2023春

模式识别与深度学习 (33-34)

深度序列建模-1

左旺孟

综合楼712

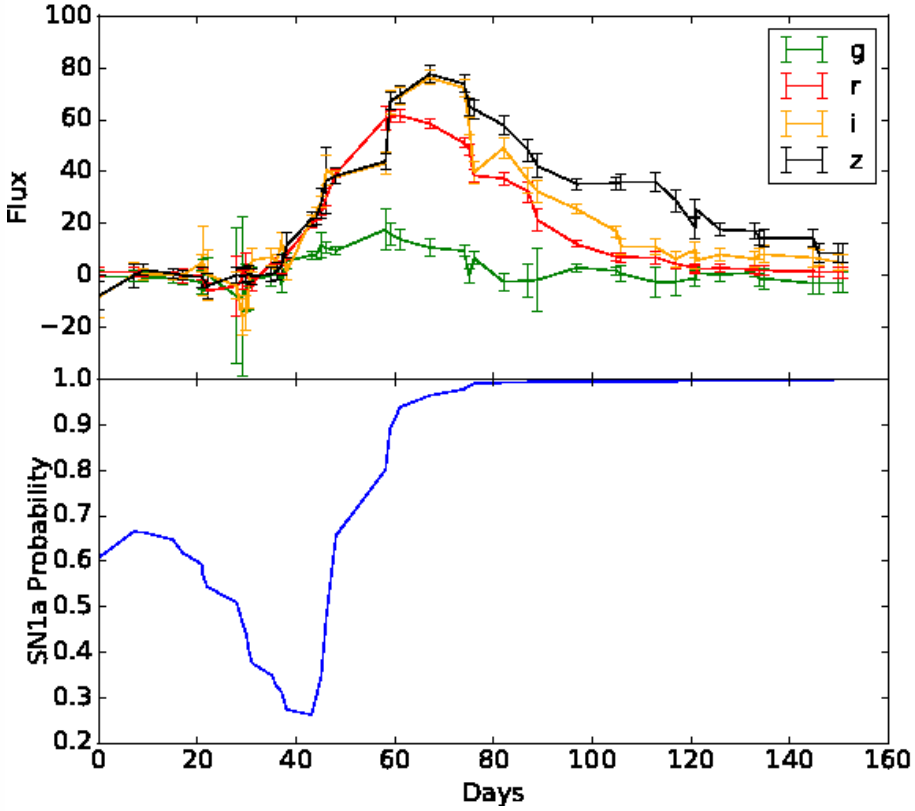
机器学习研究中心

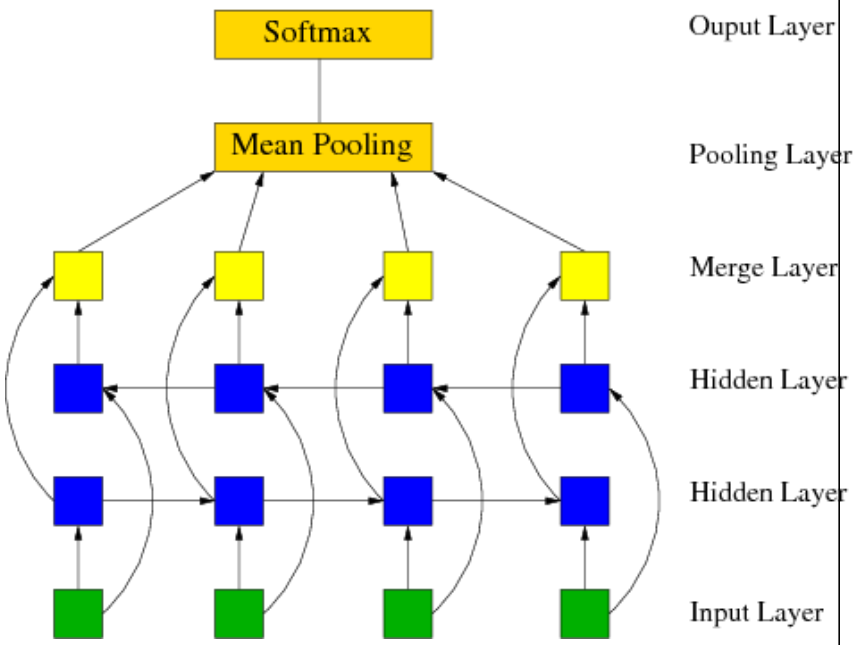
哈尔滨工业大学计算机学院

cswmzuo@gmail.com

13134506692

序列信号识别/分类


Flux
SN1a Probability
Days
g
r
i
z

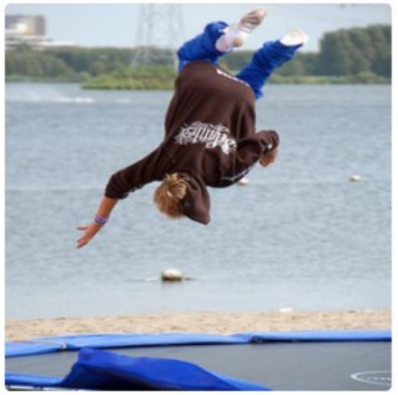

Softmax
Mean Pooling
Ouput Layer
