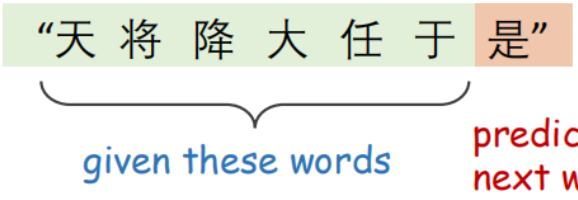
ch11: Recurrent Neural Networks

How to model a sequence（P6）

Sequence Modeling Problem（P8-9）

- **Idea #1: Counting?**

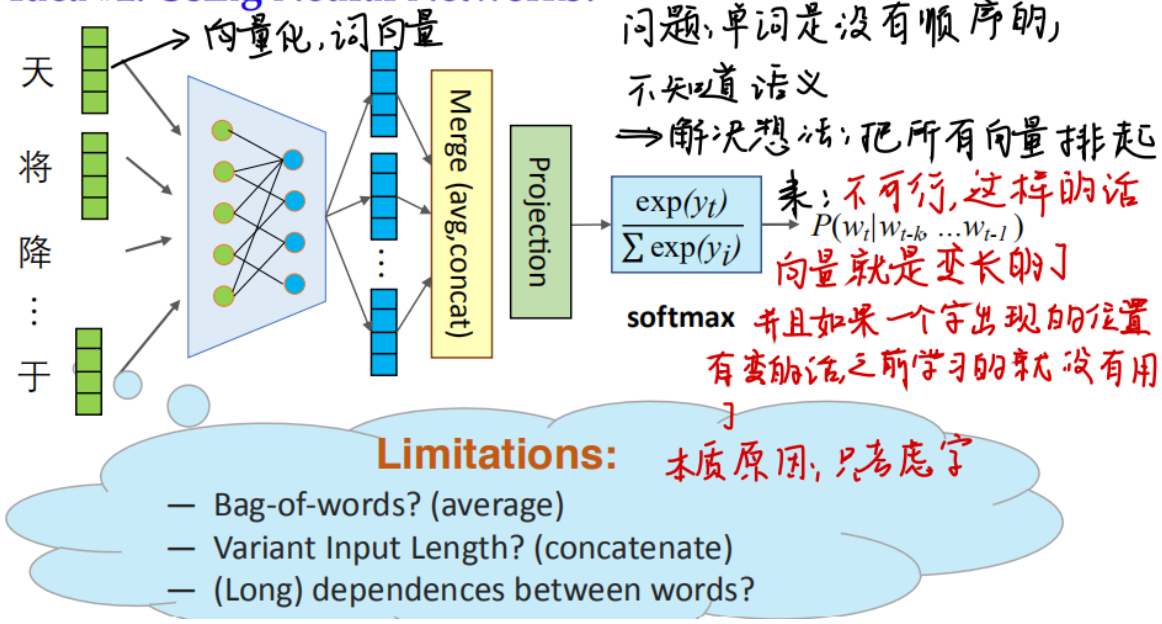


$$p(是 \mid 天将降大任于) = \frac{\text{count}(天将降大任于是)}{\text{count}(天将降大任于)}$$

Problems:
- Discrete symbols,
- No word semantics, → 丧失了语义
- Ignore similarity between words and sentences.

How about “天欲把重担给__”?

- **Idea #2: Using Neural Networks?**



向量化，词向量

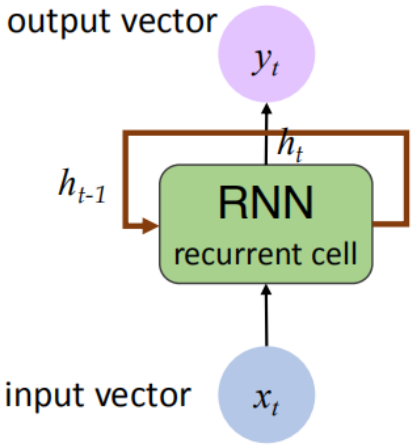
问题：单词是没有顺序的，不知道语义

⇒解决想法：把所有向量排起来；不可行，这样的话向量就是变长的了

并且如果一个字出现的位置有变的话之前学习的就没有用了

本质原因，只考虑字

Recurrent Neural Network (RNN)（P14-18）



非线性转换

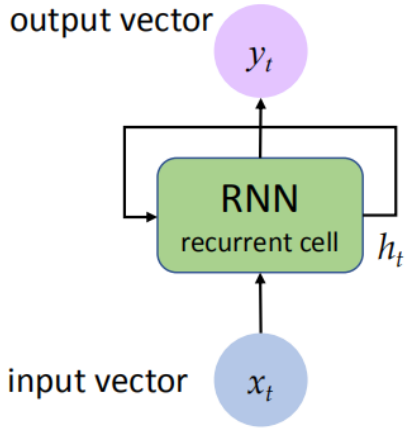
Apply a recurrent relation at every time step to process a sequence:

