Long- short term memory (LSTM) 1997.

GRU: 2016.

1. Long- term dependencies (LTD)

Q: Can this standard RNN achieve the LTD ?

Yes: → in theory

No: → in practice.

↓ If t is large (long chain),

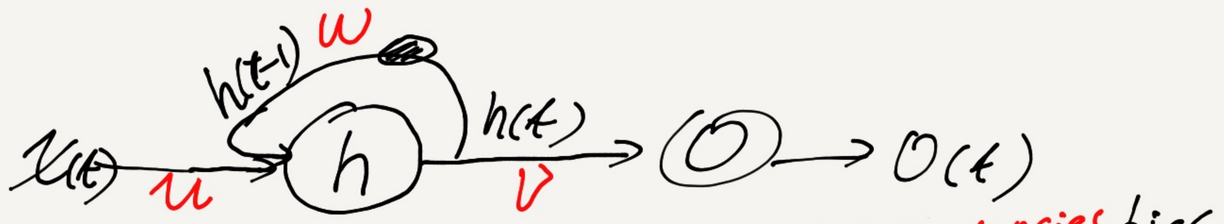
we apply g_h many times,

$h(t) \approx g_h(g_h(g_h \cdots))$

$\nabla h(t) = (\nabla g_h)^t \approx 0$

suppose g_h is sigmoid ($g_h \in [0,1]$)

We will have the vanishing gradient problem.



$h(t) = g_h(U \cdot x(t) + W \cdot h(t-1) + b_h)$

dependencies, bias

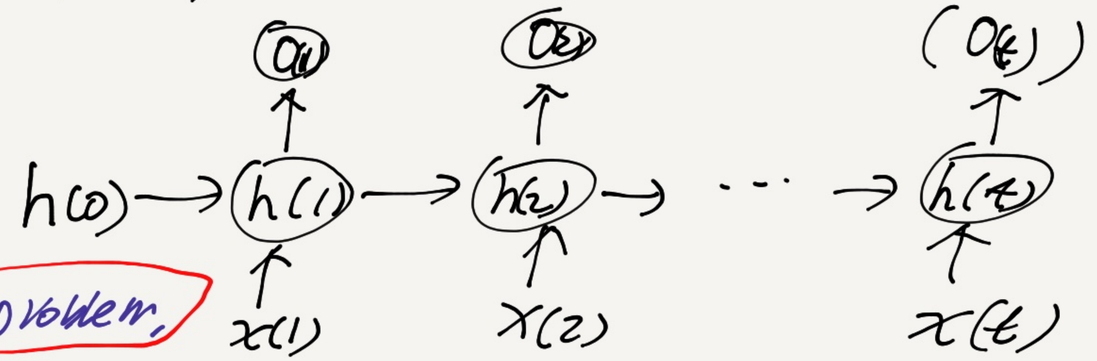
$O(t) = g_o(V \cdot h(t) + b_o)$

bias

$h(t-1)$ to encode information from past sequence.

$h(t)$ caculation is a long chain from.

