2023年春季 《深度学习导论》课程 第七章

循环神经网络

主讲：王皓 副研究员| 硕士导师

邮箱：wanghao3@ustc.edu.cn

主页：http://staff.ustc.edu.cn/~wanghao3

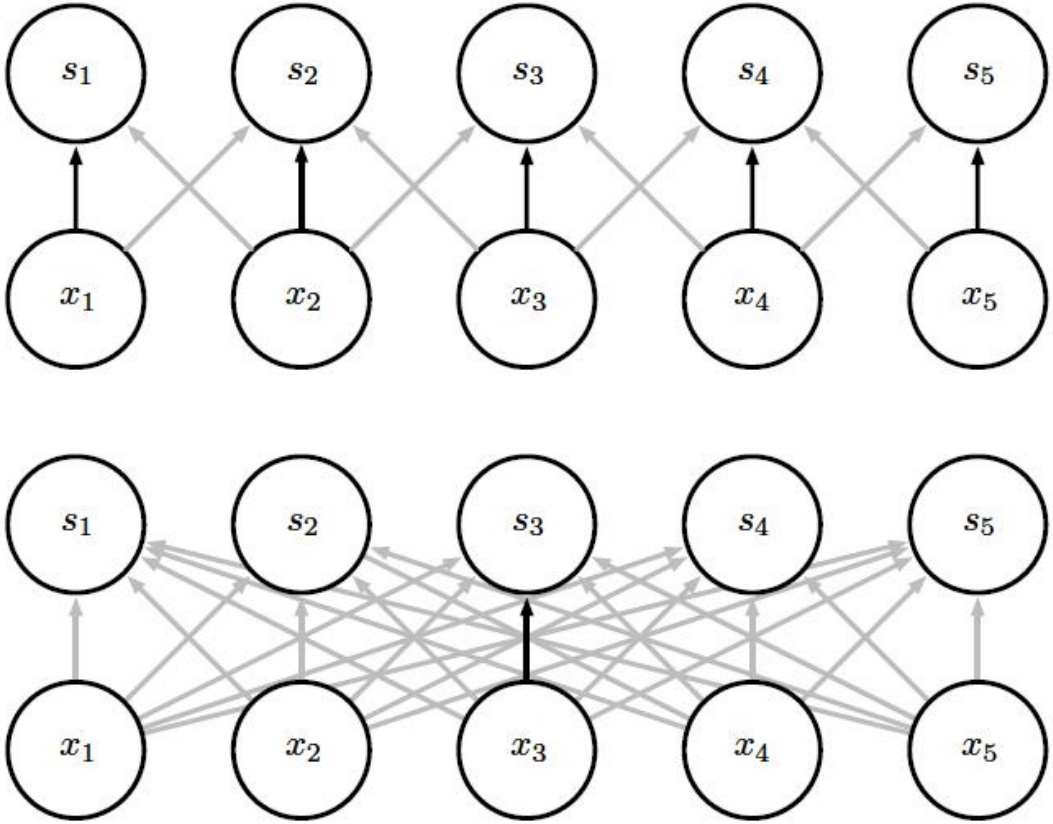
本章内容

- 循环神经网络的动机
- 循环神经网络的使用方式
- 循环神经网络的参数学习
- 循环神经网络的长程依赖
- 循环神经网络的应用

RNN动机

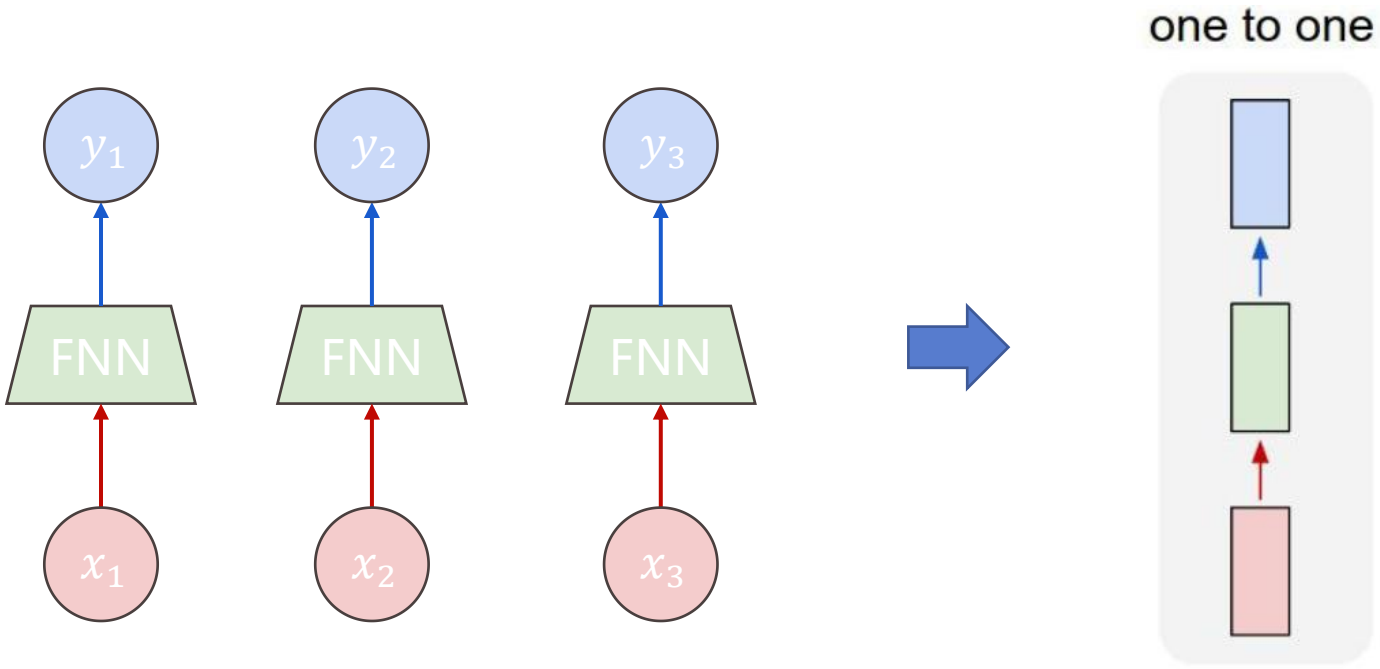
前馈神经网络

- 连接存在层与层之间，每层的节点之间是无连接的（无循环）



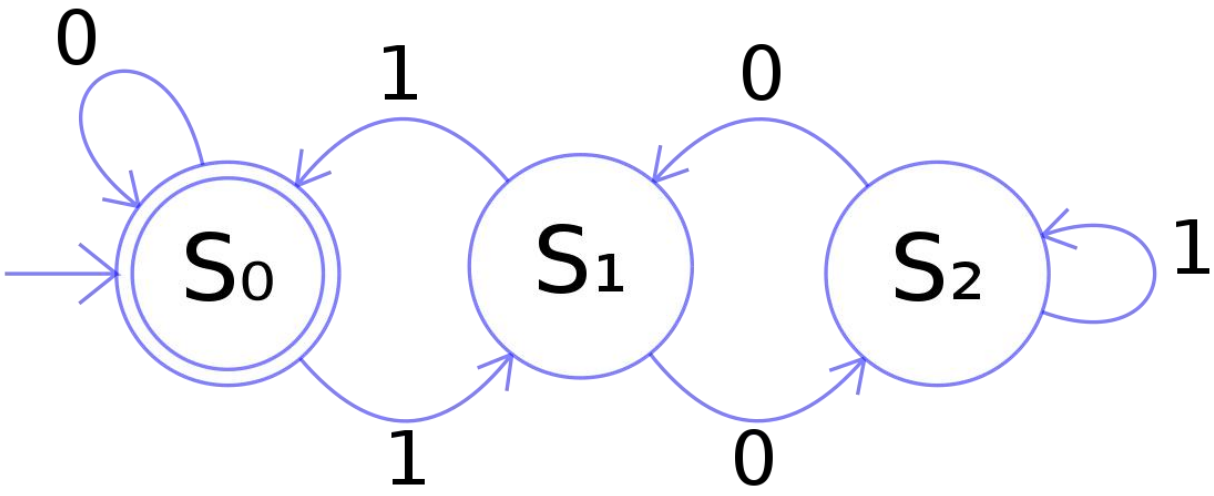
前馈神经网络

- 假设每次输入都是独立的，也就是说每次网络的输出只依赖于当前的输入



前馈神经网络存在问题

- 输入和输出维数固定，不能任意改变，无法处理变长的序列数据，比如视频、语音、文本等
- 很多现实任务，网络输出不仅和当前时刻的输入相关，也和其过去一段时间的输出相关
  - 比如，在有限状态自动机中，下一个时刻的状态不仅和当前输入相关，也和当前状态（上一个时刻的输出）相关



如何用FNN去模拟一个有限状态自动机?

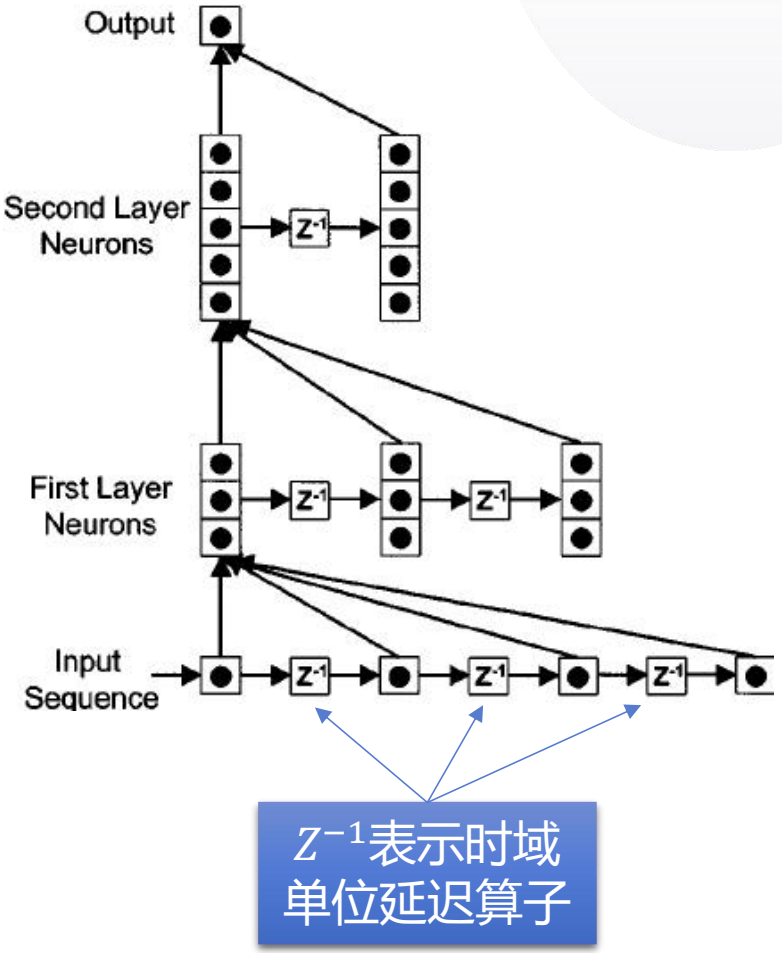
延时神经网络(Time Delay Neural Network)

- 在前馈神经网络的非输出层都添加一个延时器，记录神经元的最近几次活性值

$$\boldsymbol{h}_t^{(l)} = f\left(\boldsymbol{h}_t^{(l-1)}, \boldsymbol{h}_{t-1}^{(l-1)}, \cdots, \boldsymbol{h}_{t-K}^{(l-1)}\right)$$

$\boldsymbol{h}_t^{(l)} \in \mathbb{R}^{M_l}$表示第$l$层神经元在时刻$t$的活性值

- 这样，前馈网络就具有了短期记忆的能力。



自回归模型（Autoregressive Model，AR）

➢一类时间序列模型，用变量y_t的历史信息来预测自己

$$\boldsymbol{y}_t = w_0 + \sum_{k=1}^{K} w_k \boldsymbol{y}_{t-k} + \epsilon_t$$

- K为超参数，$w_0, \cdots, w_K$为可学习参数，$\epsilon_t \sim N(0, \sigma^2)$为第t个时刻的噪声

➢有外部输入的非线性自回归模型

- 每个时刻t都有一个输入$\boldsymbol{x}_t$，产生一个输出$\boldsymbol{y}_t$
- 通过一个延时器记录最近K_x次的外部输入和最近的K_y次输出

$$\boldsymbol{y}_t = f\left(\boldsymbol{x}_t, \boldsymbol{x}_{t-1}, \cdots, \boldsymbol{x}_{t-K_x}, \boldsymbol{y}_{t-1}, \boldsymbol{y}_{t-2}, \cdots, \boldsymbol{y}_{t-K_y}\right)$$

- $f(\cdot)$表示非线性函数，可以为前馈网络，K_x和K_y为超参数

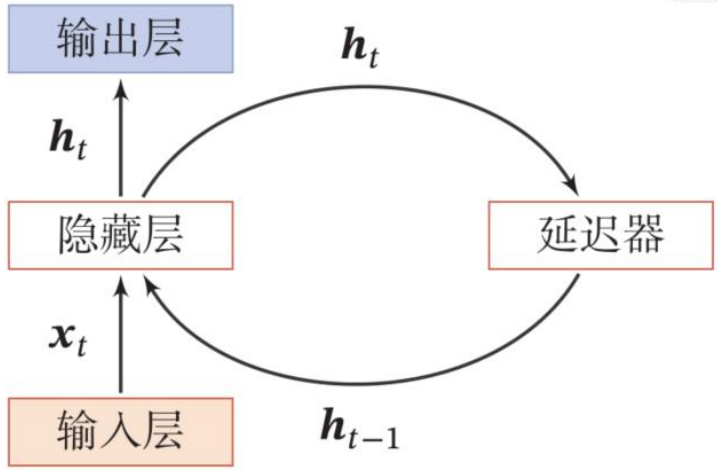
循环神经网络（Recurrent Neural Network）

- 循环神经网络通过使用带自反馈的神经元，能够处理任意长度的时序数据。

$$\boldsymbol{h}_t = f_W(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$$

新状态　　旧状态　　t时刻的输入向量

参数为W的函数



状态 $\boldsymbol{h}_t$ “存储” 直到t时刻的历史数据$\{\boldsymbol{x}_1, \cdots, \boldsymbol{x}_t\}$

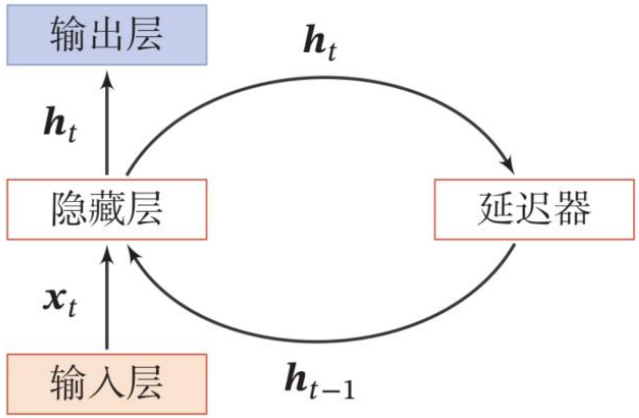
$$\boldsymbol{h}_t = f_W(f_W(f_W(\boldsymbol{h}_{t-3}, \boldsymbol{x}_{t-2}), \boldsymbol{x}_{t-1}), \boldsymbol{x}_t)$$

