Sequence Model and Recurrent Neural Network (RNN)

Changchong Sheng

Outline

This lecture introduces sequence model. The goal is to know how RNN and LSTM work, have an idea of their applications.

- Why sequence model?
- Why RNN?
- Basic RNN
- LSTM
- Applications
- Attention mechanism
- Self-attention based models

Why Sequence Model ?

➢ Some Example Tasks:

	Input		Output
Speech Recognition:		→	"Excuse me, I …"
Machine Translation:	"欢迎来到中国"	→	"Welcome to China"
Chat-bot:	How can I help you?	→	"I want to book a ticket"
Lip Reading:		→	"Best wish"

Why Sequence Model ?

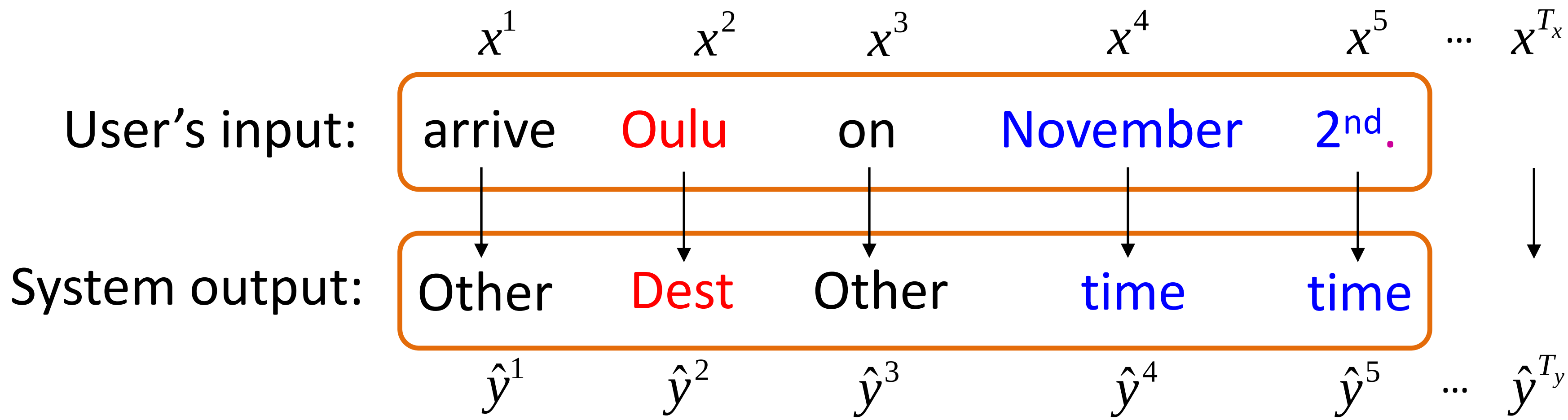
- Some Example Tasks:

	Input		Output
Music Generation:	∅	→	
Image Captioning:		→	"A Woman is running"
Sentiment Classification:	"This movie is terrible"	→	★ ☆ ☆ ☆ ☆
Activity Recognition:		→	HighJump

https://www.coursera.org/lecture/nlp-sequence-models

Why RNN?

- Slot Filling: Ticket booking system



Slot

Destination:	Oulu
time of arrival:	November 2nd
Other:	arrive, on

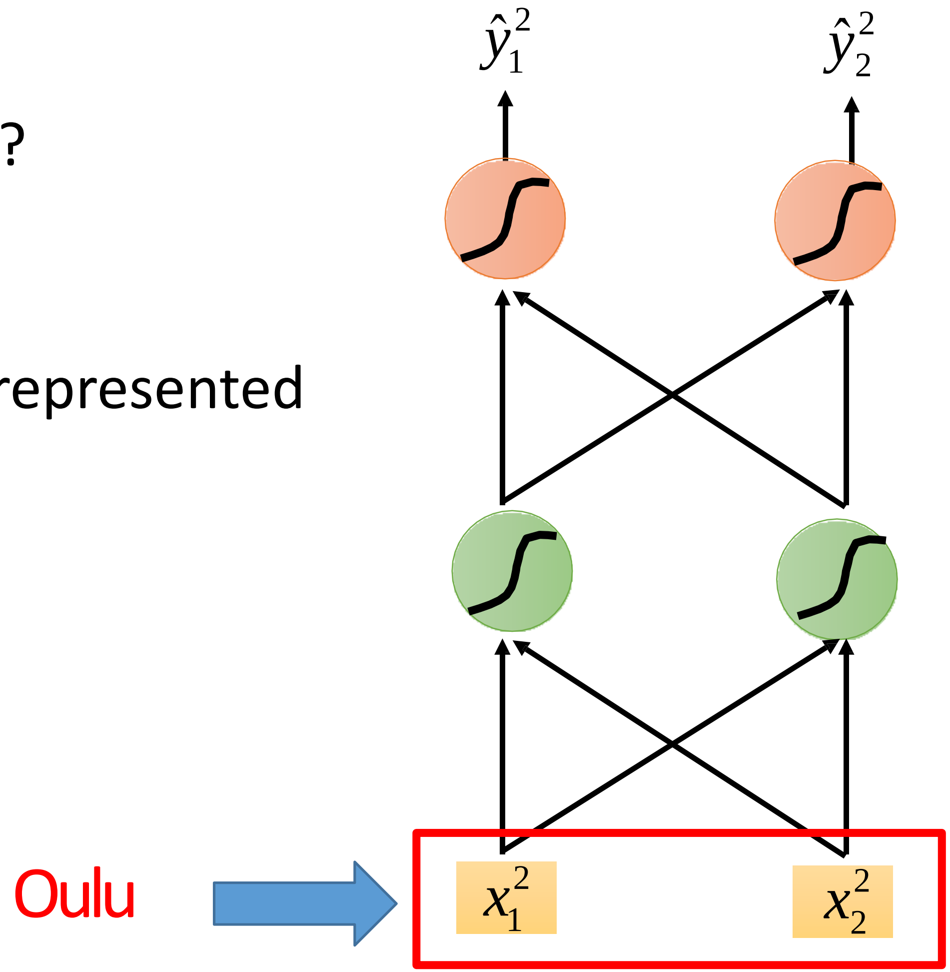
https://www.youtube.com/watch?v=xCGidAeyS4M

Why RNN?

Solving slot filling by
Feedforward network?

Input: a word

(Each word is represented
as a vector)



https://www.youtube.com/watch?v=xCGidAeyS4M

One-hot encoding

How to represent each word as a vector?

<u>One-hot Encoding</u> lexicon = {apple, bag, cat, dog, elephant}

The vector is lexicon size.

Each dimension corresponds to a word in the lexicon

The dimension for the word is 1, and others are 0

apple = [1 0 0 0 0]

bag = [0 1 0 0 0]

cat = [0 0 1 0 0]

dog = [0 0 0 1 0]

elephant = [0 0 0 0 1]

https://www.youtube.com/watch?v=xCGidAeyS4M

Why RNN?

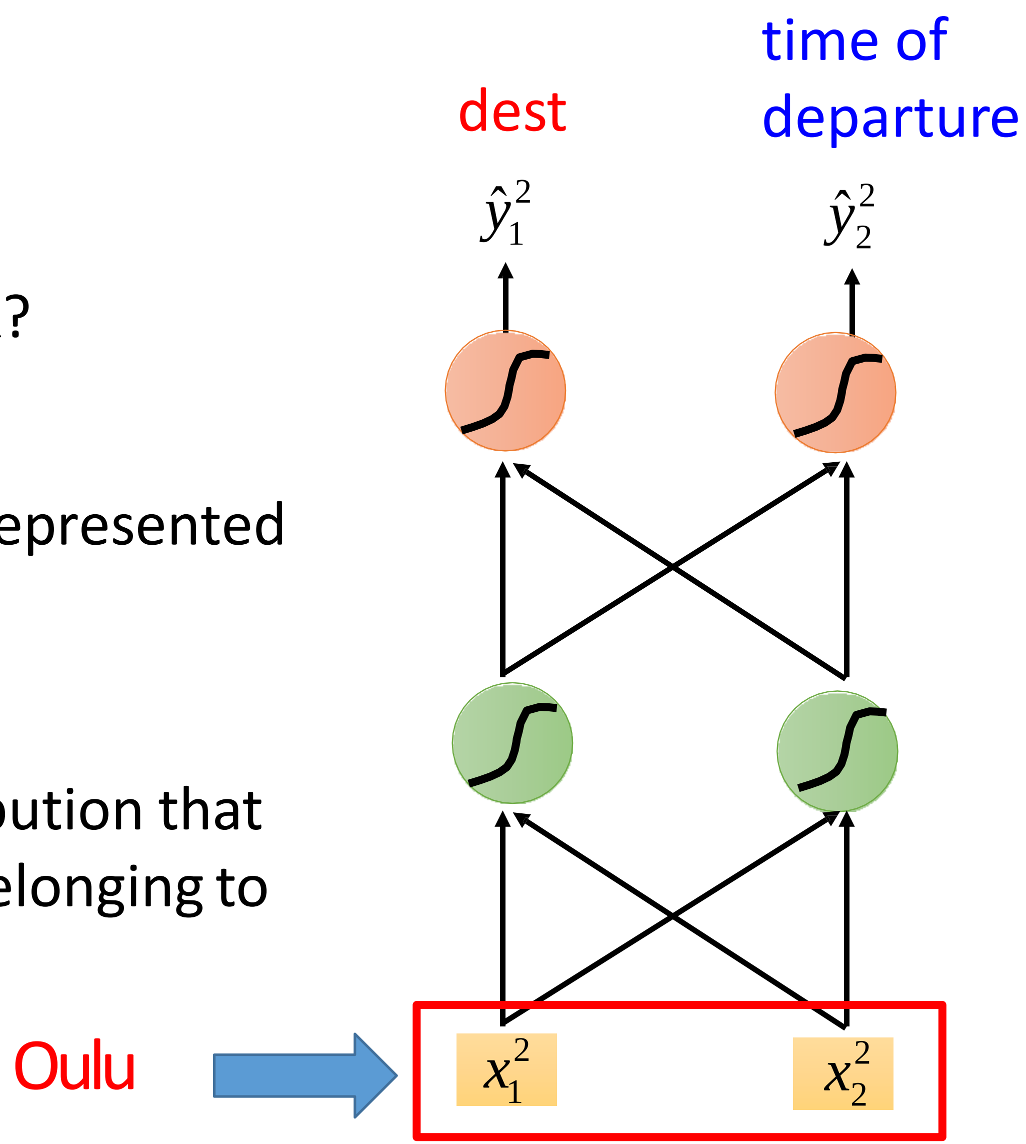
Solving slot filling by Feedforward network?

Input: a word

(Each word is represented as a vector)

Output:

Probability distribution that the input word belonging to the slots



https://www.youtube.com/watch?v=xCGidAeyS4M

Why RNN?

arrive Oulu on November 2nd

other dest other time time

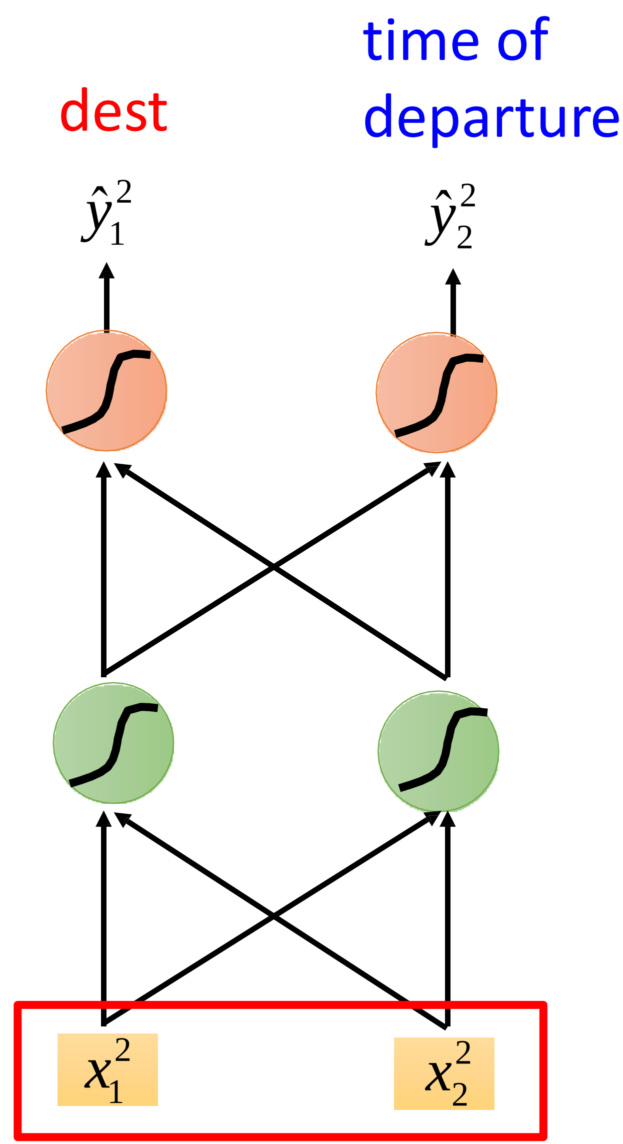
Problem?

leave Oulu on November 2nd

place of departure

Neural network needs memory!

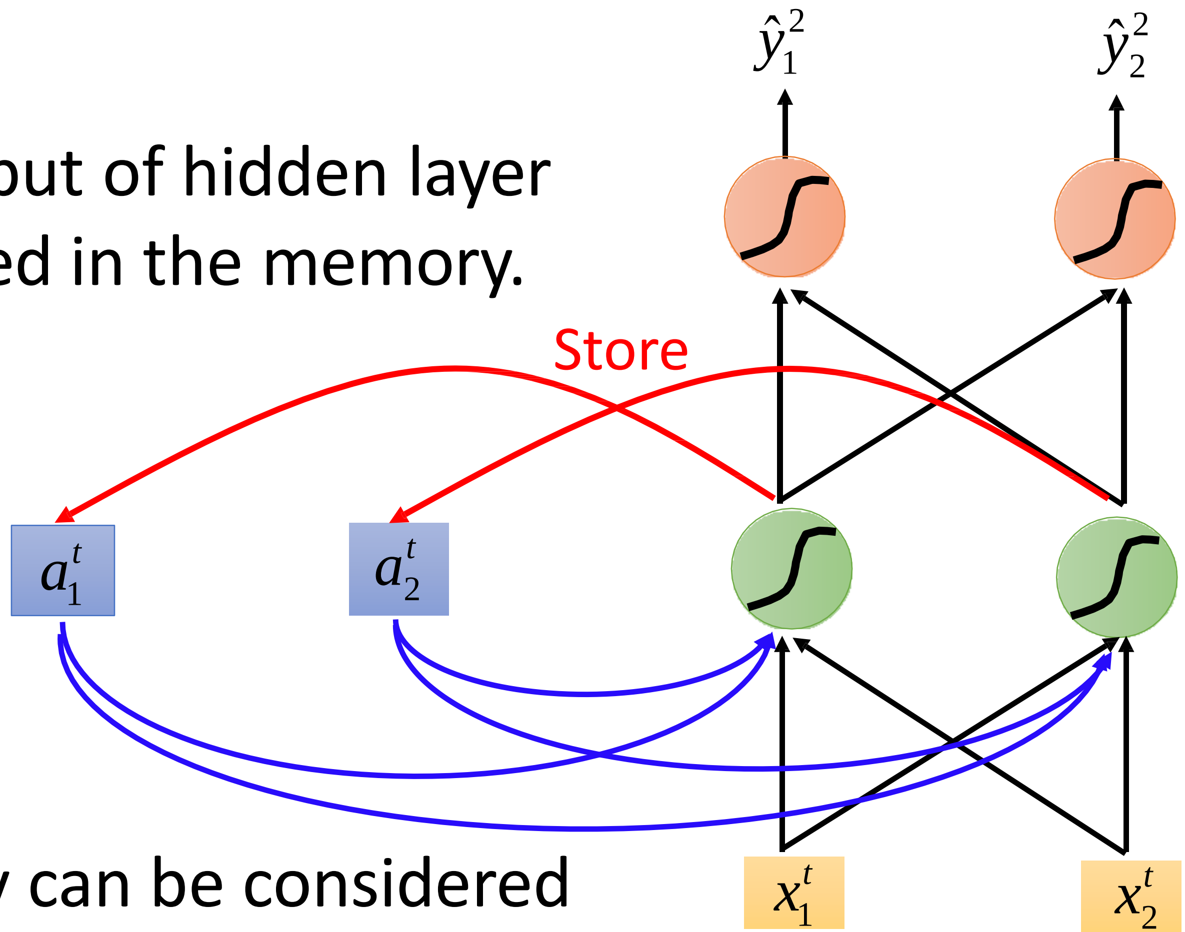
Oulu



https://www.youtube.com/watch?v=xCGidAeyS4M

Recurrent Neural Network (RNN)

The output of hidden layer are stored in the memory.



Memory can be considered as another input.