$h(0), x(1), h(1), x(2), \cdots$ to $h(t-1), x(t)$



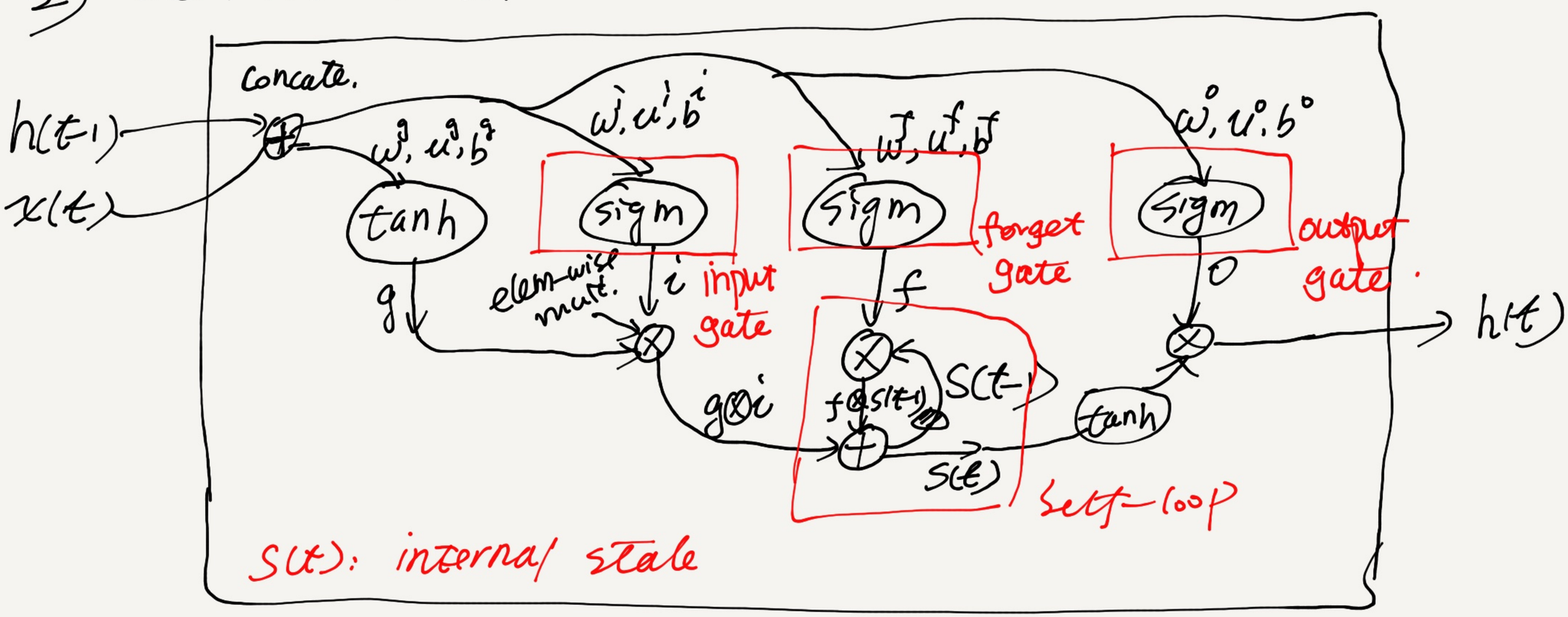
2. LSTM (1997)

D. main idea of LSTM:

① Introduced 3 gates: input gate, forget gate, output gate

② Introduced an internal state and self-loop to produce path to let gradient flow for a long term.

2) LSTM Cell.



⊕: Concatenation ⊗: element-wise multiplication

$g = \tanh(w^g \cdot h(t-1) + u^g \cdot x(t) + b^g) \in [-1, 1]$ → Candidate

new information

input gate: $i = sigm(w^i \cdot h(t-1) + u^i x(t) + b^i) \in [0, 1]$

forget gate: $f = sigm(w^f \cdot h(t-1) + u^f x(t) + b^f) \in [0, 1]$

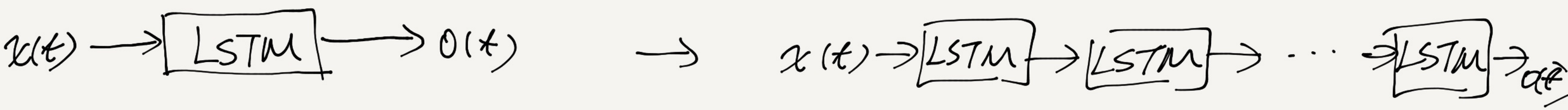
↳ controls the fraction of data to forget.

internal state. $S(t) = \underbrace{(f \otimes S(t-1))}_{\text{forgetting}} + \underbrace{(i \otimes g)}_{\text{input new information}}$

$$\begin{matrix} f & & S \\ \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} & \otimes & \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \end{matrix}$$

output gate: $O = sigm(w^o \cdot h(t-1) + u^o x(t) + b^o)$

3. LSTM-based RNN.



single LSTM cell

multiple LSTM cells