operations [G-Ops]
80
75
70
65
60
55
50
0
5
10
15
20
25
30
35
40

1. Recurrent Neural Network

1. Recurrent Neural Network

How to handle temporal data with Deep Learning?



1. Recurrent Neural Network

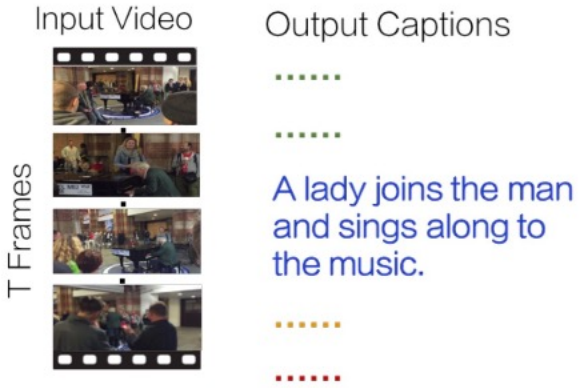
Why existing convnets are insufficient?

Variable sequence length inputs and outputs!

Example task: video captioning

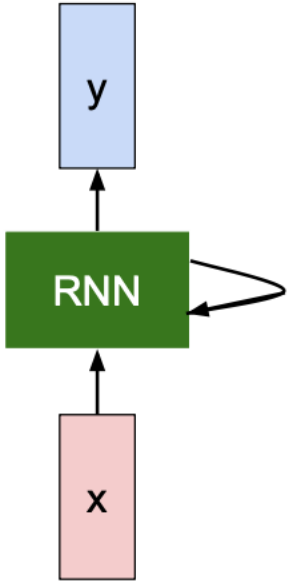
Input video can have variable number of frames

Output captions can be variable length.



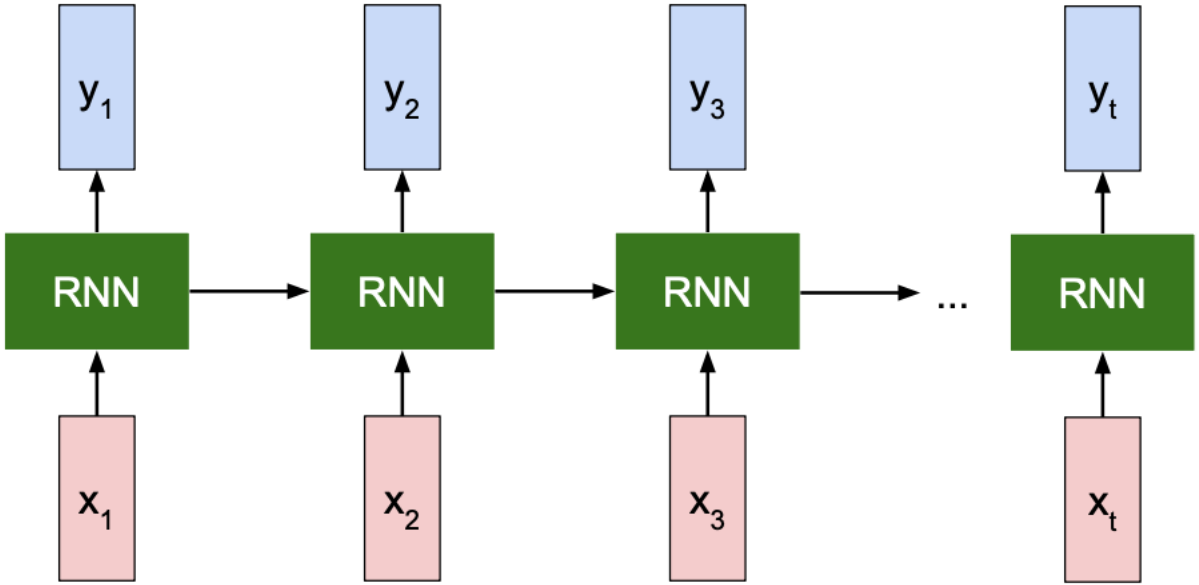
1. Recurrent Neural Network

Recurrent Neural Network



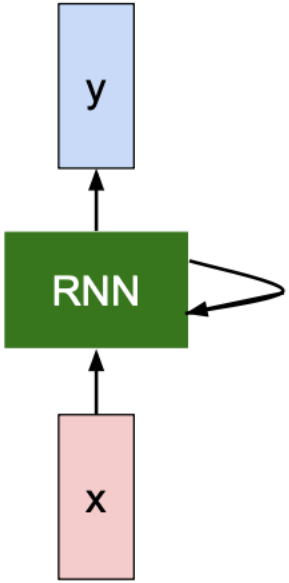
1. Recurrent Neural Network

Unfolded Recurrent Neural Network



1. Recurrent Neural Network

RNN hidden state update



$$h_t = f_W(h_{t-1}, x_t)$$

new state

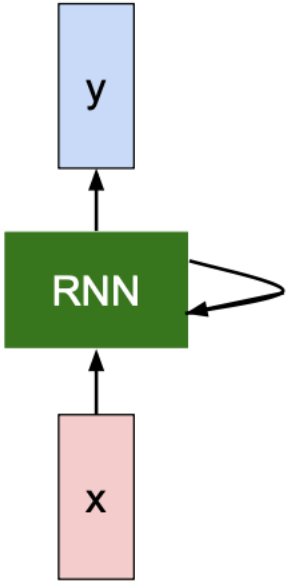
some function with parameters W

old state

input vector at some time step

1. Recurrent Neural Network

RNN output update



$$y_t = f_{W_{hy}}(h_t)$$

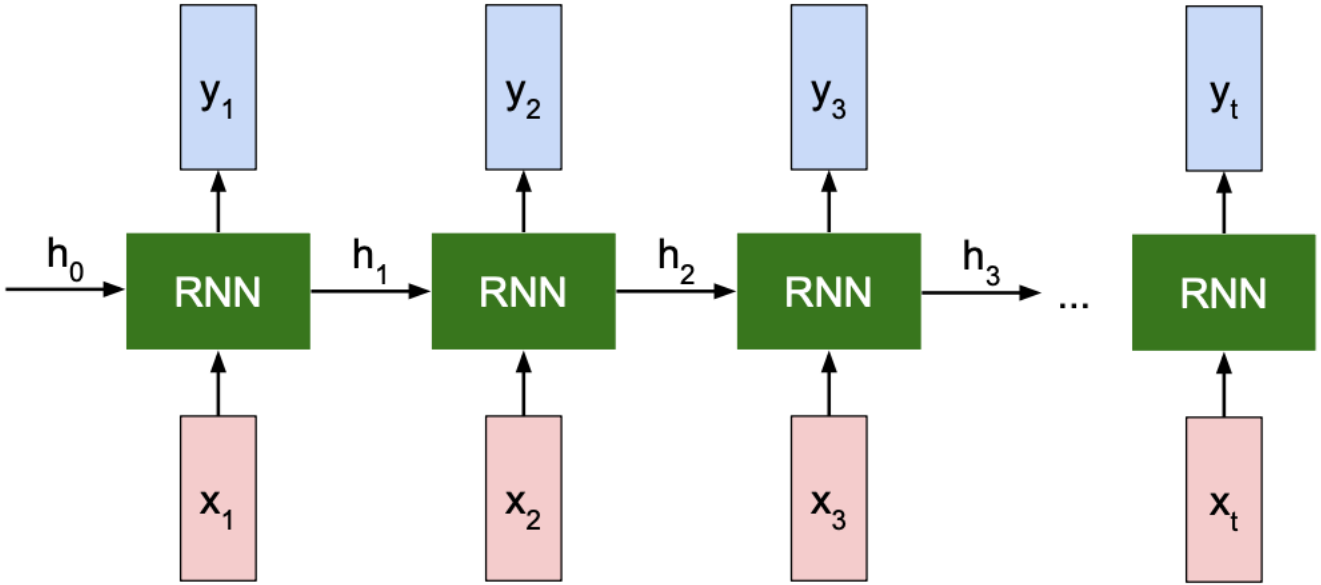
output

new state

another function
with parameters W_o

1. Recurrent Neural Network

Annotated Recurrent Neural Network



1. Recurrent Neural Network

Vanilla Recurrent Neural Network

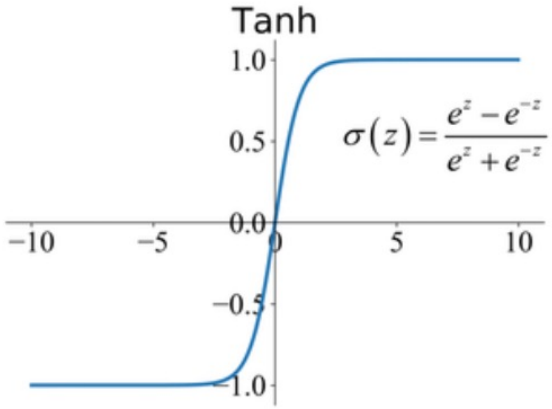
One step of RNN consists of functions with hidden state & output
(You have seen these states already)