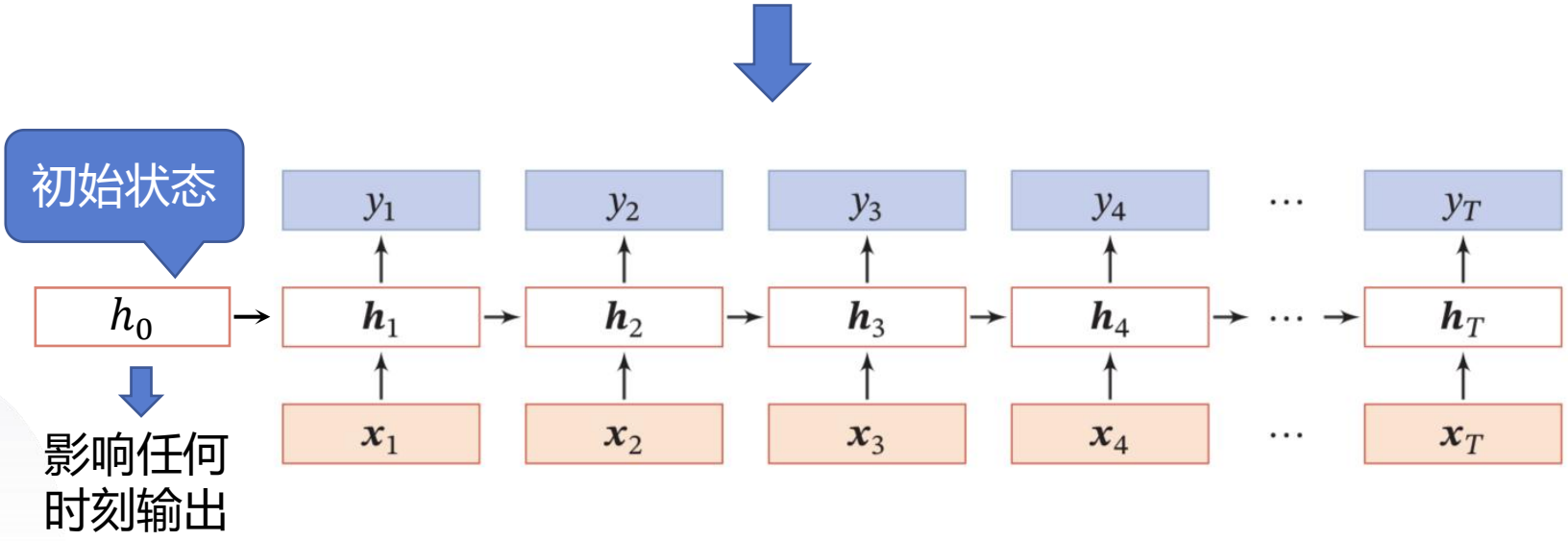
按时间展开

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

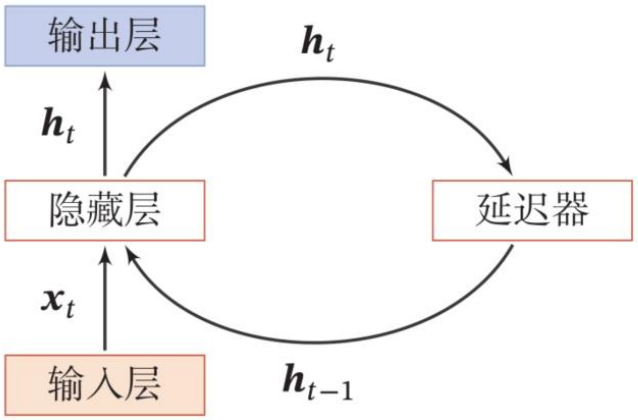
新状态　　旧状态　　t时刻的输入向量

参数为W的函数





Elman RNN



$$\boldsymbol{h}_t = f_{\boldsymbol{W}}(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t)$$

$$\boldsymbol{h}_t = f(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b})$$

$\boldsymbol{U}$为状态-状态权重矩阵，$\boldsymbol{W}$为状态-输入权重矩阵，$\boldsymbol{b}$为偏置向量

循环神经网络的计算能力

➢一个完全连接的循环网络

$$\boldsymbol{h}_t = f(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b})$$
$$\boldsymbol{y}_t = \boldsymbol{V}\boldsymbol{h}_t$$

则该网络是任何非线性动力系统的近似器

定理 6.1 – 循环神经网络的通用近似定理 [Haykin, 2009]： 如果一个完全连接的循环神经网络有足够数量的 sigmoid 型隐藏神经元，它可以以任意的准确率去近似任何一个非线性动力系统

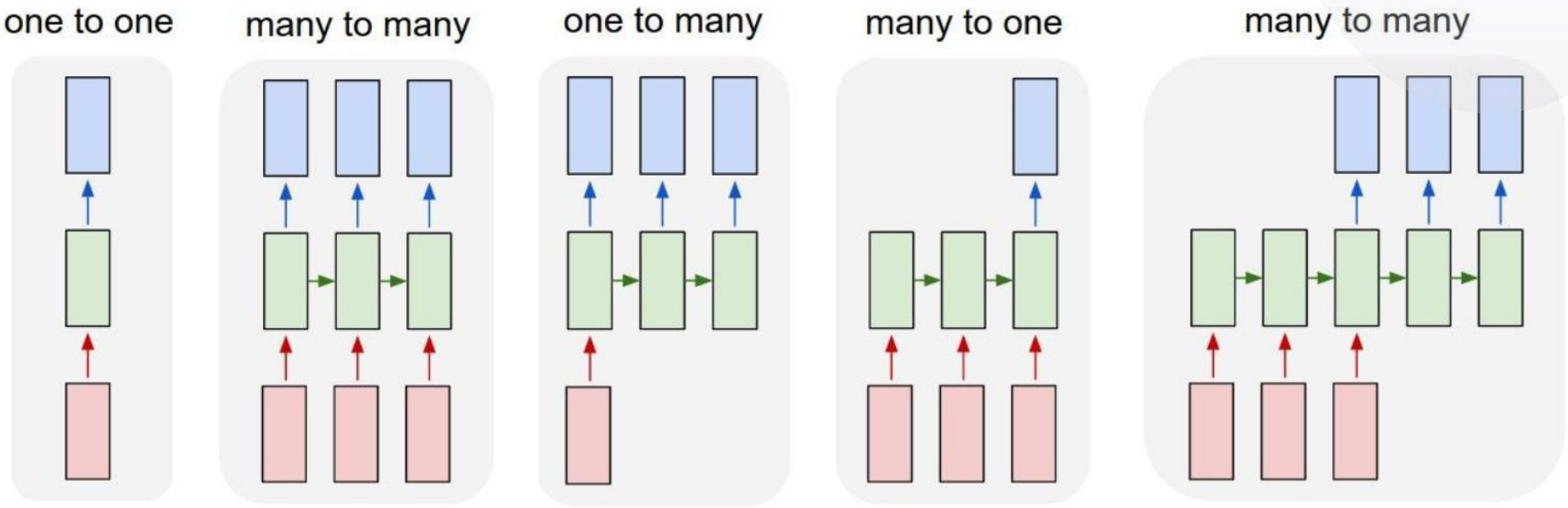
$$\boldsymbol{s}_t = g(\boldsymbol{s}_{t-1}, \boldsymbol{x}_t), \tag{6.10}$$

$$\boldsymbol{y}_t = o(\boldsymbol{s}_t), \tag{6.11}$$

其中 $\boldsymbol{s}_t$ 为每个时刻的隐状态，$\boldsymbol{x}_t$ 是外部输入，$g(\cdot)$ 是可测的状态转换函数，$o(\cdot)$ 是连续输出函数，并且对状态空间的紧致性没有限制.

RNN使用方式

循环神经网络的使用方式



前向网络

例：词性标注
句子 → 类别

例：图像描述
图像 → 句子

例：文本分类
文档 → 类别

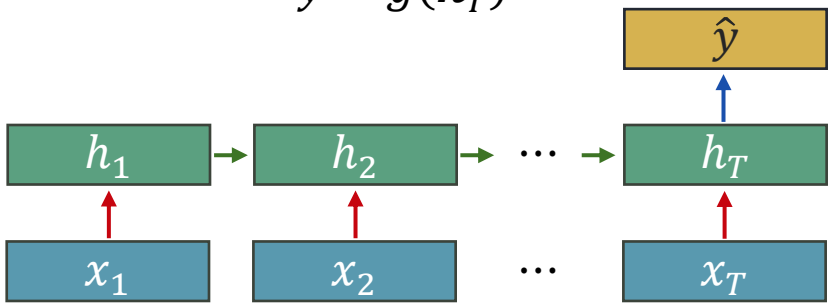
例：机器翻译
句子 → 句子

循环神经网络的使用方式

- 序列到类别
  - 情感分类
- 同步的序列到序列模式
  - 中文分词
  - 信息提取
- 异步的序列到序列模式
  - 机器翻译
- 向量到序列
  - 图像描述

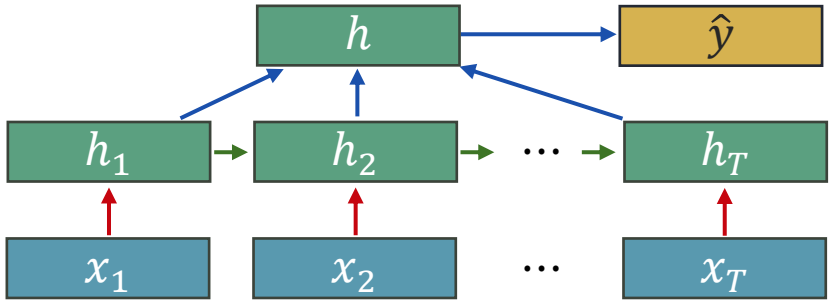
序列到类别

$$\hat{y} = g(\boldsymbol{h}_T)$$



正常模式

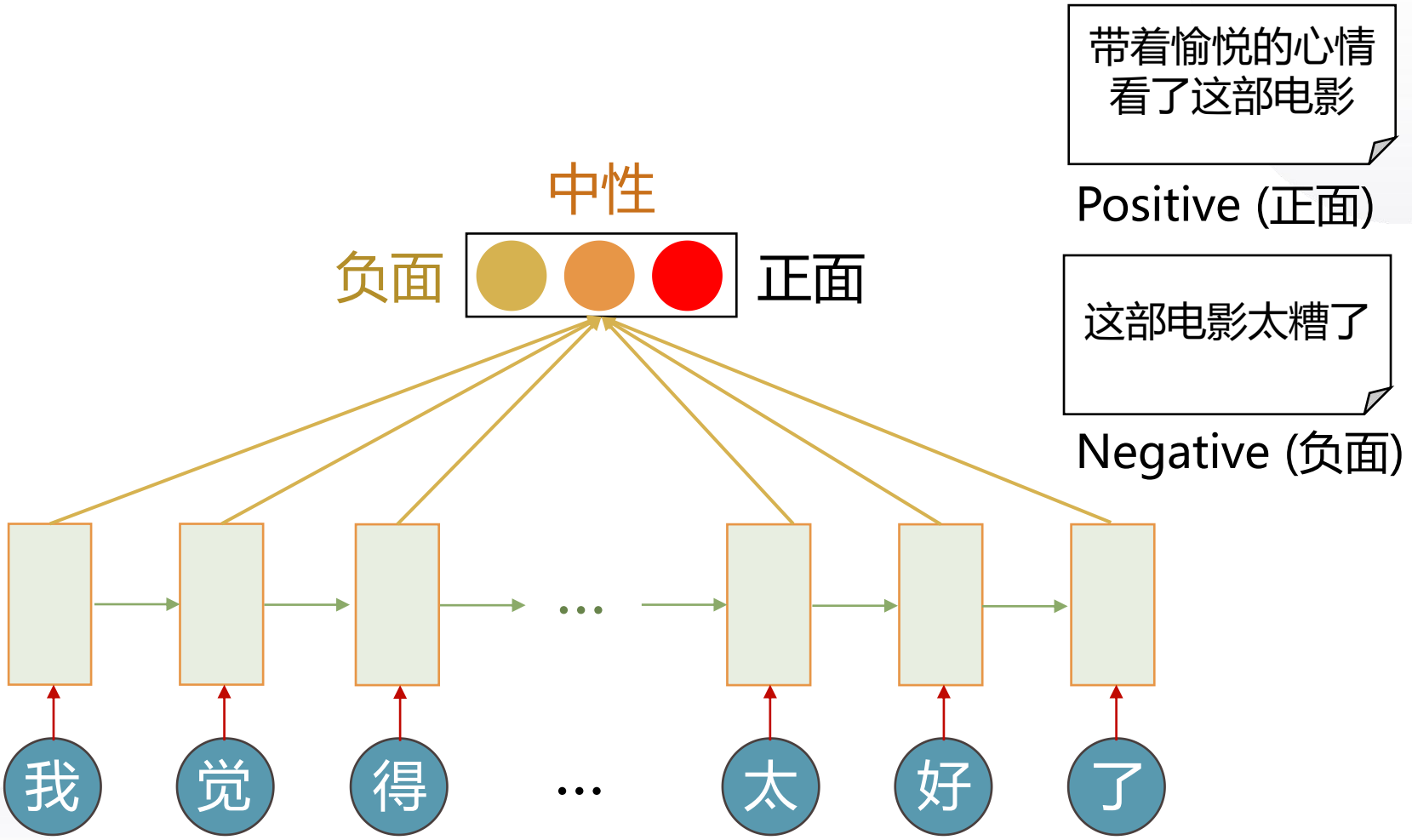
$$\hat{y} = g\left(\frac{1}{T}\sum_{t}^{T}\boldsymbol{h}_t\right)$$



按时间进行平均采样模式

$g(\cdot)$可以是简单的线性分类器（对率回归）或者复杂分类器（多层前馈神经网络）

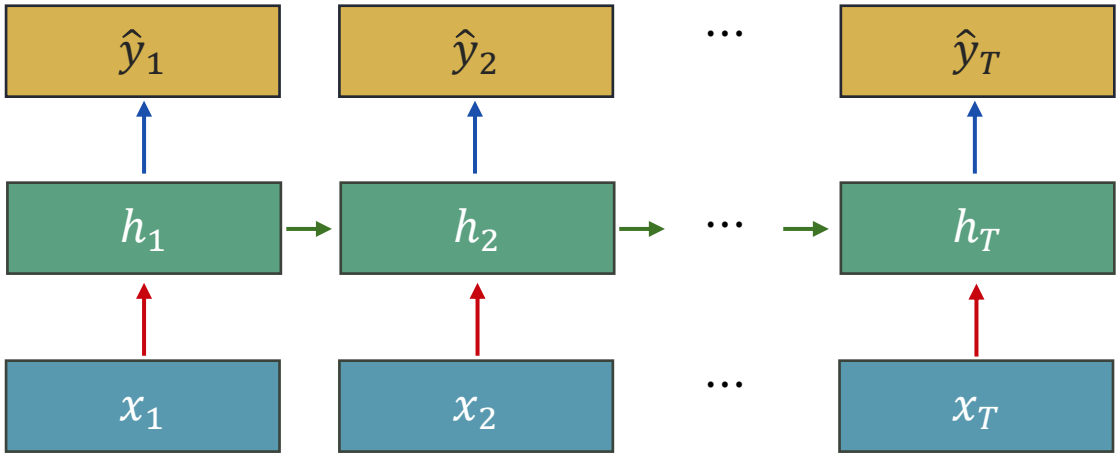
序列到类别：情感分类


带着愉悦的心情
看了这部电影
Positive (正面)
这部电影太糟了
Negative (负面)
中性
负面
正面
我
觉
得
…
太
好
了

同步的序列到序列模式

- 每一时刻都有输入和输出，输入序列和输出序列的长度相同

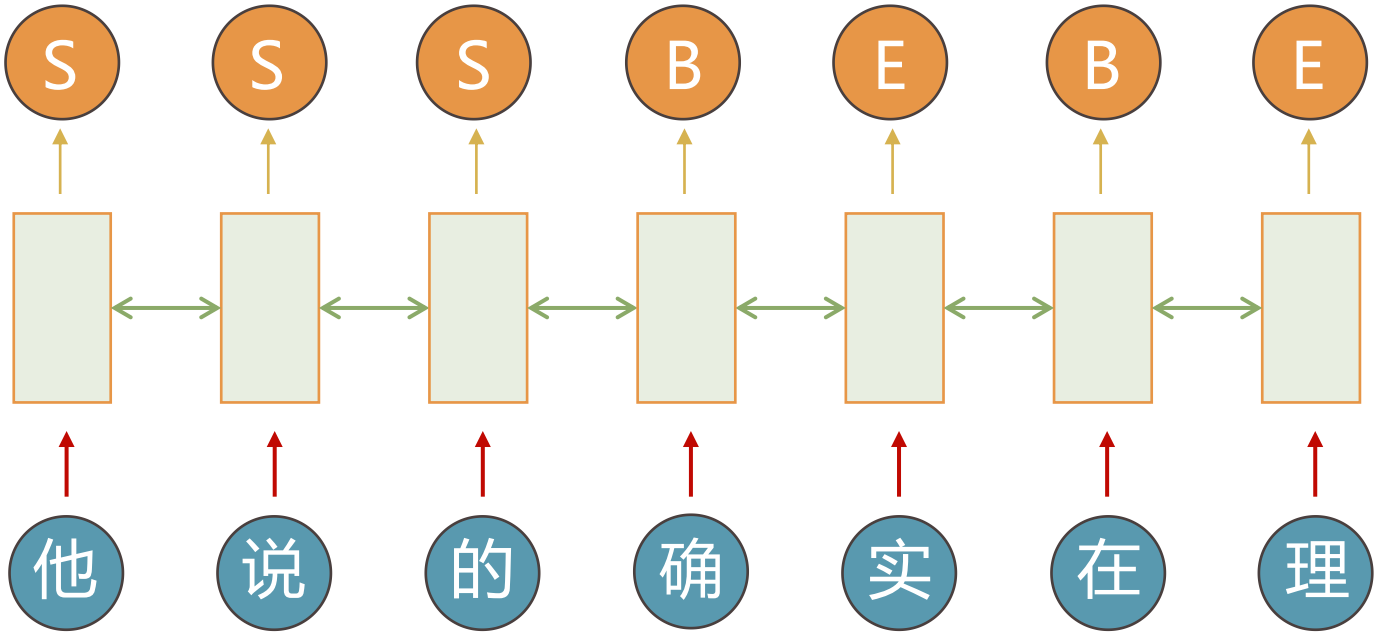
$$\hat{y}_t = g(\boldsymbol{h}_t),\ \forall t \in [1, T]$$



