RECURRENT NEURAL NETWORK

INT301 Bio-computation, Week 9, 2025

Introduction

- RNN were introduced in the late 80's.
  - Hochreiter discovers the 'vanishing gradients' problem in 1991. （RNN 有在 梯度消失现象）

- Long Short Term Memory published in 1997.
 - LSTM: a recurrent network to overcome this problem.
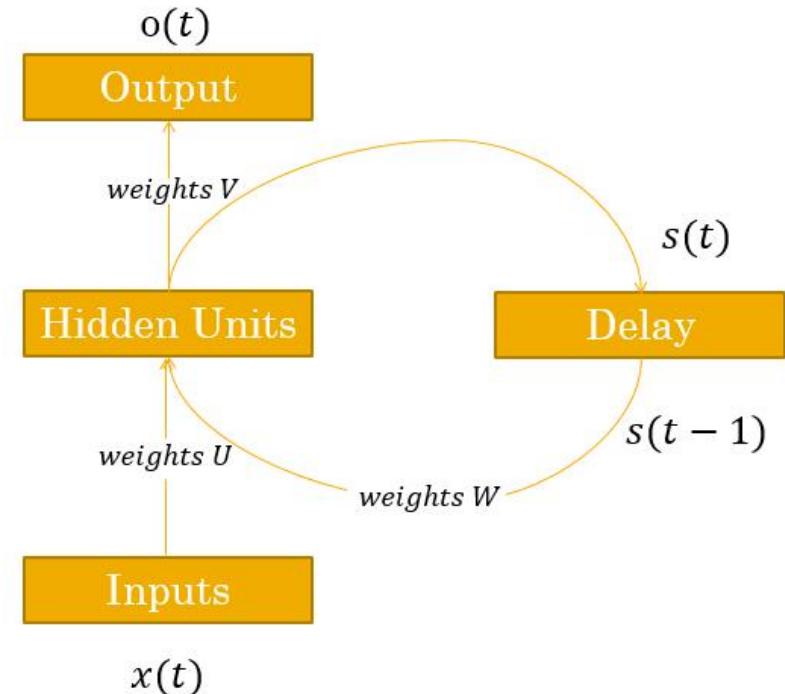
Motivation

- Feed forward networks
  - accept a fixed-sized vector as input
  - produce a fixed-sized vector as output
  - usually fixed amount of computational steps

- Recurrent networks allow us to operate over sequences of vectors
  - the idea behind RNNs is to make use of sequential information, as in a traditional neural network all inputs (and outputs) are assumed independent of each other
  - but for many tasks that's a very bad idea: if you want to predict the next word in a sentence you better know which words came before it
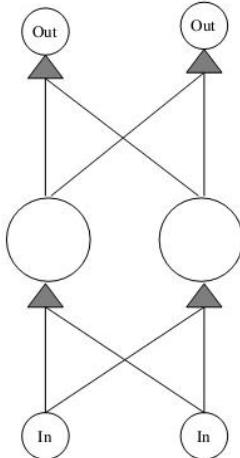
假设独立性但实际上依赖 prev

RNN Architecture
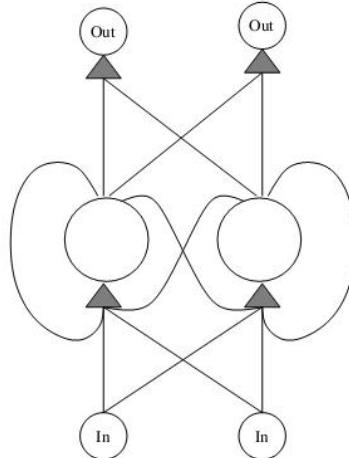
- RNNs are called recurrent because they perform the same task for every element of a sequence, with the output being depended on the previous computations (memory).
  - inputs $x(t)$
  - outputs $y(t)$
  - hidden state $s(t)$: memory of the network
  - a delay unit is introduced to hold activation until they are processed at the next step
- The decision a recurrent net reached at time step t-1 affects the decision it will reach one moment later at time step t
  - two sources of input: the present and the recent past, which combine to determine how they respond to new data
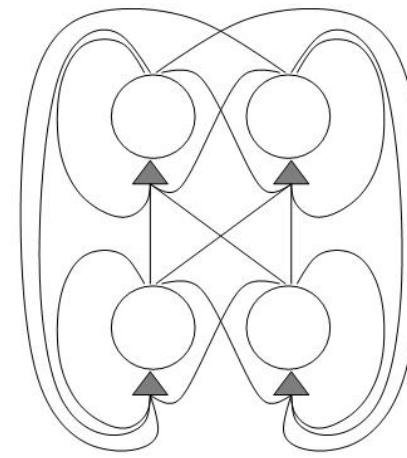
RNN Architecture


feed forward NN   a simple RNN   fully connected RNN
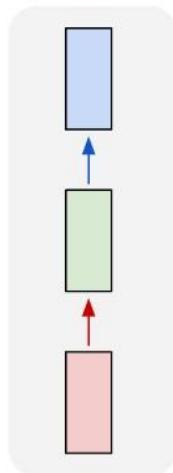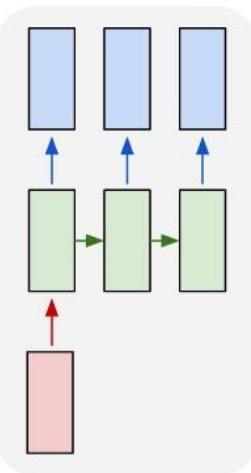
- RNN topologies range from partly recurrent to fully recurrent
  - partly recurrent is a layered network with distinct input and output layers where the recurrence is limited to the hidden layer
 - in fully recurrent networks each node gets inputs from all other nodes

RNN Architecture
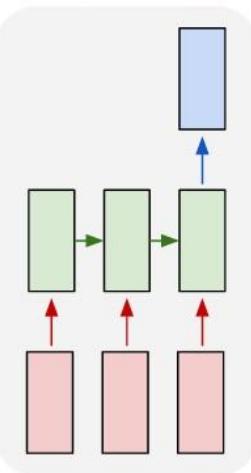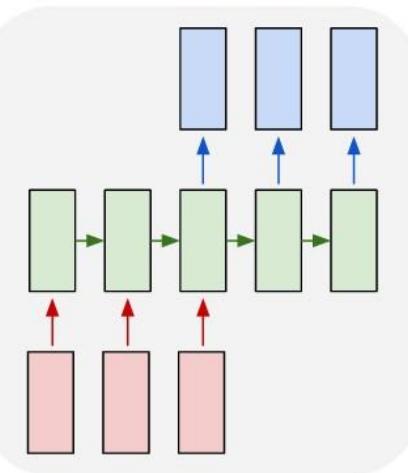
| one to one | one to many | many to one | many to many | many to many |
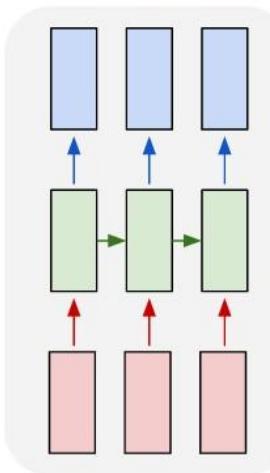
ing → 句子

句子 → 情态

中 → 英

(同步版本)

Vanilla mode of processing without RNN, from fixed-sized input to fixed-sized output (e.g. image classification).