$$h_t = f_W(h_{t-1}, x_t)$$

$$\downarrow$$

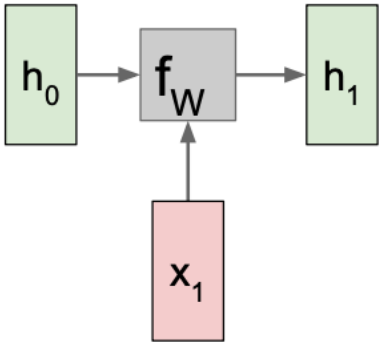
$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$

$$y_t = W_{hy}h_t$$



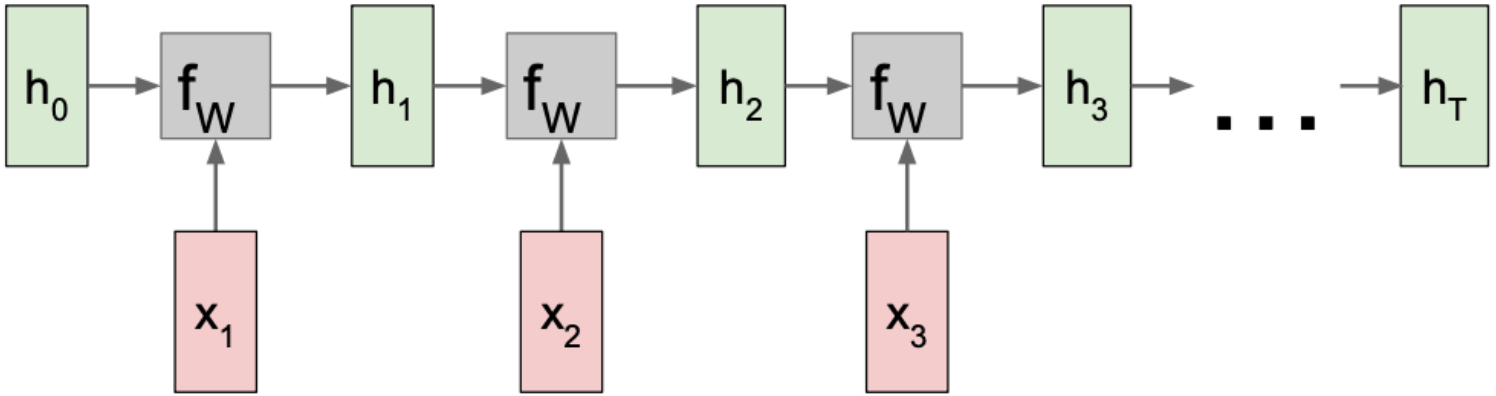
1. Recurrent Neural Network

RNN: Computational Graph



1. Recurrent Neural Network

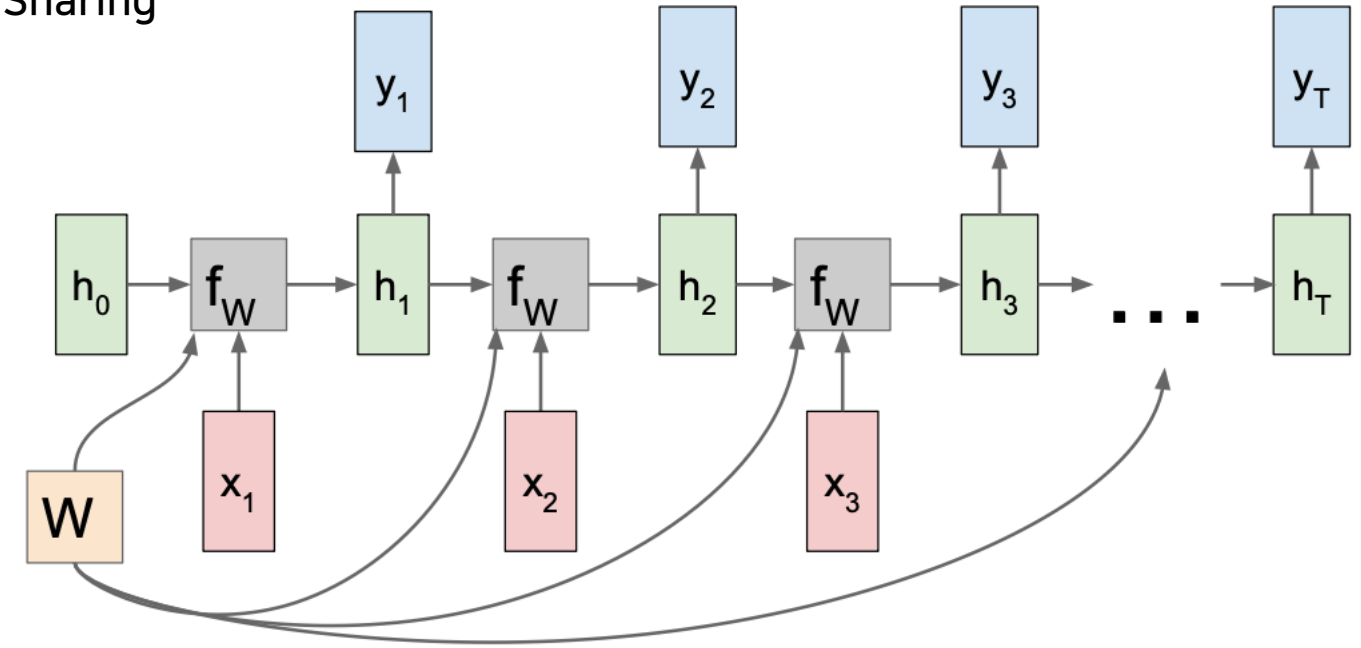
RNN: Computational Graph



1. Recurrent Neural Network

RNN: Computational Graph

Weight Sharing

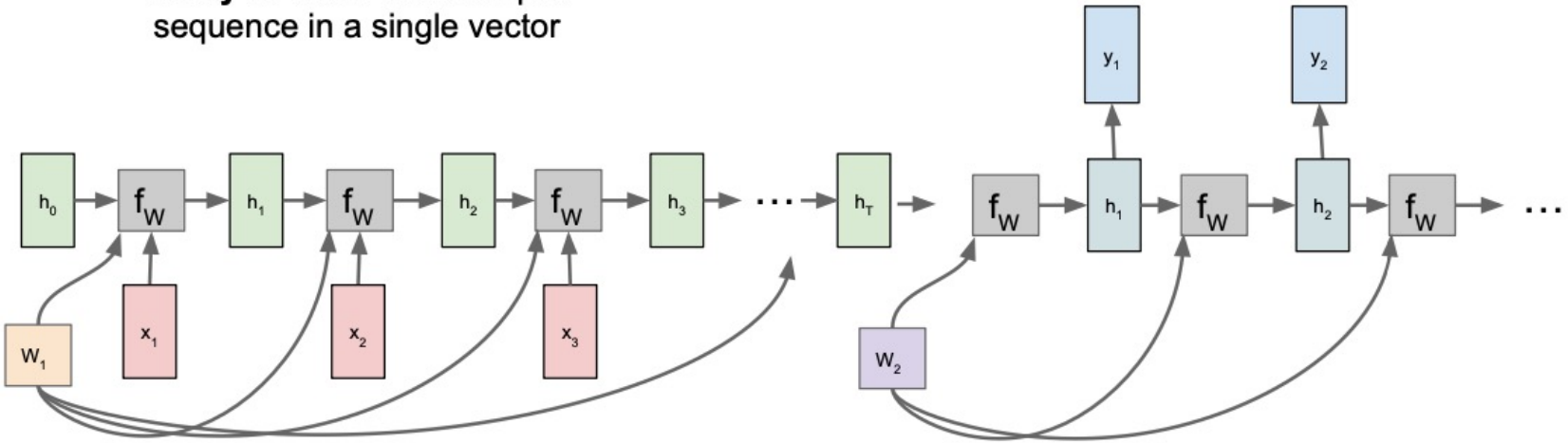


1. Recurrent Neural Network

Sequence to Sequence: Many-to-one → one-to-many

One to many: Produce output sequence from single input vector

Many to one: Encode input sequence in a single vector



1. Recurrent Neural Network