可以用于词性标注、分词、词义消歧、视频帧分类、信息抽取、语音识别等

同步的序列到序列模式：中文分词

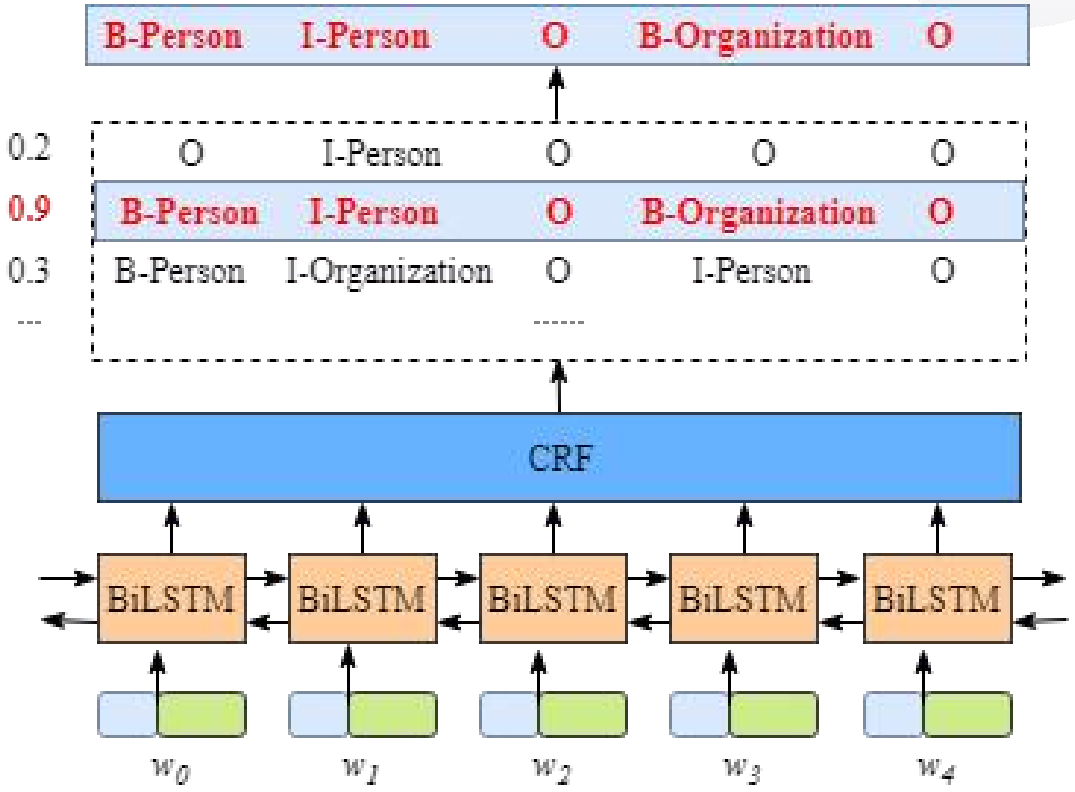
词首 B　　词中 M　　词尾 E　　单字成词 S



同步的序列到序列模式：信息抽取

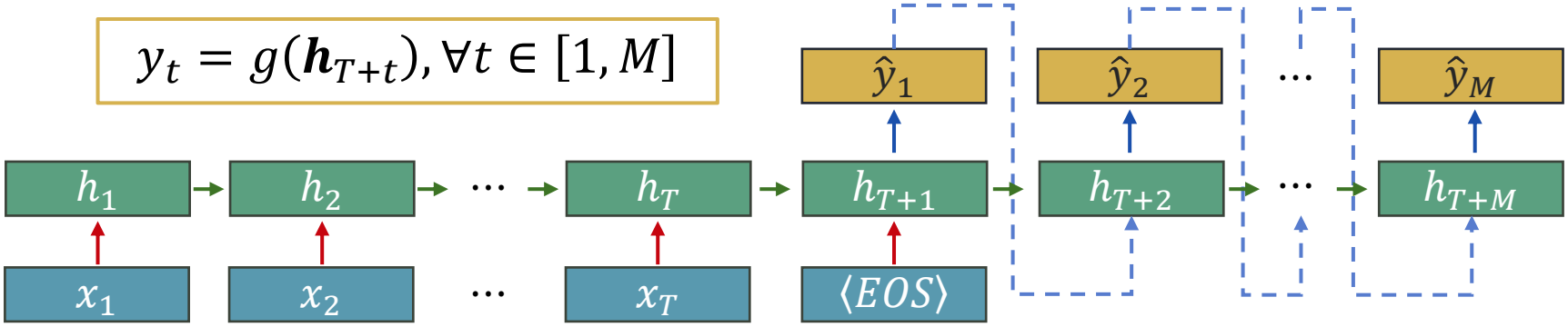
- 从无结构的文本中抽取结构化的信息，形成知识

小米创始人雷军表示，该公司2015年营收达到780亿元人民币，较2014年的743亿元人民币增长了5%。



异步的序列到序列模式

- 输入序列和输出序列不需要有严格的对应关系，也不需要保持相同的长度
- 又称为编码器-解码器（Encoder-Decoder）模型



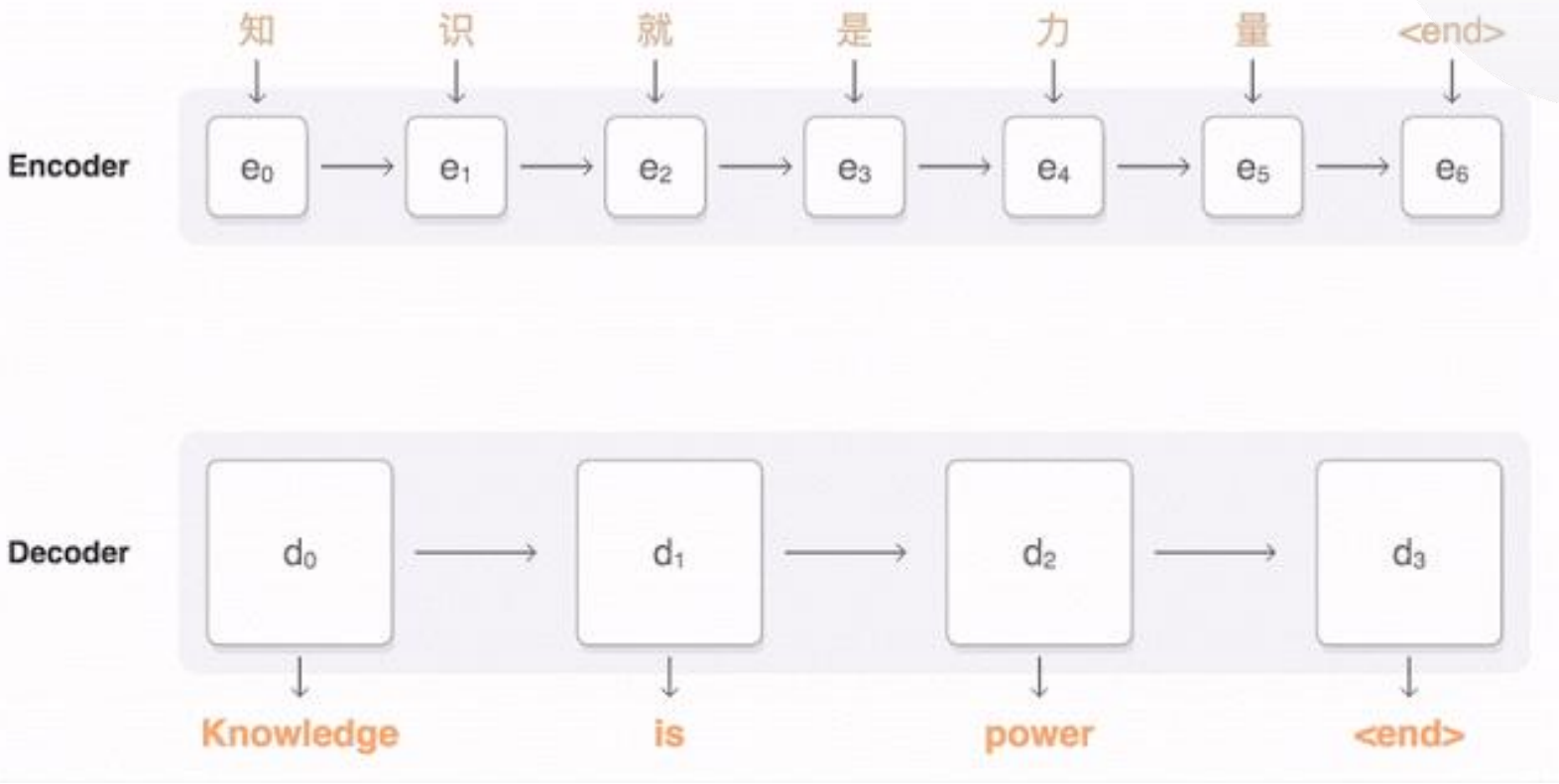
编码器

$$\boldsymbol{h}_t = f_1(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t), \forall t \in [1, T]$$

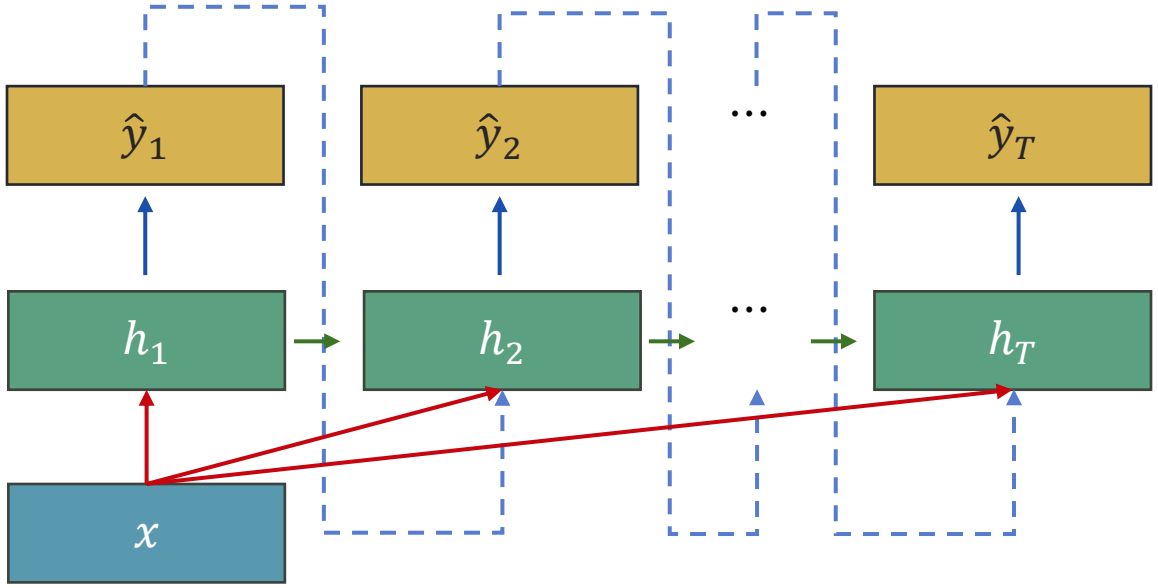
$$\boldsymbol{h}_{T+t} = f_2(\boldsymbol{h}_{T+t-1}, \hat{y}_{t-1}), \forall t \in [1, M]$$

解码器

异步的序列到序列模式：机器翻译



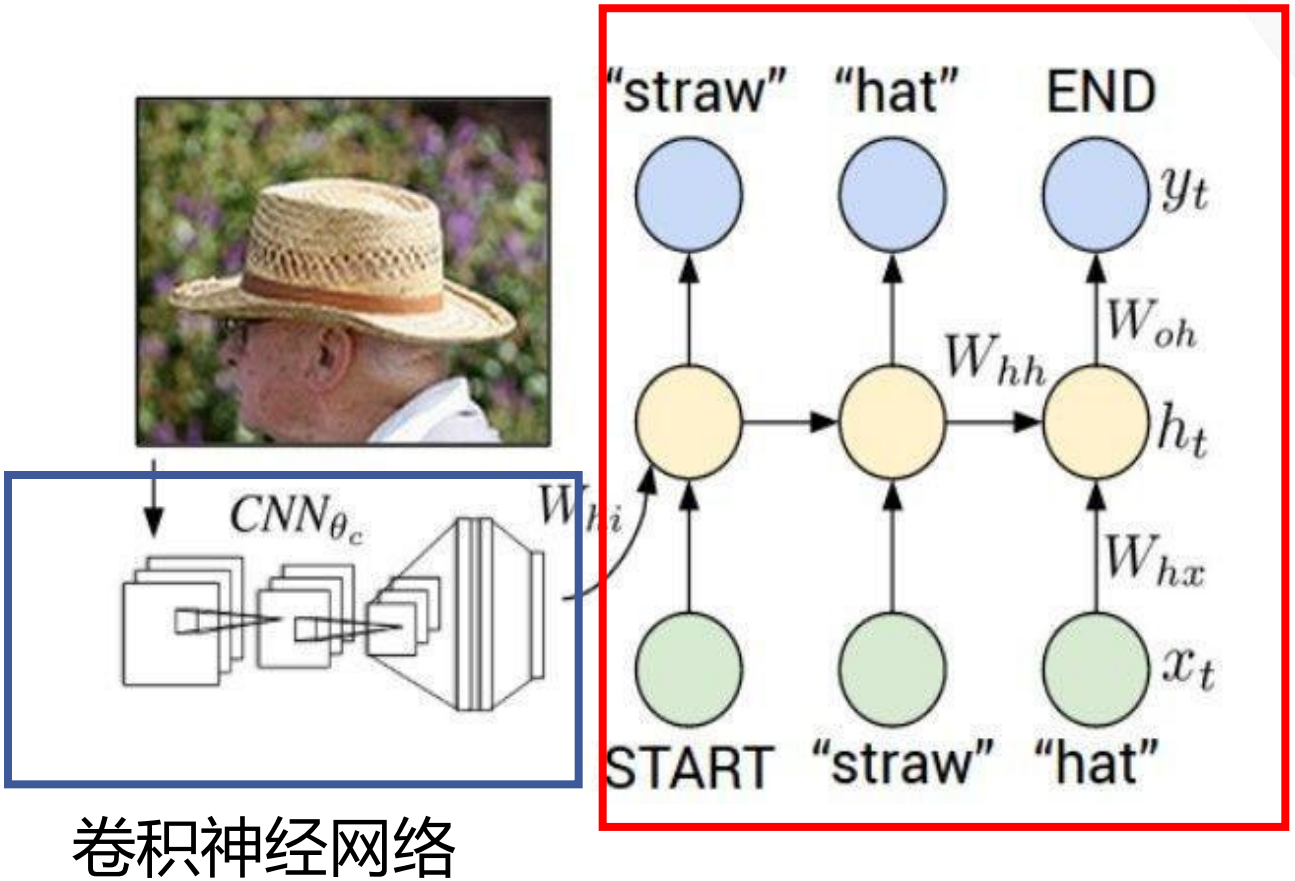
向量到序列



$$\boldsymbol{h}_t = f(\boldsymbol{h}_{t-1}, \boldsymbol{x}, \hat{y}_{t-1}), \forall t \in [1, T]$$

向量到序列：图像描述

循环神经网络



卷积神经网络

RNN参数学习

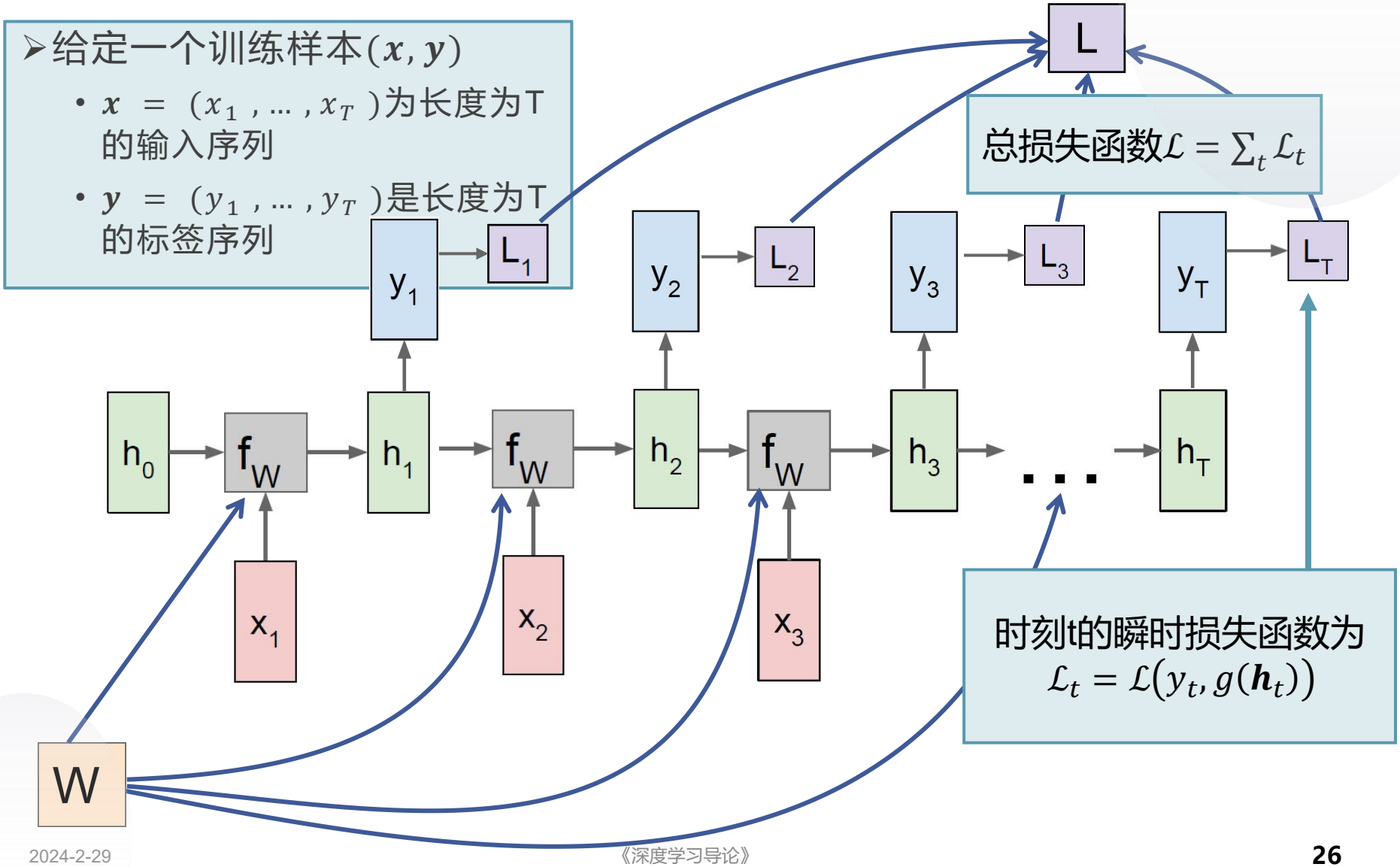
同步的序列到序列和异步的序列到序列是两种不同的机器翻译模型架构。

同步的序列到序列模型是指源语言句子和目标语言句子之间的对应关系是一一对应的，即源语言句子中的每个词都对应目标语言句子中的一个词。这种模型在训练和推理过程中都需要同时考虑源语言和目标语言的上下文信息，因此被称为同步模型。同步模型通常使用编码器-解码器结构，其中编码器将源语言句子编码为一个固定长度的向量表示，解码器根据这个向量表示生成目标语言句子。