Sequence output (e.g. image captioning takes an image and outputs a sentence of words).

Sequence input (e.g. sentiment analysis where a given sentence is classified as expressing positive or negative sentiment

Sequence input and sequence output (e.g. machine translation: an RNN reads a sentence in English and then outputs a sentence in French).

Synced sequence input and output (e.g. video classification where we wish to label each frame of the video).

RNN Forward Pass

- The network input at time t:

$$a_h(t) = Ux(t) + Ws(t-1)$$

- The activation of hidden unit at time t:
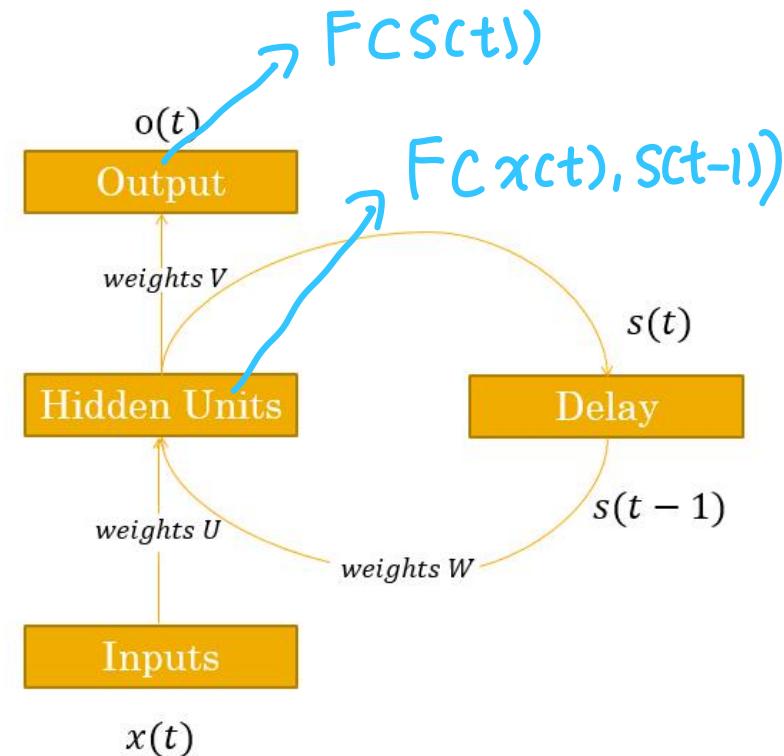
$$s(t) = f_h\big(a_h(t)\big)$$

- The hidden to the output at time t:

$$a_o(t) = Vs(t)$$

- The output of the network at time t is:

$$o(t) = f_o\big(a_o(t)\big)$$

FC S(t))

FC x(t), S(t-1))

$o(t)$

Output

weights V

Hidden Units

weights U

Inputs

$x(t)$

$s(t)$

Delay

$s(t-1)$

weights W
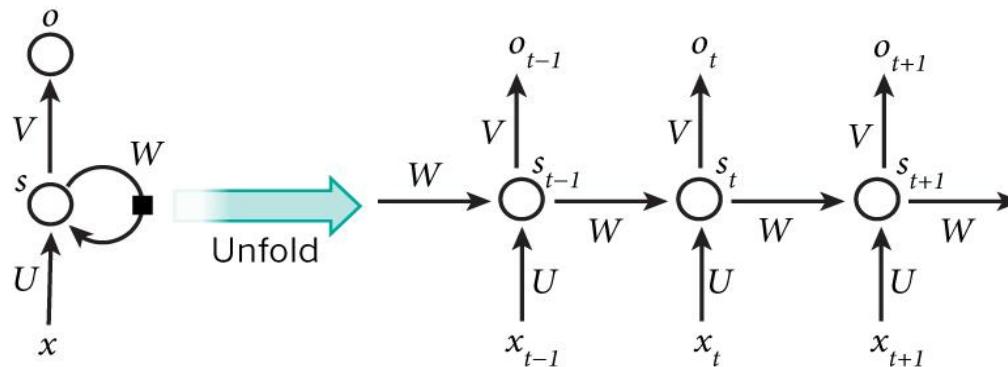
RNN Architecture

- If a network training sequence starts at time t_0 and ends at t_1, one possible choice of the total loss function is the sum over time of the square error function

$$[e(t)]_k = [d(t)]_k - [o(t)]_k \quad \rightarrow 代表\ vector\ (k*dim)$$

↑真实值

$$E(t) = \frac{1}{2} e(t)^T e(t) \quad \rightarrow 时间\ t\ 下预测值$$

$$E_{total} = \sum_{t=t_0}^{t_1} E(t)$$
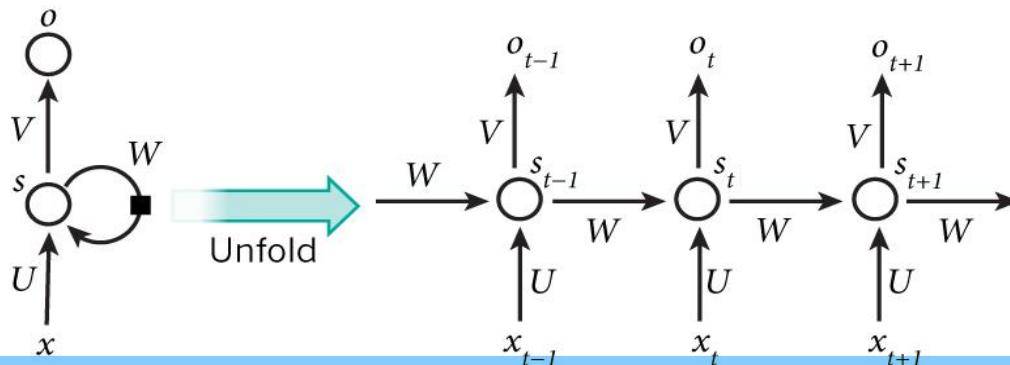
完整 t_0 到 t_1 的 loss