Pooling Layer
Merge Layer
Hidden Layer
Hidden Layer
Input Layer

序列建模问题：图像->自然语言

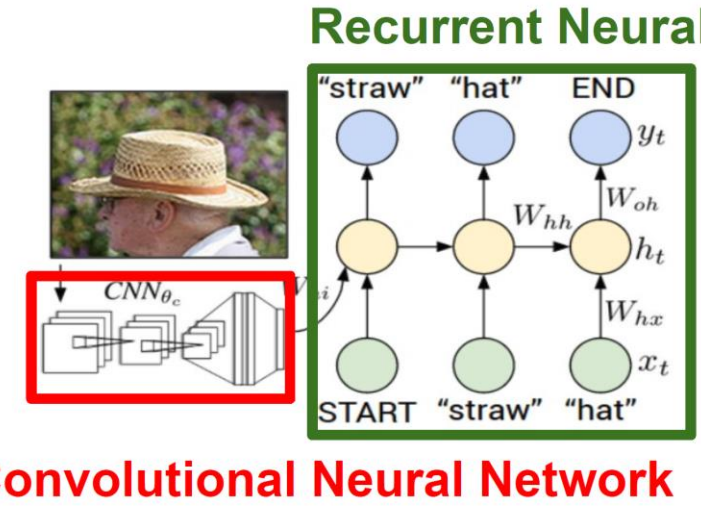
"man in black shirt is playing guitar."

"construction worker in orange safety vest is working on road."

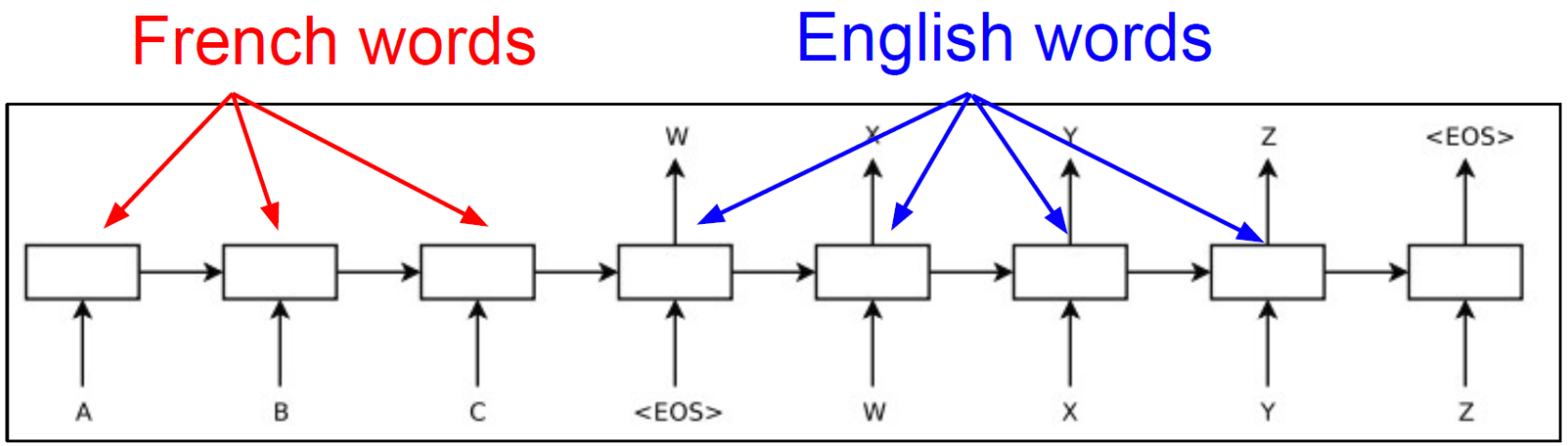
"two young girls are playing with lego toy."