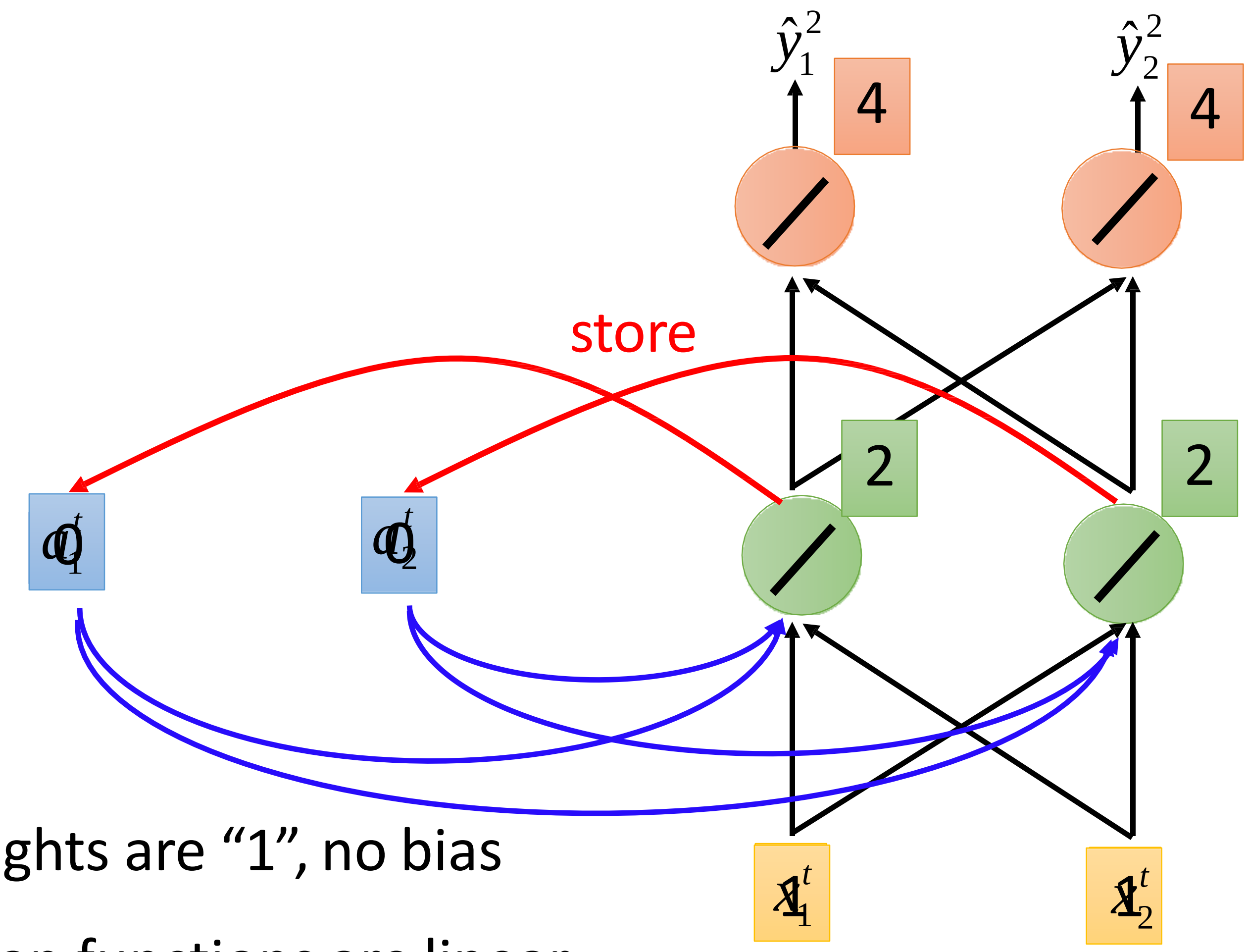
https://www.youtube.com/watch?v=xCGidAeyS4M

Example

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}$

given Initial values



All the weights are “1”, no bias

All activation functions are linear

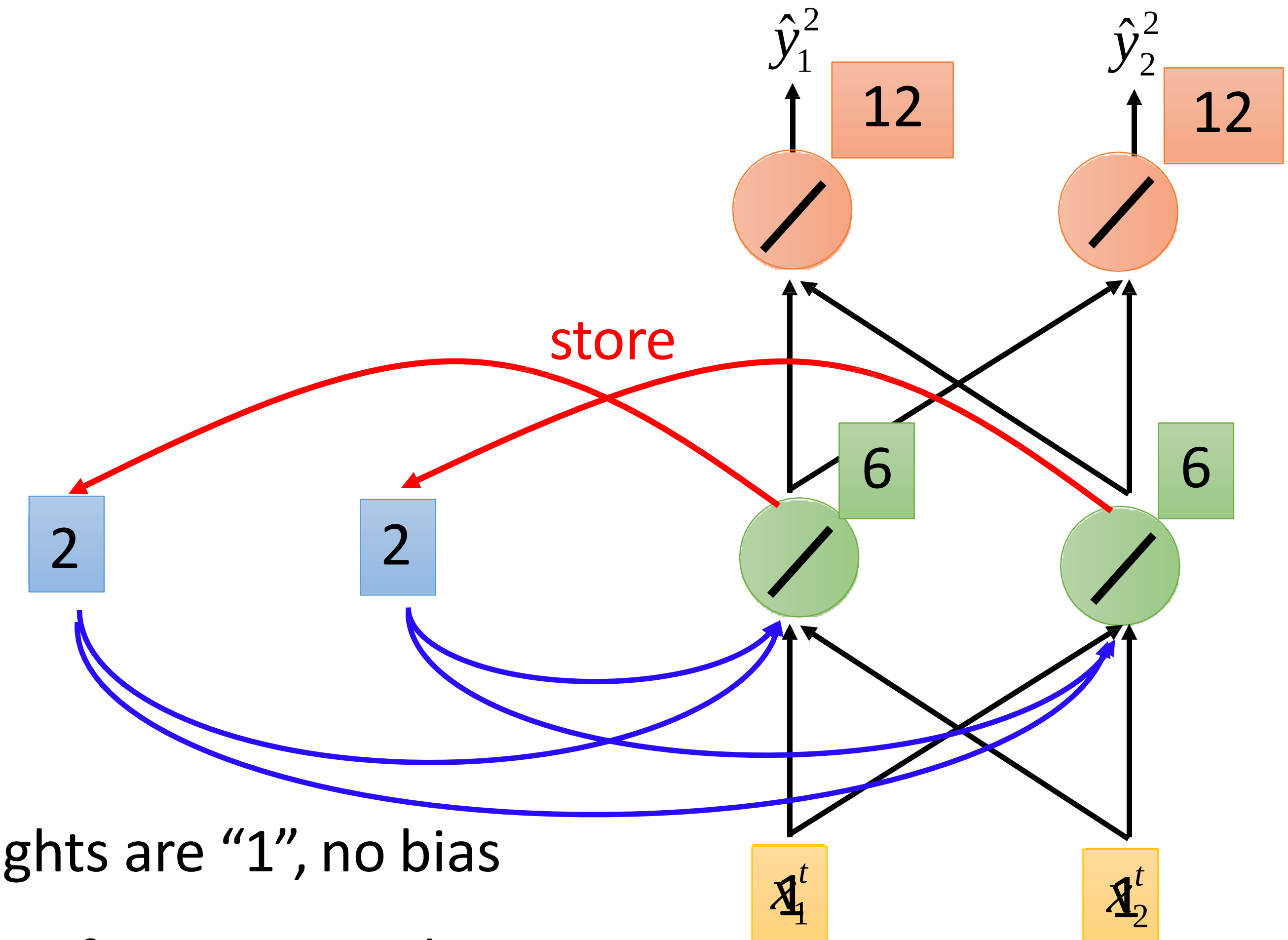
time: 1

https://www.youtube.com/watch?v=xCGidAeyS4M

Example

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}\begin{bmatrix}12\\12\end{bmatrix}$



All the weights are "1", no bias

All activation functions are linear

time: 2

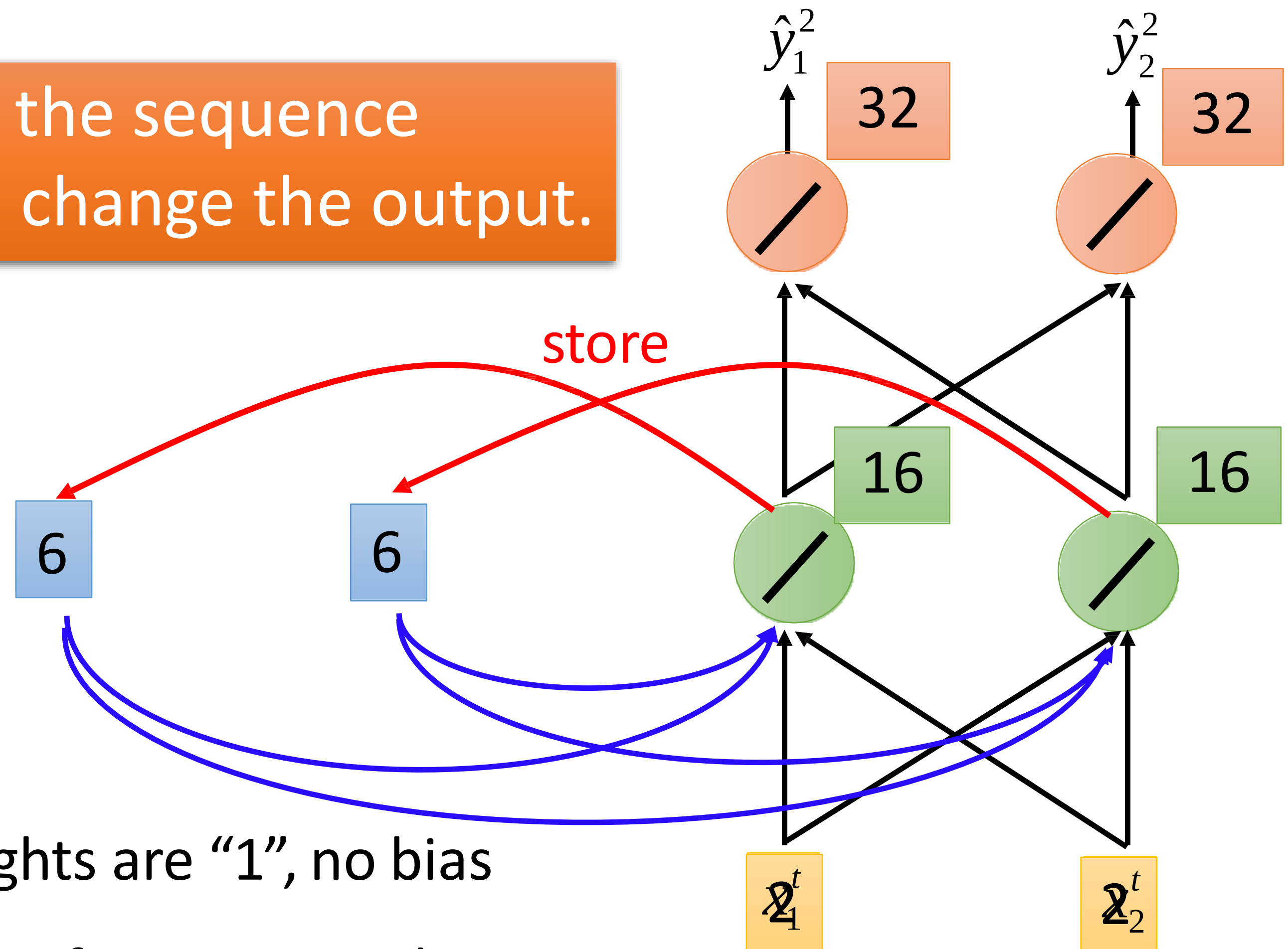
https://www.youtube.com/watch?v=xCGidAeyS4M

Example

Input sequence: $\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}1\\1\end{bmatrix}\begin{bmatrix}2\\2\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}4\\4\end{bmatrix}\begin{bmatrix}12\\12\end{bmatrix}\begin{bmatrix}32\\32\end{bmatrix}$ ……

Changing the sequence order will change the output.



All the weights are "1", no bias

All activation functions are linear

time: 3

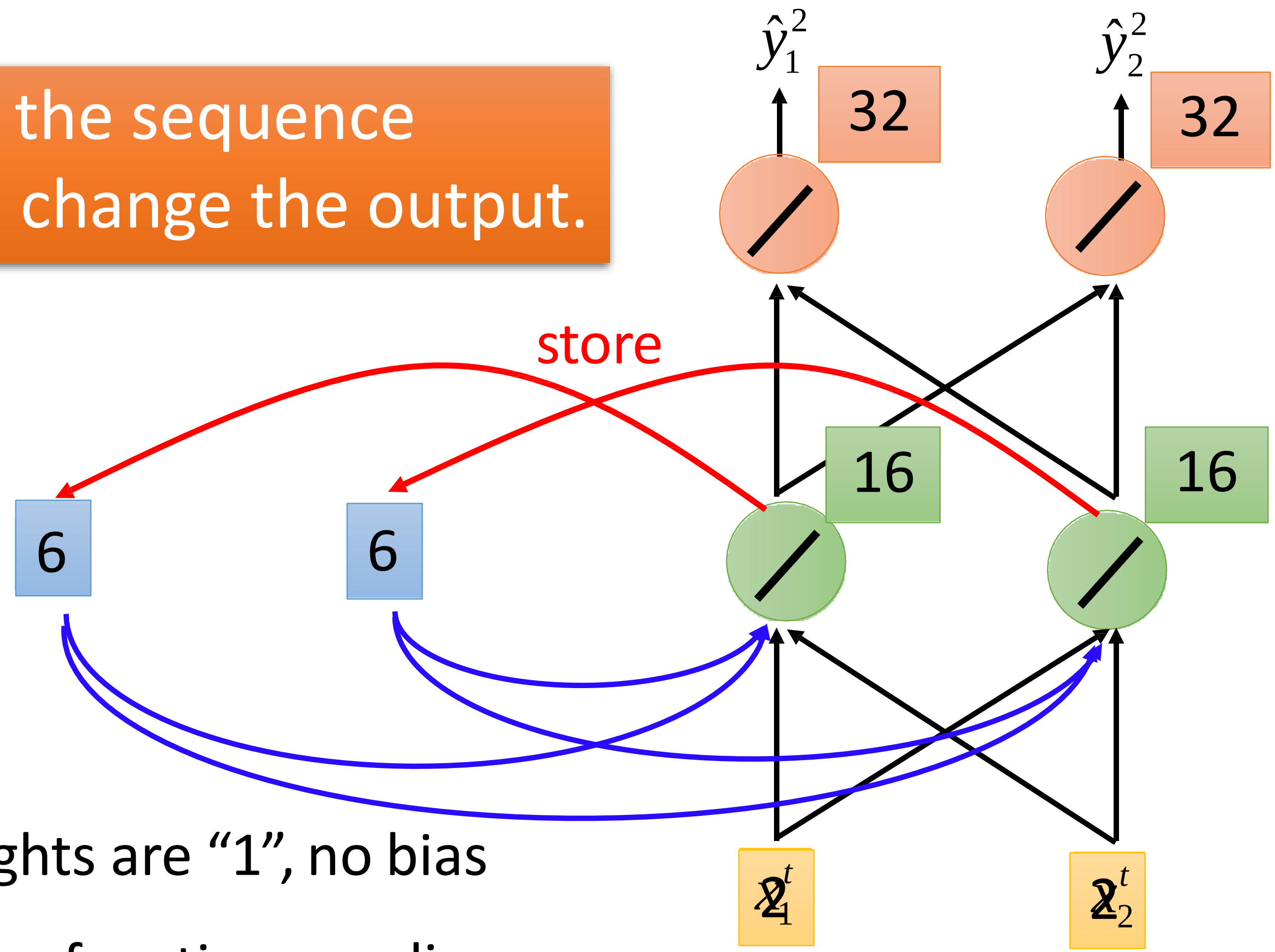
https://www.youtube.com/watch?v=xCGidAeyS4M

Example

Input sequence: $\begin{bmatrix}2\\2\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}$ $\begin{bmatrix}1\\1\end{bmatrix}$ ……

output sequence: $\begin{bmatrix}8\\8\end{bmatrix}$ $\begin{bmatrix}20\\20\end{bmatrix}$ $\begin{bmatrix}44\\44\end{bmatrix}$ ……

Changing the sequence order will change the output.