RNN Architecture

- The recurrent network can be converted into a feed forward network by unfolding over time
- The above diagram shows a RNN being unrolled (or unfolded) into a full network. By unrolling we simply mean that we write out the network for the complete sequence.
  - For example, if the sequence we care about is a sentence of 5 words, the network would be unrolled into a 5-layer neural network, one layer for each word.
- This means all the earlier theory about feed forward networks learning follows through

RNN Architecture
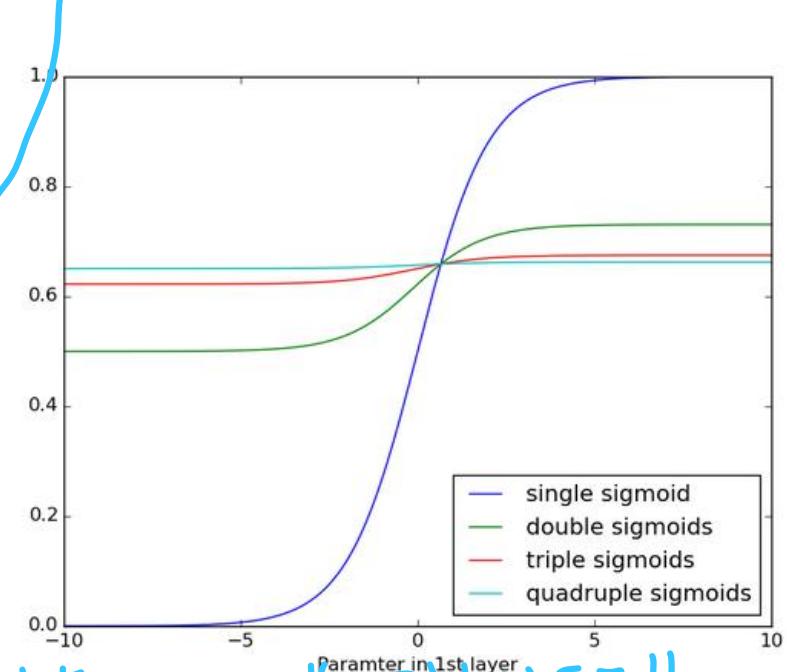
- In RNN errors can be propagated more than 2 layers in order to capture longer history information.
  - This process is usually called unfolding.
  - In an unfolded RNN, recurrent weight is duplicated for arbitrary number of time steps
- The above diagram has outputs at each time step, but depending on the task this may not be necessary.
  - For example: when predicting the sentiment of a sentence we may only care about the final output, not the sentiment after each word. Similarly, we may not need inputs at each time step. The main feature of an RNN is its hidden state, which captures some information about a sequence

Back Propagation Through Time (BPTT)

- BPTT learning algorithm is an extension of standard backpropagation that performs gradients descent on an unfolded network.

- The gradient descent weight updates have contributions from each time step.

- The errors have to be back-propagated through time as well as through the network

- For recurrent networks, the loss function depends on the activation of the hidden layer through its influence on the output layer **and** through its influence on the hidden layer at the next step.

Vanishing Gradients 梯度消失
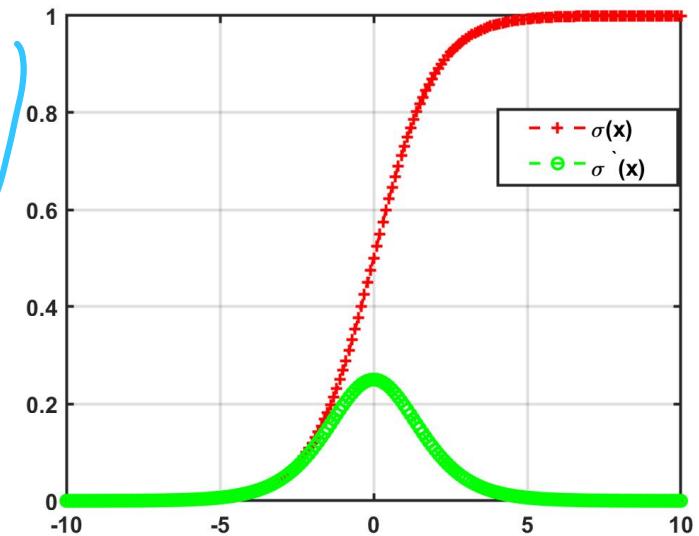
- RNNs trained with BPTT have difficulties learning long-term dependencies (e.g. dependencies between steps that are far apart) due to what is called the vanishing gradients problem.

  t很长 → RNN 问题

- The figure shows the effects of applying a sigmoid function over and over again. The data is flattened until, for large stretches, it has no detectable slope.

  →反复用 6(·). 数据被拉平了 !!

- Vanishing gradients can become too small for computers to work with or for networks to learn.

Legend:
- single sigmoid
- double sigmoids
- triple sigmoids
- quadruple sigmoids

Paramter in 1st layer
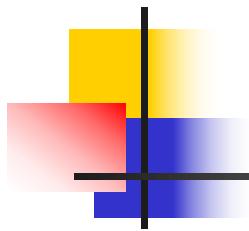
Vanishing Gradients

- Sigmoid functions have derivatives of 0 at both ends

- When this happens we say the corresponding neurons are saturated
  - they have a zero gradient and drive other gradients in previous layers towards 0.

- Gradient contributions from "far away" steps become zero, and the state at those steps doesn't contribute to what you are learning
 - forget old information: end up without learning long-range dependencies.