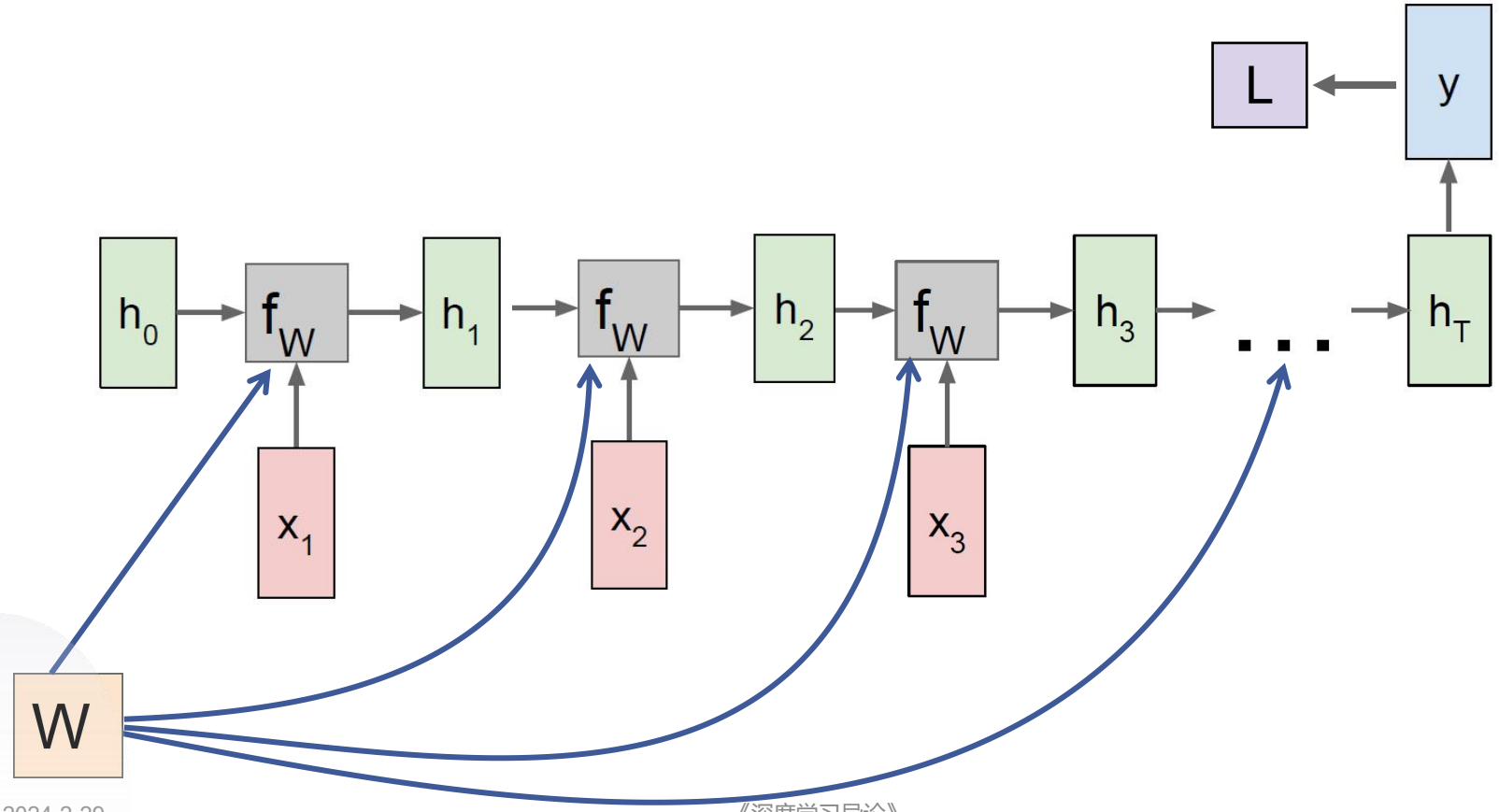
异步的序列到序列模型是指源语言句子和目标语言句子之间的对应关系不是一一对应的，即源语言句子中的一个词可能对应目标语言句子中的多个词，或者多个词对应一个词。这种模型在训练和推理过程中可以分别处理源语言和目标语言的上下文信息，因此被称为异步模型。异步模型通常使用多层编码器和解码器，其中编码器将源语言句子编码

参数学习—计算图（多对多）

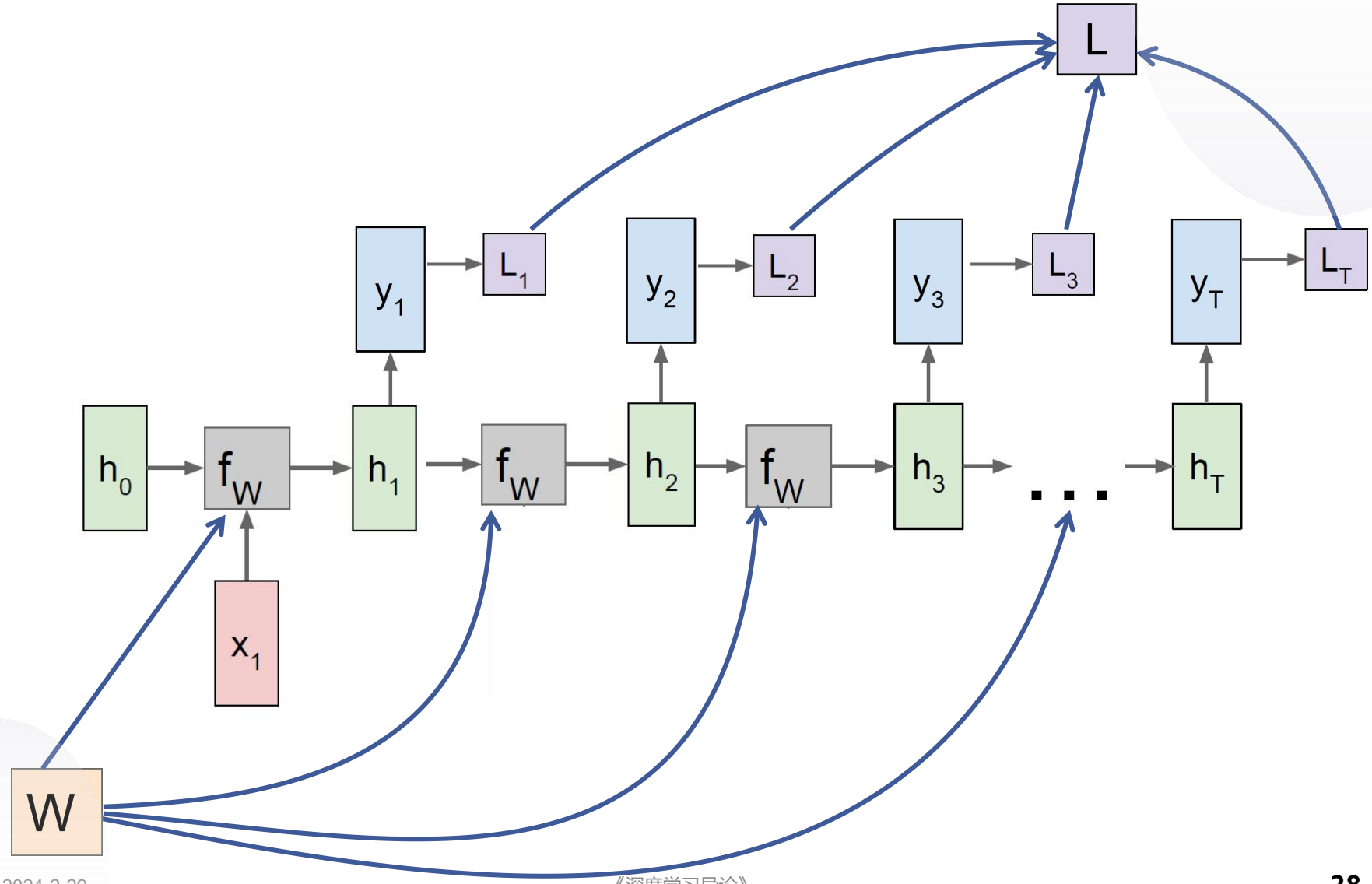
- 给定一个训练样本$(\boldsymbol{x}, \boldsymbol{y})$
  - $\boldsymbol{x} = (x_1, \ldots, x_T)$为长度为T的输入序列
  - $\boldsymbol{y} = (y_1, \ldots, y_T)$是长度为T的标签序列



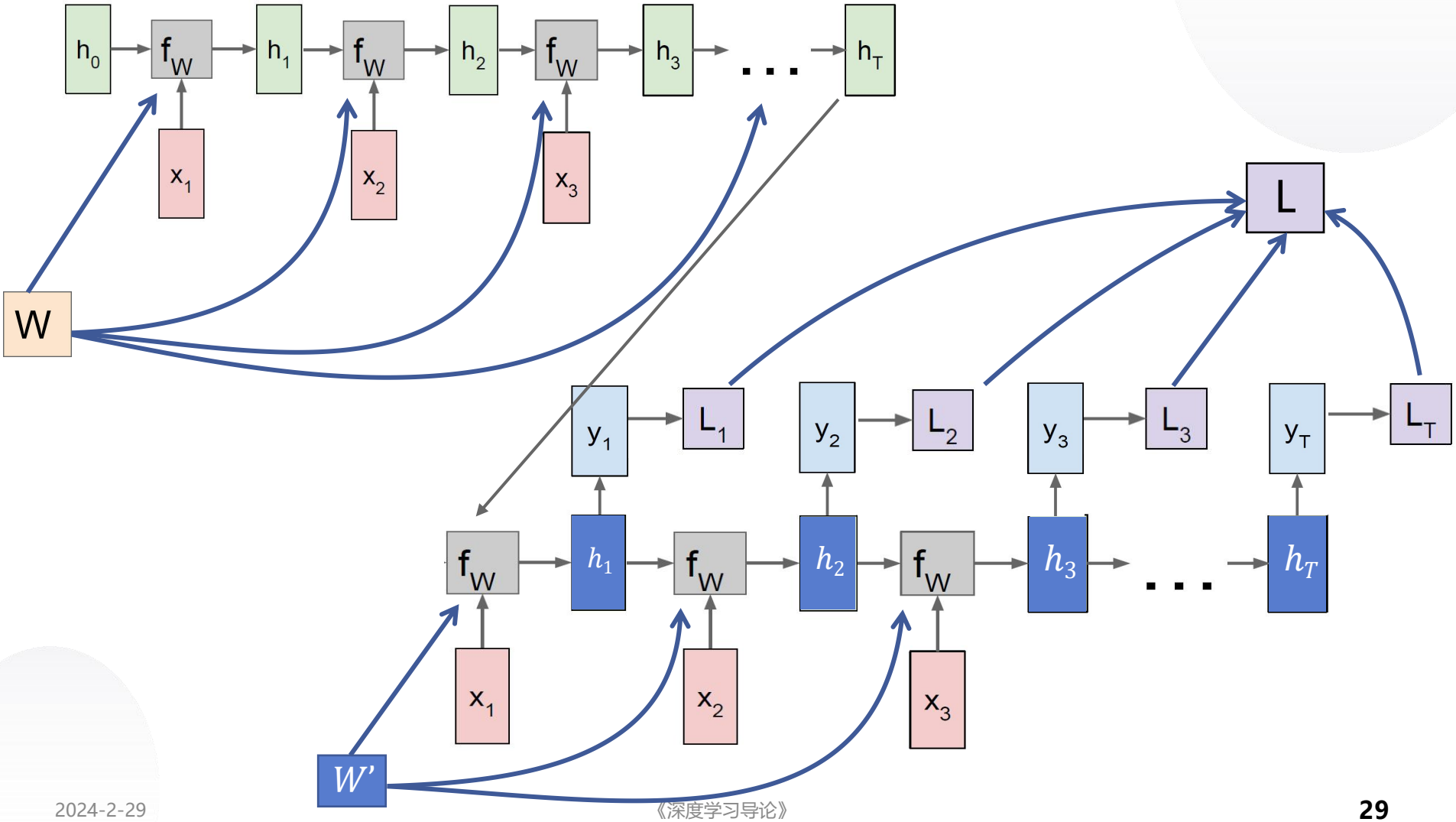
参数学习—计算图（多对一）



参数学习—计算图（一对多）



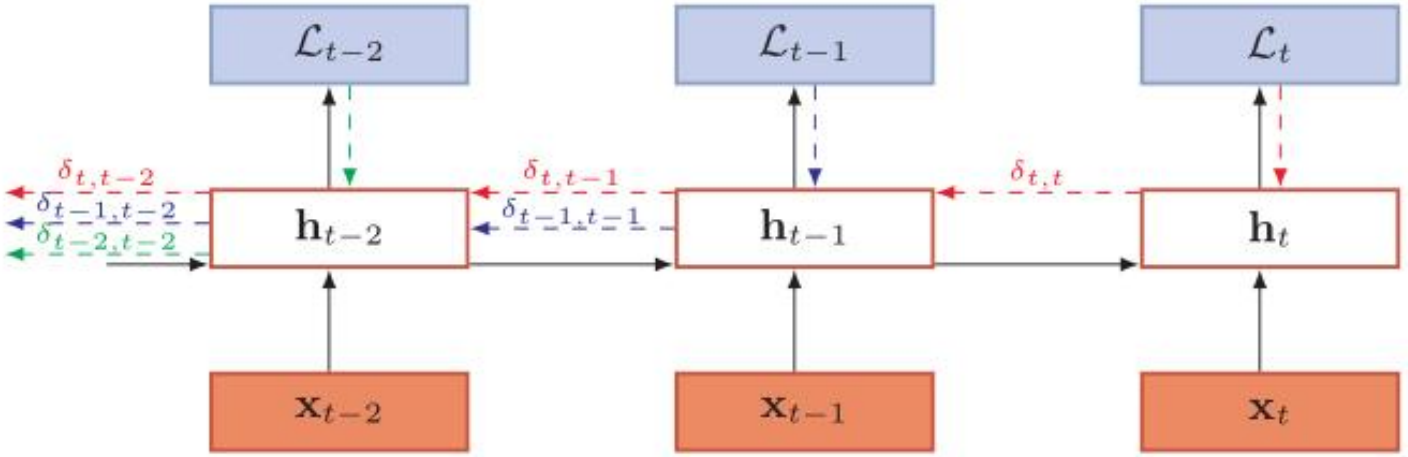
参数学习—计算图（多对一 + 一对多）



梯度计算

➢随时间反向传播算法（BPTT）

$$\boldsymbol{h}_t = f(\boldsymbol{z}_t) = f(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b})$$



$\delta_{t,k}$为第t时刻的损失对第k步隐藏神经元的净输入$\boldsymbol{z}_k$的导数

$$\boldsymbol{\delta}_{t,k} = \frac{\partial \mathcal{L}_t}{\partial \boldsymbol{z}_k} = \frac{\partial \boldsymbol{h}_k}{\partial \boldsymbol{z}_k}\frac{\partial \boldsymbol{z}_{k+1}}{\partial \boldsymbol{h}_k}\frac{\partial \mathcal{L}_t}{\partial \boldsymbol{z}_{k+1}}$$

$$= \mathrm{diag}\big(f'(\boldsymbol{z}_k)\big)\boldsymbol{U}^{\top}\boldsymbol{\delta}_{t,k+1}$$