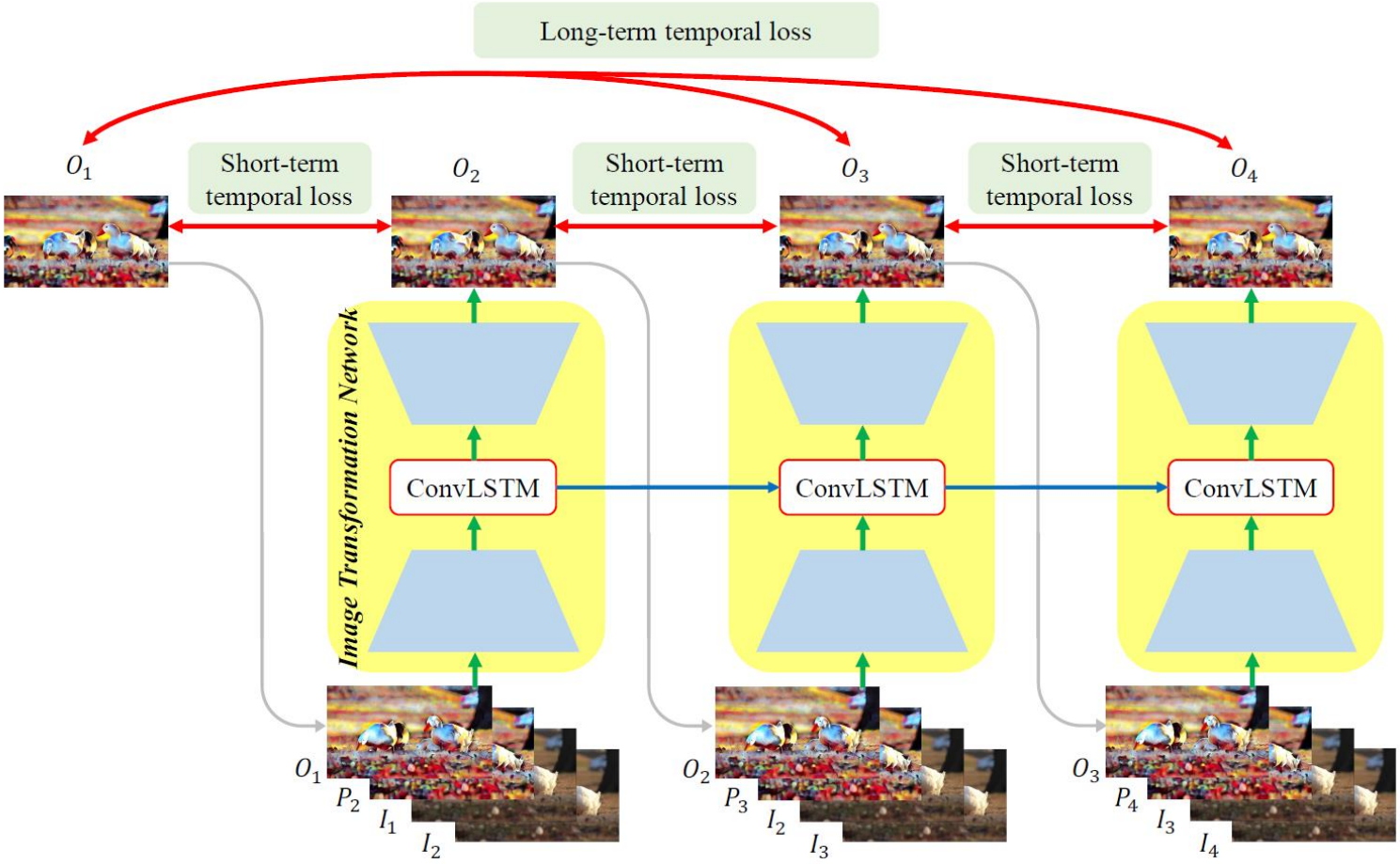
"boy is doing backflip on wakeboard."