神经元处于饱和状态
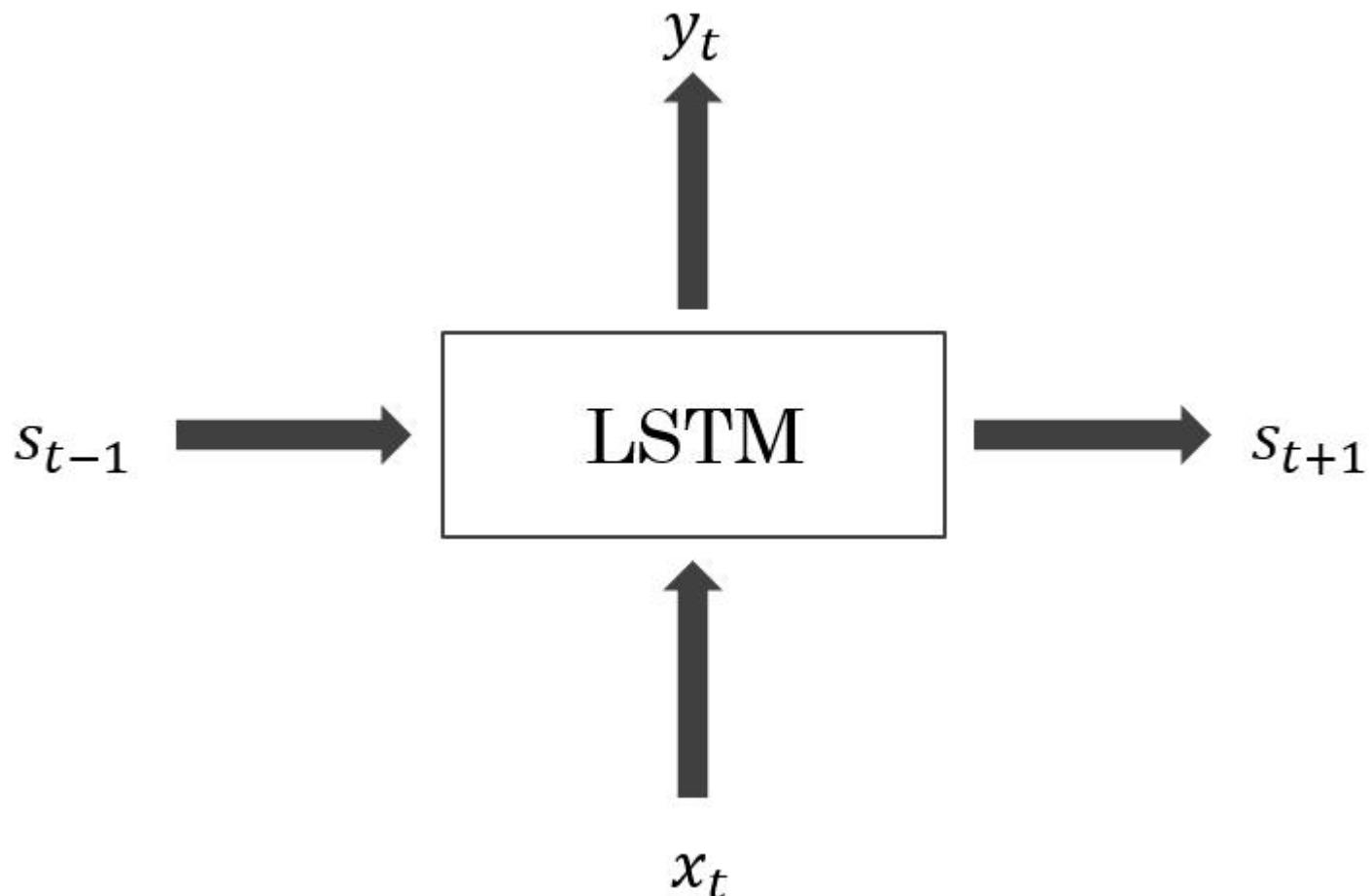
LSTM - introduction
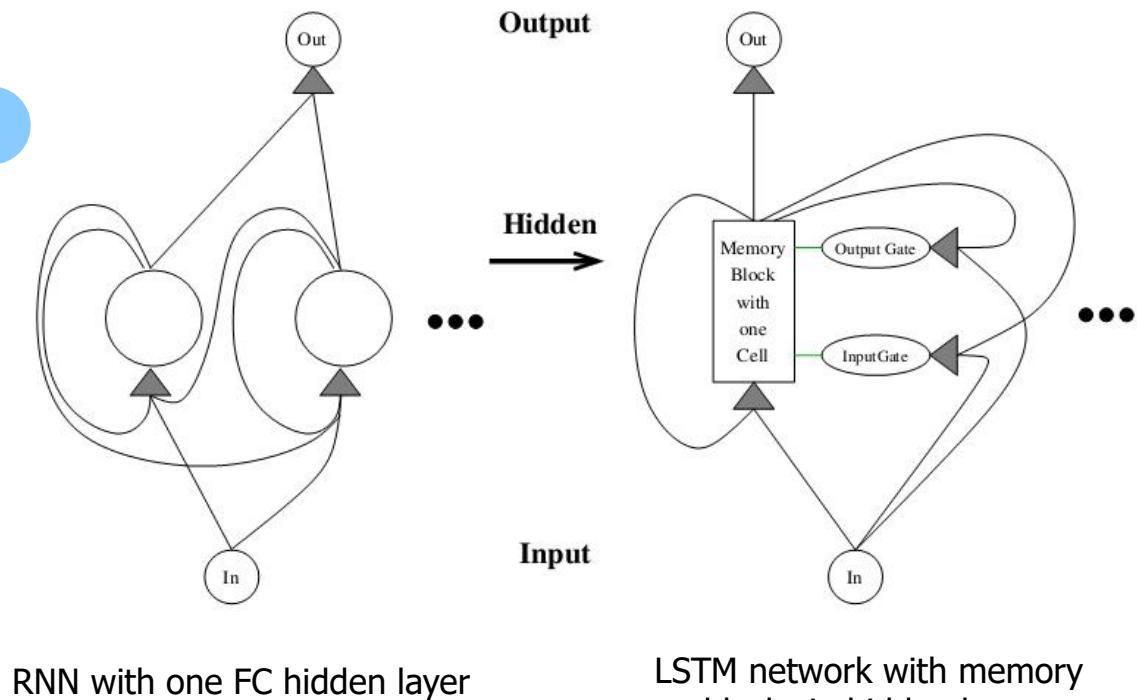
- LSTM was invented to solve the vanishing gradients problem.

- LSTM maintain a more constant error flow in the back propagation process.

- LSTM can handle large sequences that are linked remotely.