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

cell state — function parameterized by W — old state — Input vector at time step t

Note: the same function and set of parameters are used at every time step

RNNs have a **state**, h_t, that is updated **at each time step** as a sequence is processed.



Output Vector

$$y_t = \boldsymbol{W}_{hy}^T h_t$$

Update Hidden State

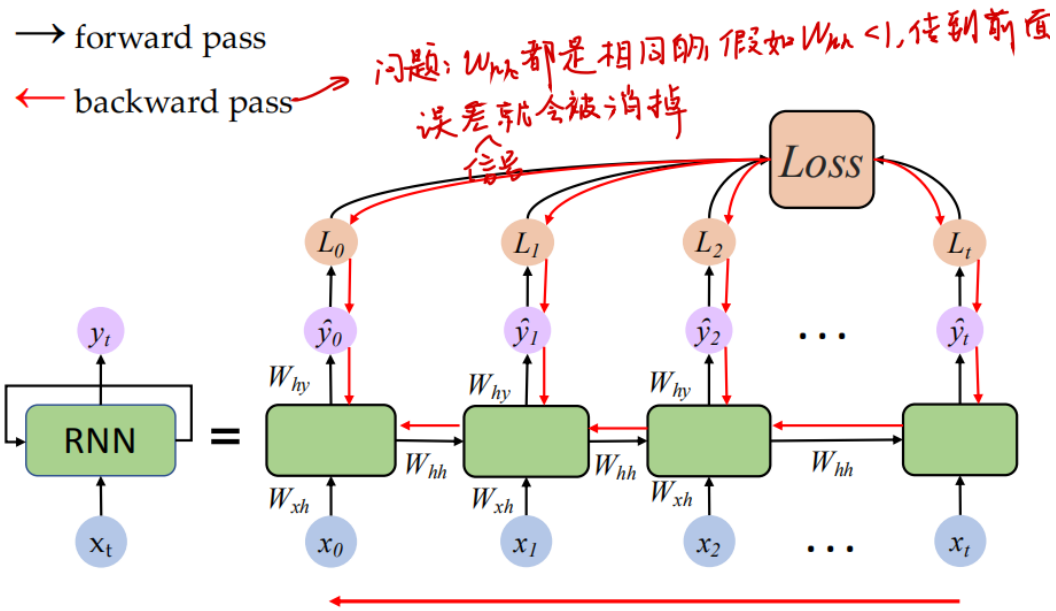
$$h_t = \tanh(\boldsymbol{W}_{hh}^T h_{t-1} + \boldsymbol{W}_{xh}^T x_t)$$

Input Vector

$$x_t$$

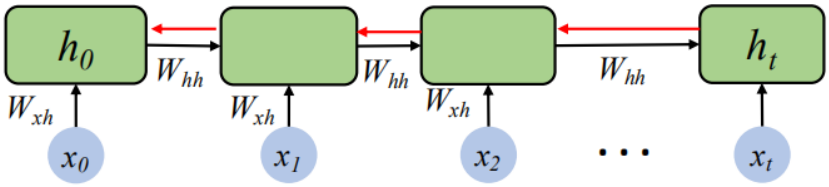
> **RNN: 网络的深度 :输入句子的长度 ;W_{hh}, W_{xh}, W_{hy}每次都一样**

- forward pass: 每个单词的loss加起来或者平均起来，或者只看句子结尾处的L_t
-



The Problems of Standard RNNs（P22）

- **Gradient Flow of Standard RNNs:**



Computing the gradient w.r.t. h_0 involves many factors of W_{hh} + repeated gradient computation!