All the weights are "1", no bias

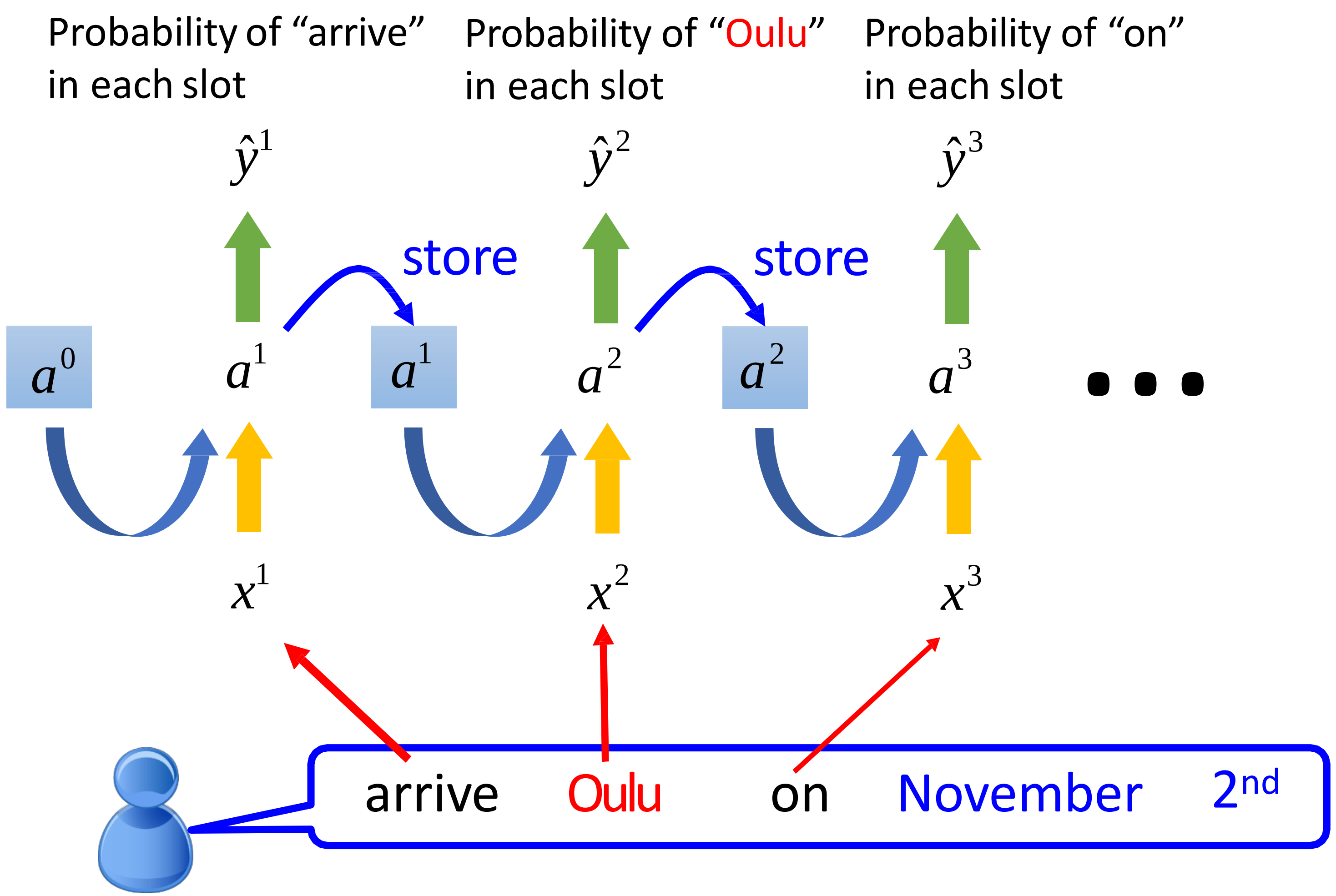
All activation functions are linear

time: 3

https://www.youtube.com/watch?v=xCGidAeyS4M

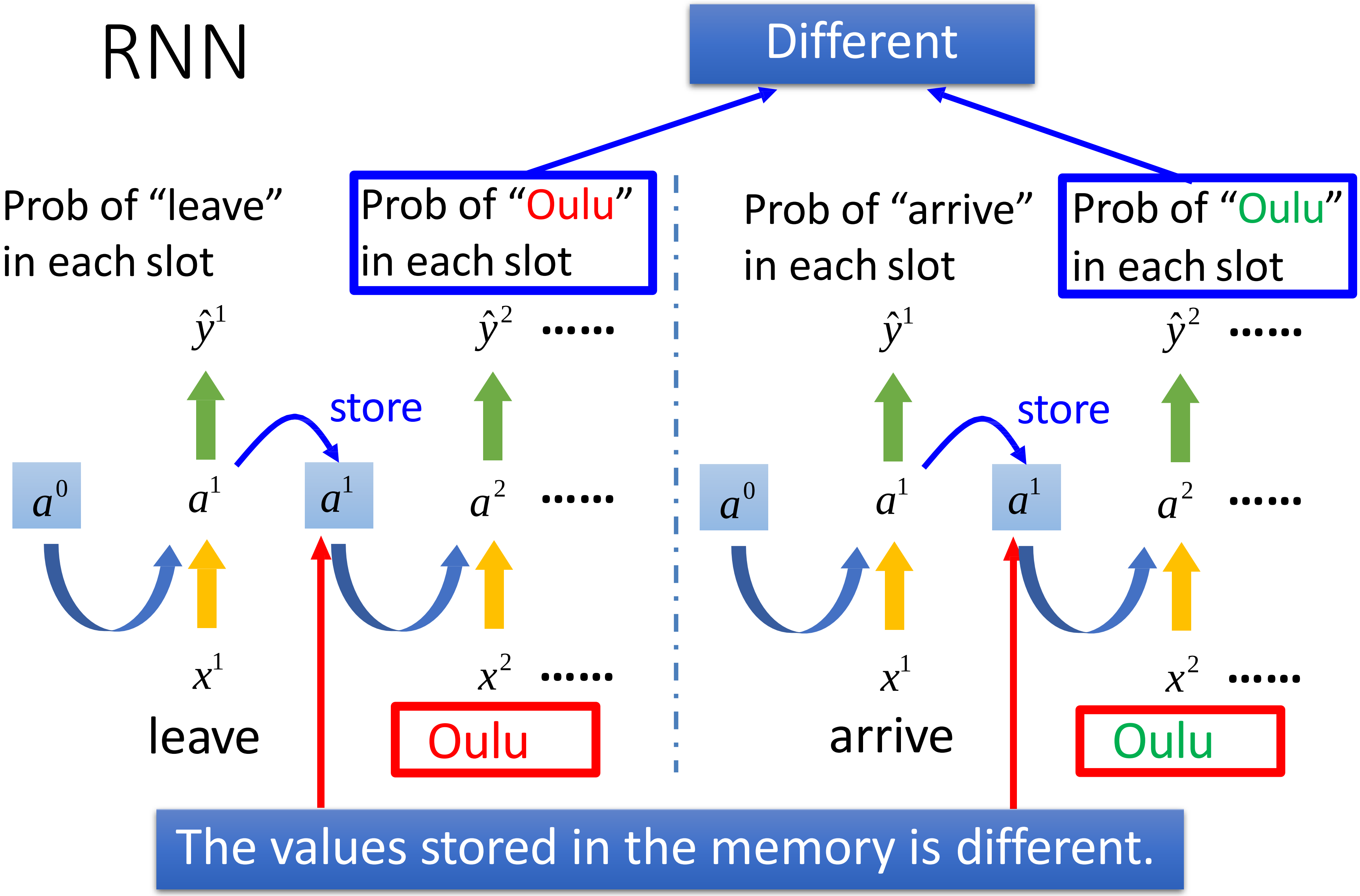
RNN

The same network is used again and again.



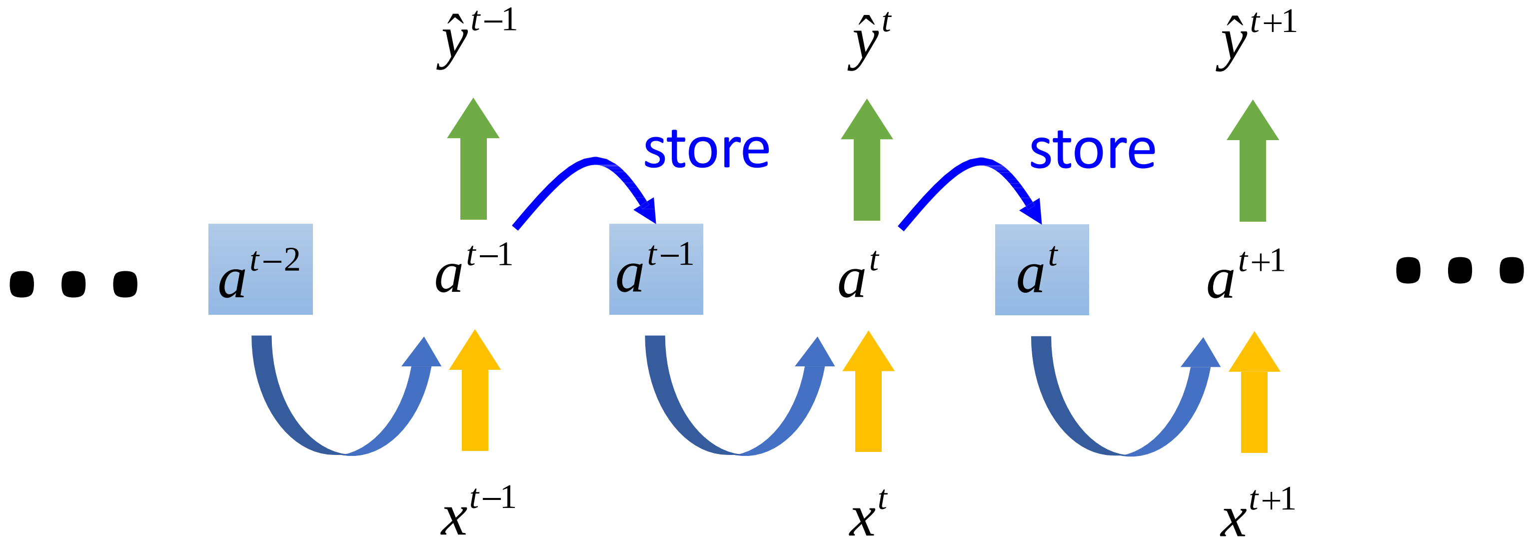
https://www.youtube.com/watch?v=xCGidAeyS4M

RNN



https://www.youtube.com/watch?v=xCGidAeyS4M

RNN Training



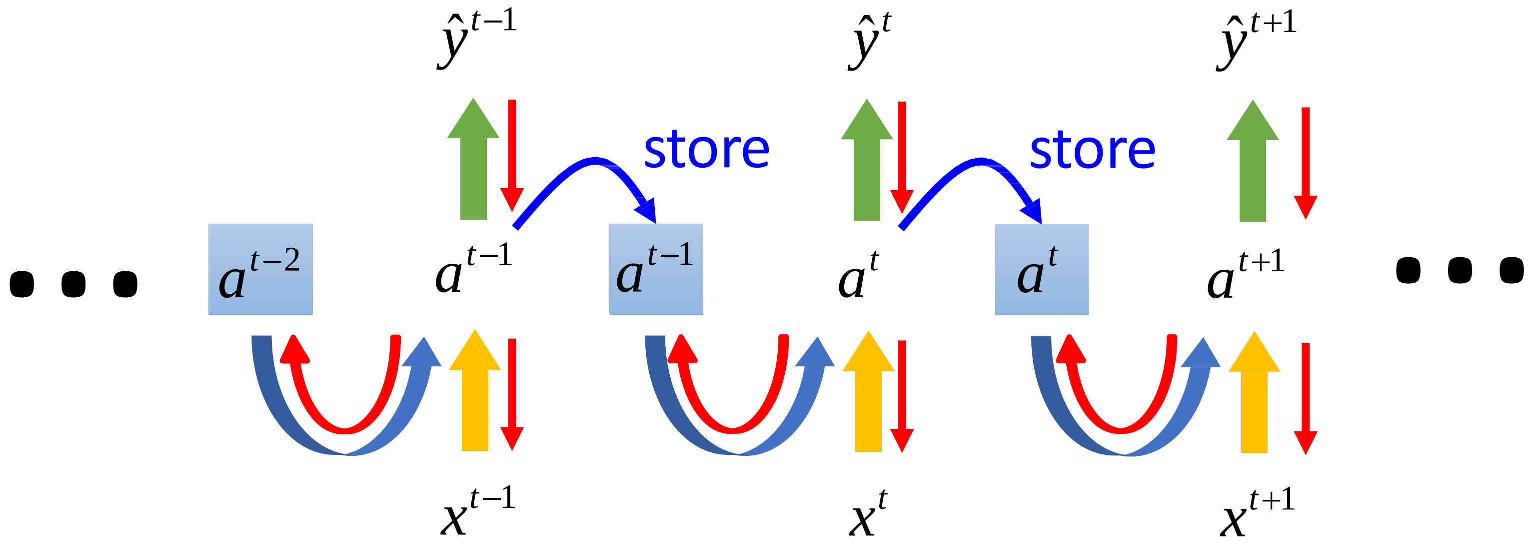
Forward Propagation:

$$a^t = g(w_{aa}a^{t-1} + w_{ax}\,x^t + b_a) \quad\Rightarrow\quad a^t = g(w_a[a^{t-1}, x^t] + b_a)$$

$$\hat{y}^t = g(w_{ya}a^t + b_y) \quad\Rightarrow\quad \hat{y}^t = g(w_y a^t + b_y)$$

RNN Training

Backpropagation through time (BPTT)



Back Propagation: $w_a \leftarrow w_a - \alpha \dfrac{\partial L}{\partial w_a}$ $\quad\alpha$: learning rate (3)

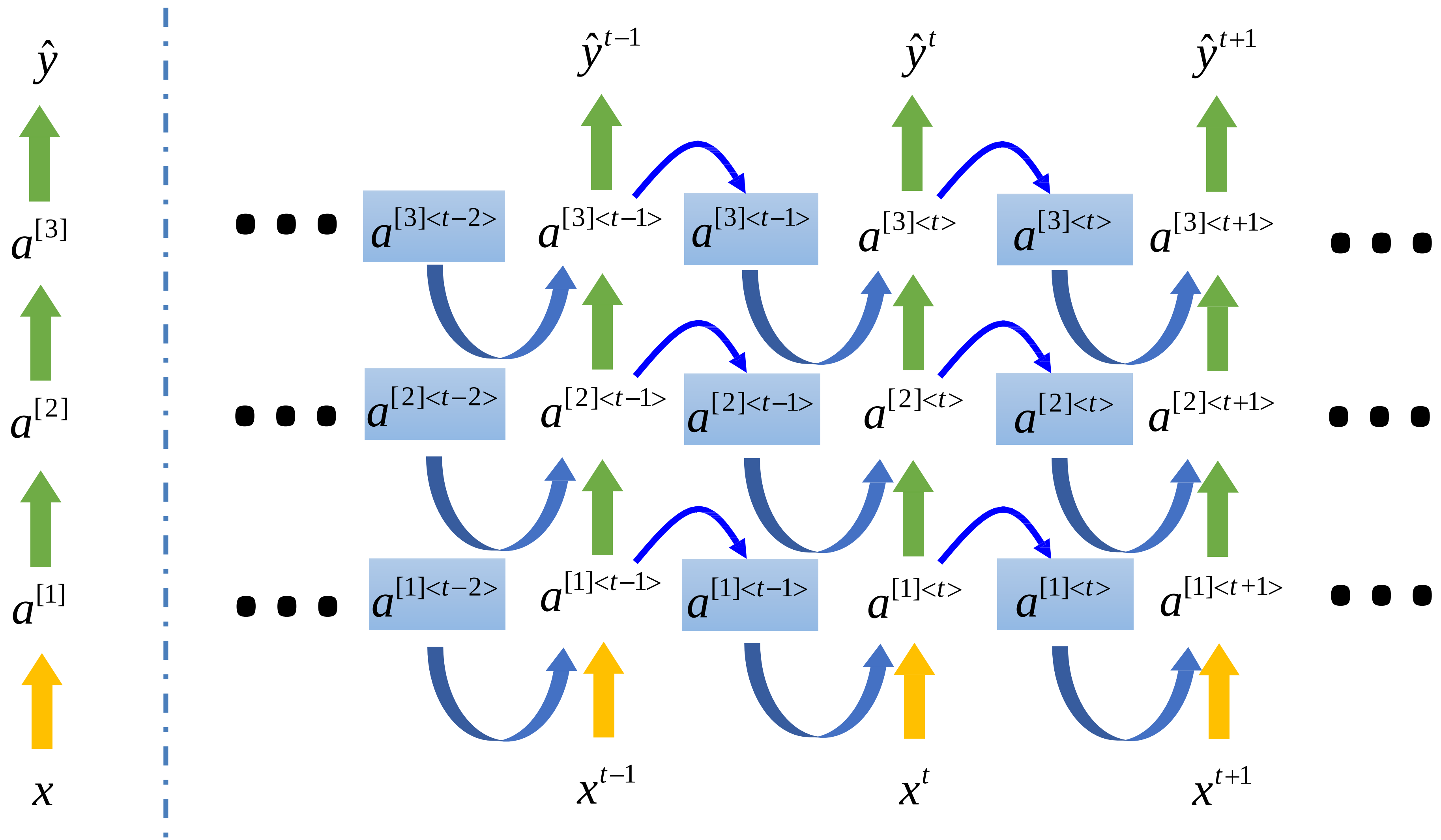
$$L^t(\hat{y}^t, y^t) = -y^t log \hat{y}^t \qquad (1)$$

$$L = \sum_{t=1}^{T_y} L^t(\hat{y}^t, y^t) \qquad (2)$$

$$\frac{\partial L}{\partial w_a} = \sum_{t=1}^{T_y} \frac{\partial L^t}{\partial w_a} \qquad (4)$$

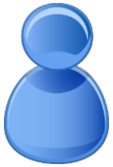
$$\frac{\partial L^t}{\partial w_a} = \sum_{k=1}^{t} \frac{\partial L^t}{\partial \hat{y}^t} \frac{\partial \hat{y}^t}{\partial a^t} \frac{\partial a^t}{\partial a^k} \frac{\partial a^k}{\partial w_a} \qquad (5)$$

Deep RNN

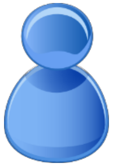


$\hat{y}$
$a^{[3]}$
$a^{[2]}$
$a^{[1]}$
x
$\hat{y}^{t-1}$
$\hat{y}^{t}$
$\hat{y}^{t+1}$
$a^{[3]<t-2>}$
$a^{[3]<t-1>}$
$a^{[3]<t-1>}$
$a^{[3]<t>}$
$a^{[3]<t>}$
$a^{[3]<t+1>}$
$a^{[2]<t-2>}$
$a^{[2]<t-1>}$
$a^{[2]<t-1>}$
$a^{[2]<t>}$
$a^{[2]<t>}$
$a^{[2]<t+1>}$
$a^{[1]<t-2>}$
$a^{[1]<t-1>}$
$a^{[1]<t-1>}$
$a^{[1]<t>}$
$a^{[1]<t>}$
$a^{[1]<t+1>}$
x^{t-1}
x^{t}
x^{t+1}

Bidirectional RNN (BRNN)

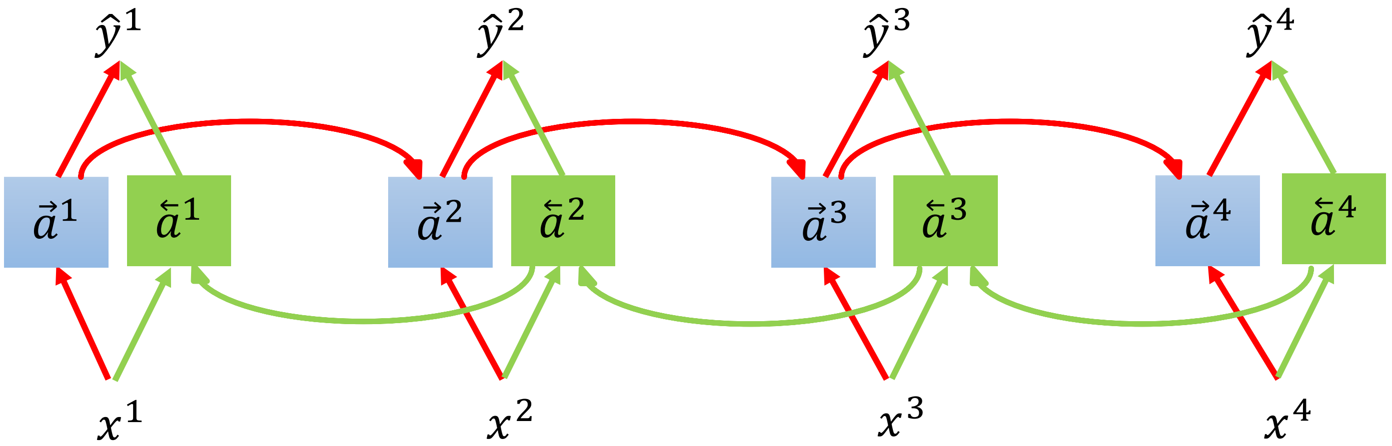


Beijing, the capital of China, I will arrive there on November 2nd.



Beijing, the capital of China, I will leave there on November 2nd.

Bidirectional RNN



Advantage: take into account information from the past and from the future.

Disadvantage: need the entire sequence of data before you can make predictions anywhere

https://www.coursera.org/lecture/nlp-sequence-models/bidirectional-rnn-fyXnn

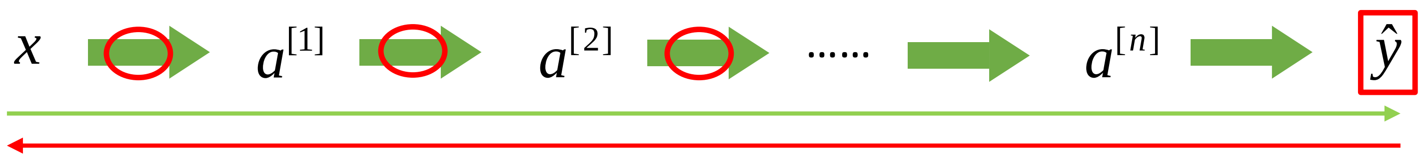
Vanishing gradients with RNNs

I will **arrive** Beijing, the capital of China and historical city…, on November 2nd.