LSTM – same idea as RNN

$$y_t$$

$$s_{t-1} \rightarrow \boxed{\text{LSTM}} \rightarrow s_{t+1}$$

$$x_t$$
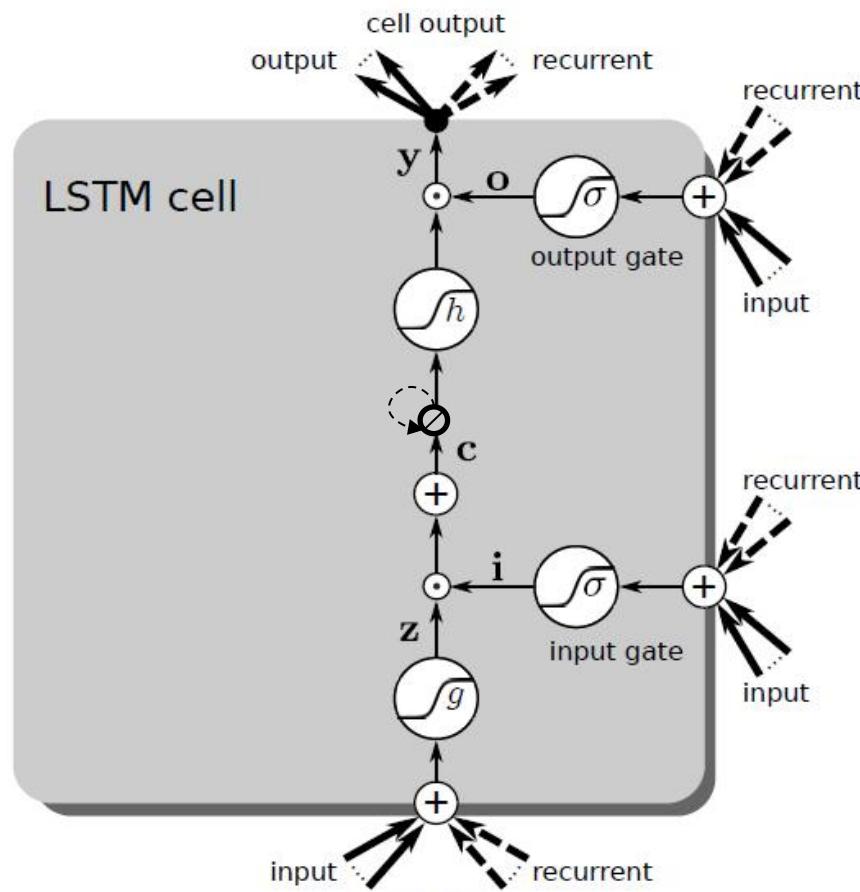
LSTM Architecture

- The basic unit in the hidden layer of an LSTM network is a memory block, it replaces the hidden unit in a traditional RNN.

- A memory block contains one or more memory cell and a pair of adaptive multiplicative gating units which gates input and output to all cells in the block.

- Memory blocks allow cells to share the same gates thus reducing the number

RNN with one FC hidden layer

LSTM network with memory blocks in hidden layer (only one is shown)
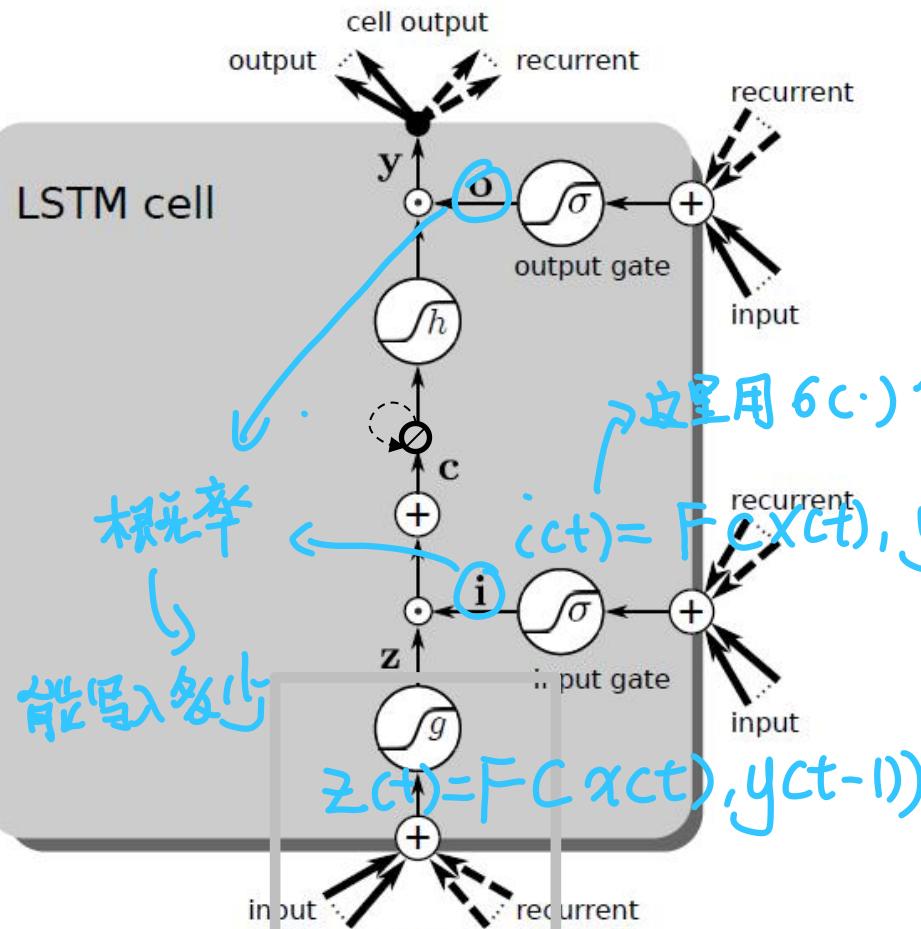
LSTM Architecture - overview