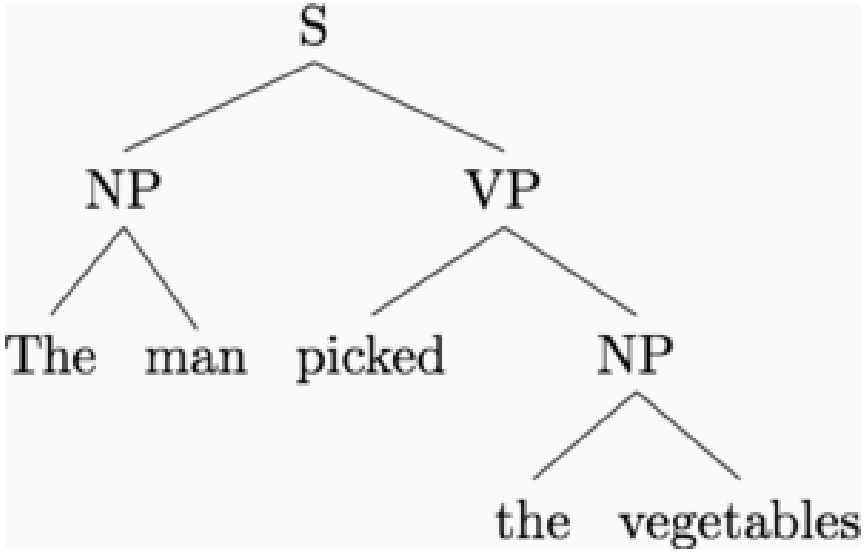
序列建模问题：机器翻译



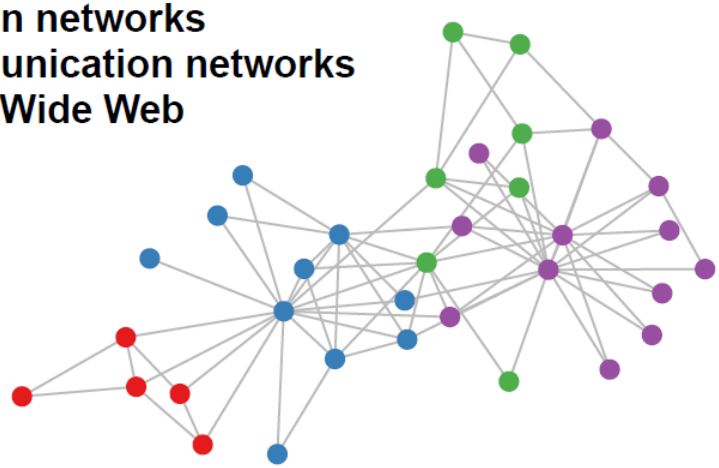
时间序列分析


Long-term temporal loss
O_1
Short-term temporal loss
O_2
Short-term temporal loss
O_3
Short-term temporal loss
O_4
Image Transformation Network
ConvLSTM
ConvLSTM
ConvLSTM
O_1 P_2 I_1 I_2
O_2 P_3 I_2 I_3
O_3 P_4 I_3 I_4

构建树和图



Social networks
Citation networks
Communication networks
World Wide Web



- 序列结构、树结构、图结构
- 输入、输出
- 结构化数据

序列建模

- 循环神经网络
- 递归神经网络
- 回声状态网络
- 记忆网络

循环神经网络

- **循环神经网络**（Recurrent NN）
- 双向RNN
- 序列到序列模型
- 长短期记忆（LSTM）、GRU

循环神经网络

- 循环神经网络
  - 用于处理序列$\boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}, \ldots, \boldsymbol{x}^{(T)}$的神经网络
  - 处理长序列的能力
  - 处理变长序列的能力

  } 参数共享

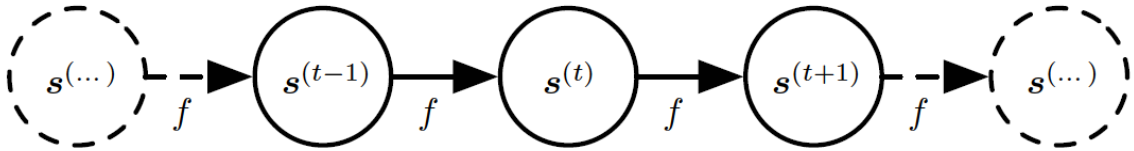
  - 通常在序列的小批量上进行
  - 时间t：序列中的位置

动态系统、计算图及其展开

- 计算图：一组计算结构的形式化表达
  - 输入、参数 -> 输出并计算损失
- 经典动态系统 $\boldsymbol{s}^{(t)} = f(\boldsymbol{s}^{(t-1)}; \boldsymbol{\theta})$
  - 展开（Unfolding）

$$\begin{aligned} \boldsymbol{s}^{(3)} &= f(\boldsymbol{s}^{(2)}; \boldsymbol{\theta}) \\ &= f(f(\boldsymbol{s}^{(1)}; \boldsymbol{\theta}); \boldsymbol{\theta}) \end{aligned}$$