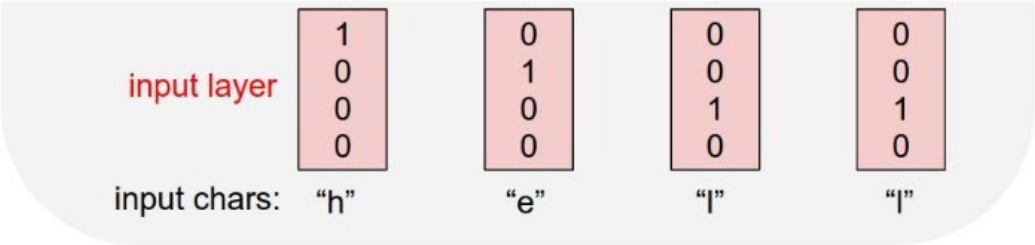
Let’s see a deeper example: How to handle NLP?

Task: Prediction of next character

Input: ‘helo’

Then, What is the output?

*Expected output: ‘ello’

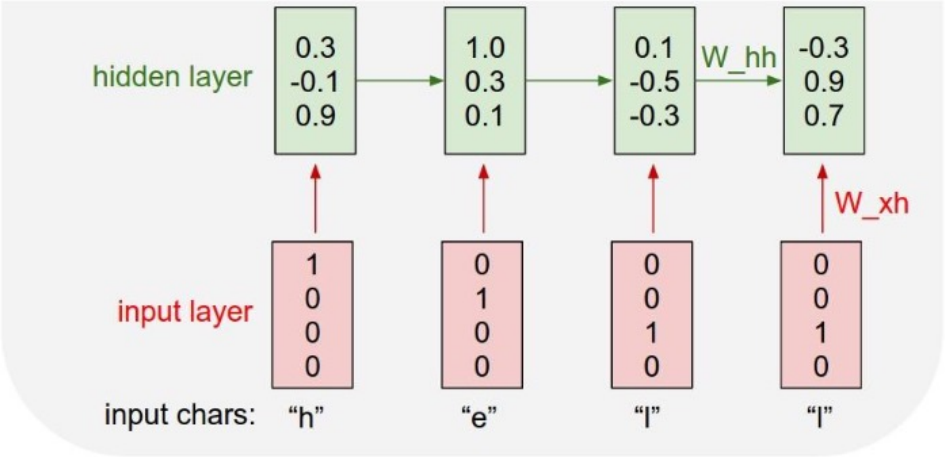


1. Recurrent Neural Network

Let's see a deeper example: How to handle NLP?

Task: Prediction of next character

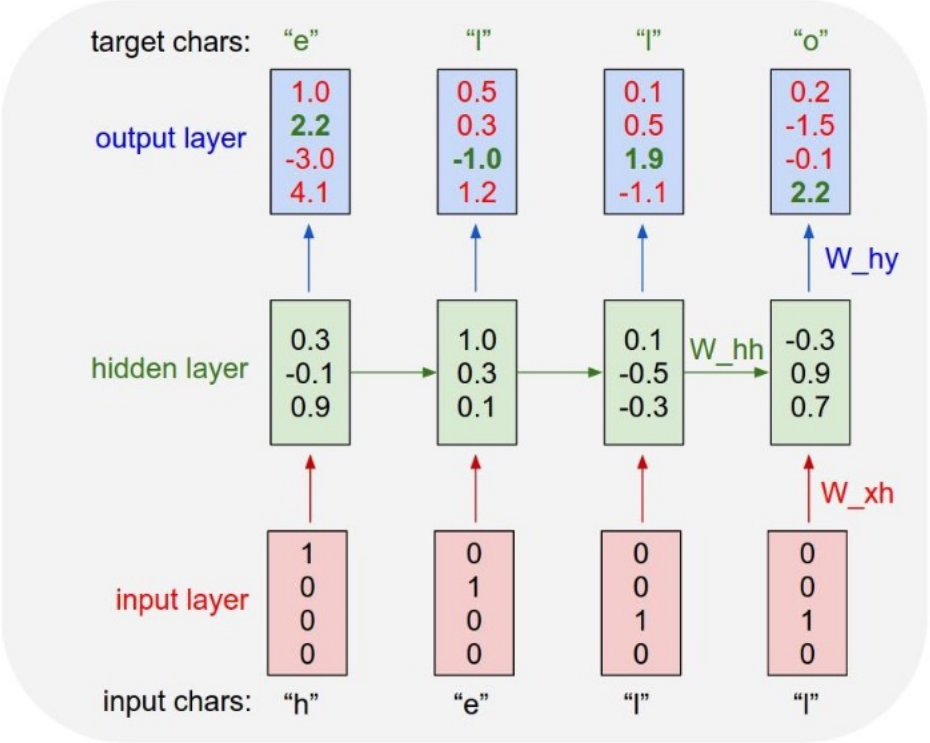
$$h_t = \tanh(W_{hh}h_{t-1} + W_{xh}x_t)$$



1. Recurrent Neural Network

Let's see a deeper exampl

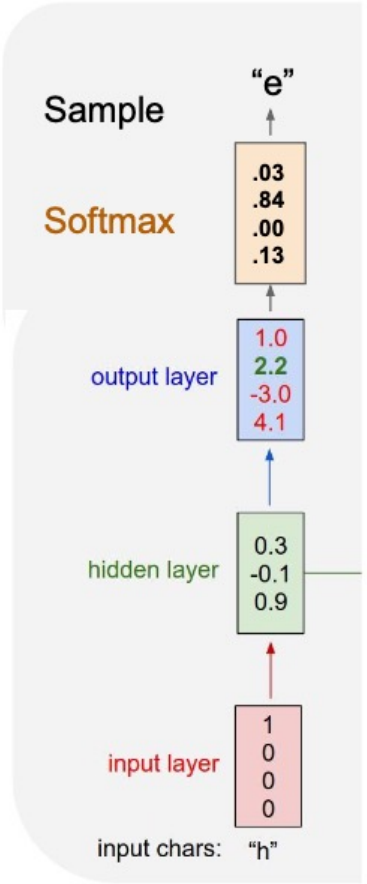
Task: Prediction of next character



1. Recurrent Neural Network

Let's see a deeper example: H ?

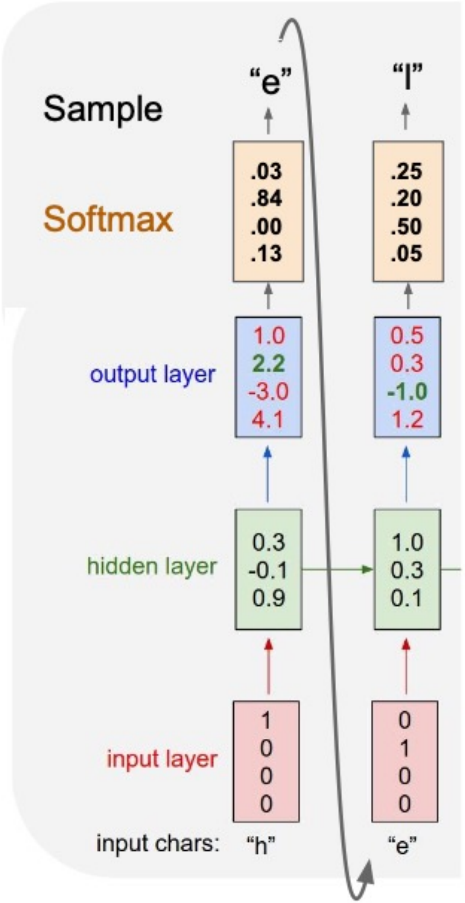
How about Test-time?