LSTM Architecture – input
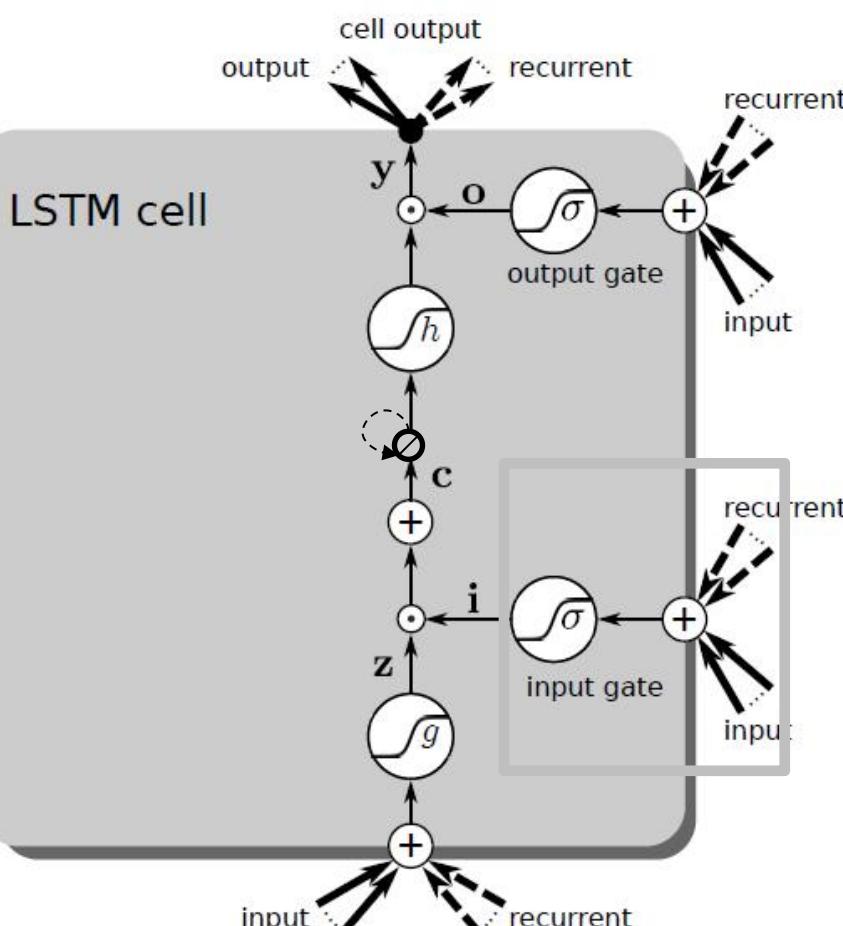
- The layer decides which information from the input should be forwarded into the LSTM cell.

- By multiplying y(t−1) with the recurrent weight matrix Rz, the layer decides which information from the previous time step should be forwarded into the LSTM cell

$$a_z(t) = W_z x(t) + R_z y(t-1)$$
$$z(t) = g(a_z(t))$$

Handwritten annotations:

概率
能写入多少
z(t)=FC(x(t), y(t-1))
(c(t)= FC(x(t), y(t-1))
立望用 6(·) 代表权限

LSTM Architecture — input gate

- The input gate controls write accesses to memory cells.
- This is realized by using the result of the squashing function σ as a factor which will be later multiplied by the squashed cell input z(t).
- The range of function σ is (0,1) and can be interpreted in case of 0 as write access denied and in case of 1 as write access granted.
- Note that all values in between are also possible.
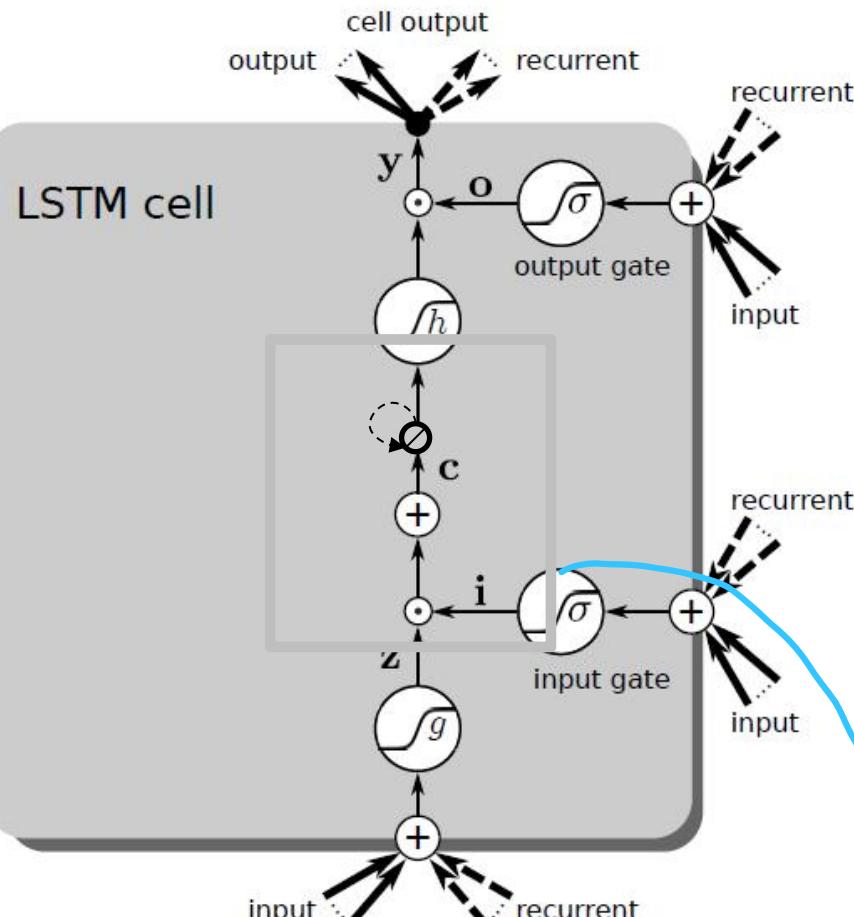
$$a_{in}(t) = W_{in}x(t) + R_{in}y(t-1)$$