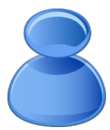
Time of arrival

I will **leave** Beijing, the capital of China and historical city…, on November 2nd.

Time of departure

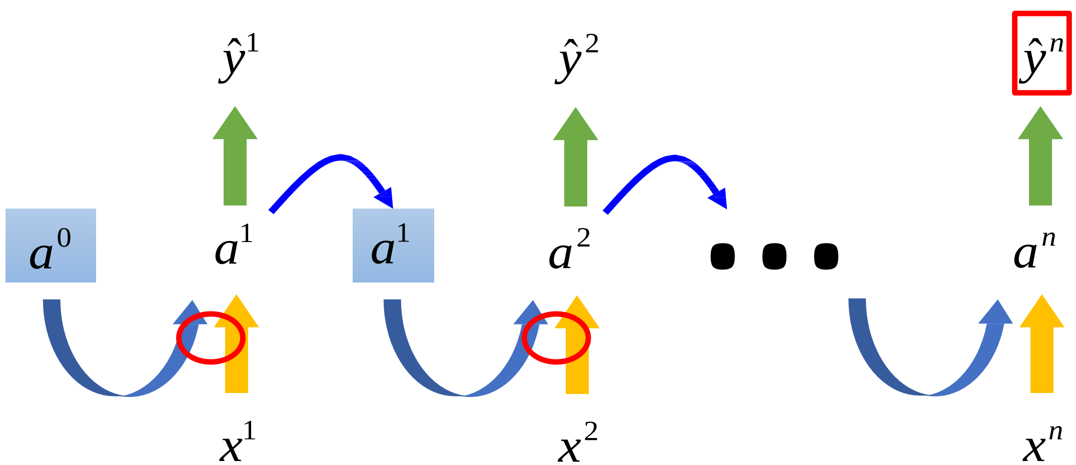


Vanishing gradients with RNNs



I will **arrive** Beijing, the capital of China and historical city, on November 2nd.

I will **leave** Beijing, the capital of China and historical city, on November 2nd.

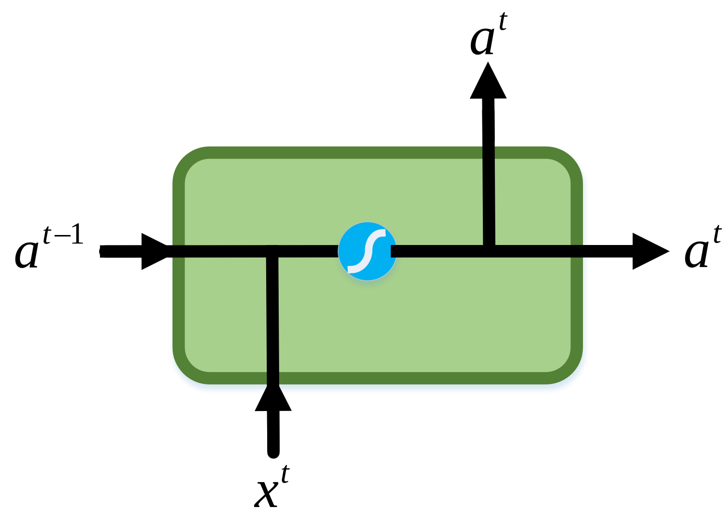


Exploding gradients?

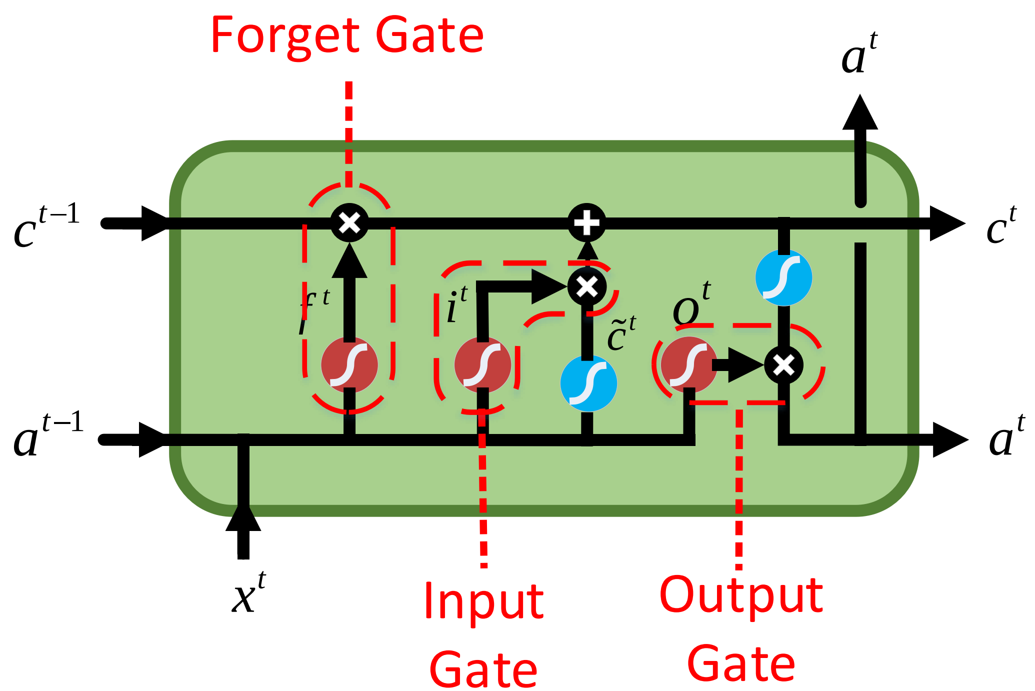
gradient clipping

Long Short-term Memory (LSTM)

Basic RNN unit

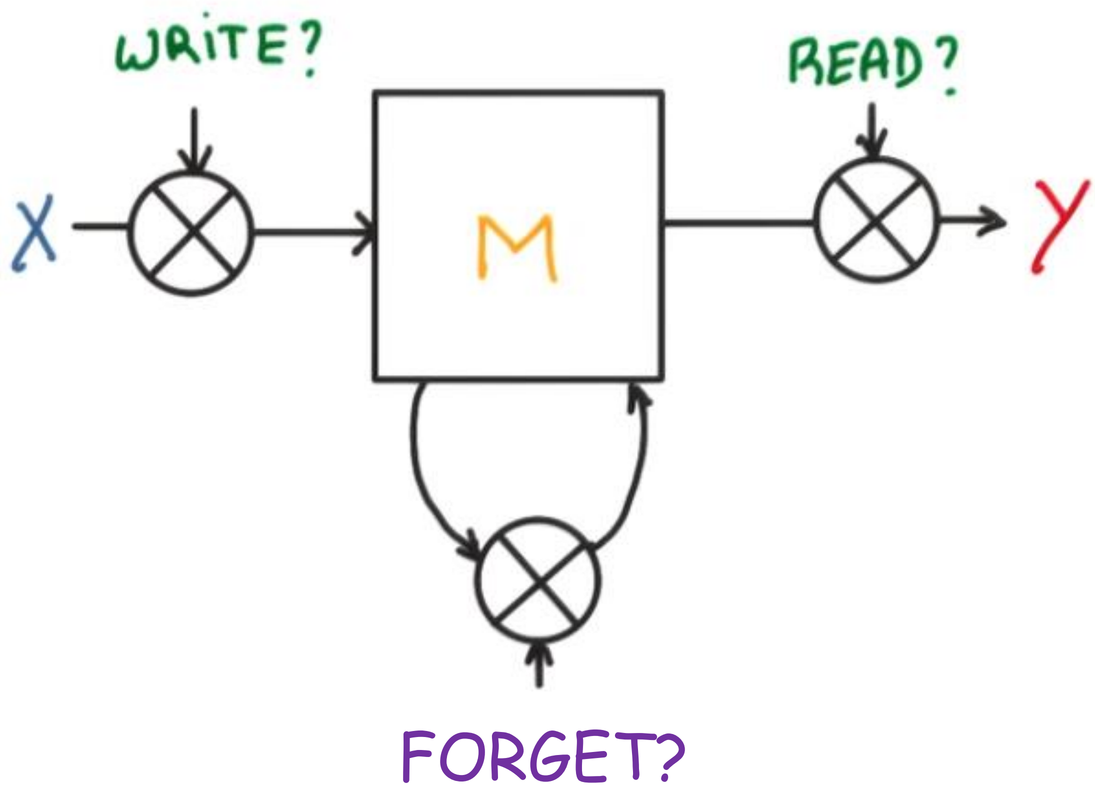


LSTM Unit

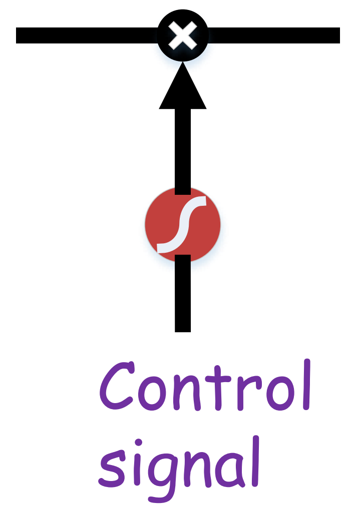


[1] Hochreiter S, Schmidhuber J. Long short-term memory. Neural computation, 1997, 9(8): 1735-1780.

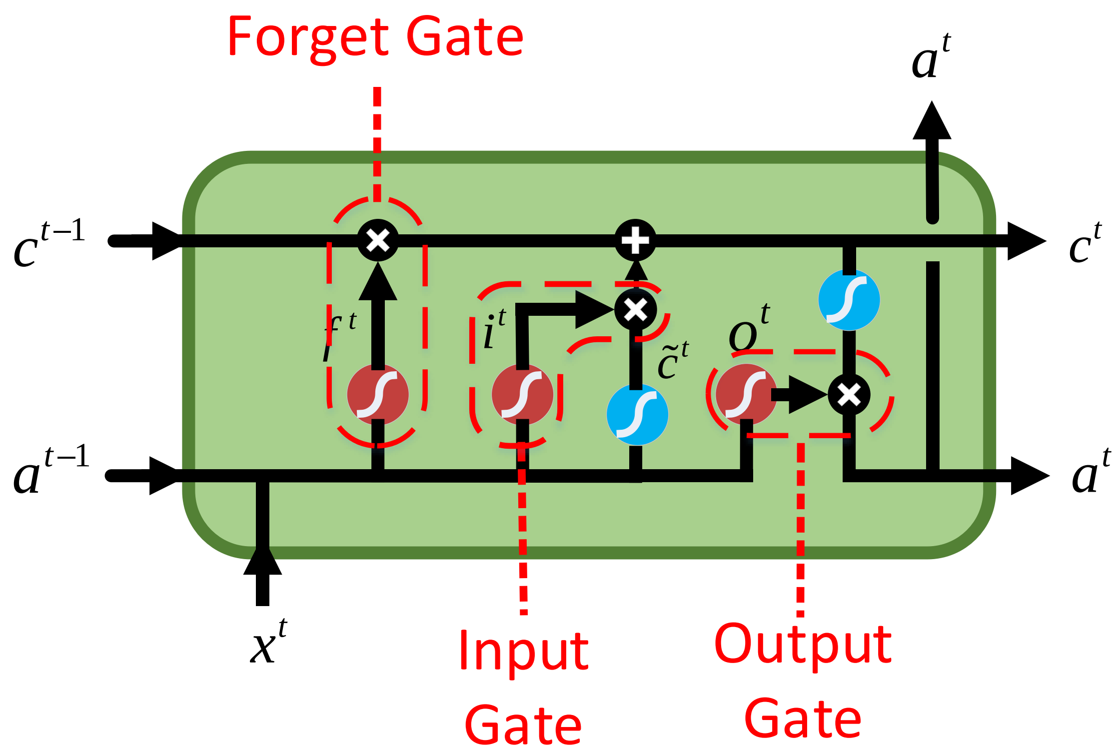
Long Short-term Memory (LSTM)

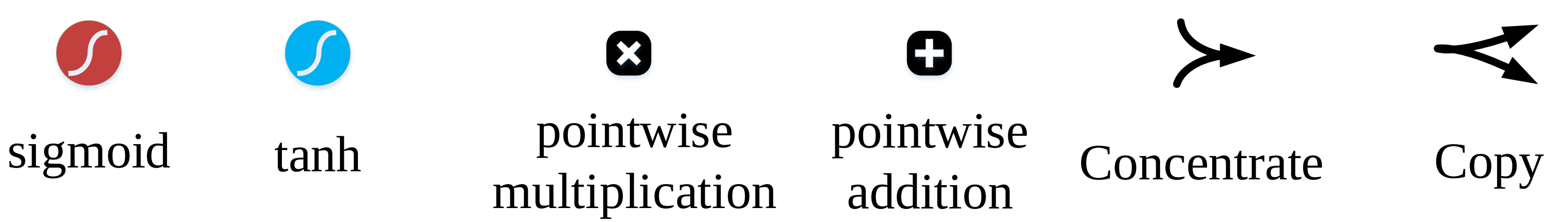


Gate Unit:



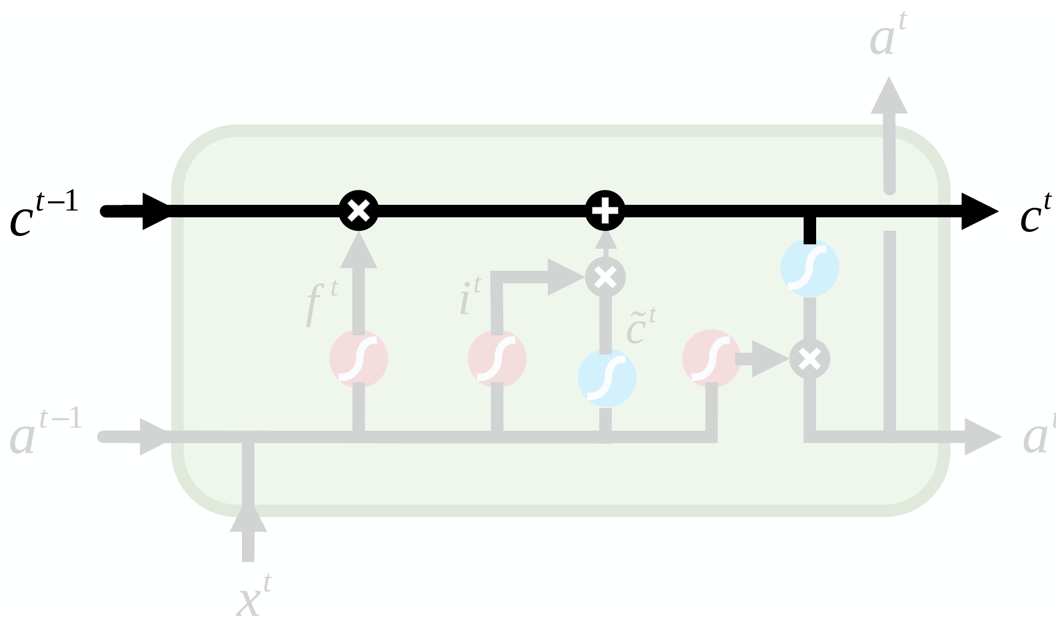
Long Short-term Memory (LSTM)





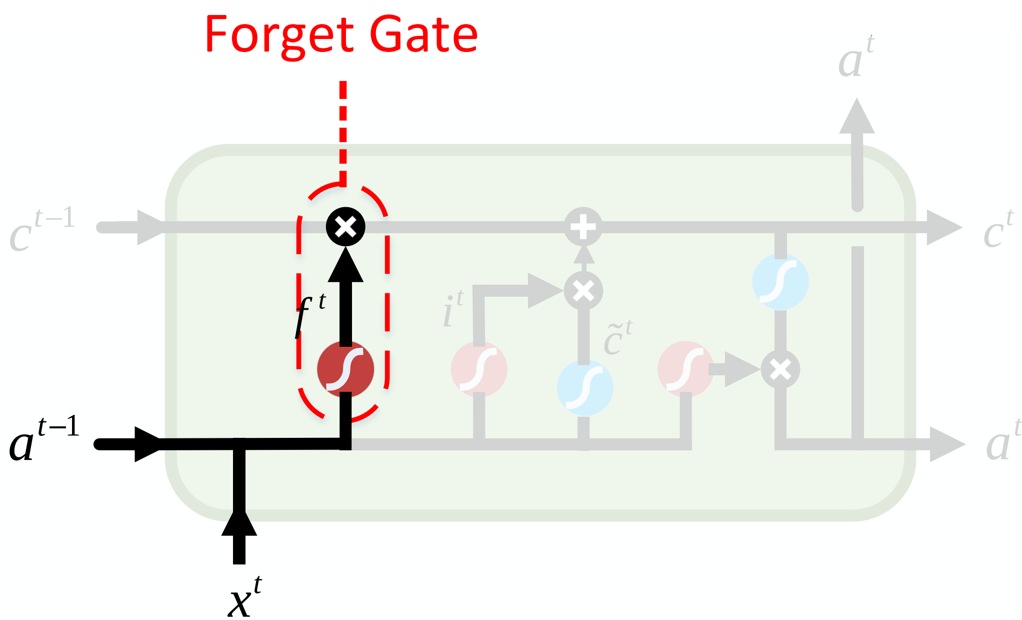
Long Short-term Memory (LSTM)

Memory Cell State:



Long Short-term Memory (LSTM)