Many values > 1:
Exploding gradients
梯度爆炸
Gradient clipping to scale big gradients

Many values <1:
Vanishing gradients
梯度消失
Gradient clipping to 0 gradients

- 为什么梯度的消失是个问题？
  - 后 文对于前面没有影响, 后面的loss无法用于前面纠错
- 解决问题的trick:
  - **Using activation functions that have larger derivatives(**避免激活函数的导数太小，比如reLU)（P24）
  - **Trick #2: Parameter Initialization（P25）**

    - **Initialize weights**

      Example
      - Initialize weights to identify matrix
      - Initial biases to zero

      $$W_{init} = \begin{pmatrix} 1 & 0 & 0 & & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & & 0 \\ & \vdots & & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \end{pmatrix}$$

      This helps prevent the weights from shrinking to zero.

  - **Gated Cells（P26）**

- Idea: use a more complex recurrent unit with gates to control what information is passed through.

用门控制信息流通

gated cell
LSTM, GRU, etc.

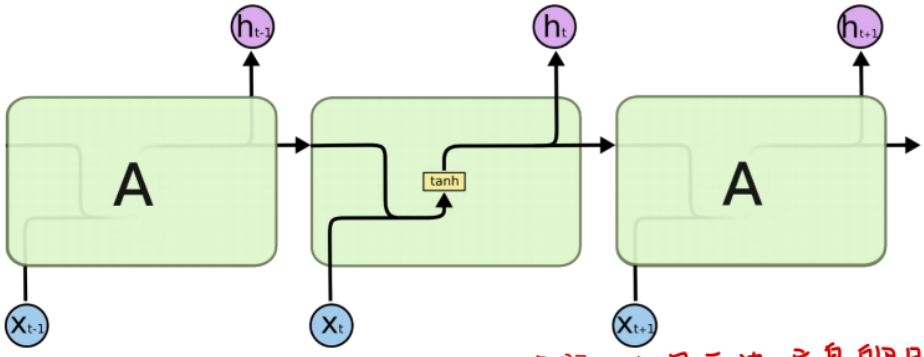
长短记忆神经网络

Long Short Term Memory (LSTM) networks rely on a gated cell to track information through many time steps.

Long Short Term Memory (LSTM) Networks（P27-38）

- **Standard RNN的问题**

- In a standard RNN, repeating modules contain a **simple computation node**.



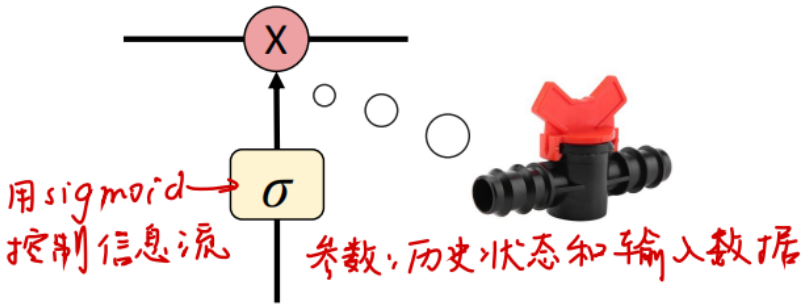
问题：使用历史信息的时候在做乘法

⇒历史"记忆"应该做加(减)法；

把历史"记忆"淡化一点，加入新数据

- 解决方案：Gates

- Information is **added** or **removed** through structures called gates.



门可以选择性地让信息通过

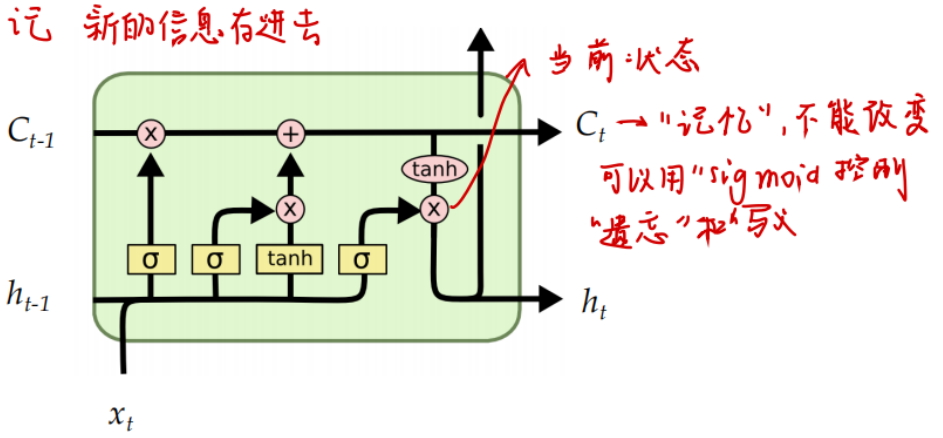
Gates optionally let information through, for example via a sigmoid neural net layer and pointwise multiplication

how LSTM work:

- How do LSTMs work? 更新长期记忆

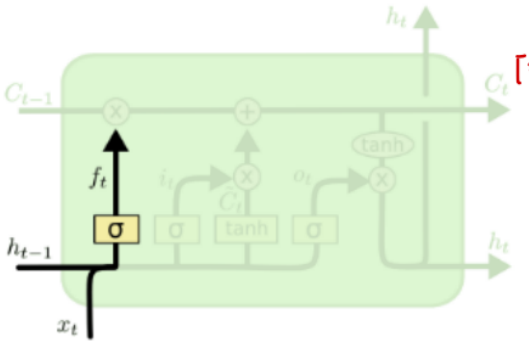
1) Forget 2) Store 3) Update 4) Output 有选择地传到下一部分

忘记原先的记忆一部分　新的信息有进去



1) Forget 2) Store 3) Update 4) Output

- LSTMs **forget irrelevant** parts of the previous state

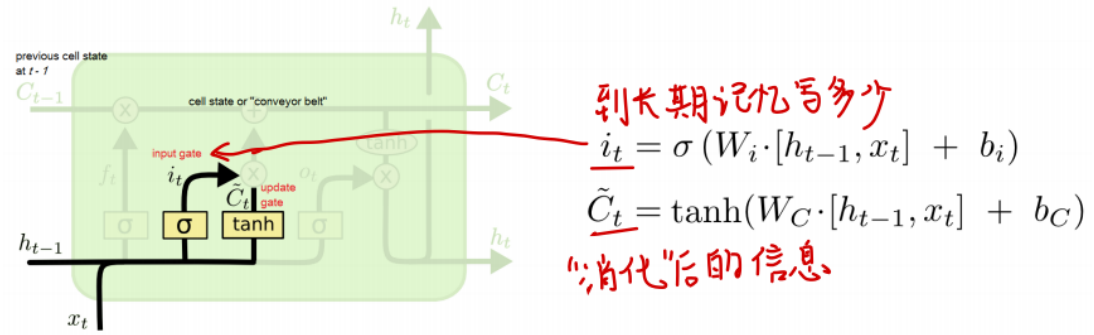


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right)$$

[0,1]
当前的状态
sigmoid
观察到的信息

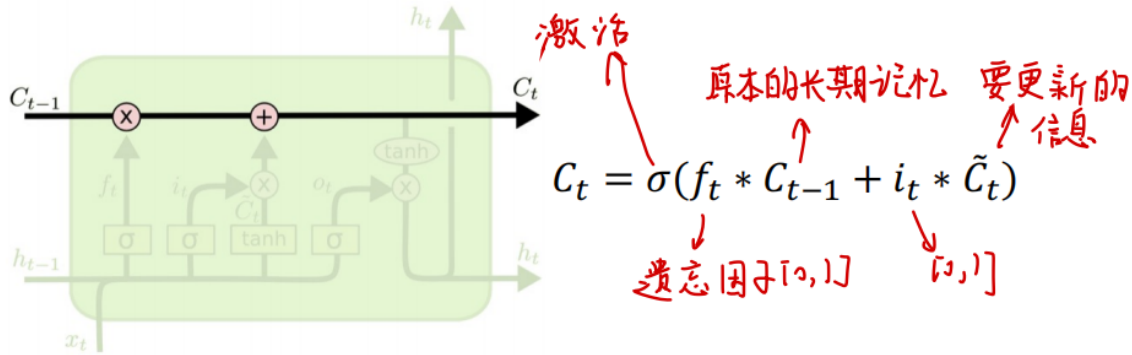
1) Forget **2) Store** 3) Update 4) Output

- LSTMs **store relevant** new information into the cell state.