  - 计算图展开



计算图及其展开

- 外部信号驱动的动态系统
  - 当前状态

$$\boldsymbol{s}^{(t)} = f(\boldsymbol{s}^{(t-1)}, \boldsymbol{x}^{(t)}; \boldsymbol{\theta})$$

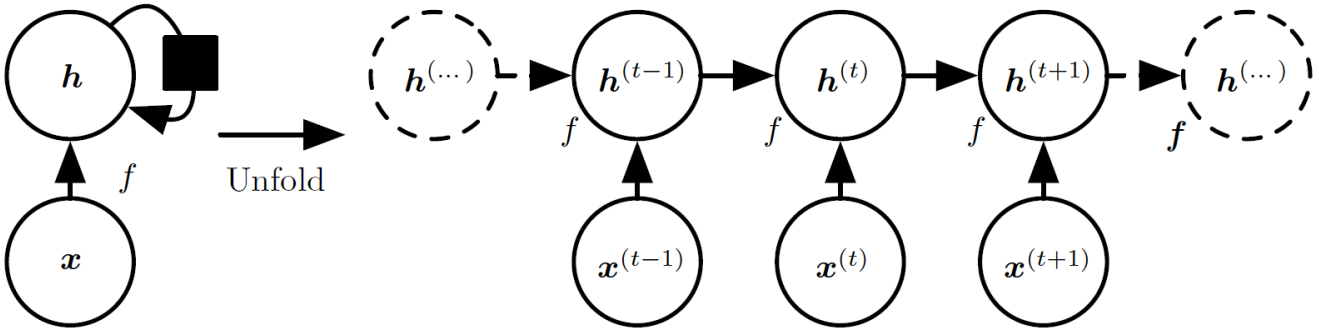
  - 可以包含整个过去系列的信息
  - 隐藏单元$\boldsymbol{h}$

$$\boldsymbol{h}^{(t)} = f(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)}; \boldsymbol{\theta})$$