LSTM Architecture – CEC

确实像Resnet！

（C记忆细胞），递归权重为1，不受非线性影响

↓

误差反向传播不会衰减

cell output

output · · · recurrent

recurrent

LSTM cell

y

o

σ

output gate

input

∫h

c

+

recurrent

i

σ

input gate

input

z

∫g

+

input · · · recurrent
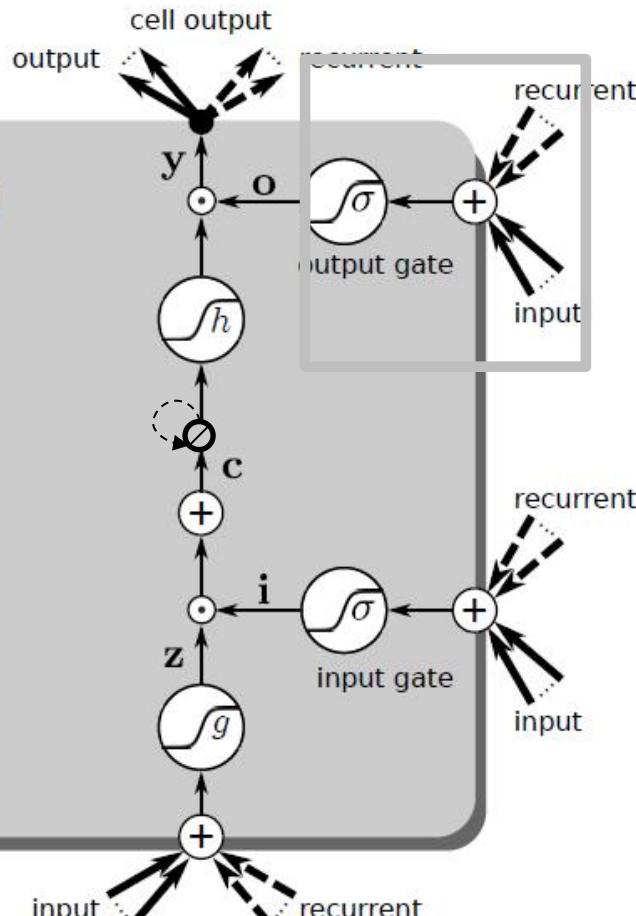
- Each memory cell contains a node with a self-connected recurrent edge of fixed weight one, ensuring that the gradient can pass across many time steps without vanishing – which is called CEC (constant error carousel)

- The CEC solves the vanishing error problem. In the absence of new input or error signals to the cell the CEC local error back flow remains constant, neither growing or decaying.

$$c(t) = z(t) \odot i(t) + c(t-1)$$

⊙ is the Hadamard product

LSTM Architecture — output gate
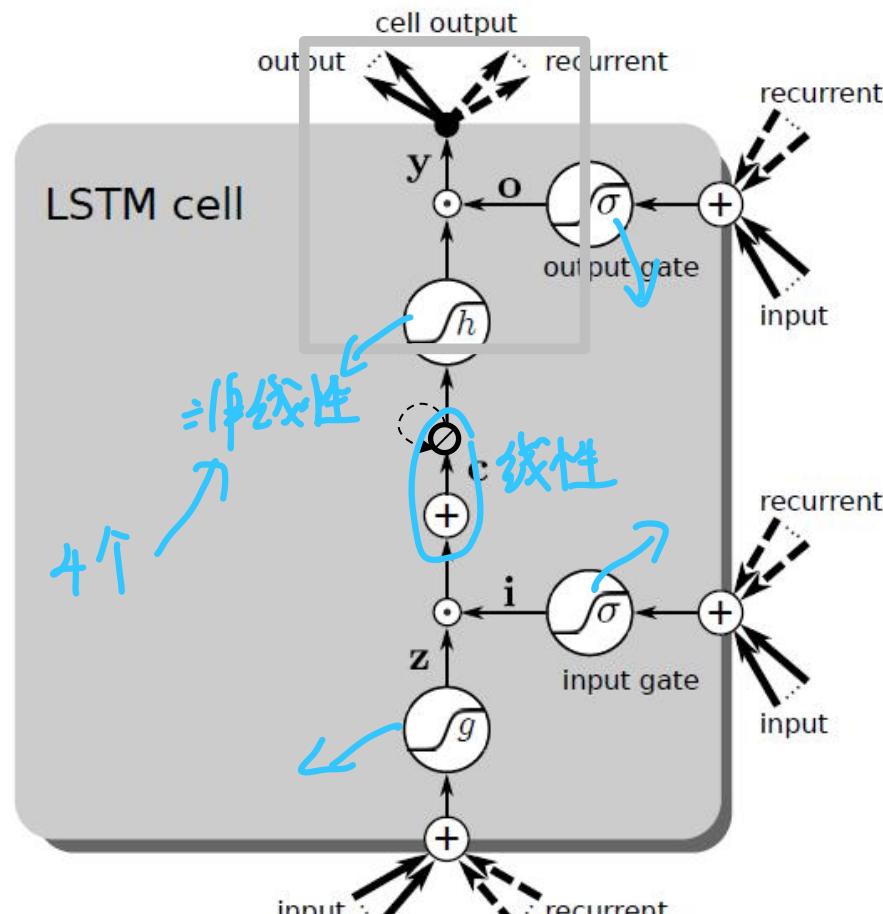
- The output gate controls read accesses from memory cells.

- This is realized by using the result of the squashing function σ as a factor which will be later multiplied by the squashed cell content h(c(t)).

- The range of function σ is (0,1) and can be interpreted in case of 0 as read access denied and in case of 1 as read access granted.

- Note that all values in between are also possible.

$$a_{out}(t) = W_{out}x(t) + R_{out}y(t-1)$$
$$o(t) = \sigma\big(a_{out}(t)\big)$$