梯度计算

➢随时间反向传播算法（BPTT）

- 计算偏导数$\frac{\partial \mathcal{L}_t}{\partial \boldsymbol{U}}$

$$\boldsymbol{z}_k = \boldsymbol{U}\boldsymbol{h}_{k-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b}$$

$$\frac{\partial \mathcal{L}_t}{\partial u_{ij}} = \sum_{k}^{t} \frac{\partial \boldsymbol{z}_k}{\partial u_{ij}} \frac{\partial \mathcal{L}_t}{\partial \boldsymbol{z}_k} = \sum_{k=1}^{t} \sum_{l} \frac{\partial z_{k,l}}{\partial u_{ij}} \frac{\partial \mathcal{L}_t}{\partial z_{k,l}}$$

$$= \sum_{k=1}^{t} \sum_{l} \mathbb{I}(l=i)[\boldsymbol{h}_{k-1}]_j [\boldsymbol{\delta}_{t,k}]_l$$

$$= \sum_{k=1}^{t} [\boldsymbol{h}_{k-1}]_j [\boldsymbol{\delta}_{t,k}]_i$$

$$\frac{\partial \mathcal{L}_t}{\partial \boldsymbol{U}} = \sum_{k=1}^{t} \boldsymbol{\delta}_{t,k} \boldsymbol{h}_{k-1}^{\top}$$

同理可得

$$\frac{\partial \mathcal{L}_t}{\partial \boldsymbol{W}} = \sum_{k=1}^{t} \boldsymbol{\delta}_{t,k} \boldsymbol{x}_k^{\top}$$

$$\frac{\partial \mathcal{L}_t}{\partial \boldsymbol{b}} = \sum_{k=1}^{t} \boldsymbol{\delta}_{t,k}$$

梯度计算

- 随时间反向传播算法（BPTT）

$$\boldsymbol{\delta}_{t,k} = \frac{\partial \mathcal{L}_t}{\partial \boldsymbol{z}_k} = \mathrm{diag}\big(f'(\boldsymbol{z}_k)\big)\boldsymbol{U}^{\top}\boldsymbol{\delta}_{t,k+1} = \prod_{\tau=k}^{t-1}\big(\mathrm{diag}\big(f'(\boldsymbol{z}_\tau)\big)\boldsymbol{U}^{\top}\big)\,\boldsymbol{\delta}_{t,t}$$

为了便于分析，假设$f'(\boldsymbol{z}_\tau) = \gamma$

$$= \gamma^{t-k}\boldsymbol{U}^{t-k\top}\boldsymbol{\delta}_{t,t}$$

最大特征值大于1，$t-k$比较大时，**梯度爆炸，**造成系统不稳定；
最大特征值小于1，$t-k$比较大时，梯度变得非常小，出现**梯度消失**

如果$f'(\boldsymbol{z}_\tau) = 1$

$$= \boldsymbol{U}^{t-k\top}\boldsymbol{\delta}_{t,t}$$

谱半径$\rho(\boldsymbol{U})$大于1，$t-k$比较大时，**梯度爆炸，**造成系统不稳定；
谱半径$\rho(\boldsymbol{U})$小于1，$t-k$比较大时，梯度变得非常小，出现**梯度消失**

RNN长程依赖

长程依赖问题

- 由于梯度爆炸或消失问题，实际上只能学习到短周期的依赖关系。这就是所谓的长程依赖问题！

- 循环神经网络中的梯度消失不是$\frac{\partial \mathcal{L}_t}{\partial \boldsymbol{U}}$的梯度消失，而是$\frac{\partial \mathcal{L}_t}{\partial \boldsymbol{h}_k}$的梯度消失

$$\frac{\partial \mathcal{L}_t}{\partial \boldsymbol{h}_k} = \boldsymbol{U}^{\top} \mathrm{diag}\left(f'(\boldsymbol{z}_{k+1})\right) \frac{\partial \mathcal{L}_t}{\partial \boldsymbol{h}_{k+1}}$$

- U的更新主要靠当前时刻t的几个相邻状态$\boldsymbol{h}_k$来更新，长距离的状态对参数几乎没有影响

长程依赖问题

- 循环神经网络在时间维度上非常深！
  - 梯度消失或梯度爆炸

- 如何改进？

梯度爆炸问题	梯度消失问题
• 权重衰减 • 梯度截断	• 改进模型，产生新的RNN结构

Gradient clipping: Scale gradient if its norm is too big

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

长程依赖问题——改进方法

- 可以通过调参，尽量使得 $\boldsymbol{U}^{\top} diag\left(f'(\boldsymbol{z}_{k+1})\right) \approx 1$，但是需要经验
- 循环边改为线性依赖关系

$$\boldsymbol{h}_t = f(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b}) \Rightarrow \boldsymbol{h}_t = \boldsymbol{h}_{t-1} + g(\boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b}) \qquad \frac{\partial \mathcal{L}_t}{\partial \boldsymbol{h}_t} = \frac{\partial \mathcal{L}_t}{\partial \boldsymbol{h}_{t-1}}$$

- 但上述模型只刻画线性依赖关系，降低了模型的表示能力
- 解决办法：增加非线性

$$\boldsymbol{h}_t = \boldsymbol{h}_{t-1} + g(\boldsymbol{U}\boldsymbol{h}_{t-1} + \boldsymbol{W}\boldsymbol{x}_t + \boldsymbol{b})$$

$\boldsymbol{h}_t$和$\boldsymbol{h}_{t-1}$既有线性关系也有非线性关系

缺点

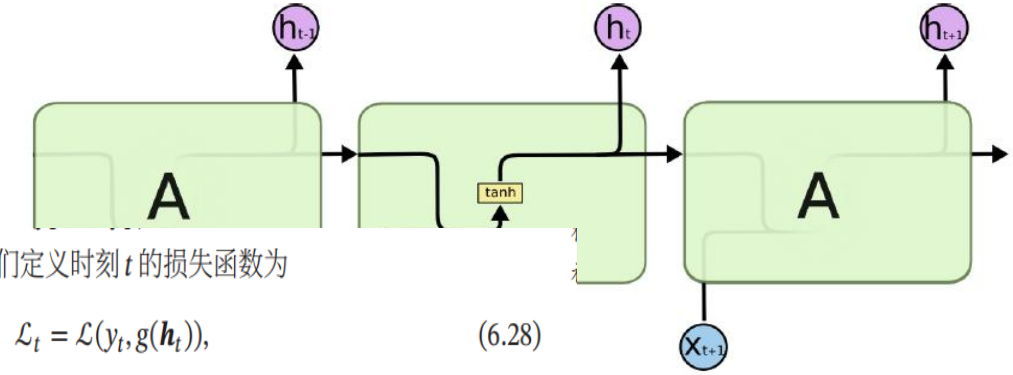
- 梯度爆炸仍存在
- 记忆容量问题，导致h饱和

为了解决这两个问题，可以通过门控机制来进一步改进模型

长程依赖问题——门控机制

➢长短期记忆神经网络（Long Short-Term Memory, LSTM）

Vanilla RNN



$$\boldsymbol{h}_t = f\left(\boldsymbol{W}\begin{pmatrix}\boldsymbol{h}_{t-1}\\ \boldsymbol{x}_t\end{pmatrix}\right)$$

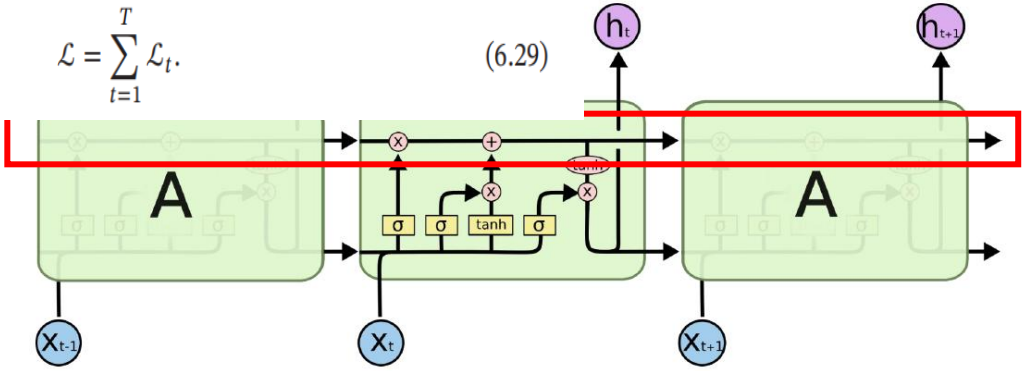
刻t，都有一个监督信息y_t，我们定义时刻t的损失函数为

$$\mathcal{L}_t = \mathcal{L}(y_t, g(\boldsymbol{h}_t)), \qquad (6.28)$$

其中$g(\boldsymbol{h}_t)$为第t时刻的输出，$\mathcal{L}$为可微分的损失函数，比如交叉熵. 那么整个序列的损失函数为

$$\mathcal{L} = \sum_{t=1}^{T} \mathcal{L}_t. \qquad (6.29)$$

LSTM



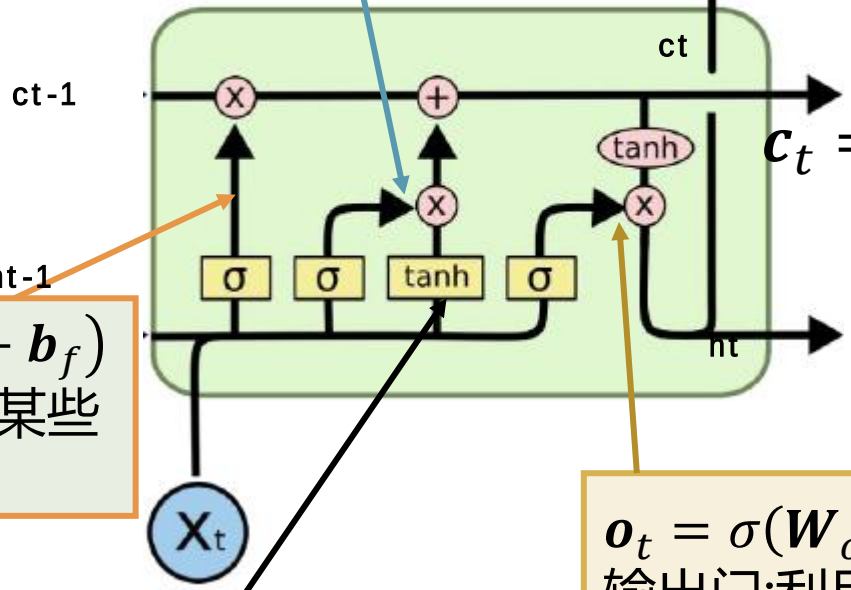
引入一个新的内部状态$\boldsymbol{c}_t$，隐状态$\boldsymbol{h}_t$由该内部状态导出

长短期记忆神经网络（LSTM）

ht与ct的区别
why ht not ct is applied to represent history

$i_t = \sigma(W_i[h_{t-1}, x_t] + b_i)$
输入门:记忆现在的某些信息

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



$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t$

$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f)$
遗忘门:选择忘记过去某些信息

$o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$:
输出门:利用多少存储信息输出

$\tilde{c}_t = \tanh(W_c[h_{t-1}, x_t] + b_c)$
感知层:确定关键和新的信息

长短期记忆神经网络（LSTM）

- 可以更加简洁的描述为

$$\begin{bmatrix} \tilde{\boldsymbol{c}}_t \\ \boldsymbol{o}_t \\ \boldsymbol{i}_t \\ \boldsymbol{f}_t \end{bmatrix} = \begin{bmatrix} tanh \\ \sigma \\ \sigma \\ \sigma \end{bmatrix} \left(\boldsymbol{W} \begin{bmatrix} \boldsymbol{h}_{t-1} \\ \boldsymbol{x}_t \end{bmatrix} + \boldsymbol{b} \right)$$

$$\boldsymbol{c}_t = \boldsymbol{f}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t$$

$$\boldsymbol{h}_t = \boldsymbol{o}_t \odot \tanh(\boldsymbol{c}_t)$$

长短期记忆神经网络（LSTM）

- 循环神经网络的隐状态$\boldsymbol{h}$ 记录了历史信息，看作一种记忆
- 简单循环神经网络中，$\boldsymbol{h}$每个时刻都会被重写，看作是短期记忆
- 神经网络中的参数隐含了从数据中学到的经验，看作是长期记忆
- LSTM网络中，记忆单元$\boldsymbol{c}$可以在某个时刻捕捉到关键信息，并有能力保存一定的时间间隔，生命周期要长期短期记忆，称为长短期记忆

LSTM的各种变体

- 没有遗忘门 $\boldsymbol{c}_t = \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t$

- 耦合输入门和遗忘门 $\boldsymbol{c}_t = (1 - \boldsymbol{i}_t) \odot \boldsymbol{c}_{t-1} + \boldsymbol{i}_t \odot \tilde{\boldsymbol{c}}_t$

- peephole连接：三个门不但依赖于输入$\boldsymbol{x}_t$和$\boldsymbol{h}_{t-1}$，还依赖于$\boldsymbol{c}_{t-1}$

$$\boldsymbol{i}_t = \sigma(\boldsymbol{W}_i[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t, \boldsymbol{c}_{t-1}] + \boldsymbol{b}_i)$$

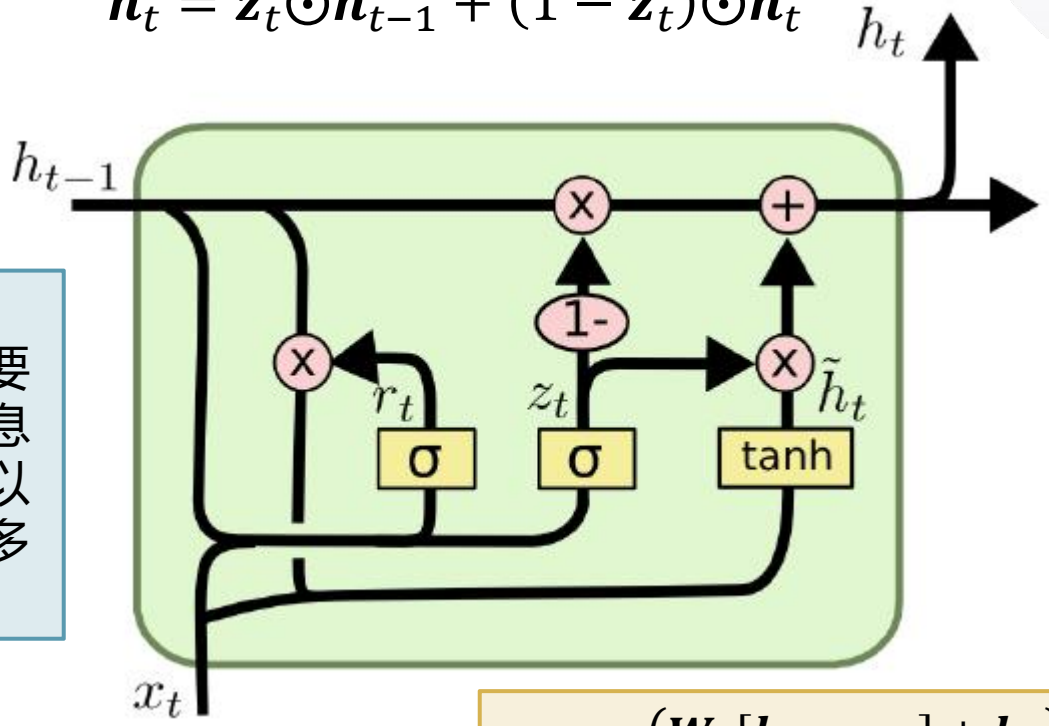
$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_f[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t, \boldsymbol{c}_{t-1}] + \boldsymbol{b}_f\right)$$

$$\boldsymbol{o}_t = \sigma(\boldsymbol{W}_o[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t, \boldsymbol{c}_t] + \boldsymbol{b}_o)$$

门控循环单元（GRU）

- 统一输出单元($\boldsymbol{h}_t$) 和记忆单元($\boldsymbol{c}_t$)，引入更新门

$$\boldsymbol{h}_t = \boldsymbol{z}_t \odot \boldsymbol{h}_{t-1} + (1 - \boldsymbol{z}_t) \odot \tilde{\boldsymbol{h}}_t$$