1. Recurrent Neural Network

Let's see a deeper example: Ho

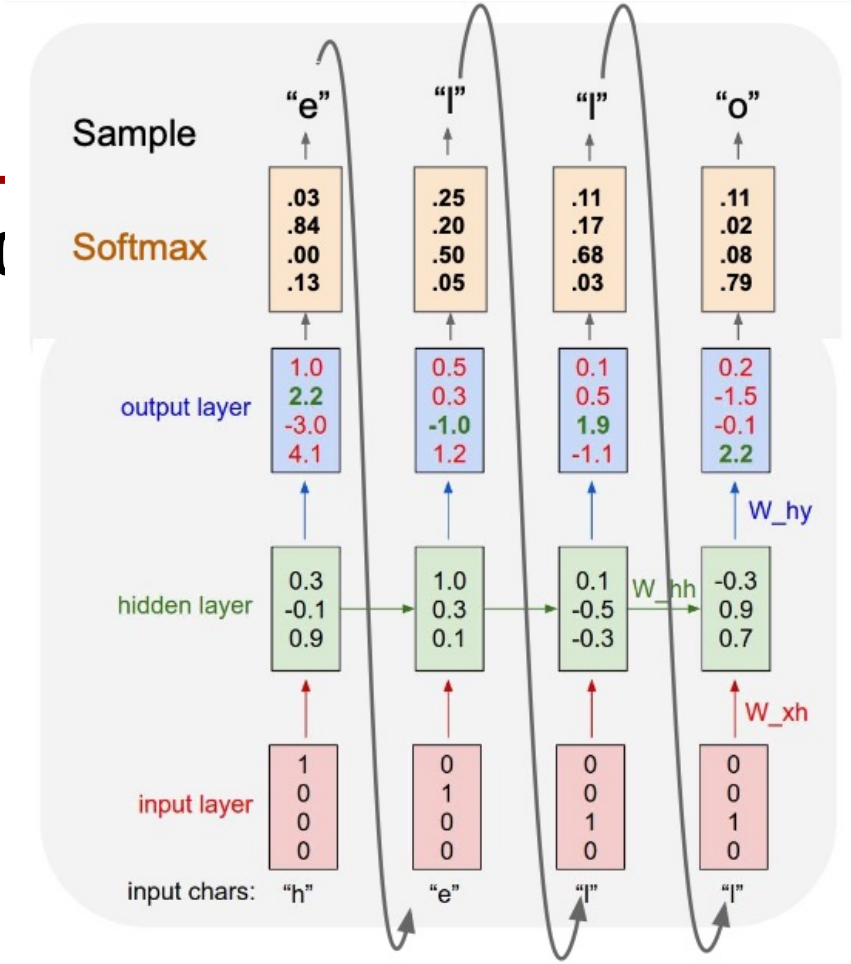
How about Test-time?



1. Recurrent Neural Network

Let's see a deeper example: H

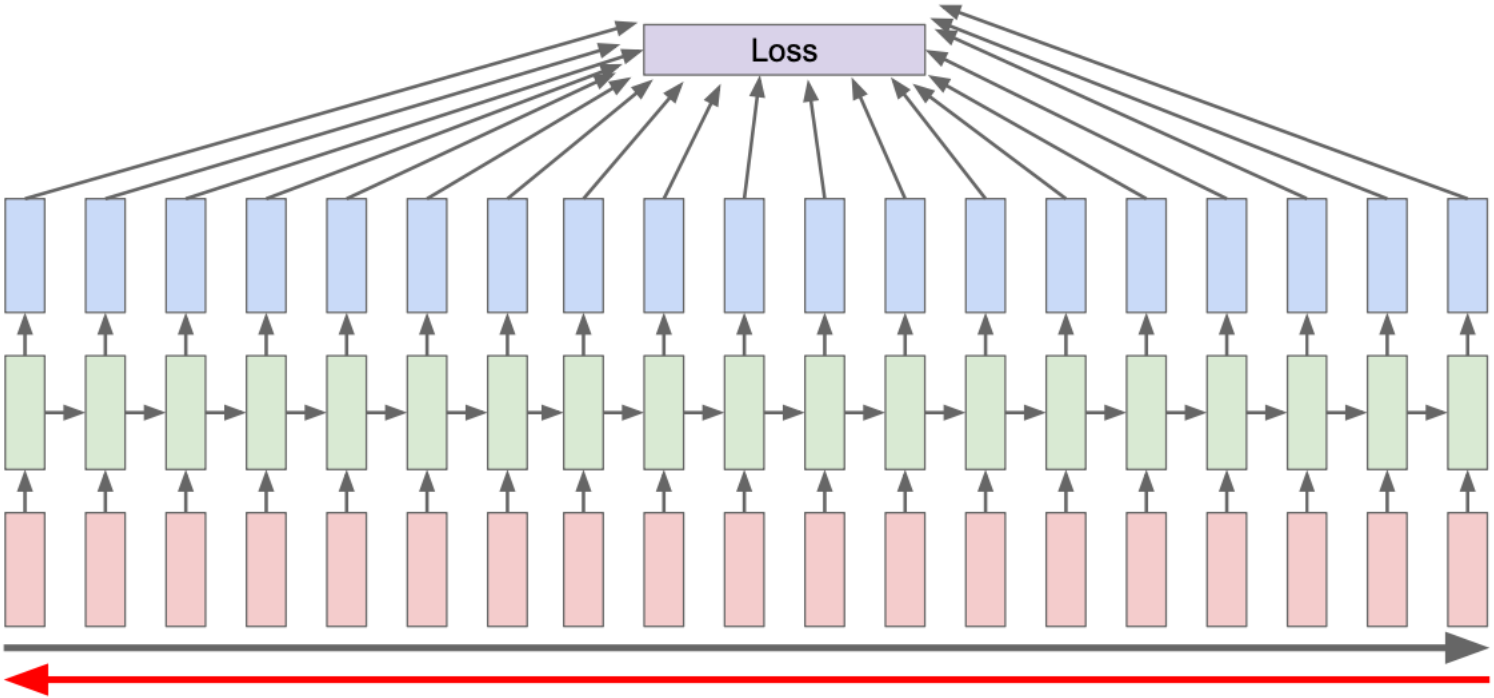
How about Test-time?



1. Recurrent Neural Network

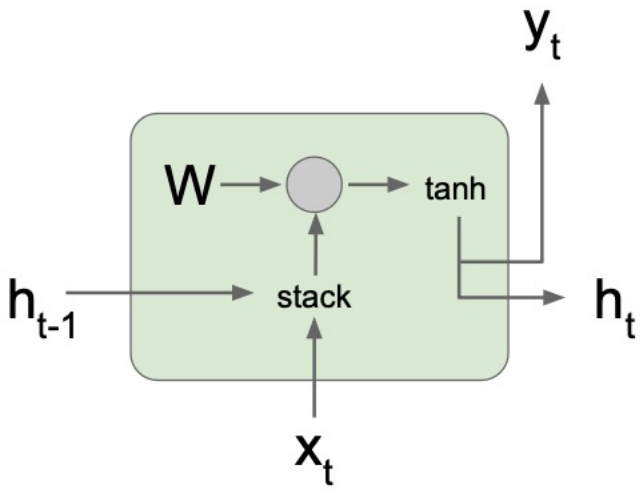
Then, how about Backpropagation?

Backward through entire sequence to compute gradient



1. Recurrent Neural Network

Vanilla RNN Gradient Flow

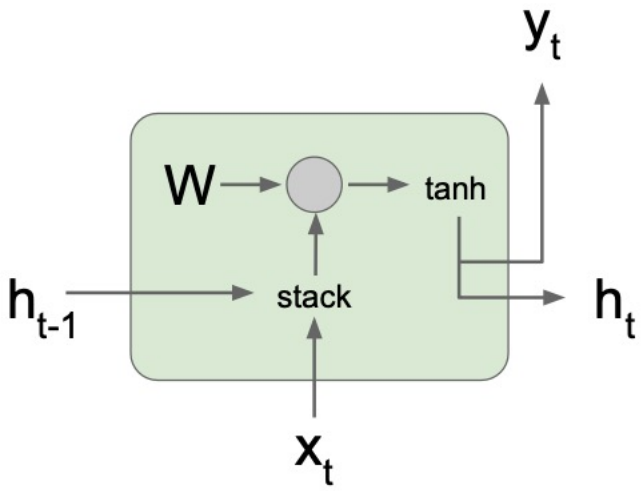


$$
\begin{aligned}
h_t &= \tanh(W_{hh}h_{t-1} + W_{xh}x_t) \\
&= \tanh\left(\begin{pmatrix} W_{hh} & W_{hx} \end{pmatrix} \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right) \\
&= \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix} \right)
\end{aligned}
$$