LSTM Architecture – output
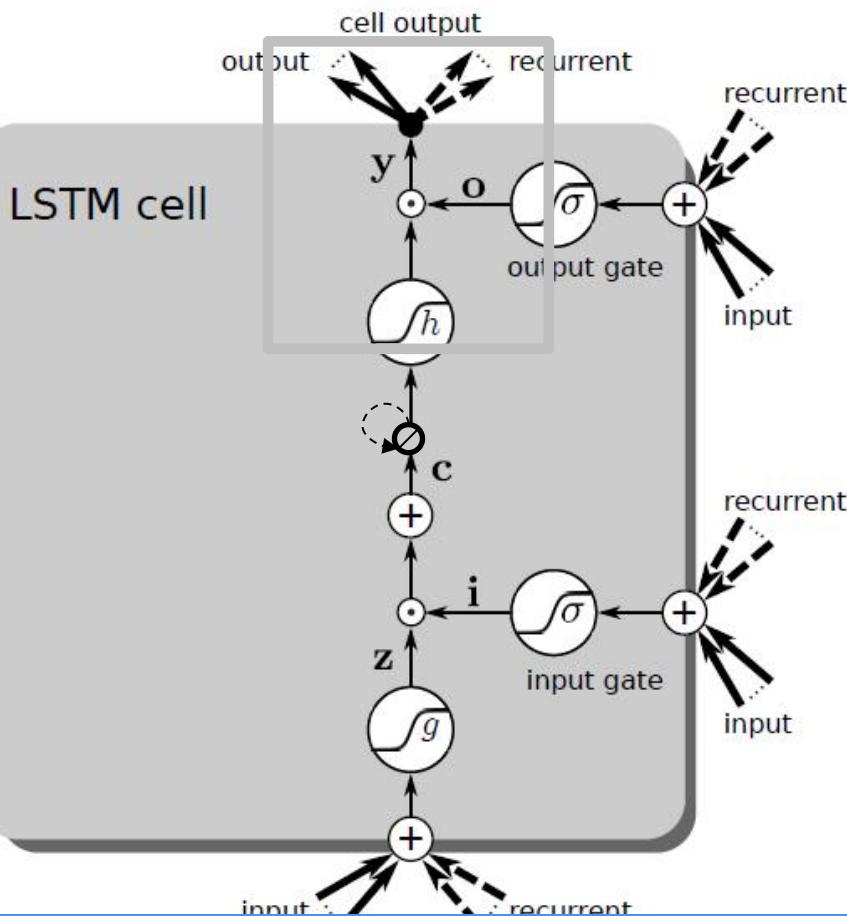


- The value stored in the cell c(t) is squashed by function h, and whether this information gets output is decided by the output gate via o(t).

$$y(t) = h\big(c(t)\big) \odot o(t)$$

$\odot$ is the Hadamard product

LSTM Forward Pass

- The cell state c is updated based on its current state and 3 inputs: a_z , a_{in} , a_{out}
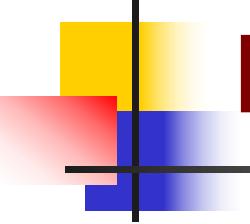
$$a_z(t) = W_z x(t) + R_z(y(t-1)), z(t) = g(a_z(t))$$
$$a_{in}(t) = W_{in} x(t) + R_{in}(y(t-1)), i(t) = \sigma(a_{in}(t))$$
$$c(t) = z(t) \odot i(t) + c(t-1)$$
$$a_{out}(t) = W_{out} x(t) + R_{out}(y(t-1)), o(t) = \sigma(a_{out}(t))$$
$$y(t) = h(c(t)) \odot o(t)$$

LSTM Backward Pass

- Errors arriving at cell outputs are propagated to the CEC

- Errors can stay for a long time inside the CEC

- This ensures non-decaying error

- Can bridge time lags between input events and target signals

Advantages of LSTM

↗CEC ② 循环由于线性向过去t更新元衰减！
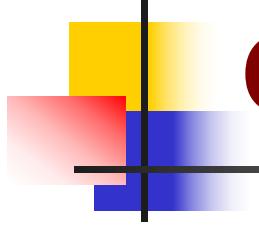
- Non-decaying error backpropagation.

- For long time lag problems, LSTM can handle noise and continuous values.

- No parameter fine tuning.

- Memory for long time periods

Limitations of LSTM
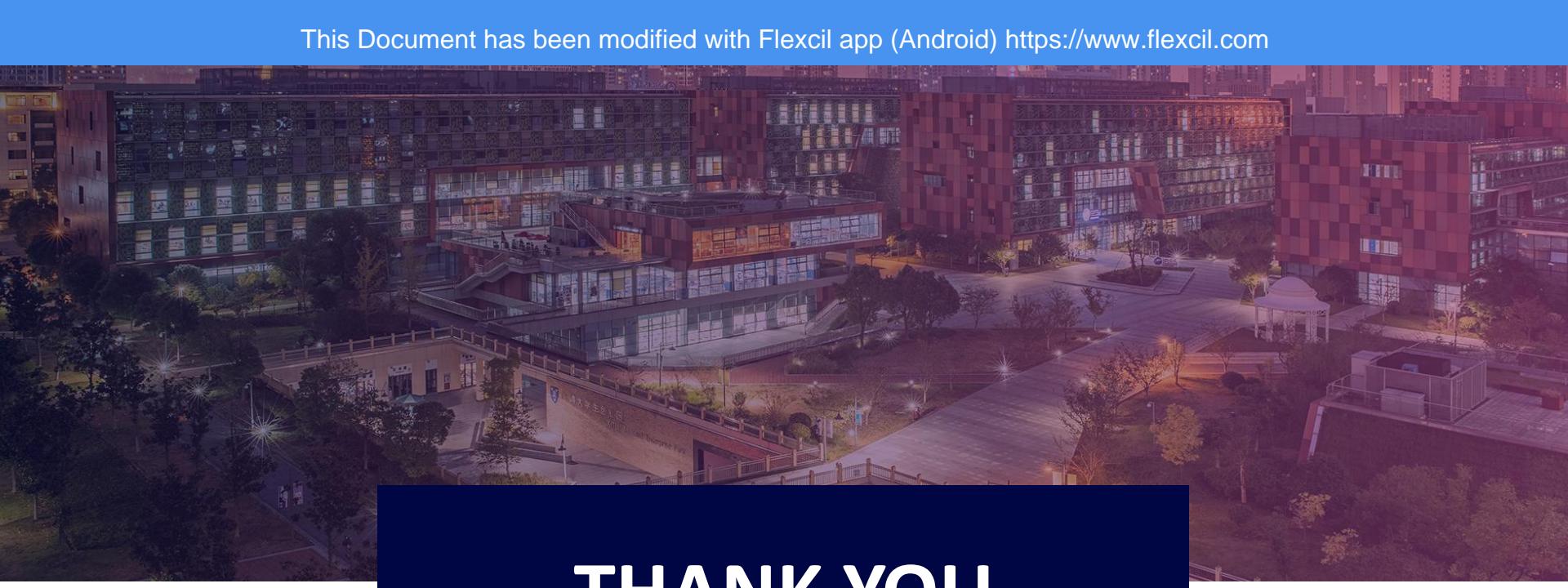
$$c(t) = z(t) \odot i(t) + c(t-1) \rightarrow \text{ grows linearly}$$

- LSTM allows information to be stored across arbitrary time lags and error signals to be carried far back in time.

- This potential strength however, can contribute to a weakness: the cells states c(t) often tend to grow linearly during the presentation of a time series.

- If we present a continuous input stream, the cell state may grow in unbounded fashion causing saturation of the output squashing function h(c(t)).

h(c(t)) 太大出问题

Conclusions

- RNNs - self connected networks
- Vanishing gradients and long memory problems
- LSTM - solves the vanishing gradients and the long memory limitation problem

THANK YOU

VISIT US

WWW.XJTLU.EDU.CN

FOLLOW US

@XJTLU

Xi'an Jiaotong-Liverpool University
西交利物浦大学