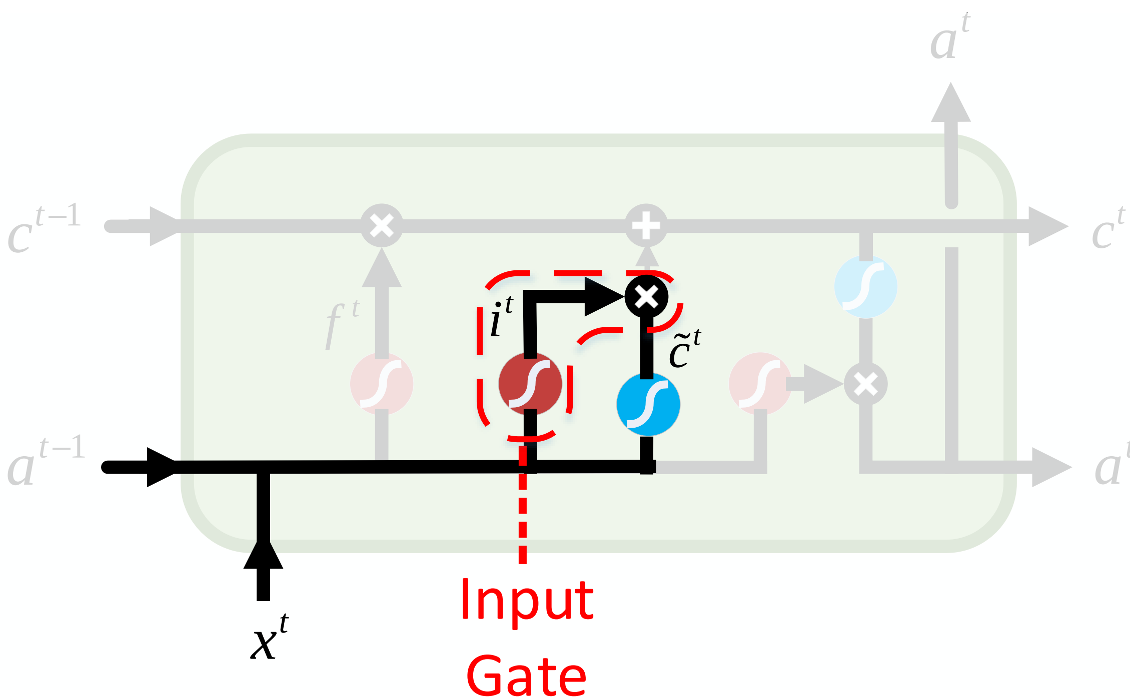
Step 1:



$$f^t = \sigma(w_f[a^{t-1}, x^t] + b_f)$$

Long Short-term Memory (LSTM)

Step 2:

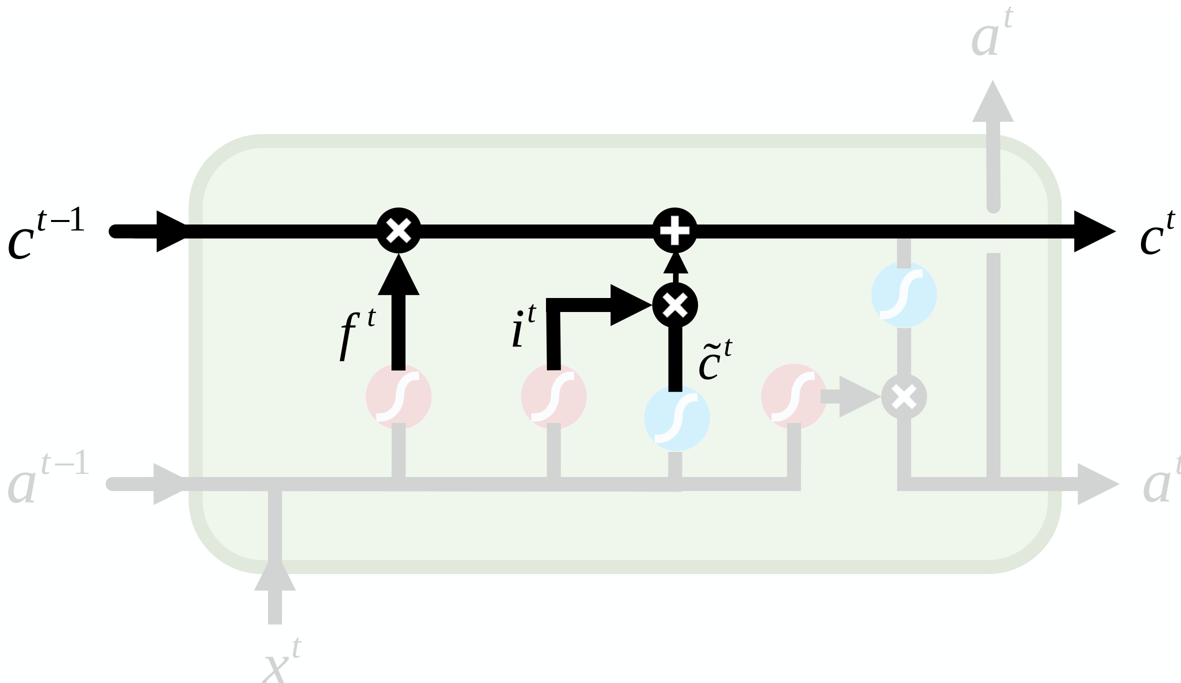


$$i^t = \sigma(w_i[a^{t-1}, x^t] + b_i)$$

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

Long Short-term Memory (LSTM)

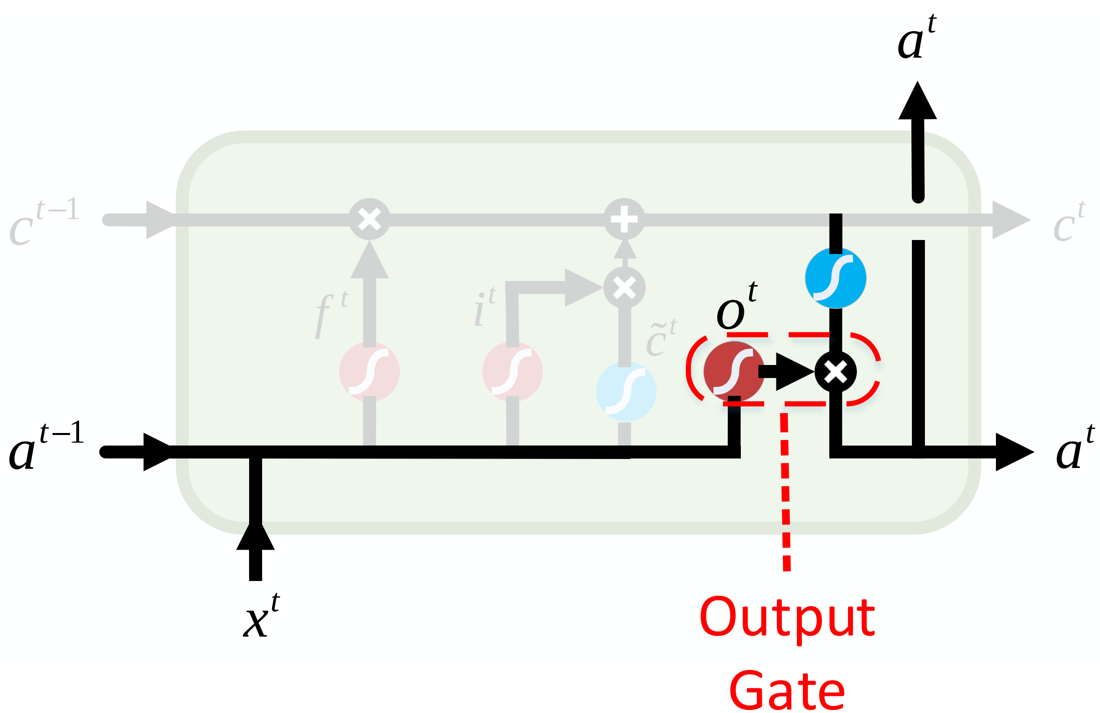
Step 3:



$$c^t = f^t * c^{t-1} + i^t * \tilde{c}^t$$

Long Short-term Memory (LSTM)

Step 4:

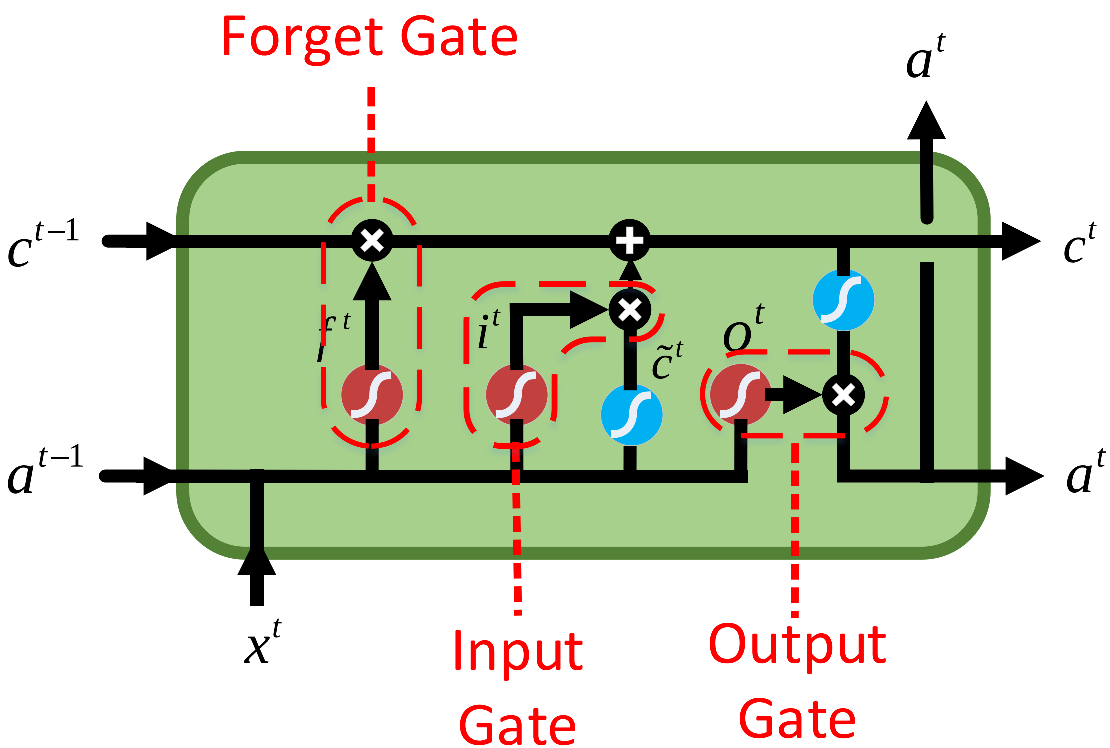


$$o^t = \sigma(w_o[a^{t-1}, x^t] + b_o)$$

$$a^t = o^t * \tanh(c^t)$$

Long Short-term Memory (LSTM)

Overall:



$$f^t = \sigma(w_f[a^{t-1}, x^t] + b_f)$$

$$i^t = \sigma(w_i[a^{t-1}, x^t] + b_i)$$

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

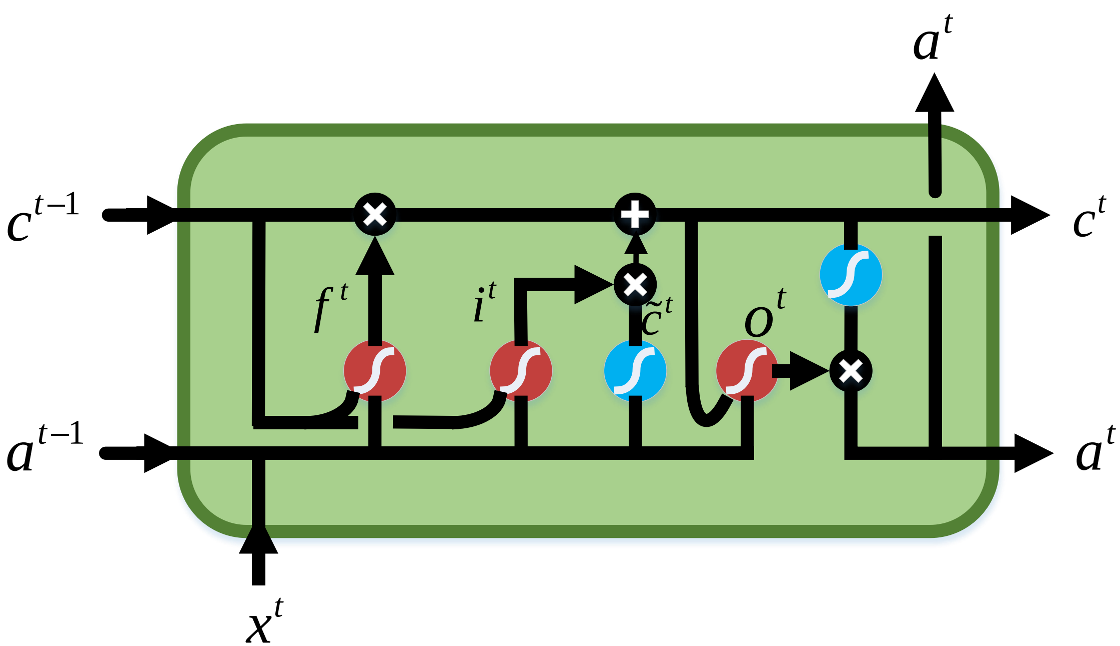
$$c^t = f^t * c^{t-1} + i^t * \tilde{c}^t$$

$$o^t = \sigma(w_o[a^{t-1}, x^t] + b_o)$$

$$a^t = o^t * \tanh(c^t)$$

Long Short-term Memory (LSTM)

Some Variants of LSTM :

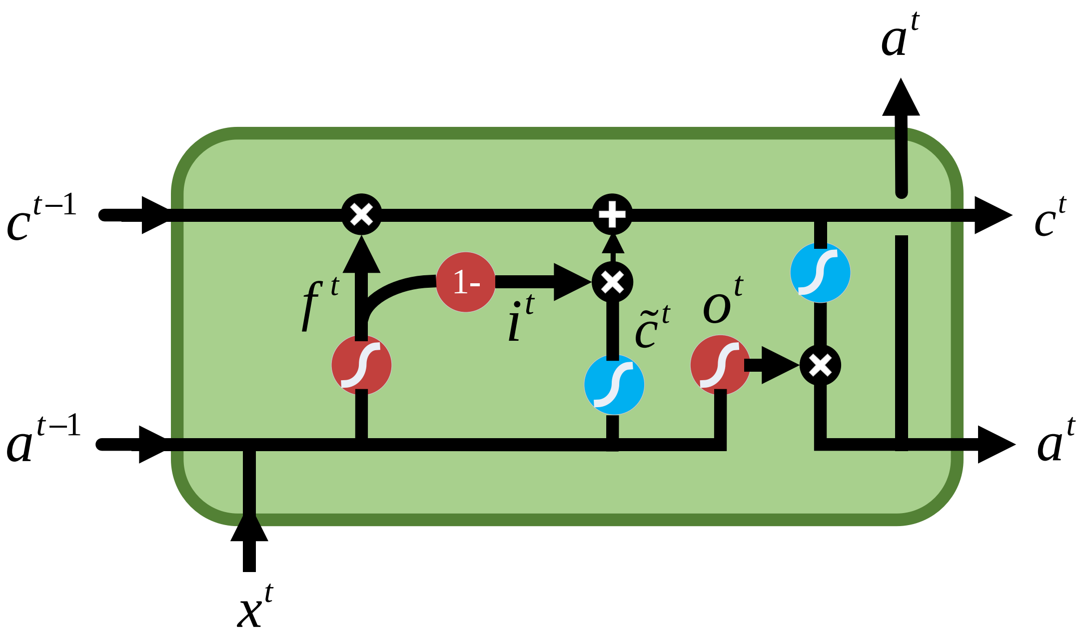


$$f^t = \sigma(w_f[c^{t-1}, a^{t-1}, x^t] + b_f)$$
$$i^t = \sigma(w_i[c^{t-1}, a^{t-1}, x^t] + b_i)$$
$$o^t = \sigma(w_o[c^t, a^{t-1}, x^t] + b_o)$$

[1] Gers F A, Schmidhuber J. Recurrent nets that time and count. Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks. IJCNN 2000.

Long Short-term Memory (LSTM)

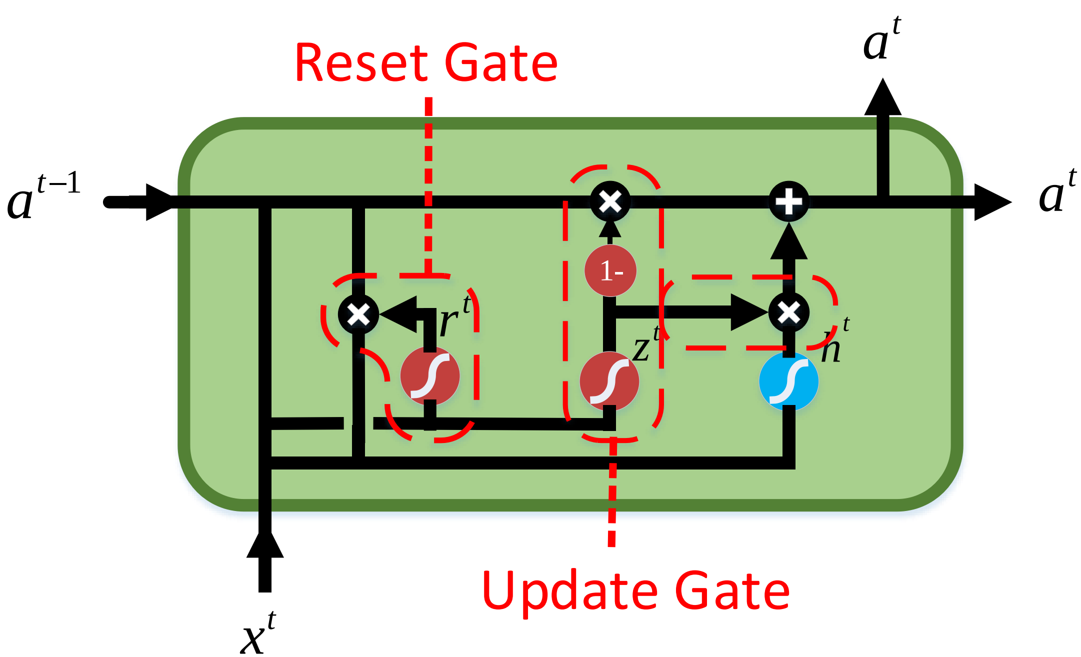
Some Variants of LSTM:



$$f^t = \sigma(w_f[c^{t-1}, a^{t-1}, x^t] + b_f)$$
$$i^t = 1 - f^t$$
$$c^t = f^t * c^{t-1} + (1 - f^t) * \tilde{c}^t$$

[1] Greff K, Srivastava R K, Koutník J, et al. LSTM: A search space odyssey[J]. IEEE transactions on neural networks and learning systems, 2016, 28(10): 2222-2232.