1. Recurrent Neural Network

Vanilla RNN Gradient Flow

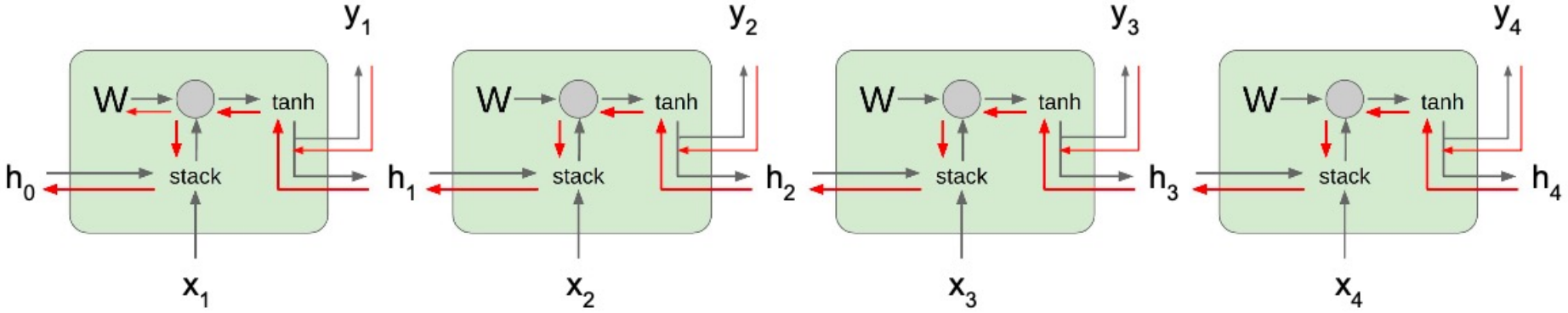
$$\frac{\partial h_t}{\partial h_{t-1}} = tanh'(W_{hh}h_{t-1} + W_{xh}x_t)W_{hh}$$



$$\begin{aligned} h_t &= \tanh(W_{hh}h_{t-1} + W_{xh}x_t) \\ &= \tanh\left((W_{hh} \quad W_{hx})\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right) \\ &= \tanh\left(W\begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right) \end{aligned}$$

1. Recurrent Neural Network

Vanilla RNN Gradient Flow For Multi time step:

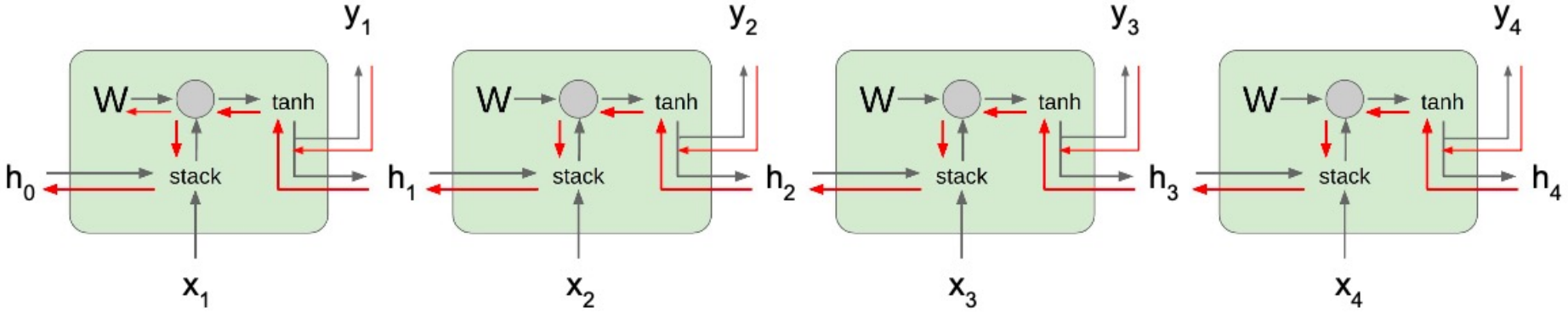


$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} \frac{\partial h_t}{\partial h_{t-1}} \cdots \frac{\partial h_1}{\partial W}$$

1. Recurrent Neural Network

Vanilla RNN Gradient Flow For Multi time step:

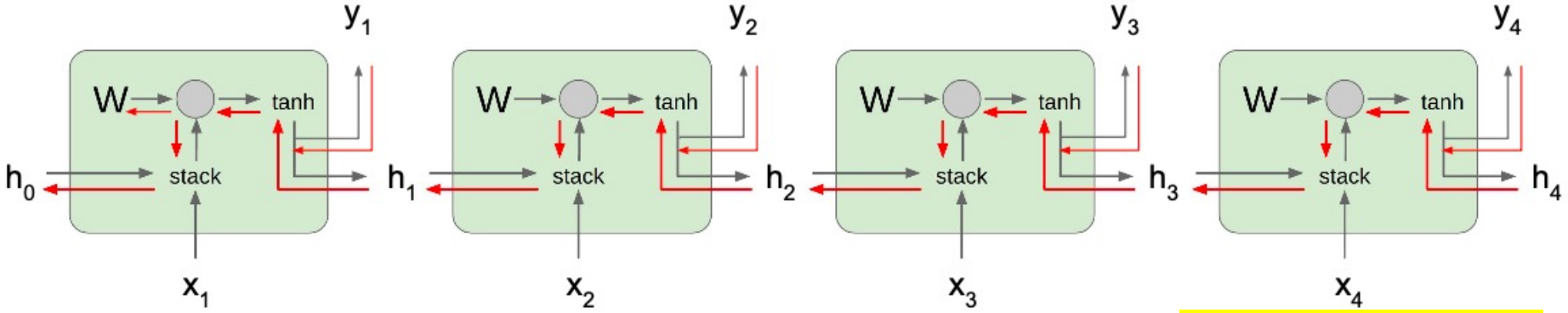


$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} \frac{\partial h_t}{\partial h_{t-1}} \cdots \frac{\partial h_1}{\partial W} = \frac{\partial L_T}{\partial h_T} \left(\prod_{t=2}^{T} \frac{\partial h_t}{\partial h_{t-1}} \right) \frac{\partial h_1}{\partial W}$$

1. Recurrent Neural Network

Vanilla RNN Gradient Flow For Multi time step:



$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

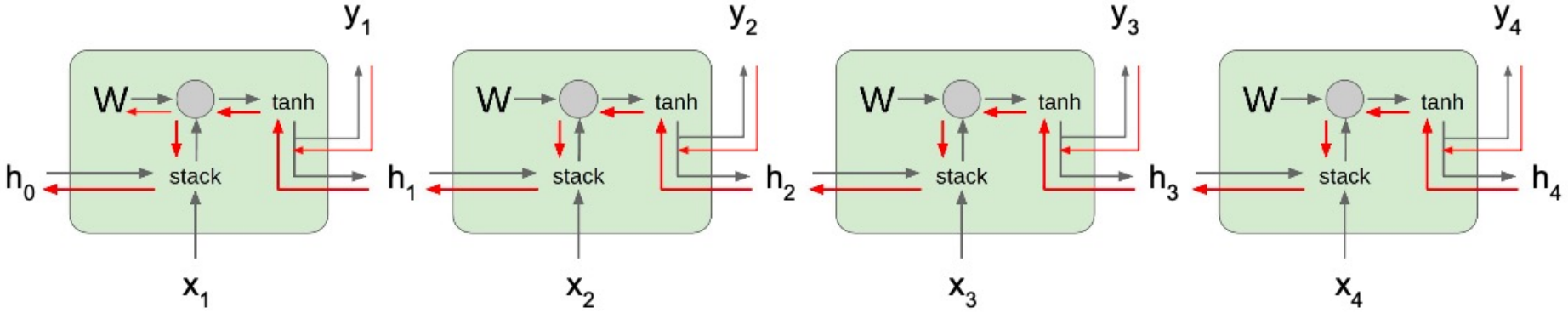
$$\boxed{\frac{\partial h_t}{\partial h_{t-1}}} = \boxed{tanh'}(W_{hh} h_{t-1} + W_{xh} x_t) W_{hh}$$