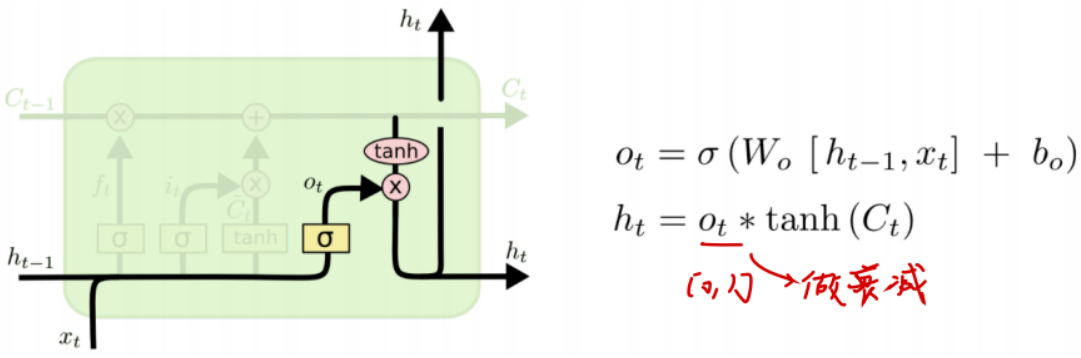
1) Forget 2) Store **3) Update** 4) Output

- LSTMs **selectively update** cell state values

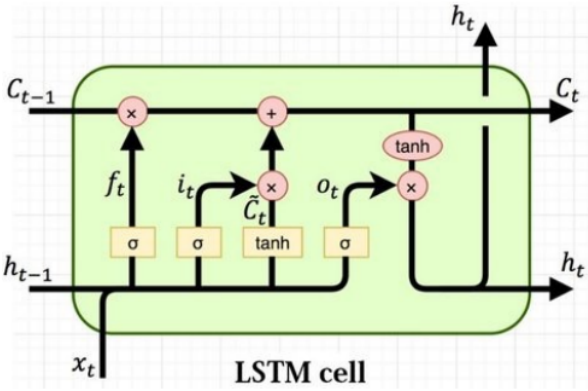


1) Forget 2) Store 3) Update **4) Output**

- The **output gate** controls what information is sent to the next time step.



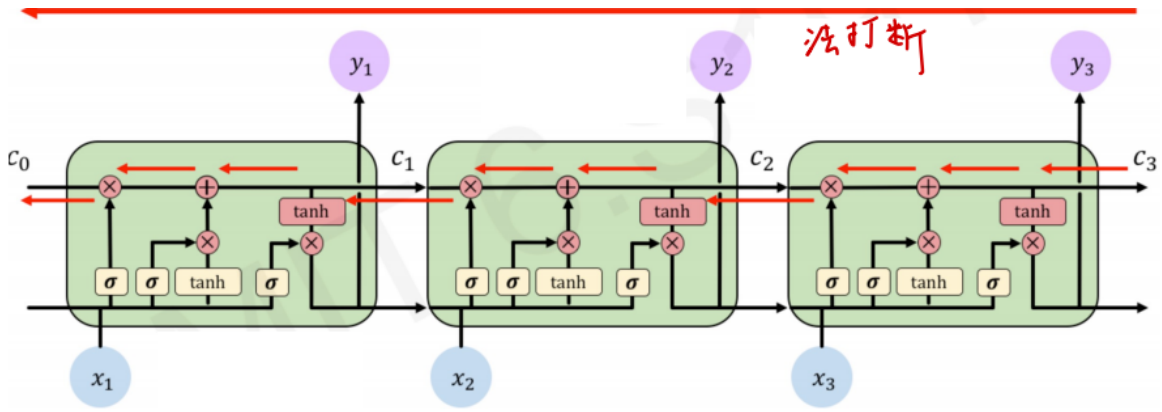
- overview



线性转换 + sigmoid

$$i_t = \sigma\left(x_t U^i + h_{t-1} W^i\right)$$
$$f_t = \sigma\left(x_t U^f + h_{t-1} W^f\right)$$
$$o_t = \sigma\left(x_t U^o + h_{t-1} W^o\right)$$
$$\tilde{C}_t = \tanh\left(x_t U^g + h_{t-1} W^g\right)$$
$$C_t = \sigma\left(f_t * C_{t-1} + i_t * \tilde{C}_t\right)$$
$$h_t = \tanh(C_t) * o_t$$

长期记忆不会被矩阵乘法打断



LSTMs: Key Concepts（P38）

1. Maintain a **separate cell state** from what is outputted
2. Use **gates** to control the flow of information
   - **Forget** gate get rid of irrelevant information
   - **Store** relevant information from current input
   - Selectively **update** cell state
   - **Output** gate returns a filtered version of the cell state
3. Backpropagation through time with **uninterrupted gradient flow**

RNN Applications（P39-46）

- **RNN for Sequence Classification**

- **RNN for Sequence Generation**