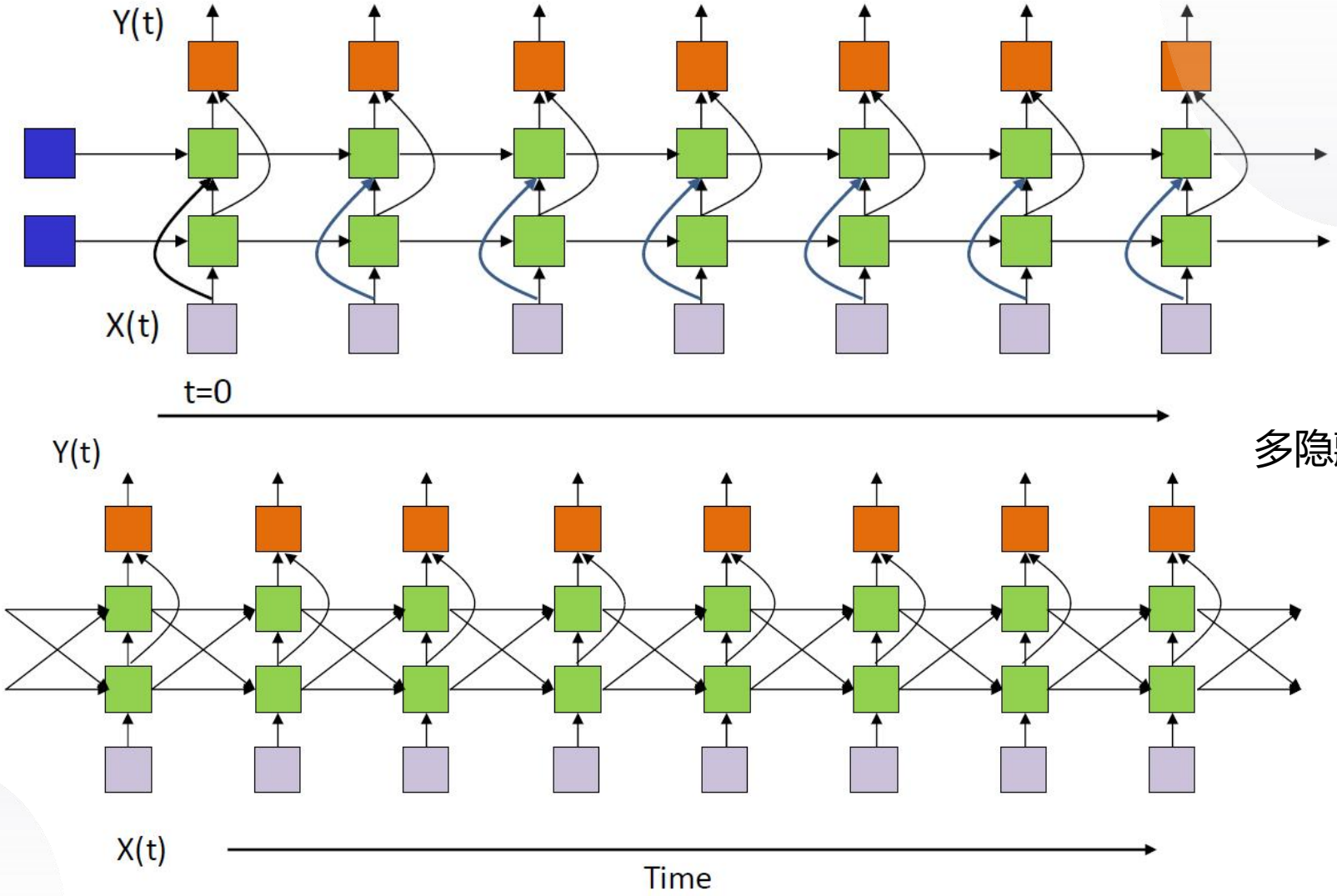


$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_z[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_i)$
更新门：控制当前状态需要从历史状态中保留多少信息（不经过非线性变换），以及需要从候选状态中接受多少新信息

$$\tilde{\boldsymbol{h}}_t = \tanh\left(\boldsymbol{W}_h[\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_c\right)$$

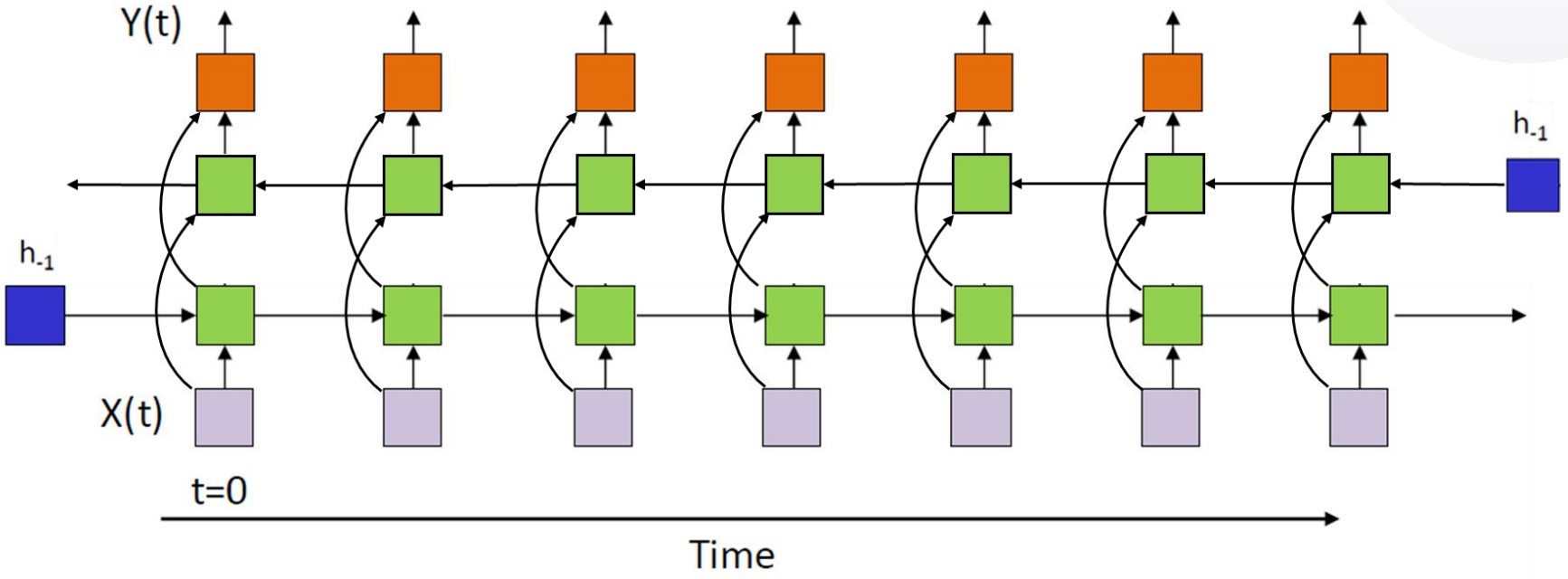
$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_r[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_f)$
重置门：控制候选状态的计算是否依赖上一时刻的状态

深层循环神经网络



多隐藏层

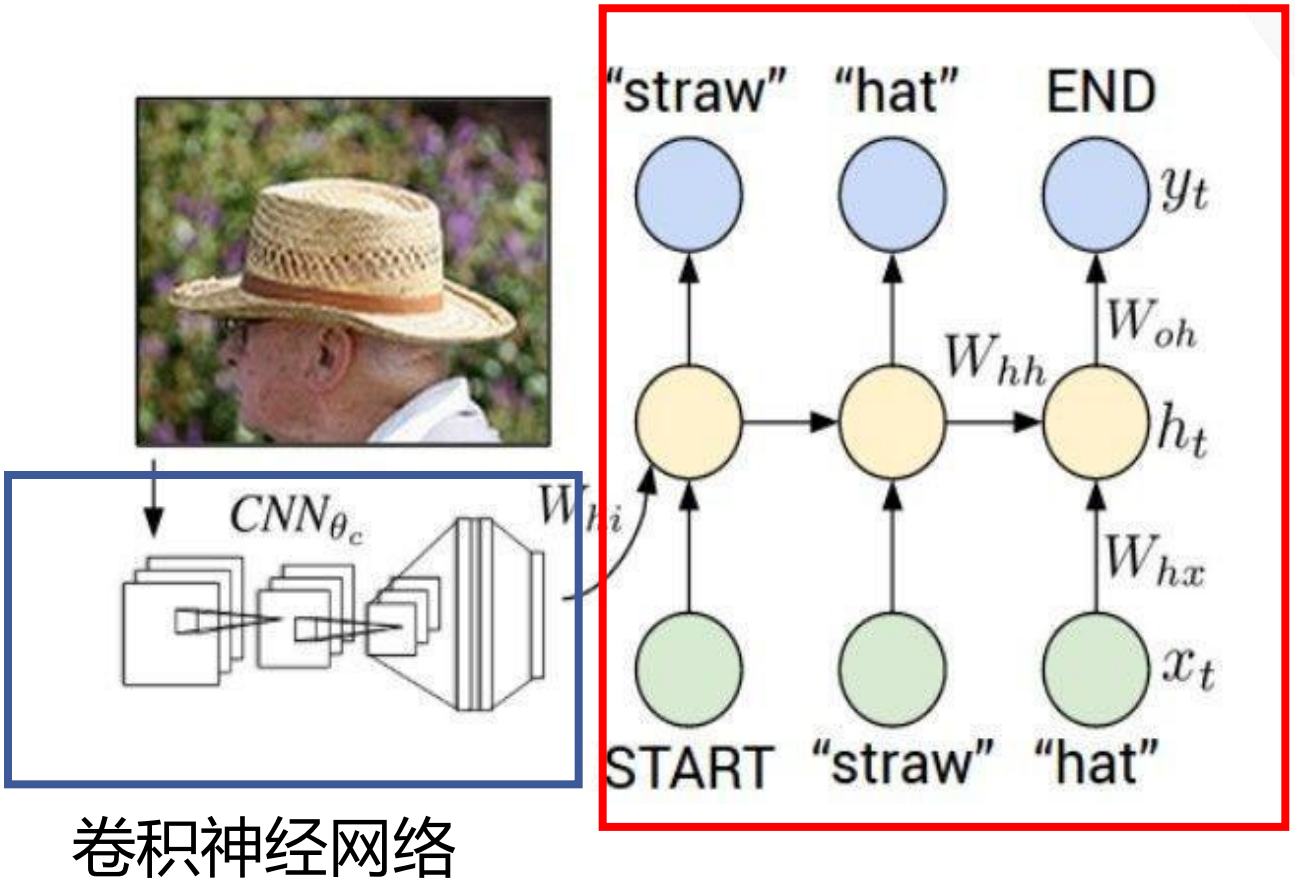
双向循环神经网络



RNN应用

RNN的应用——图像描述

循环神经网络



卷积神经网络

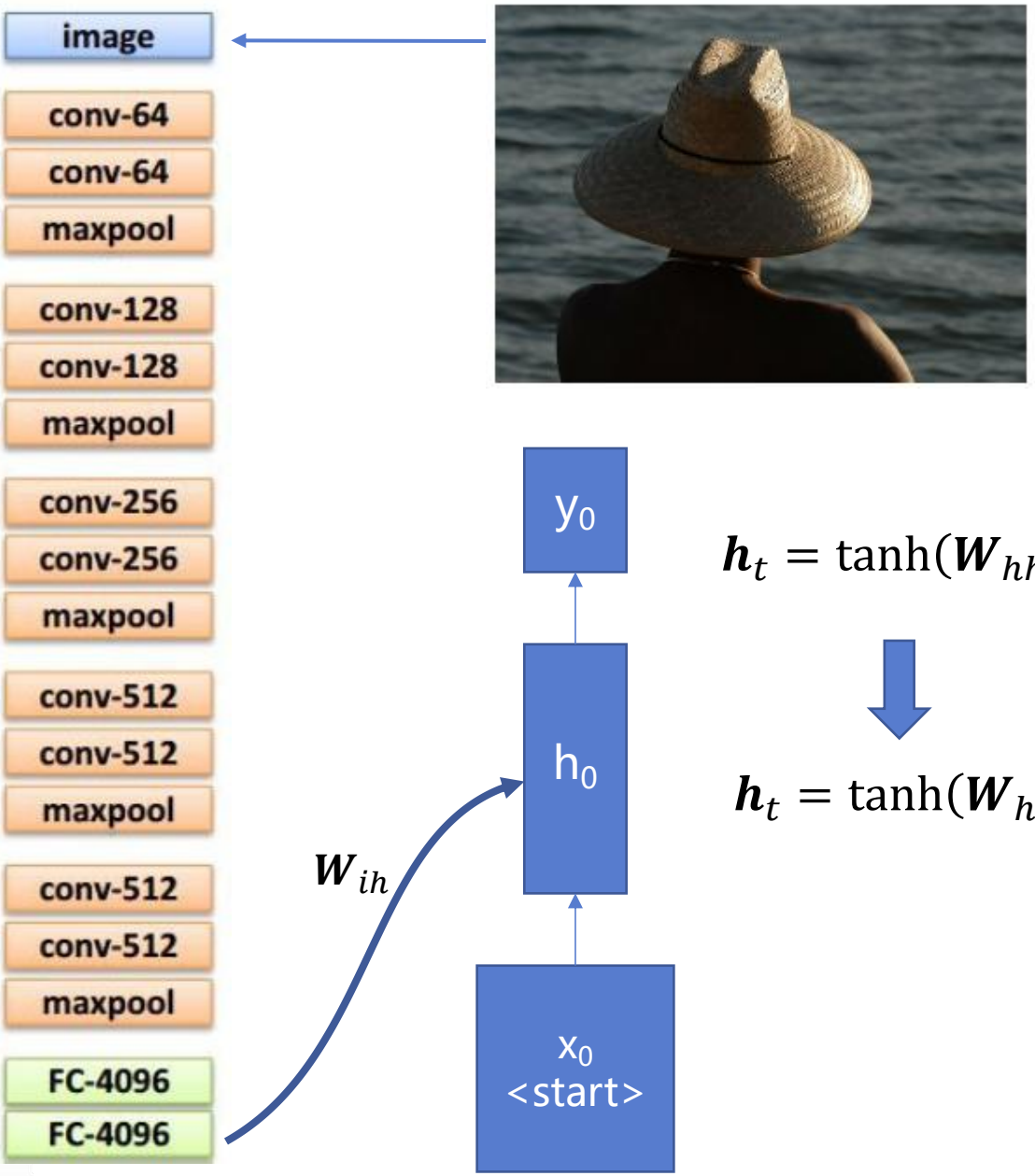


image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
maxpool
conv-512
conv-512
maxpool
conv-512
conv-512
maxpool
FC-4096
FC-4096
$\boldsymbol{v}$
$\boldsymbol{W}_{ih}$
y_0
h_0
x_0
<start>
$\boldsymbol{h}_t = \tanh(\boldsymbol{W}_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{xh}\boldsymbol{x}_t)$
$\boldsymbol{h}_t = \tanh(\boldsymbol{W}_{hh}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{xh}\boldsymbol{x}_t + \boldsymbol{W}_{ih}\boldsymbol{v})$

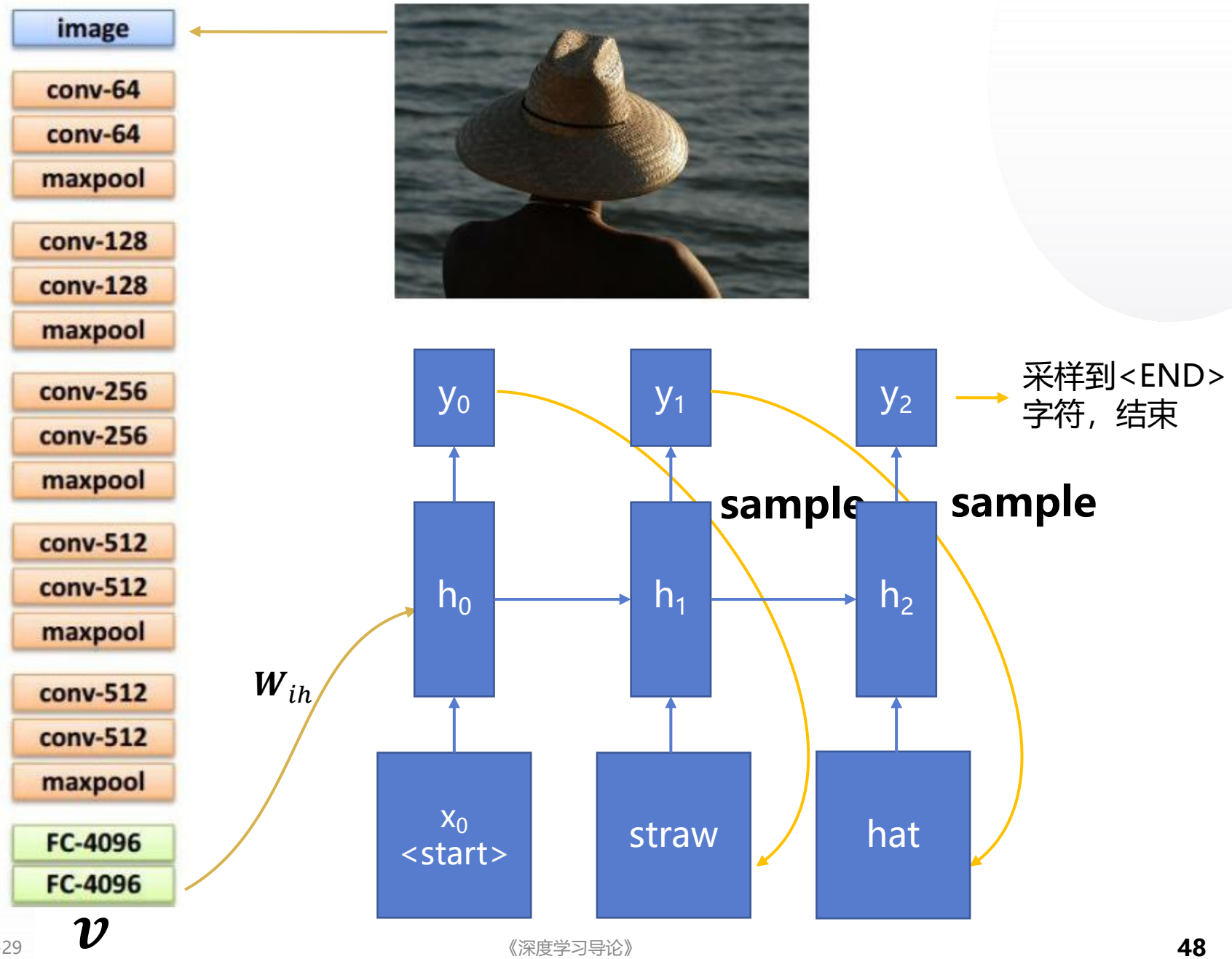


image
conv-64
conv-64
maxpool
conv-128
conv-128
maxpool
conv-256
conv-256
maxpool
conv-512
conv-512
maxpool
conv-512
conv-512
maxpool
FC-4096
FC-4096
v
W_{ih}
y_0
y_1
y_2
采样到<END>
字符，结束
sample
sample
h_0
h_1
h_2
x_0
<start>
straw
hat