Almost < 1 ➔ Vanishing Gradient

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} \frac{\partial h_t}{\partial h_{t-1}} \cdots \frac{\partial h_1}{\partial W} = \frac{\partial L_T}{\partial h_T} \left(\prod_{t=2}^{T} \boxed{\frac{\partial h_t}{\partial h_{t-1}}} \right) \frac{\partial h_1}{\partial W}$$

1. Recurrent Neural Network

Vanilla RNN Gradient Flow For Multi-time step:



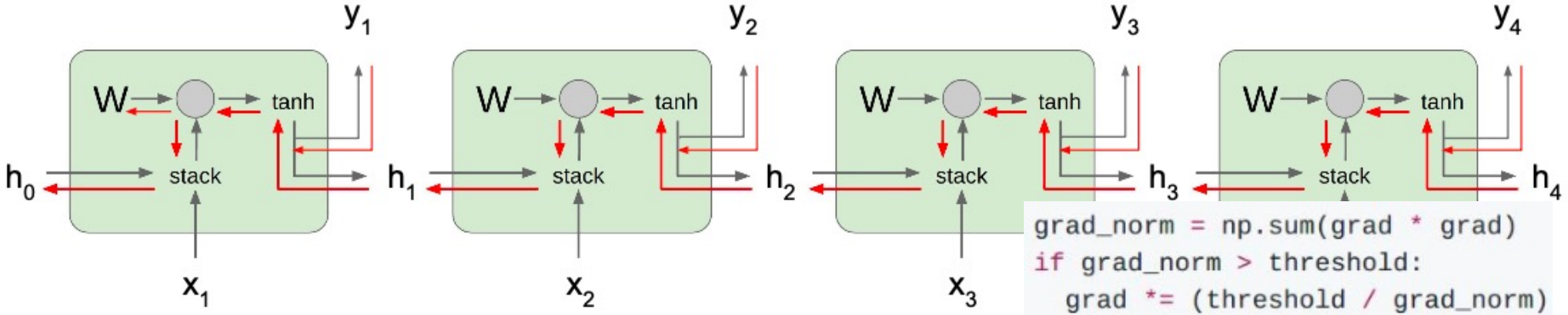
$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

If no non-linearity?➔ Exploding Gradient

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T}\frac{\partial h_t}{\partial h_{t-1}}\cdots\frac{\partial h_1}{\partial W} = \frac{\partial L_T}{\partial h_T}\left(\prod_{t=2}^{T}\frac{\partial h_t}{\partial h_{t-1}}\right)\frac{\partial h_1}{\partial W}$$

1. Recurrent Neural Network

Vanilla RNN Gradient Flow For Multi-time step:




```
grad_norm = np.sum(grad * grad)
if grad_norm > threshold:
  grad *= (threshold / grad_norm)
```

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

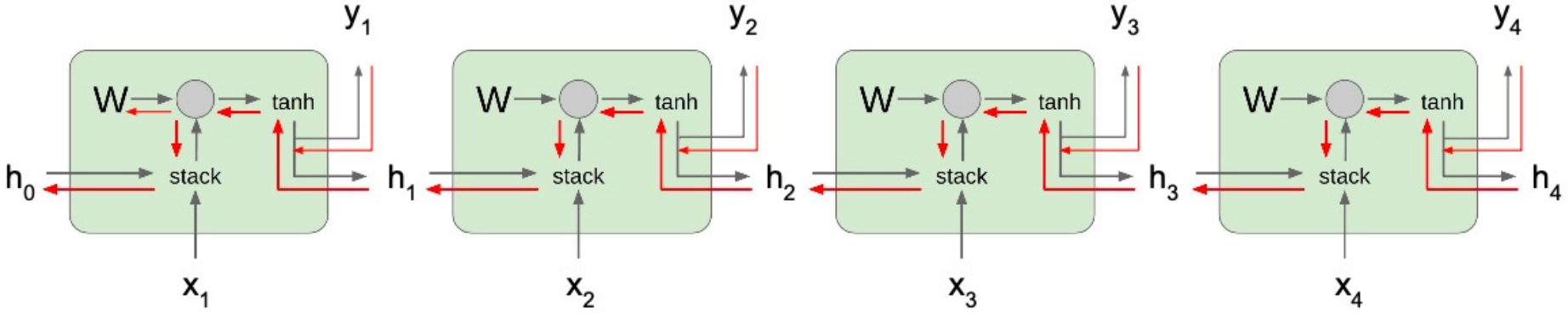
How to handle?➔ Exploding Gradient: Gradient Clipping

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} \frac{\partial h_t}{\partial h_{t-1}} \cdots \frac{\partial h_1}{\partial W}$$

Vanishing Gradient: New RNN Structure

1. Recurrent Neural Network

Vanilla RNN Gradient Flow For Multi-time step:



$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial W}$$

How to handle?➔ Exploding Gradient: Gradient Clipping

$$\frac{\partial L_T}{\partial W} = \frac{\partial L_T}{\partial h_T} \frac{\partial h_t}{\partial h_{t-1}} \cdots \frac{\partial h_1}{\partial W}$$

Vanishing Gradient: New RNN Structure
➔ LSTM !

2. LSTM

Comparison between Vanilla RNN & LSTM

Vanilla RNN

$$h_t = \tanh\left(W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}\right)$$

LSTM

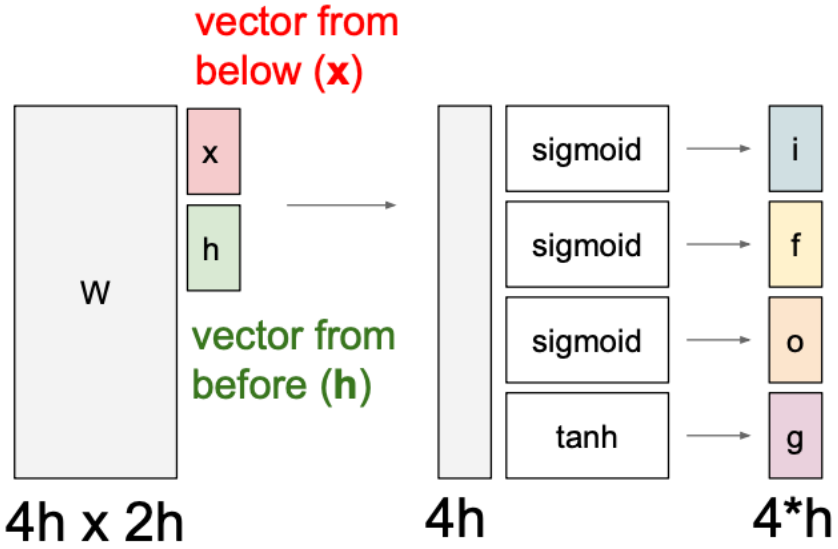
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

2. LSTM

LSTM Architecture



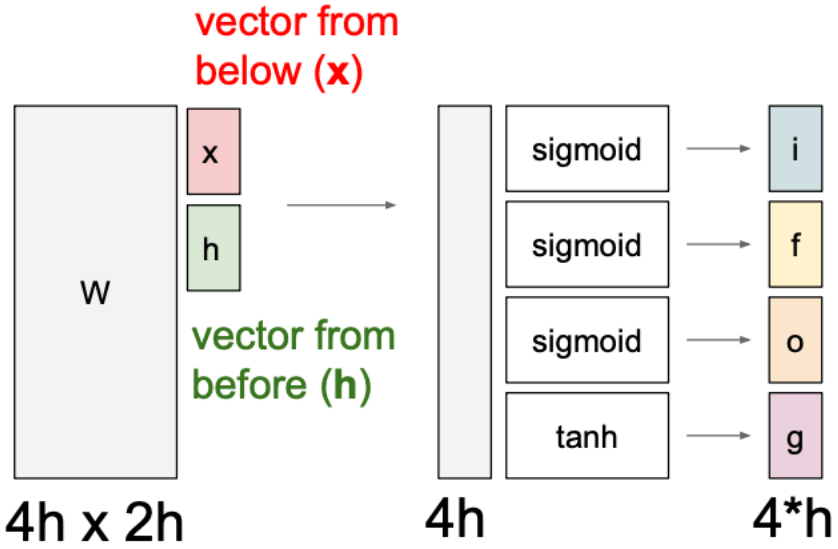
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