Gated Recurrent Unit (GRU)



$$z^t = \sigma(w_z[a^{t-1}, x^t])$$

$$r^t = \sigma(w_r[a^{t-1}, x^t])$$

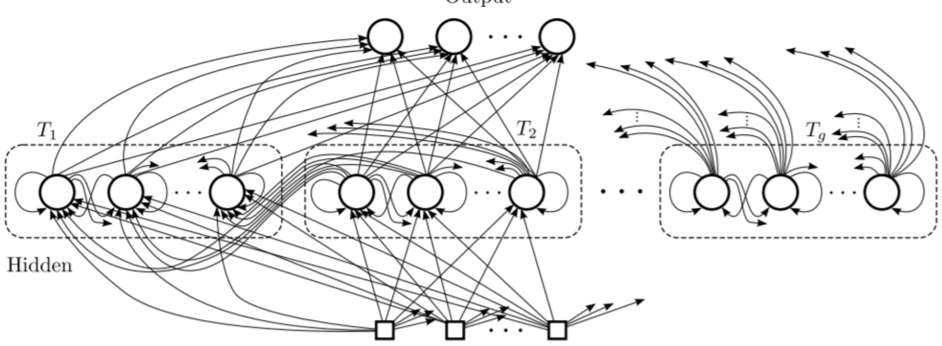
$$\tilde{a}^t = \tanh(w_{\tilde{a}}[r^t * a^{t-1}, x^t])$$

$$a^t = (1 - z^t) * a^{t-1} + z^t * \tilde{a}^t$$

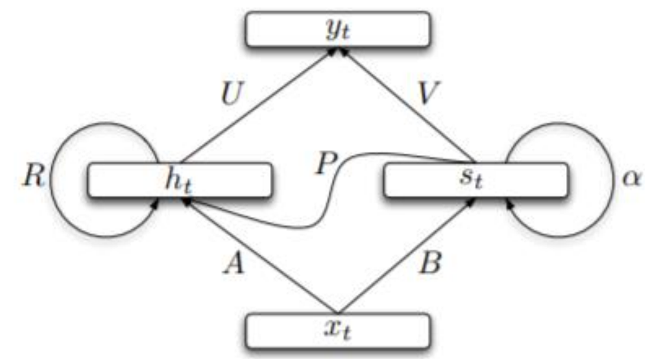
[1] Cho K, Van Merriënboer B, Gulcehre C, et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. arXiv preprint arXiv:1406.1078, 2014.

Other variants of RNN

Clockwise RNN:



Structually Constrained Recurrent Network(SCRN):



[1] Koutnik J, Greff K, Gomez F, et al. A clockwork rnn. arXiv preprint arXiv:1402.3511, 2014.

[2] Mikolov T, Joulin A, Chopra S, et al. Learning longer memory in recurrent neural networks. arXiv preprint arXiv:1412.7753, 2014.

RNN, LSTM, GRU in Pytorch

```
nn.RNN(input_size=512,hidden_size=1024,num_layers=3,
       bias=False, batch_first=True, dropout=0.1, bidirectional=True)
nn.LSTM(input_size=512,hidden_size=1024,num_layers=3,
       bias=False, batch_first=True, dropout=0.1, bidirectional=True)
nn.GRU(input_size=512,hidden_size=1024,num_layers=3,
       bias=False, batch_first=True, dropout=0.1, bidirectional=True)
```

Main Arguments:

input_size: the dimension of x^t

hidden_size: the dimension of a^t and c^t

num_layers: number of recurrent layers

bias: the layer use bias or not

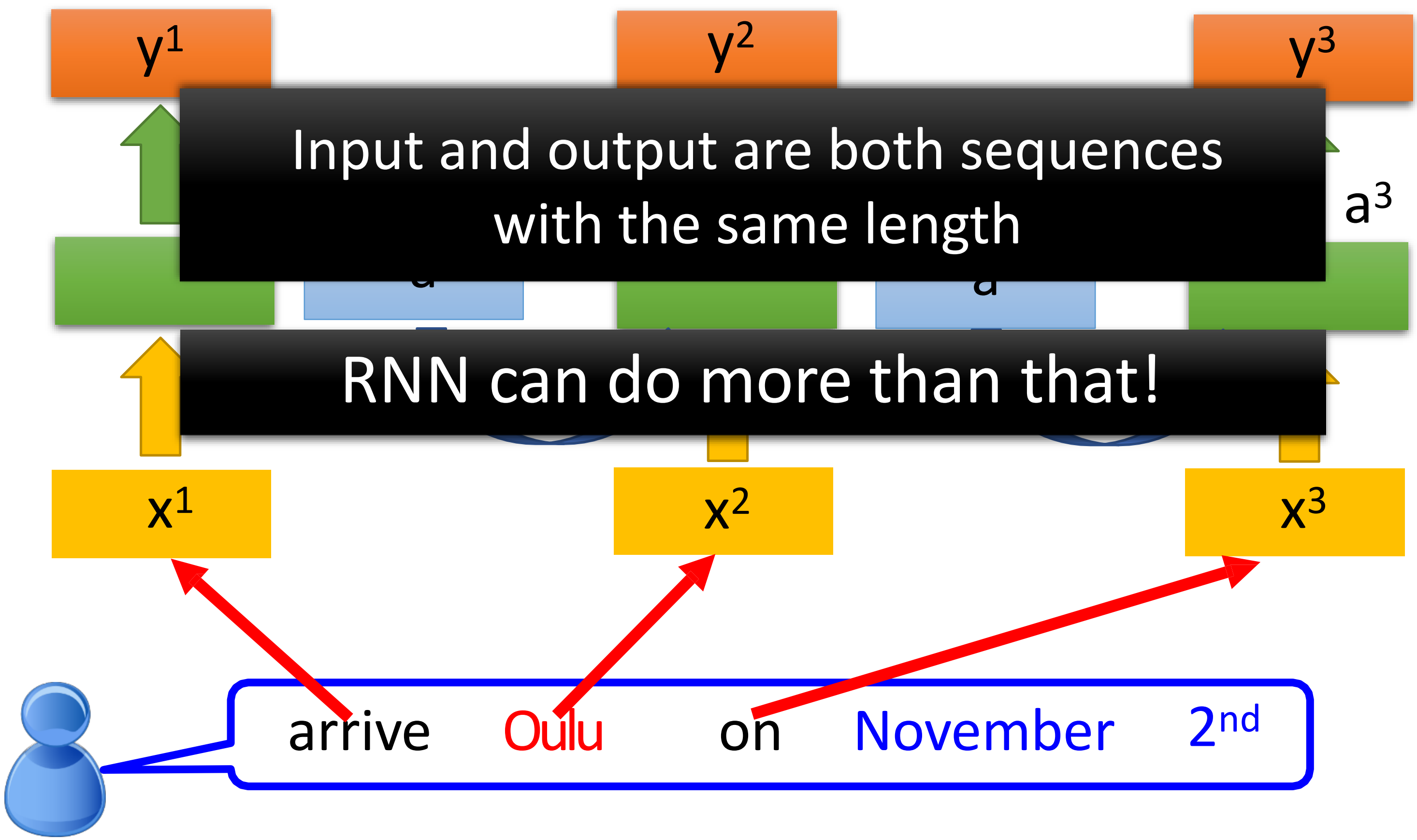
bidirectional: unidirectional or bidirectional

More Applications ……

Probability of "arrive" in each slot

Probability of "Oulu" in each slot

Probability of "on" in each slot

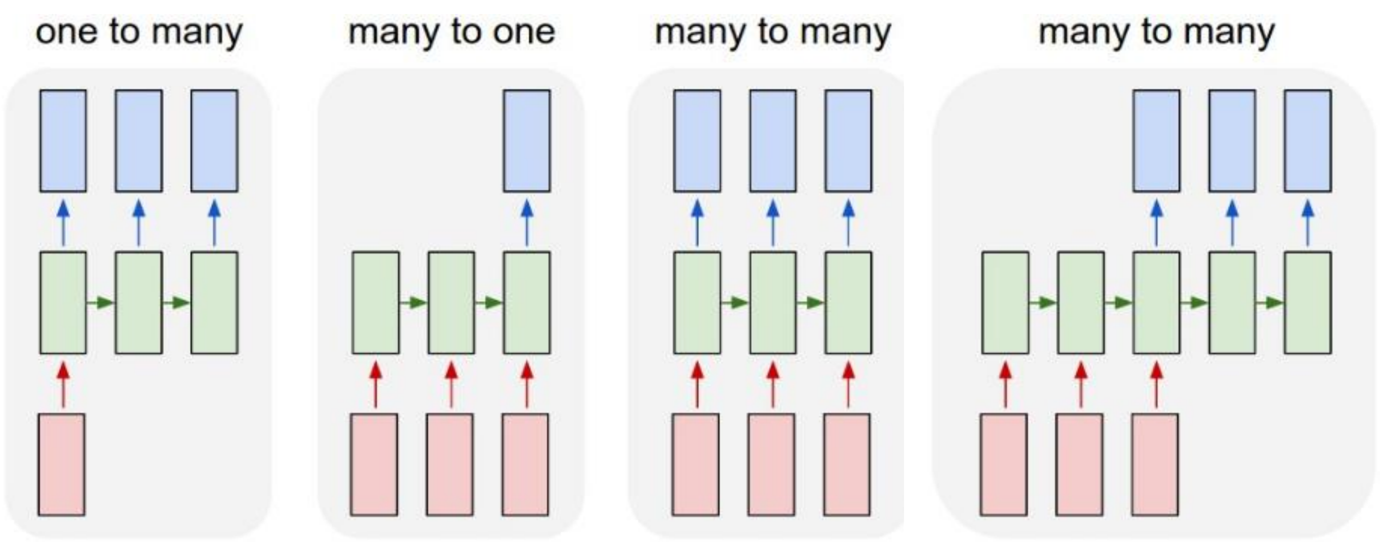


Input and output are both sequences with the same length

RNN can do more than that!

https://www.youtube.com/watch?v=xCGidAeyS4M

Applications

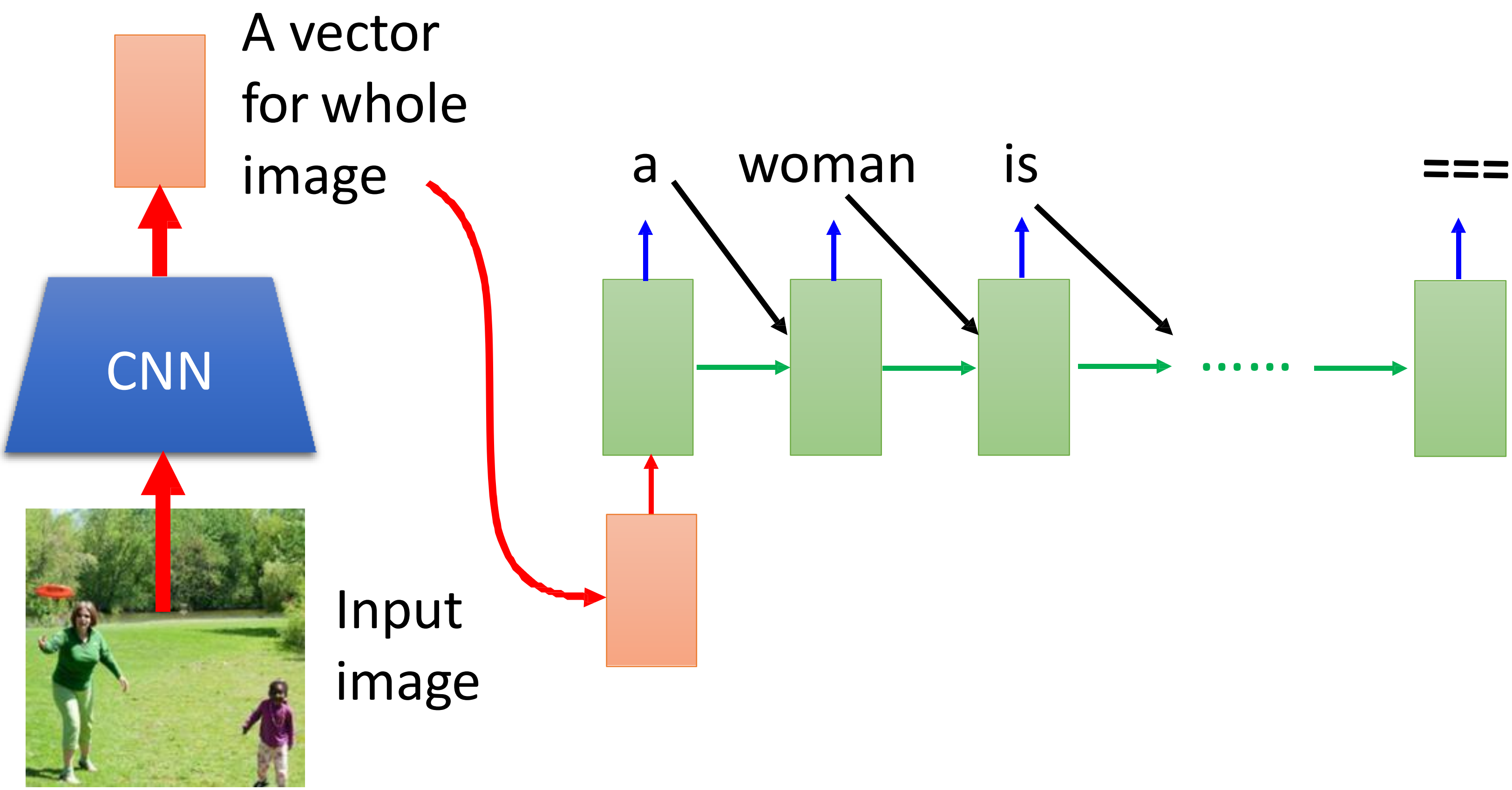


(a) (b) (c) (d)

(a) Sequence output (e.g. image captioning). **(b)** Sequence input (e.g. sentiment analysis). **(c)** Synced sequence input and output (e.g. speech recognition) **(d)** Sequence input and sequence output (e.g. Machine Translation).

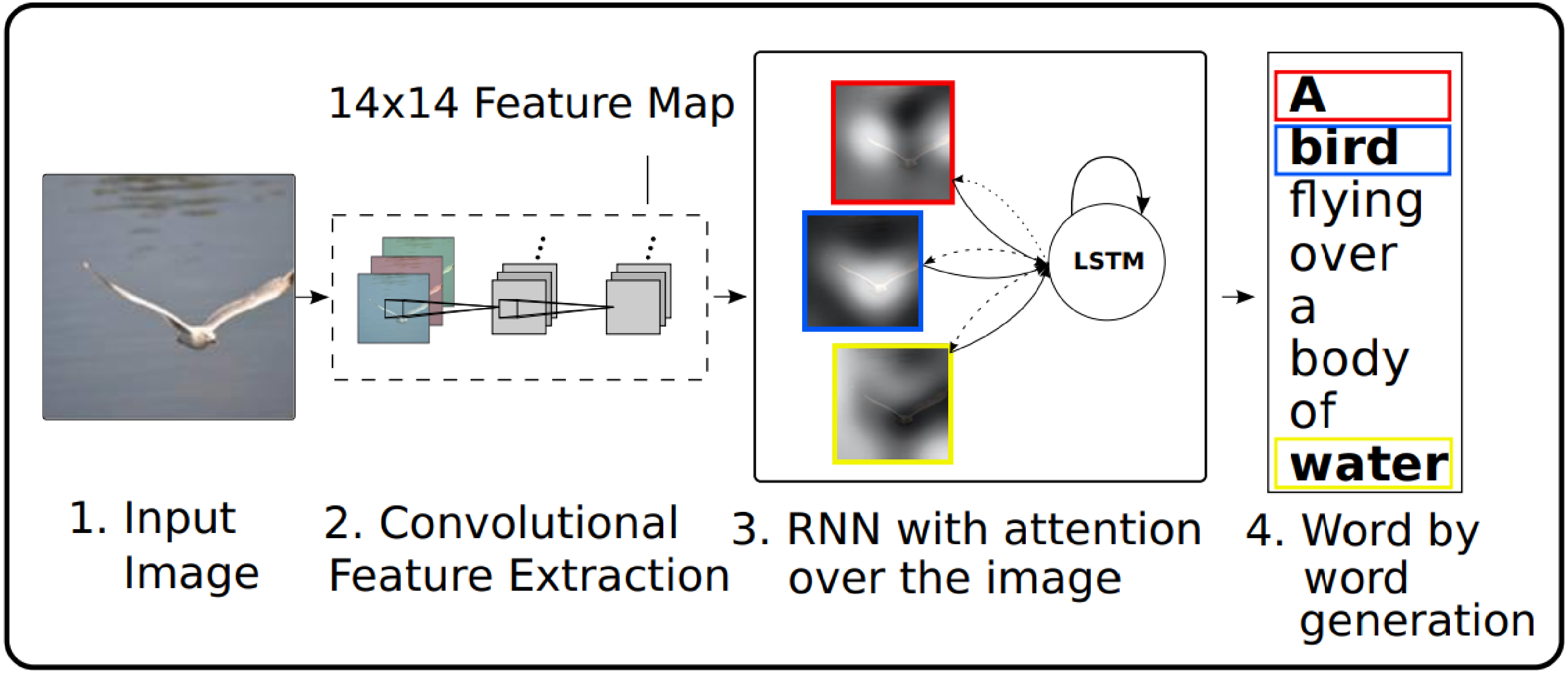
One to Many (Image Captioning)

- Input an image, but output a sequence of words



One to Many (Image Captioning)

- Input an image, but output a sequence of words



[1] Xu K, Ba J, Kiros R, et al. Show, attend and tell: Neural image caption generation with visual attention. ICML 2015.

One to Many (Image Captioning)

- Reference paper list:

[1] Xu K, Ba J, Kiros R, et al. Show, attend and tell: Neural image caption generation with visual attention. ICML 2015.

[2] Karpathy A, Fei-Fei L. Deep visual-semantic alignments for generating image descriptions. CVPR 2015.

[3] You Q, Jin H, Wang Z, et al. Image captioning with semantic attention. CVPR 2016.

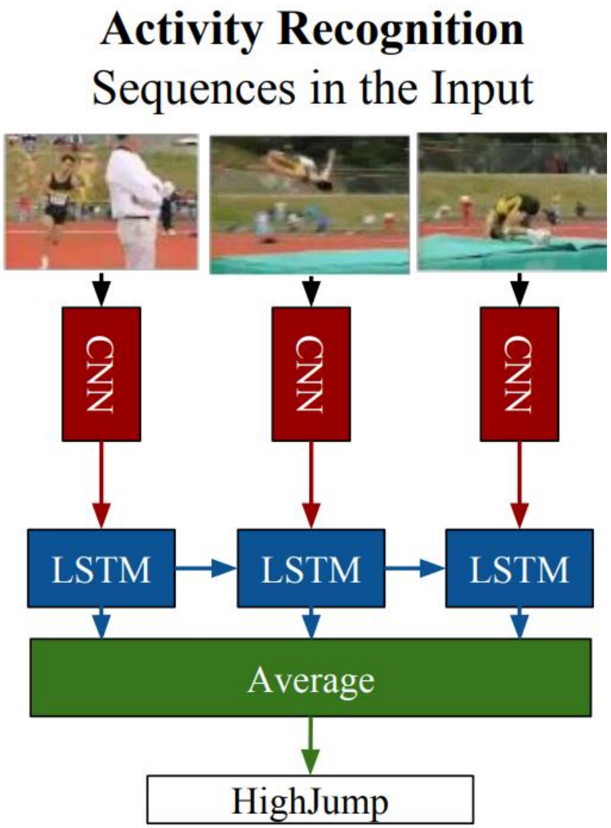
[4] Gan Z, Gan C, He X, et al. Semantic compositional networks for visual captioning. CVPR 2017.