RNN的应用——图像描述

A cat sitting on a suitcase on the floor

A cat is sitting on a tree branch

A dog is running in the grass with a frisbee

A white teddy bear sitting in the grass

Two people walking on the beach with surfboards

A tennis player in action on the court

Two giraffes standing in a grassy field

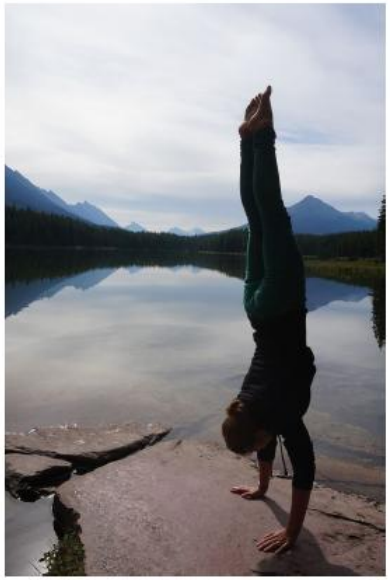
A man riding a dirt bike on a dirt track

RNN的应用——图像描述

失败例子

A woman is holding a cat in her hand

A person holding a computer mouse on a desk



A woman standing on a beach holding a surfboard

A bird is perched on a tree branch

A man in a baseball uniform throwing a ball

RNN的应用——语言模型

- 自然语言理解→一个句子的可能性/合理性

！在报那猫告做只
那只猫在作报告！
那个人在作报告！

$$P(x_1, x_2, \cdots, x_T) = \prod_{i=1}^{T} P(x_i | x_1, \cdots, x_{i-1})$$

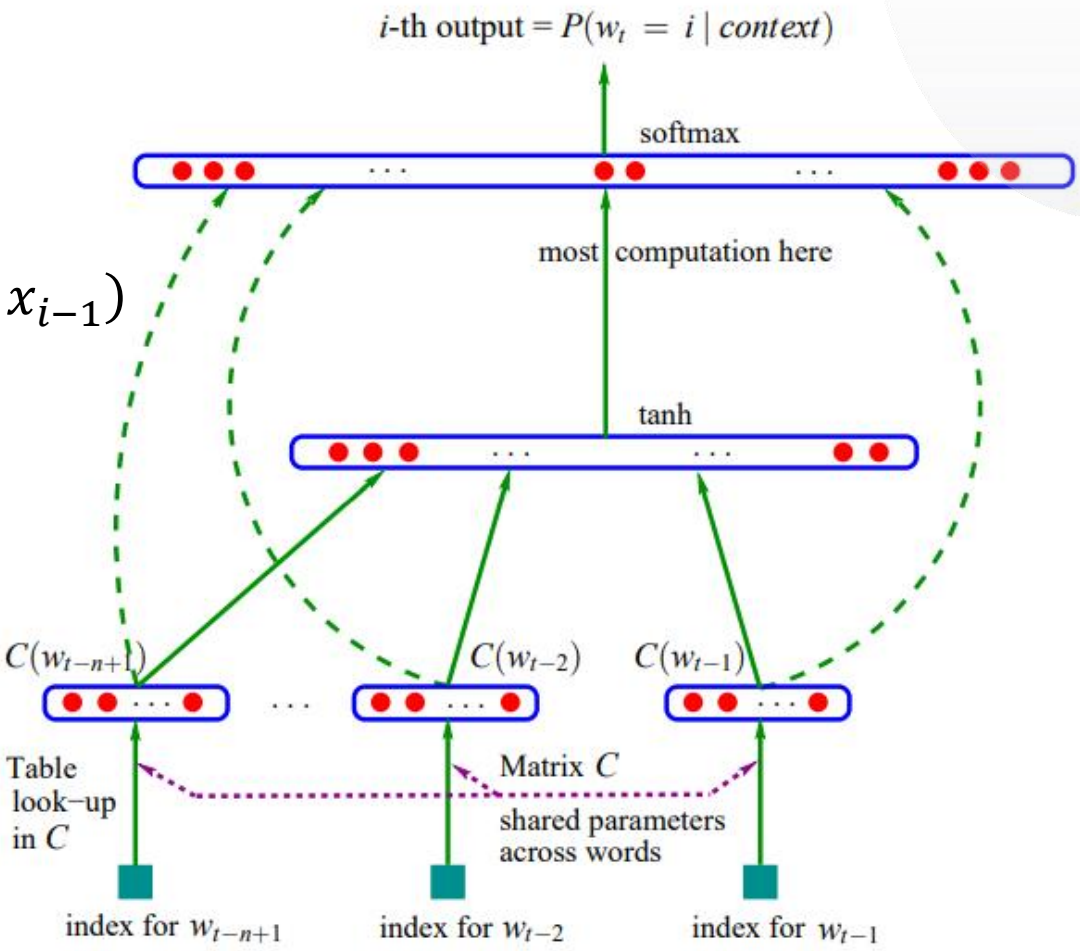
$$\approx \prod_{i=1}^{T} P(x_i | x_{i-n+1}, \cdots, x_{i-1})$$

N元语言模型

RNN的应用——语言模型

- N元语言模型

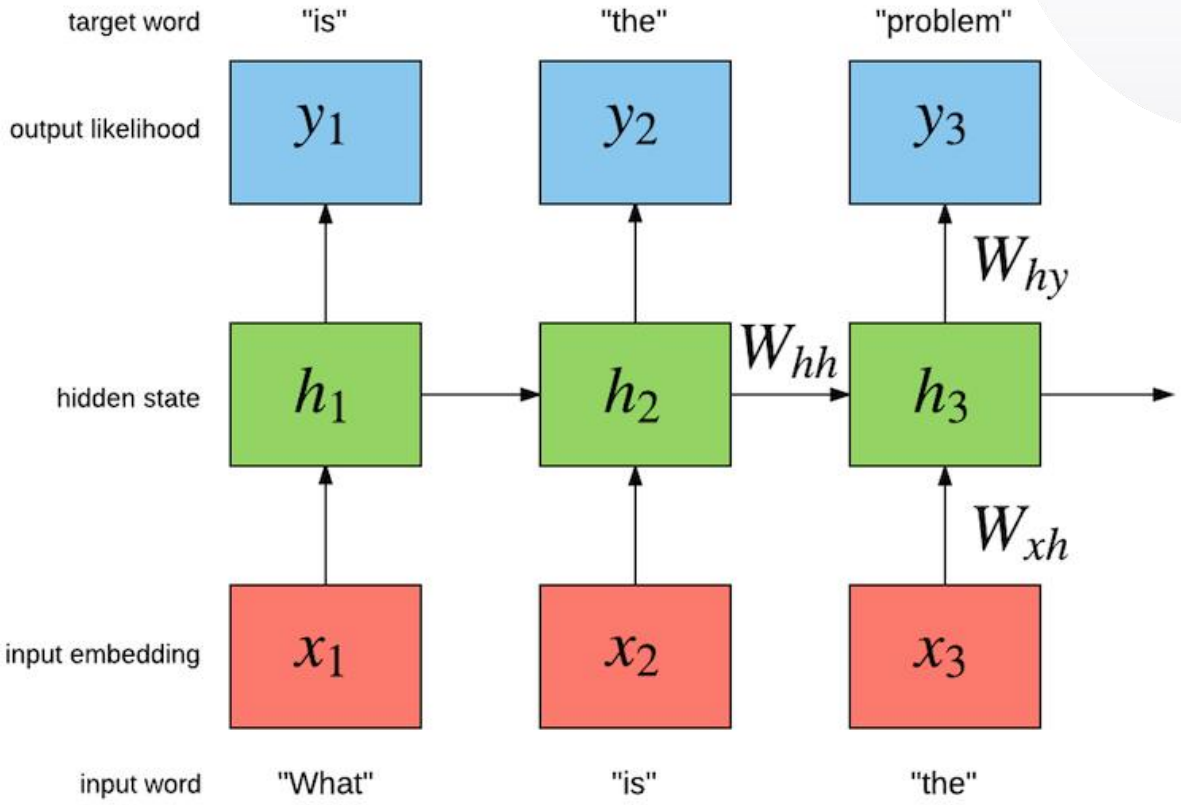
$$P(x_1, x_2, \cdots, x_T)$$
$$\approx \prod_{i=1}^{T} P(x_i | x_{i-n+1}, \cdots, x_{i-1})$$



Bengio, Y., Ducharme, R., Vincent, P., & Janvin, C. (2003). A neural probabilistic language model. The journal of machine learning research, 3, 1137-1155.

RNN的应用——语言模型

- 基于RNN的语言模型



Mikolov, T., Karafiát, M., Burget, L., Černocký, J., & Khudanpur, S. (2010). Recurrent neural network based language model. In Eleventh annual conference of the international speech communication association.

RNN的应用——生成LINUX内核代码

```
/*
 * If this error is set, we will need anything right after that BSD.
 */
static void action_new_function(struct s_stat_info *wb)
{
  unsigned long flags;
  int lel_idx_bit = e->edd, *sys & ~((unsigned long) *FIRST_COMPAT);
  buf[0] = 0xFFFFFFFF & (bit << 4);
  min(inc, slist->bytes);
  printk(KERN_WARNING "Memory allocated %02x/%02x, "
    "original MLL instead\n"),
    min(min(multi_run - s->len, max) * num_data_in),
    frame_pos, sz + first_seg);
  div_u64_w(val, inb_p);
  spin_unlock(&disk->queue_lock);
  mutex_unlock(&s->sock->mutex);
  mutex_unlock(&func->mutex);
  return disassemble(info->pending_bh);
}

static void num_serial_settings(struct tty_struct *tty)
{
  if (tty == tty)
    disable_single_st_p(dev);
  pci_disable_spool(port);
```

RNN的应用——作词

- RNN在“学习”过汪峰全部作品后自动生成的歌词

- https://github.com/phunterlau/wangfeng-rnn

我在这里中的夜里
就像一场是一种生命的意叶
就像我的生活变得在我一样
可我们这是一个知道
我只是一天你会怎吗
可我们这是我们的是不要为你

我们想这有一种生活的时候

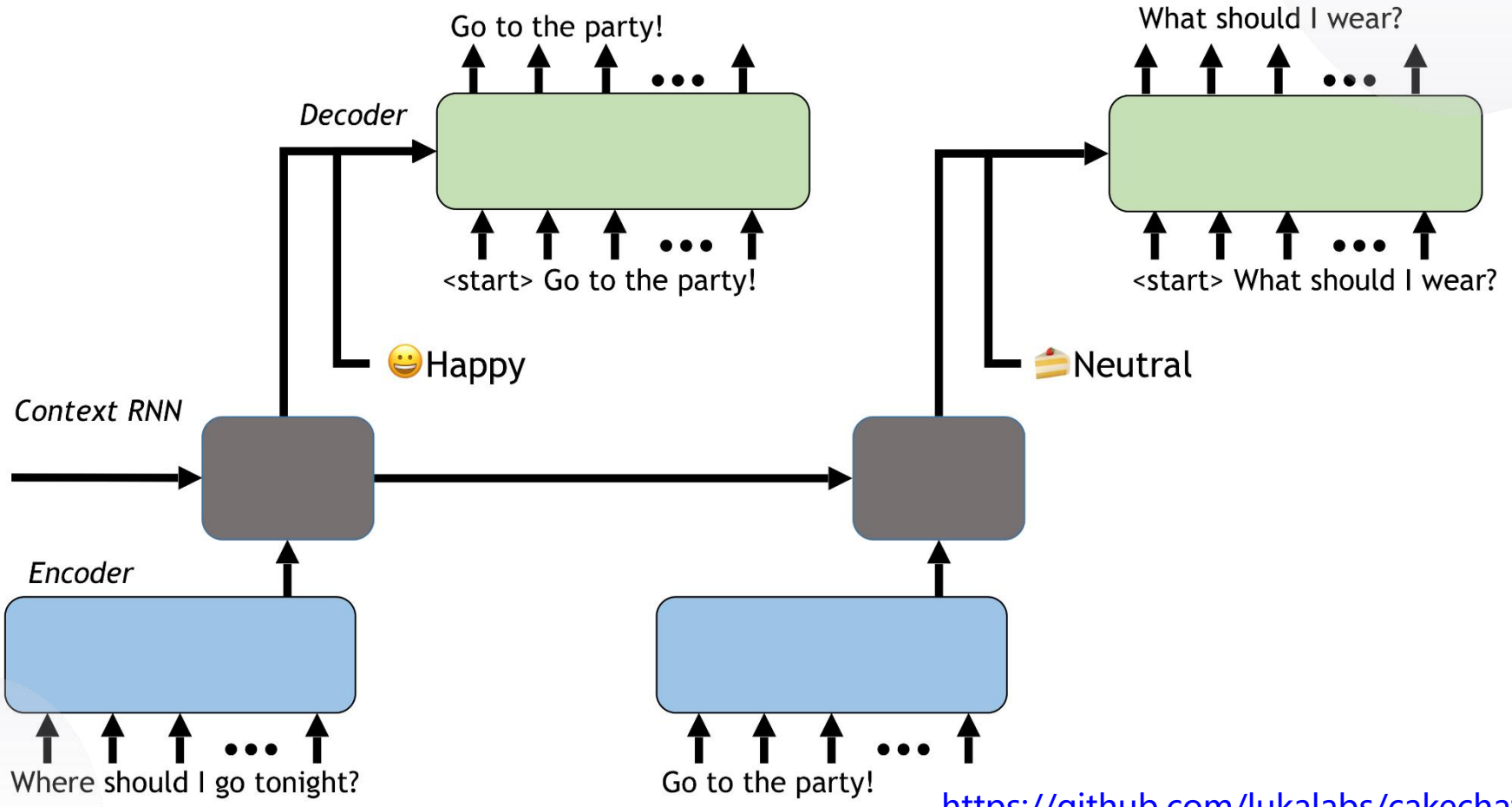
我在哭泣
我不能及你的时光
我是我们在这是一种无法少得可以没有一天
我们想这样
我们远在一场我在我一个相多地在此向
可我是个想已经把我的时候
我看到在心悄一种痛定的时候

我看着我的感觉在飞歌
我不能在这多在心中
就像在我一瞬间
我的生命中的感觉
我在我一次到孤独所
我们在这是一看可以是一场我们在这样
我在这里失命
我不能在这感觉在天里
夜里
夜里
没有一天 我想心的是不吗到了

RNN的应用——作诗

白鹭窥鱼立， Egrets stood, peeping fishes. 青山照水开. Water was still, reflecting mountains. 夜来风不动， The wind went down by nightfall, 明月见楼台. as the moon came up by the tower.	满怀风月一枝春， Budding branches are full of romance. 未见梅花亦可人. Plum blossoms are invisible but adorable. 不为东风无此客， With the east wind comes Spring. 世间何处是前身. Where on earth do I come from?