2. LSTM

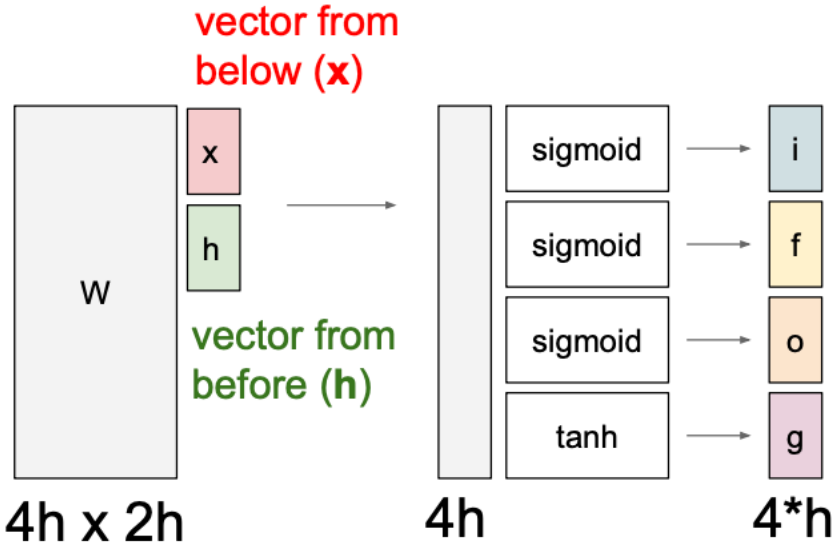
LSTM Architecture



i: input gate
f: forget gate
o: output gate
g: gate(?) gate
← personally called generation gate

2. LSTM

LSTM Architecture



i: input gate
f: forget gate
o: output gate
g: gate(?) gate
← personally called generation gate

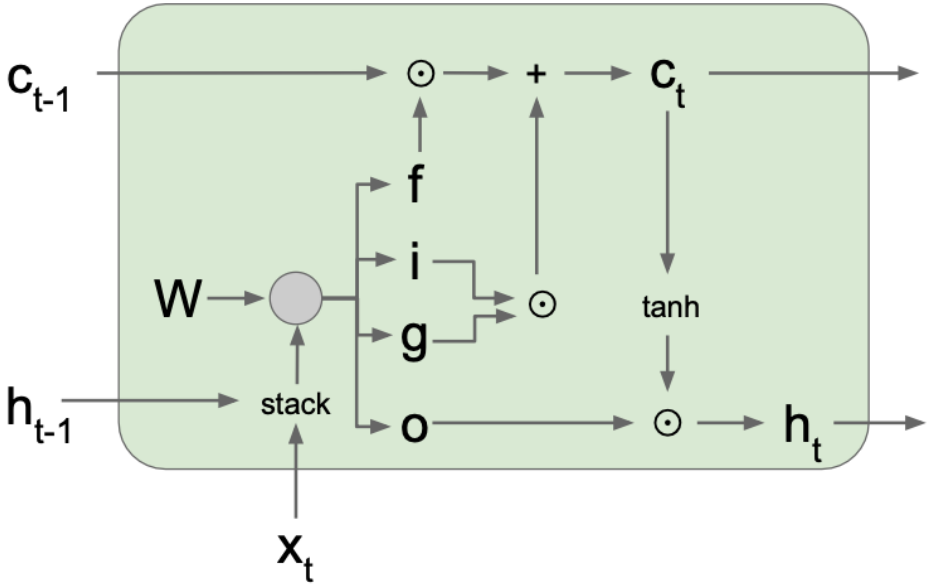
$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

2. LSTM

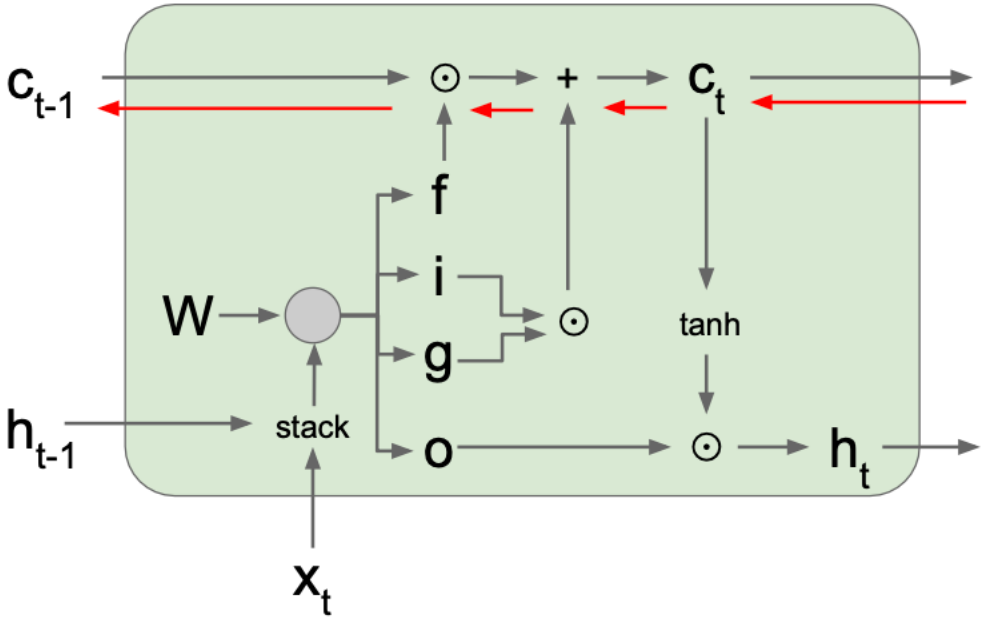
LSTM Block Architecture



How about Gradient?

2. LSTM

LSTM Block Architecture

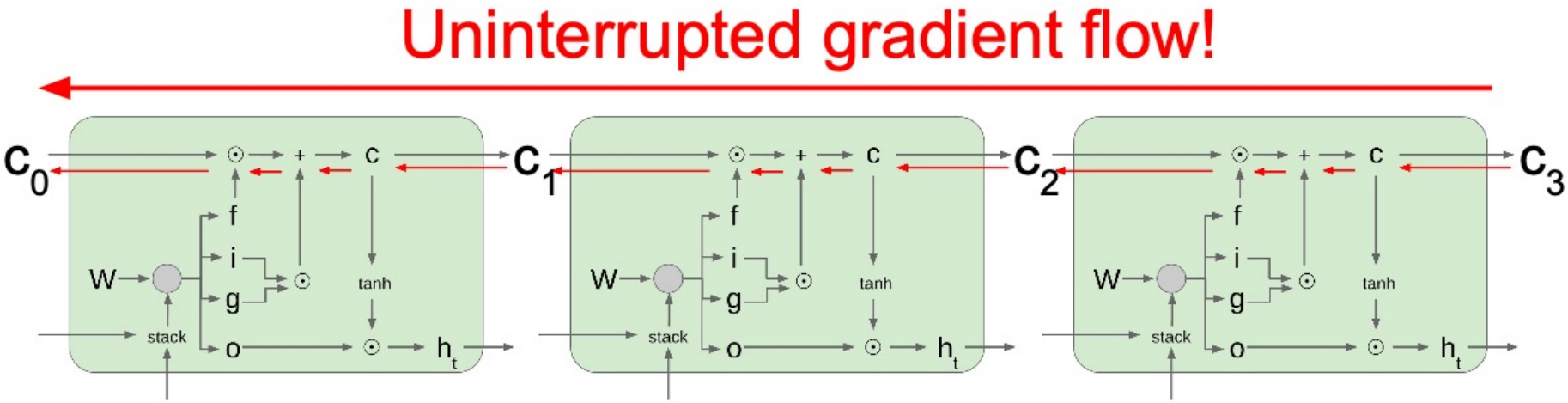


$$\begin{pmatrix} i \\ f \\ o \\ g \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} W \begin{pmatrix} h_{t-1} \\ x_t \end{pmatrix}$$