[5] Anderson P, He X, Buehler C, et al. Bottom-up and top-down attention for image captioning and visual question answering. CVPR 2018.

[6] Dai B, Fidler S, Lin D. A neural compositional paradigm for image captioning. NIPS 2018.

Many to One (Video Classification)

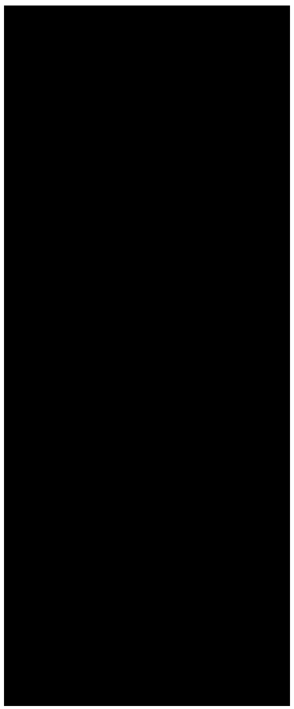
- Input an video, but output a class score



[1] Donahue J, Anne Hendricks L, Guadarrama S, et al. Long-term recurrent convolutional networks for visual recognition and description. CVPR 2015.

Many to One (Video Classification)

- Input an video, but output class scores



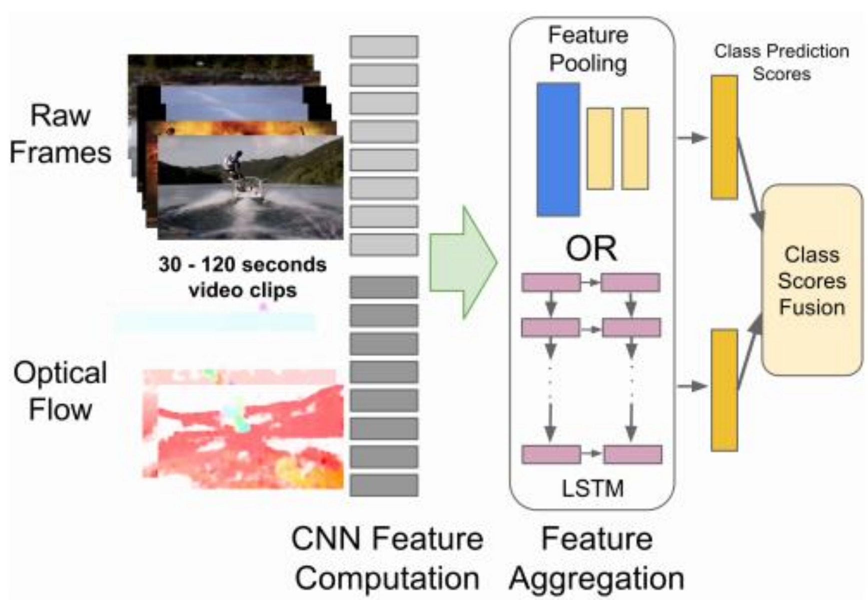


Figure 1: Overview of our approach.

[1] Yue-Hei Ng J, Hausknecht M, Vijayanarasimhan S, et al. Beyond short snippets: Deep networks for video classification. CVPR 2015.

Many to One (Video Classification)

- Reference paper list:

[1] Donahue J, Anne Hendricks L, Guadarrama S, et al. Long-term recurrent convolutional networks for visual recognition and description. CVPR 2015.

[2] Yue-Hei Ng J, Hausknecht M, Vijayanarasimhan S, et al. Beyond short snippets: Deep networks for video classification. CVPR 2015.

[3] Simonyan K, Zisserman A. Two-stream convolutional networks for action recognition. NIPS 2015.

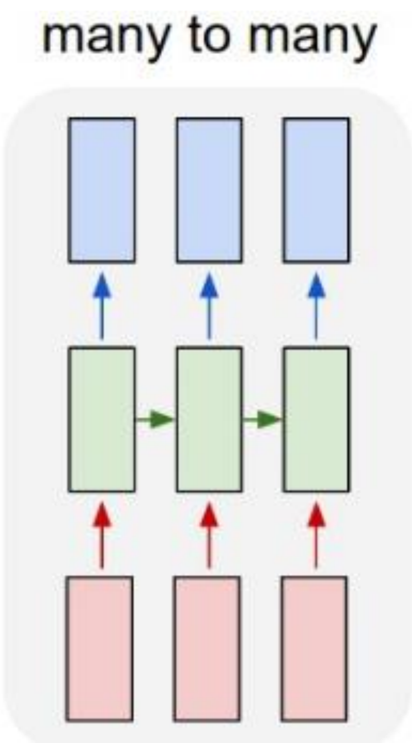
[4] Feichtenhofer C, Pinz A, Zisserman A. Convolutional two-stream network fusion for video action recognition. CVPR 2016.

[5] Wang L, Xiong Y, Wang Z, et al. Temporal segment networks: Towards good practices for deep action recognition. ECCV 2016.

[6] Xu H, Das A, Saenko K. R-c3d: Region convolutional 3d network for temporal activity detection. ICCV 2017.

Many to Many (Synced)

- Both input and output are sequences, ***Synchronizing in temporal.***
- Output sequence is shorter or the same length as input sequence.



Same length: slot filling

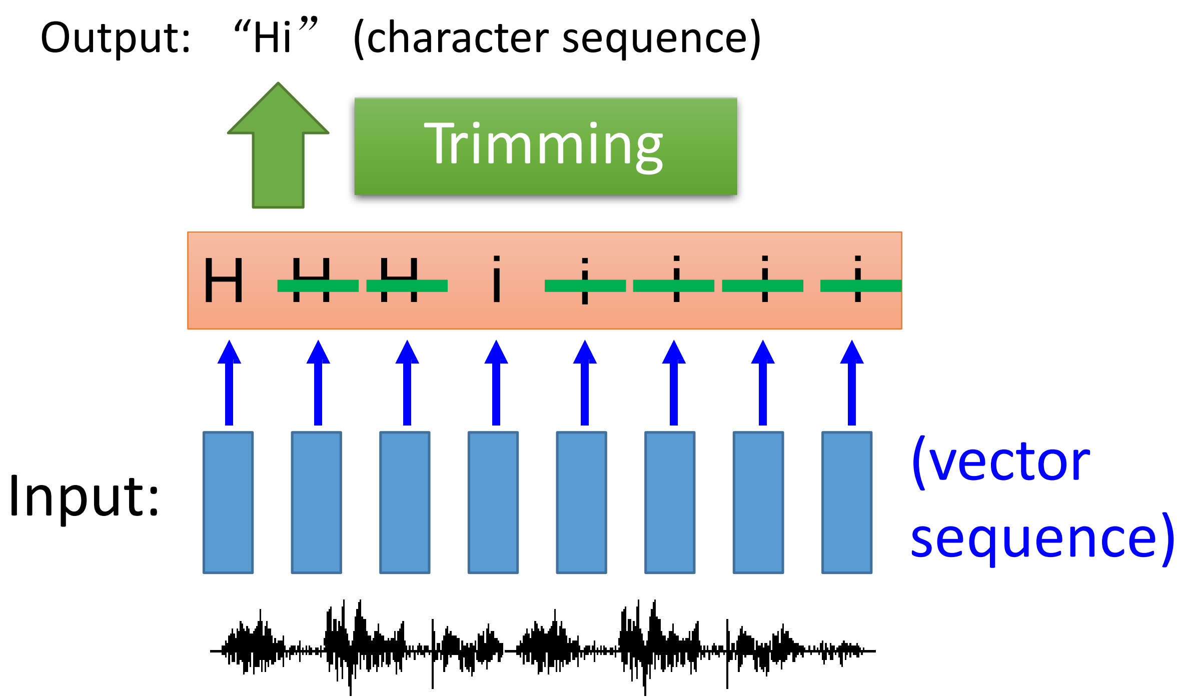
different length: speech recognition

Many to Many (Speech Recognition)

- Both input and output are both sequences, ***but the output is shorter.***

Problem?

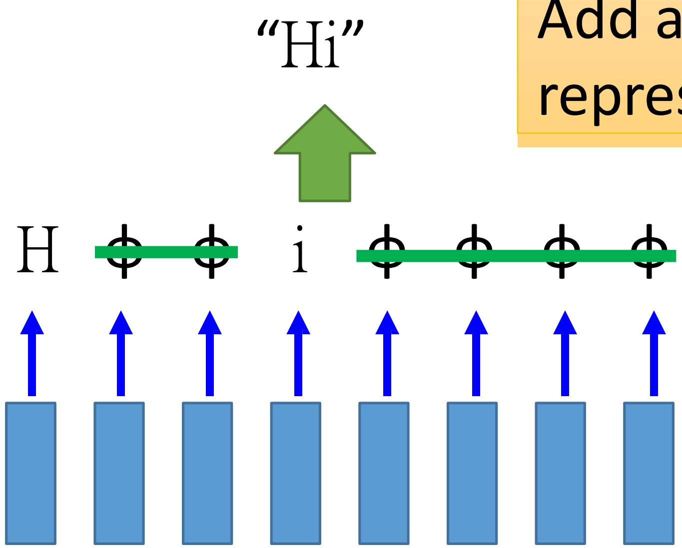
"Helo", Why can't it be "Hello"



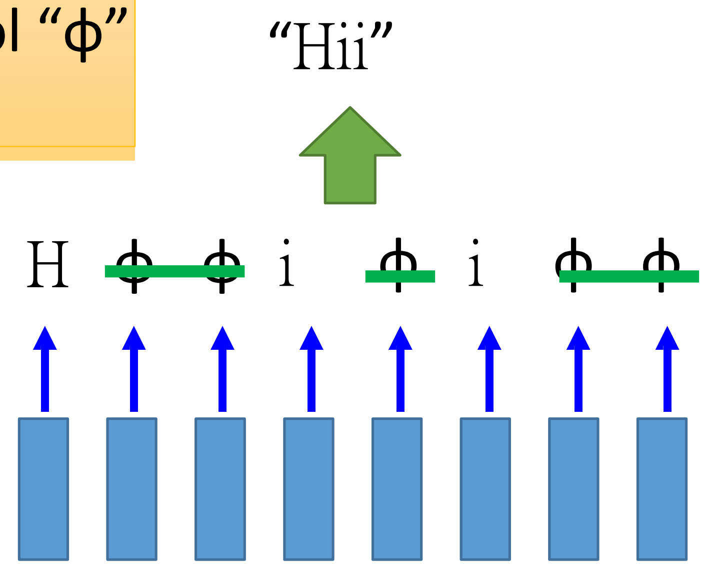
https://www.youtube.com/watch?v=xCGidAeyS4M

Many to Many (Speech Recognition)

- Both input and output are both sequences, ***but the output is shorter.***
- Connectionist Temporal Classification (CTC)



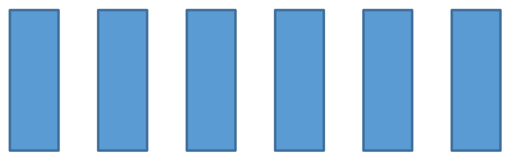
Add an extra symbol “ϕ” representing “null”



[1] Graves A, Fernández S, Gomez F, et al. Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks. ICML 2006.

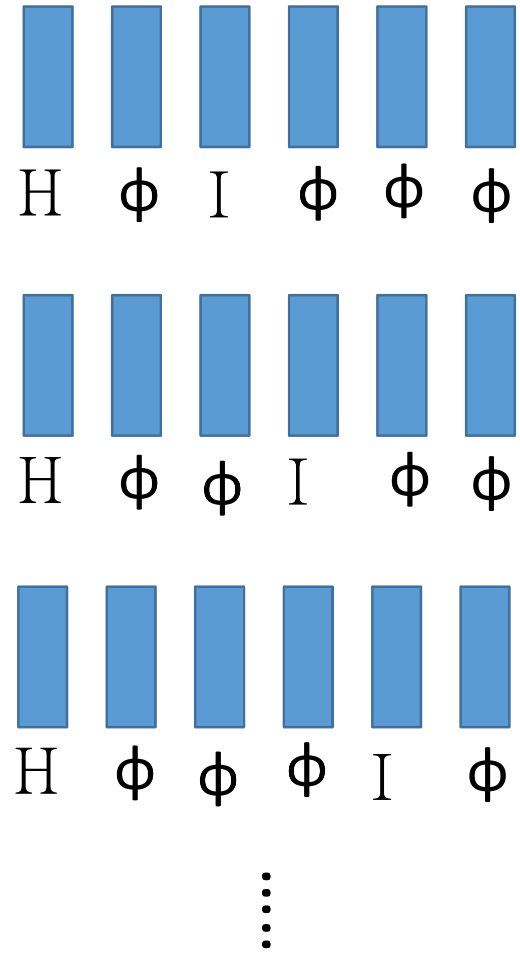
Many to Many (Speech Recognition)

- CTC: Training

Acoustic Features: 

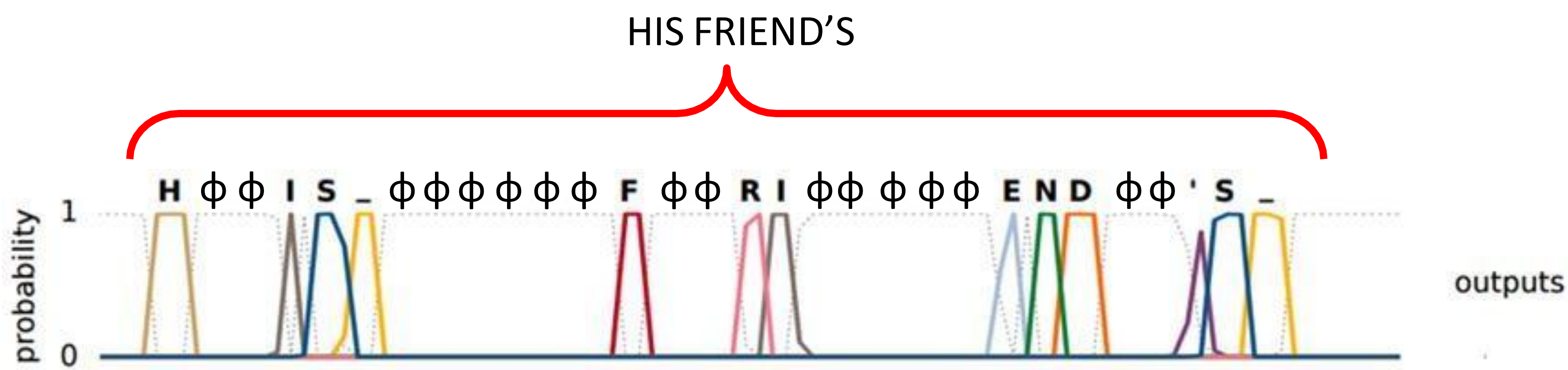
Label: H I

All possible alignments are considered as correct.



Many to Many (Speech Recognition)

- CTC: example



[1] Graves, Alex, and Navdeep Jaitly. "Towards end-to-end speech recognition with recurrent neural networks." ICML 2014.

Many to Many (Lip Reading)

[1] Assael Y M, Shillingford B, Whiteson S, et al. Lipnet: End-to-end sentence-level lipreading. arXiv:1611.01599, 2016.

Many to Many (Lip Reading)

Method	Unseen Speakers CER	Unseen Speakers WER	Overlapped Speakers CER	Overlapped Speakers WER
Hearing-Impaired Person (avg)	–	47.7%	–	–
Baseline-LSTM	38.4%	52.8%	15.2%	26.3%
Baseline-2D	16.2%	26.7%	4.3%	11.6%
Baseline-NoLM	6.7%	13.6%	2.0%	5.6%
LipNet	**6.4%**	**11.4%**	**1.9%**	**4.8%**

[1] Assael Y M, Shillingford B, Whiteson S, et al. Lipnet: End-to-end sentence-level lipreading. arXiv:1611.01599, 2016.

Many to Many (Lip Reading)

- Reference paper list:

[1] Assael Y M, Shillingford B, Whiteson S, et al. Lipnet: End-to-end sentence-level lipreading. arXiv:1611.01599, 2016.

[2] Chung J S, Senior A, Vinyals O, et al. Lip reading sentences in the wild. CVPR 2017.

[3] Stafylakis T, Tzimiropoulos G. Combining residual networks with LSTMs for lipreading. arXiv:1703.04105, 2017.

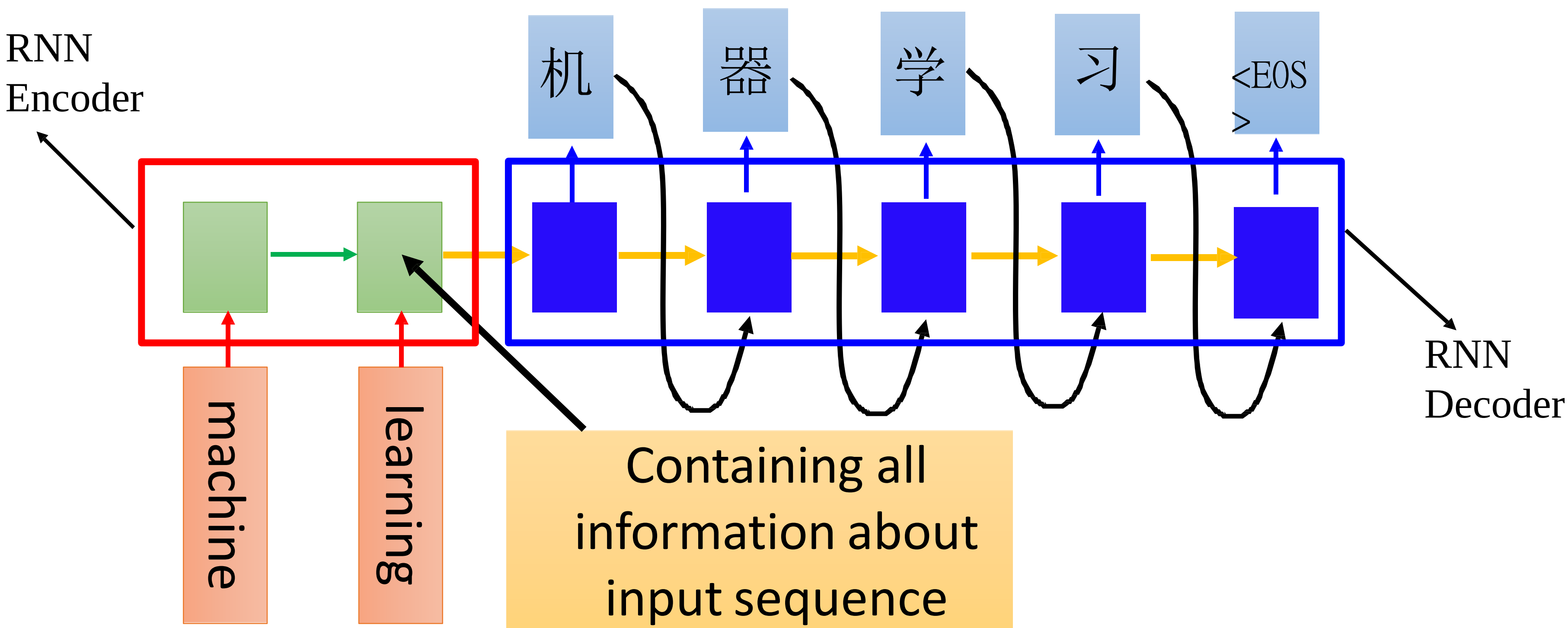
[4] Afouras T, Chung J S, Senior A, et al. Deep audio-visual speech recognition. IEEE TPAMI, 2018.