RNN的应用——对话系统



https://github.com/lukalabs/cakechat

RNN的应用——机器翻译

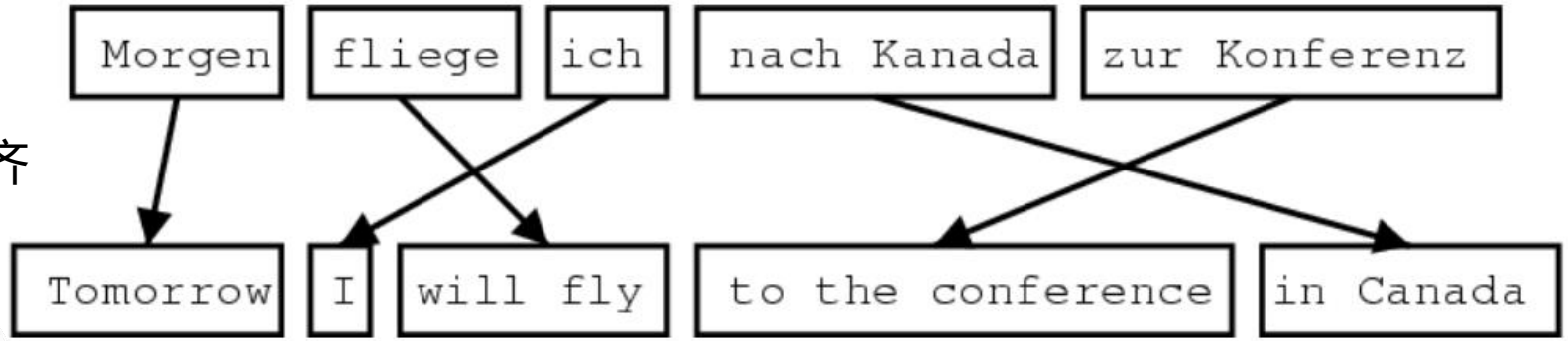
- 假设给定源语言句子：f、目标语言句子：e
- 统计机器翻译模型

$$\hat{e} = \text{argmax}_e \, P(e|f)$$

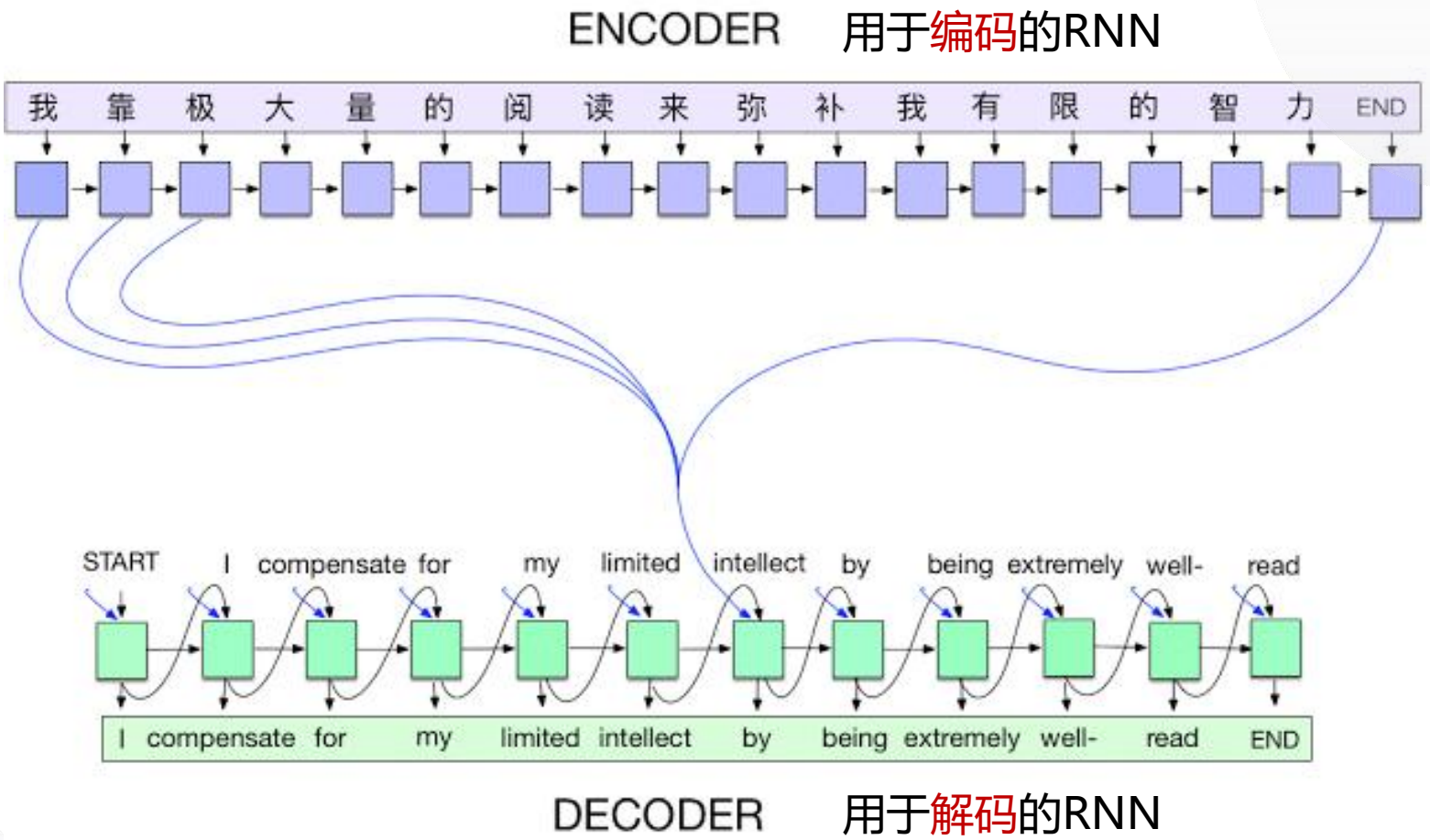
$$= \text{argmax}_e \, P(f|e)P(e)$$

保证翻译的意义 ← 翻译模型　　语言模型 → 保证翻译的流畅

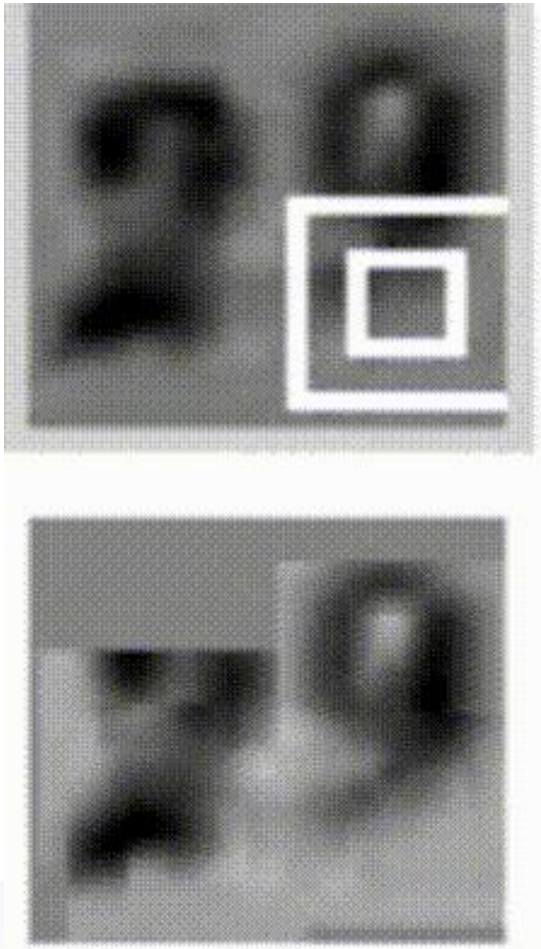
词对齐



RNN的应用——机器翻译


ENCODER
用于编码的RNN
我 靠 极 大 量 的 阅 读 来 弥 补 我 有 限 的 智 力 END
START I compensate for my limited intellect by being extremely well- read
I compensate for my limited intellect by being extremely well- read END
DECODER
用于解码的RNN

非序列数据的序列化处理——图片分类



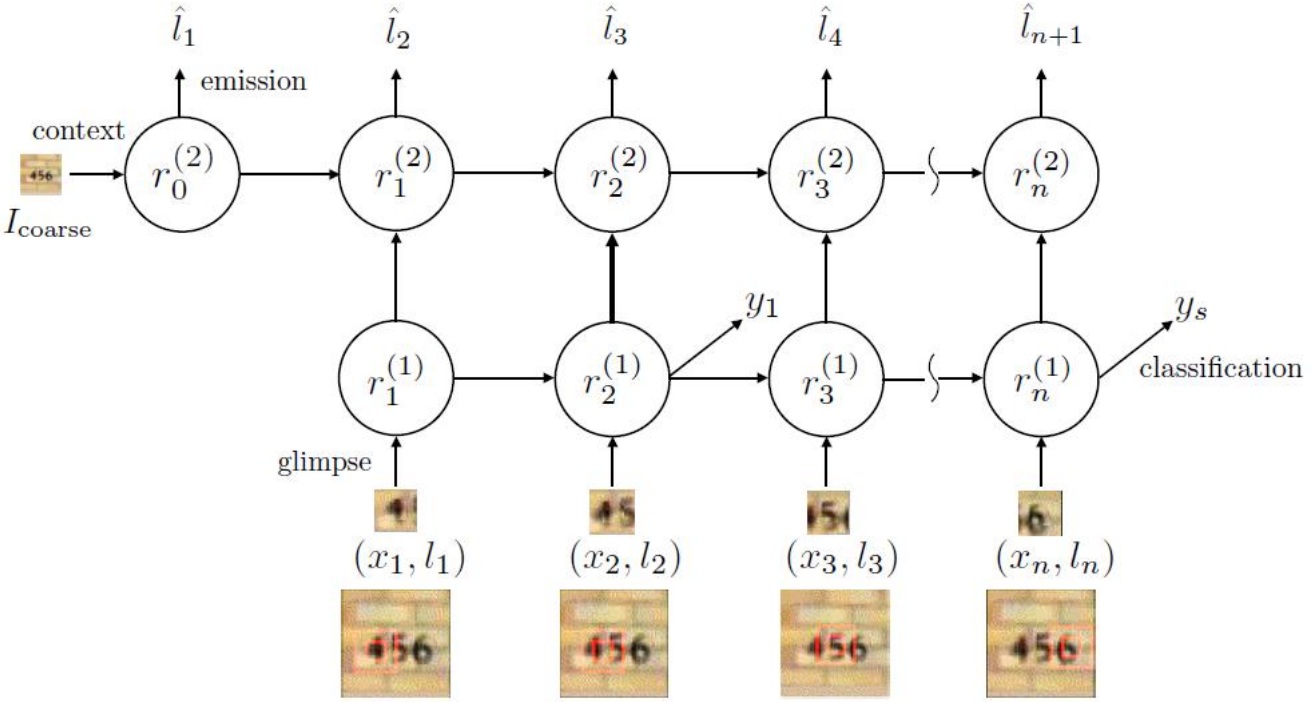
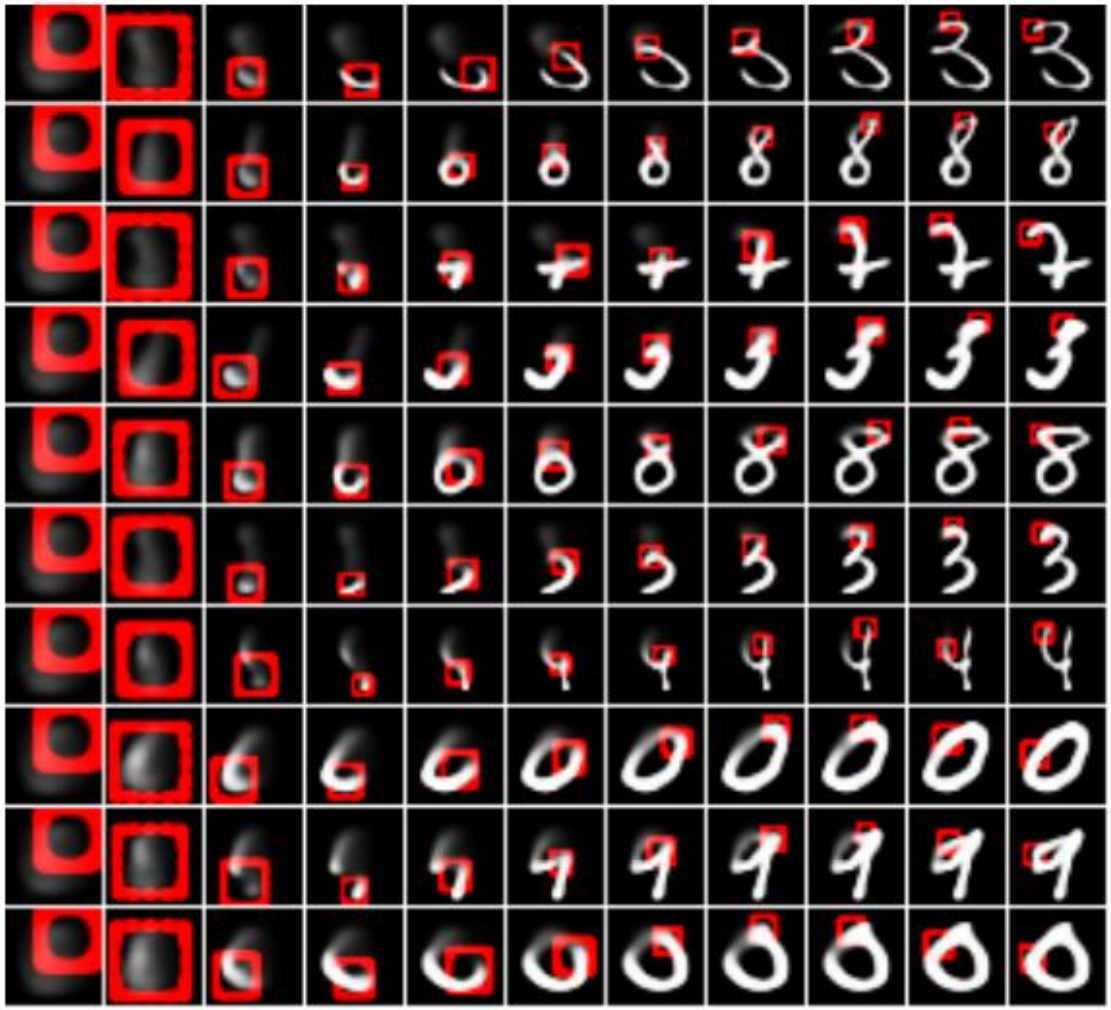


Figure 1: The deep recurrent attention model.

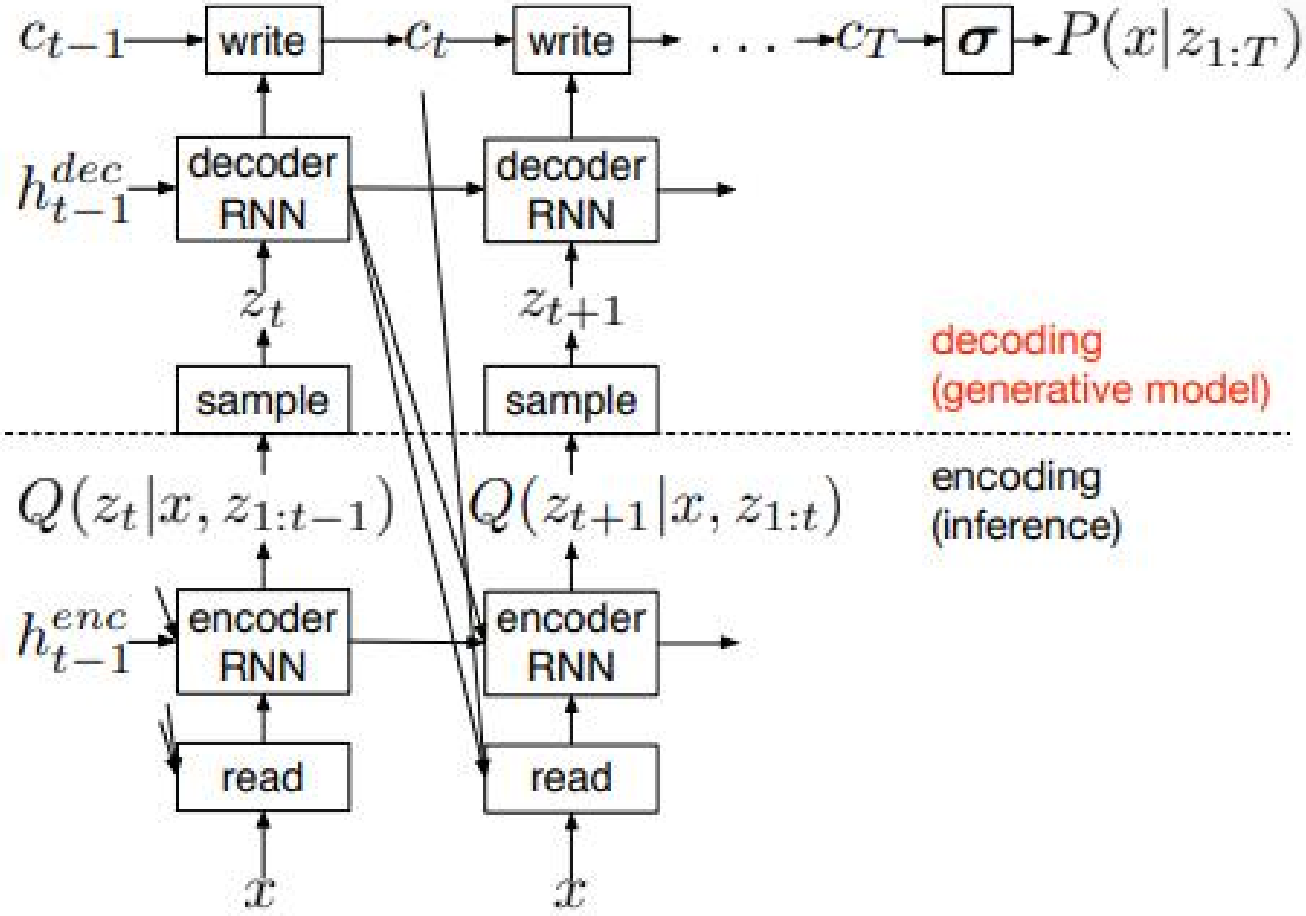
Classify images by taking a series of "glimpses"

Ba, J., Mnih, V., & Kavukcuoglu, K. (2014). Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*.

非序列数据的序列化处理——图片生成



非序列数据的序列化处理——图片生成



非序列数据的序列化处理

Ba, Mnih, and Kavukcuoglu, “Multiple Object Recognition with Visual Attention”, ICLR 2015.
Gregor et al, “DRAW: A Recurrent Neural Network For Image Generation”, ICML 2015