$$c_t = f \odot c_{t-1} + i \odot g$$

$$h_t = o \odot \tanh(c_t)$$

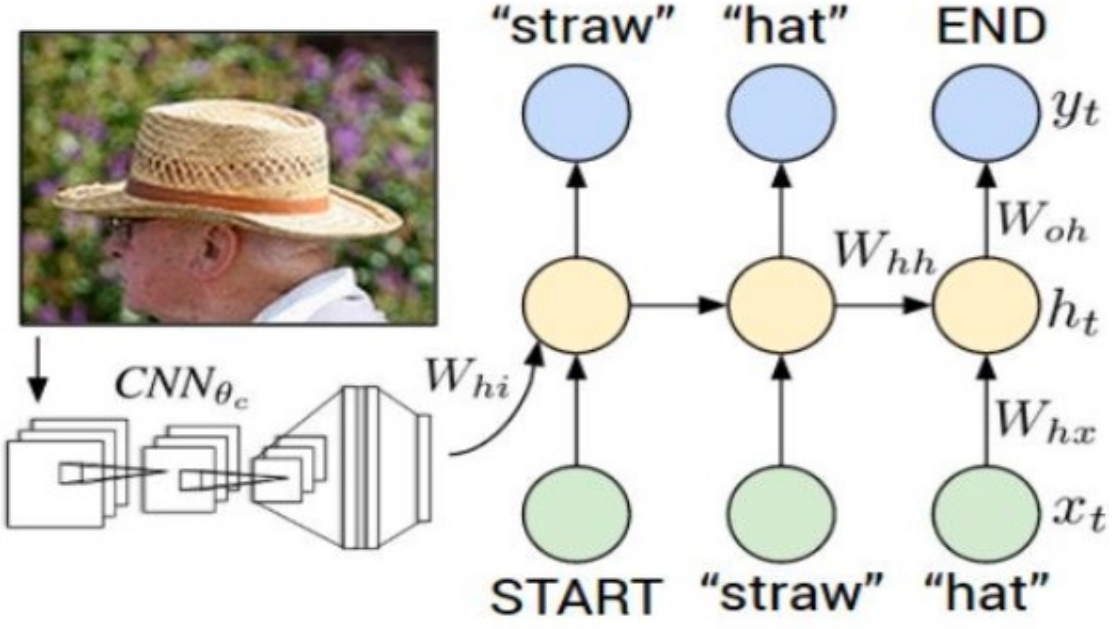
2. LSTM



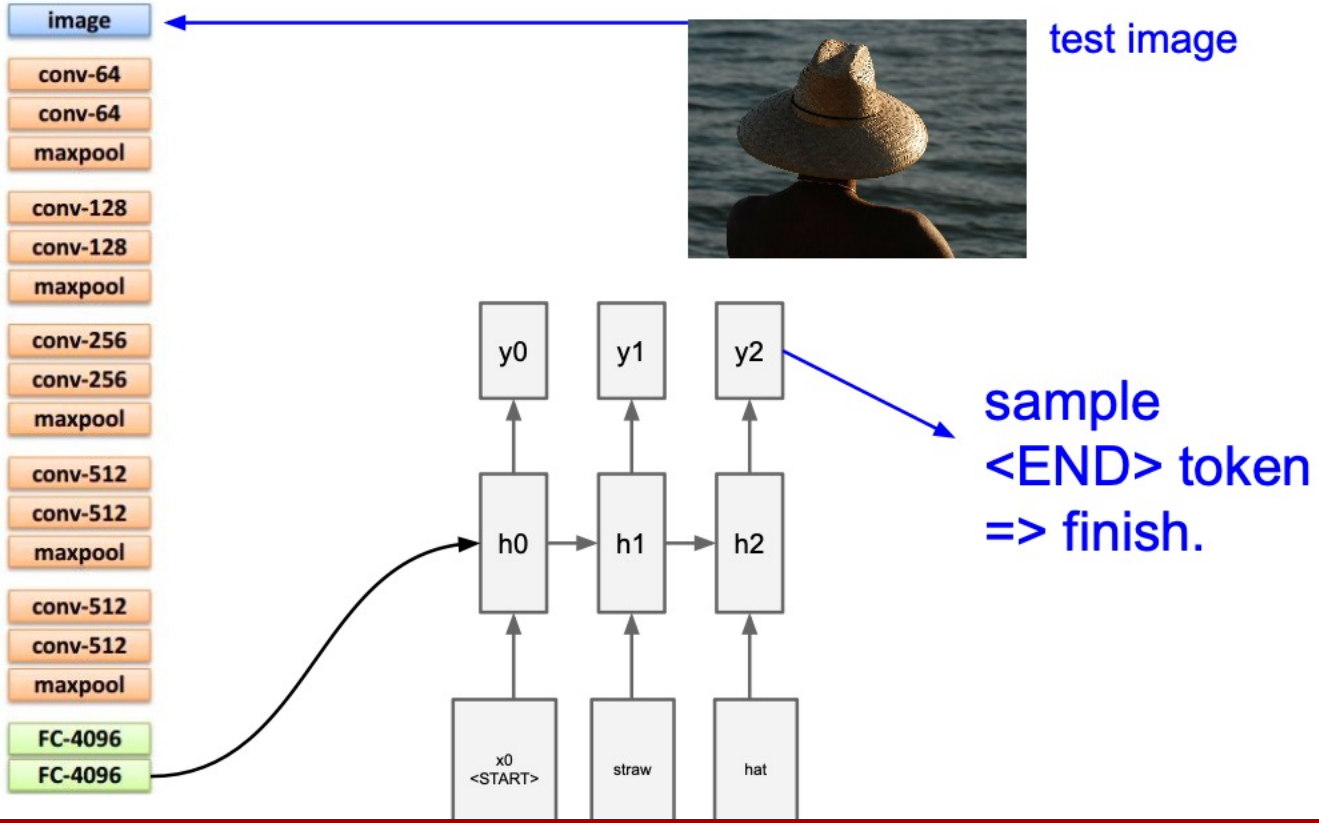
But this process doesn't guarantee the stability of Gradient of other gates : i,f,o and g

3. Image Captioning

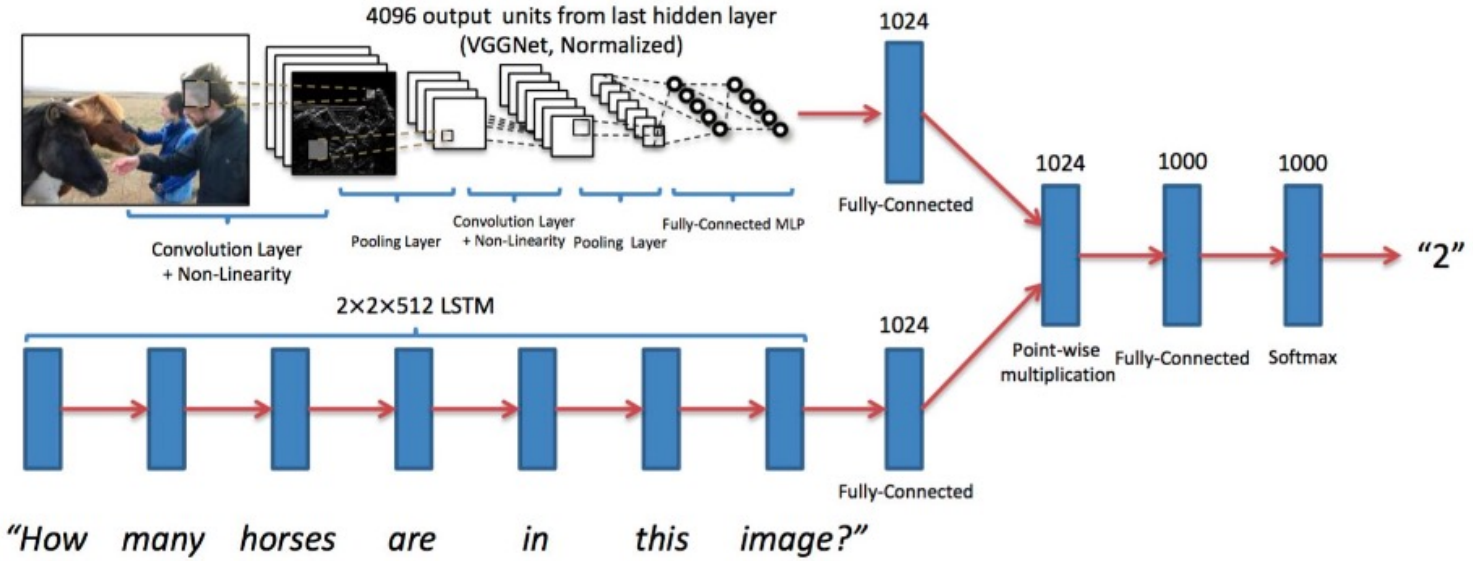
Like MultiModal



3. Image Captioning



3. Image Captioning



Q&A

Have a nice week🥰