[5] Shillingford B, Assael Y, Hoffman M W, et al. Large-scale visual speech recognition. arXiv:1807.05162, 2018.

[6] Zhang X, Cheng F, Wang S. Spatio-Temporal Fusion based Convolutional Sequence Learning for Lip Reading. ICCV 2019.

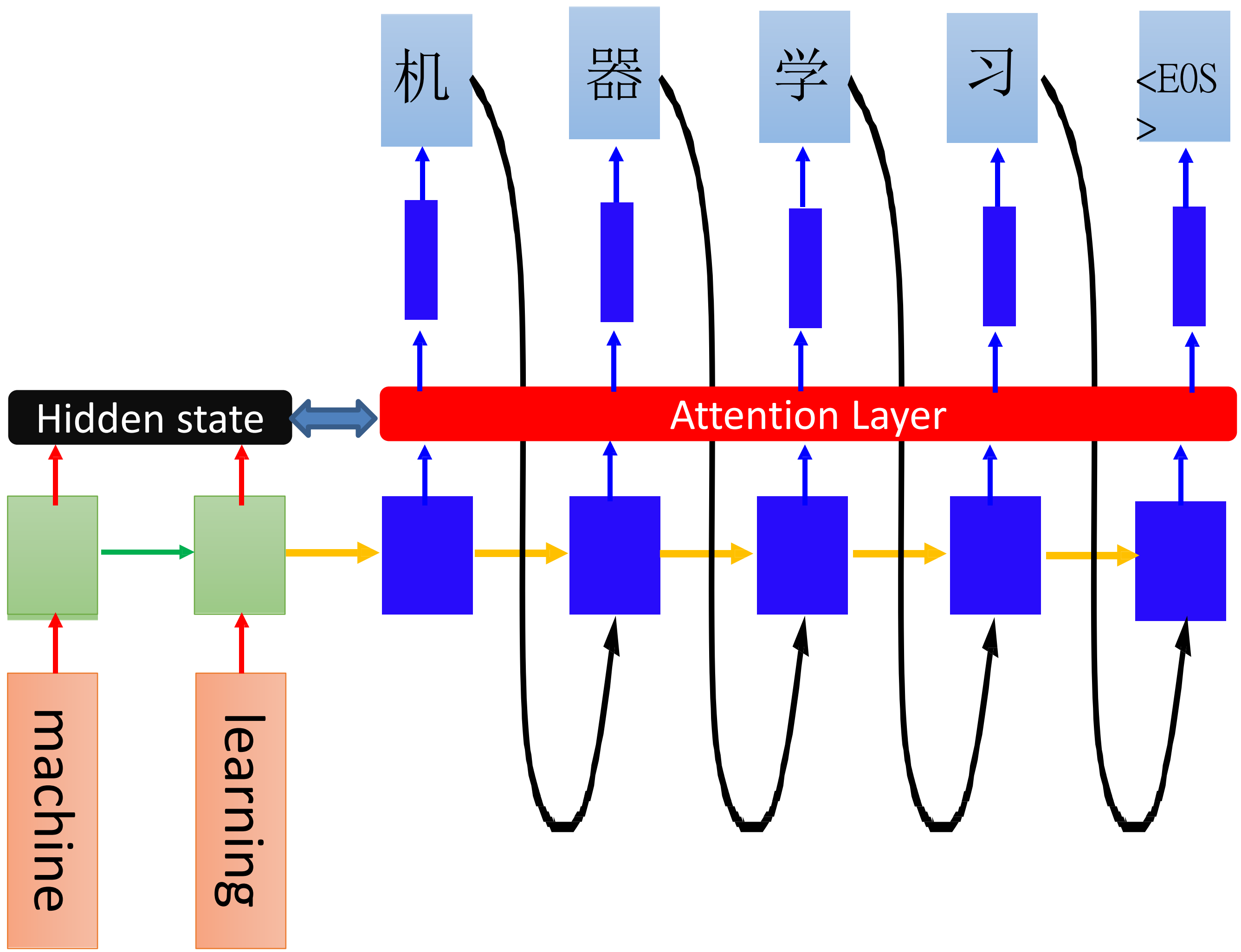
Many to Many (Seq2seq)

- Both input and output are both sequences ***<u>with different lengths</u>***. → ***<u>Sequence to sequence learning</u>***
  - E.g. ***<u>Machine Translation</u>*** (machine learning→机器学习)



[1] Sutskever I, Vinyals O, Le Q V. Sequence to sequence learning with neural networks. Advances in NIPS, 2014.

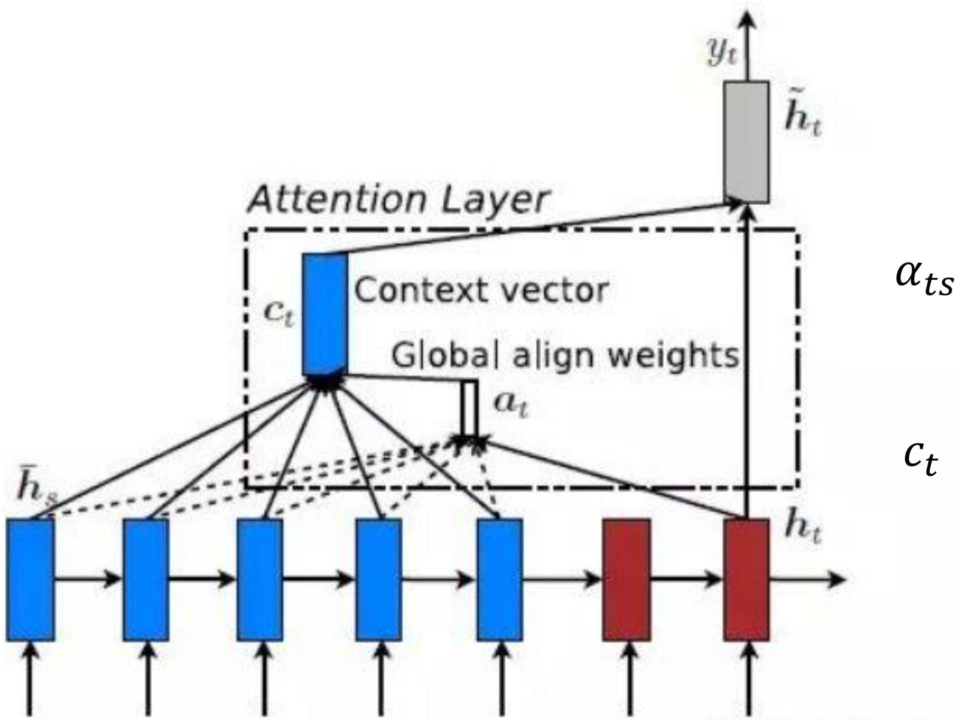
Many to Many (Seq2seq-Att)



[1] Bahdanau D, Cho K, Bengio Y. Neural machine translation by jointly learning to align and translate. arXiv:1409.0473, 2014.

Many to Many (Seq2seq-Att)

- Attention layer:

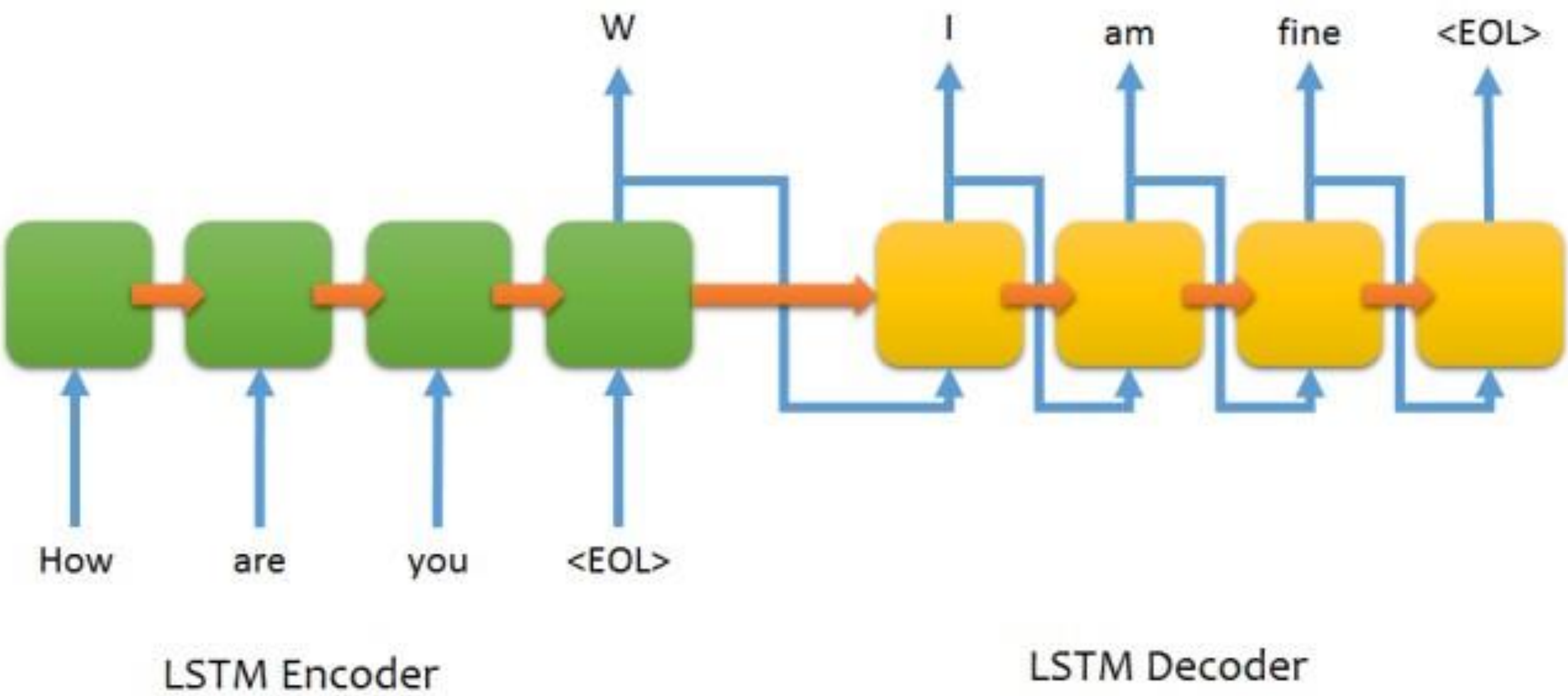


$$\alpha_{ts} = \frac{\exp(score(h_t, \bar{h}_s))}{\sum_{s'=1}^{S} \exp(h_t, \bar{h}_{s\prime})}$$ Attention weight

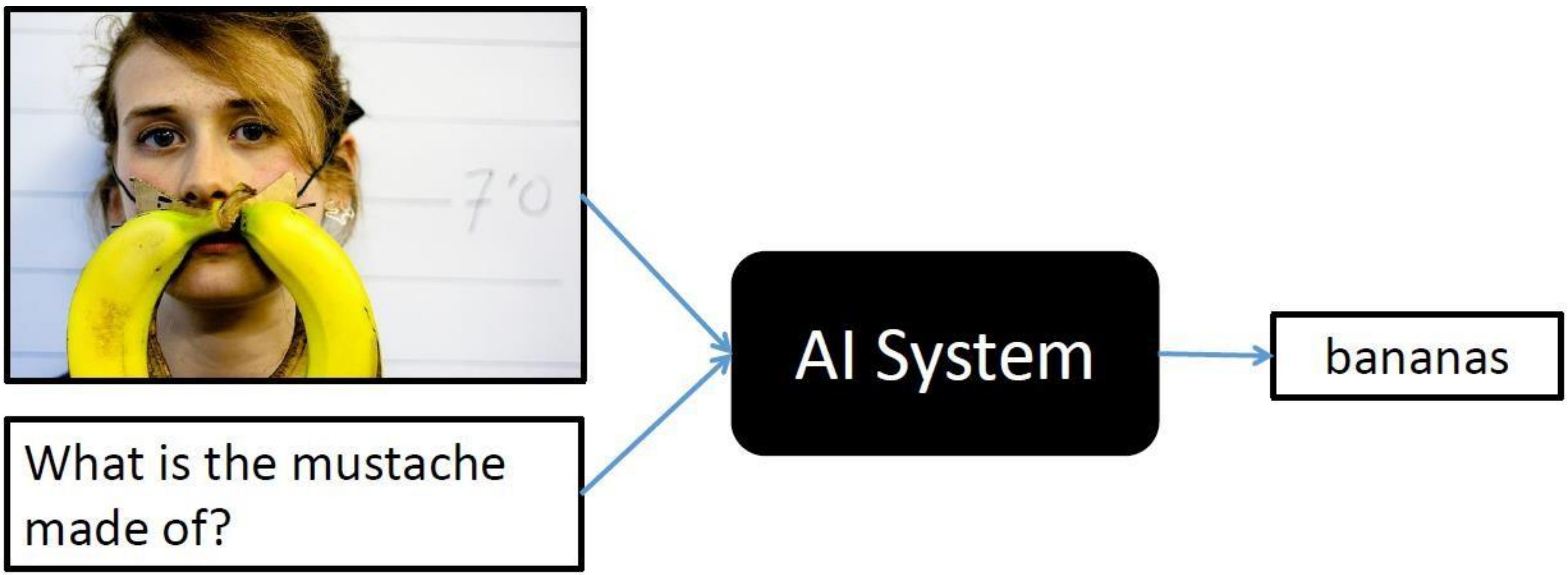
$$c_t = \sum_s \alpha_{ts}\,\bar{h}_s$$ Context vector

[1] Chaudhari S, Polatkan G, Ramanath R, et al. An attentive survey of attention models. arXiv:1904.02874, 2019.

Demo: Chat-bot



Demo:Visual Question Answering



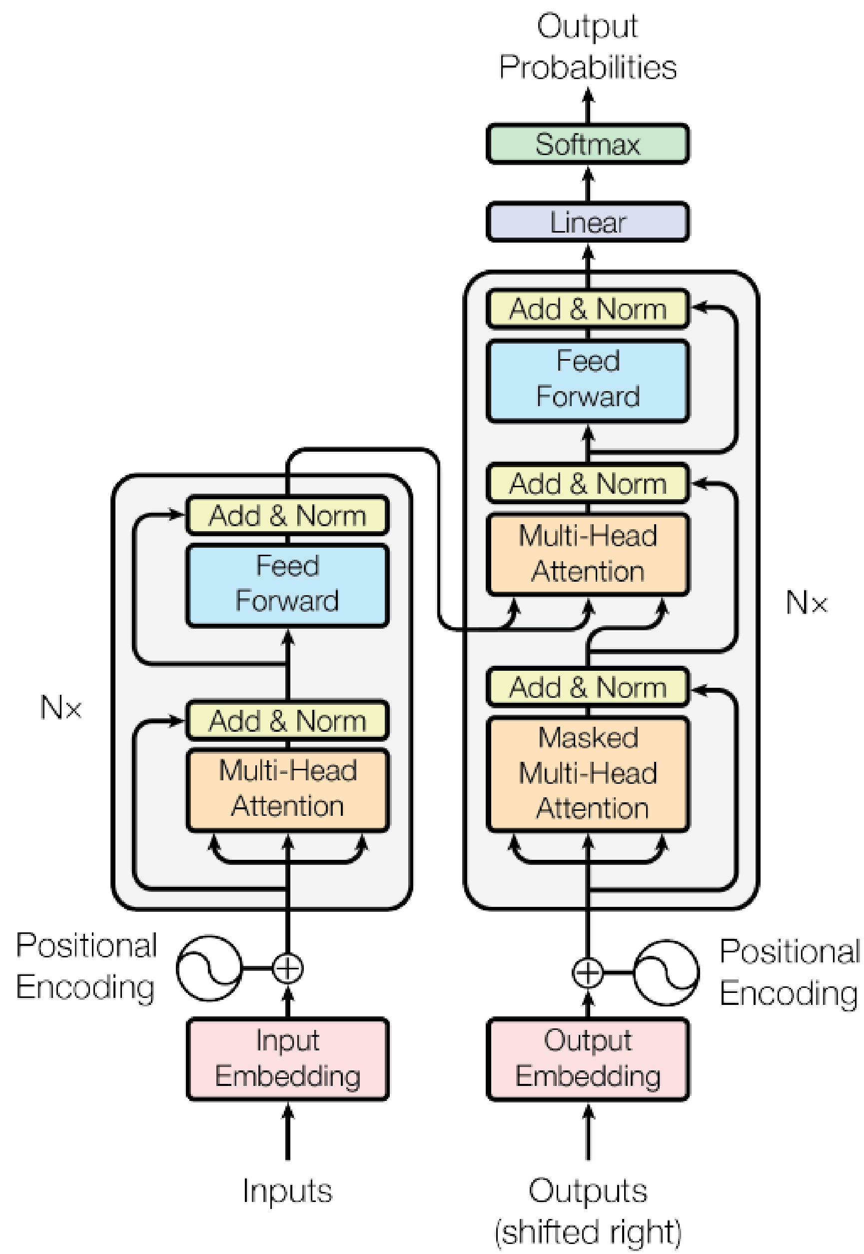
source: http://visualqa.org/

Some drawbacks of RNN

- Memory cost
- Can’t train in parallel
- Short-term dependency

Beyond RNN

Transformer



[1] Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need. NIPS 2017.

Thank you!