  - 固定长度
  - 过去序列与任务有关的有损摘要
  - *f*: 转移函数

计算图及其展开

- 外部信号驱动的动态系统
  - 计算图及其展开

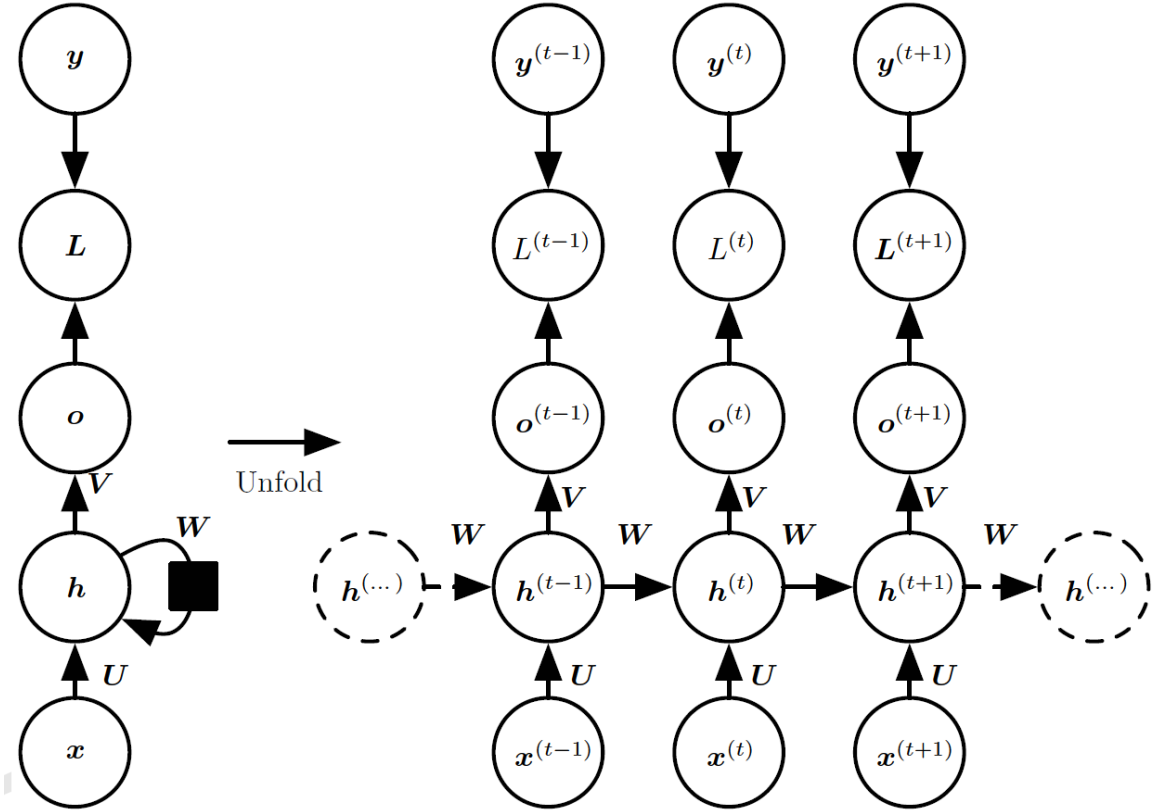


$$\boldsymbol{h}^{(t)} = g^{(t)}(\boldsymbol{x}^{(t)}, \boldsymbol{x}^{(t-1)}, \boldsymbol{x}^{(t-2)}, \ldots, \boldsymbol{x}^{(2)}, \boldsymbol{x}^{(1)})$$
$$= f(\boldsymbol{h}^{(t-1)}, \boldsymbol{x}^{(t)}; \boldsymbol{\theta}).$$

  - 为变长序列学习单一的共享模型
  - 没有考虑最终输出

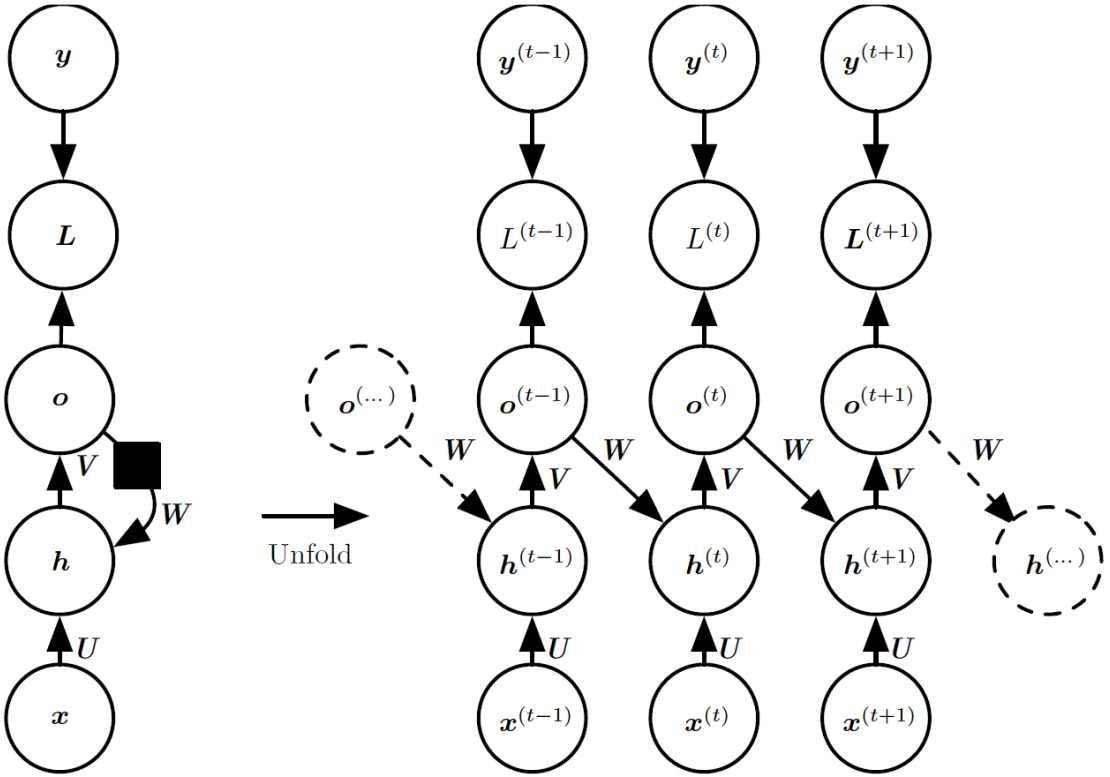
典型RNN设计模式1：考虑输出及损失

- 每个时间步都有输出，并且隐藏单元之间有循环连接的循环网络



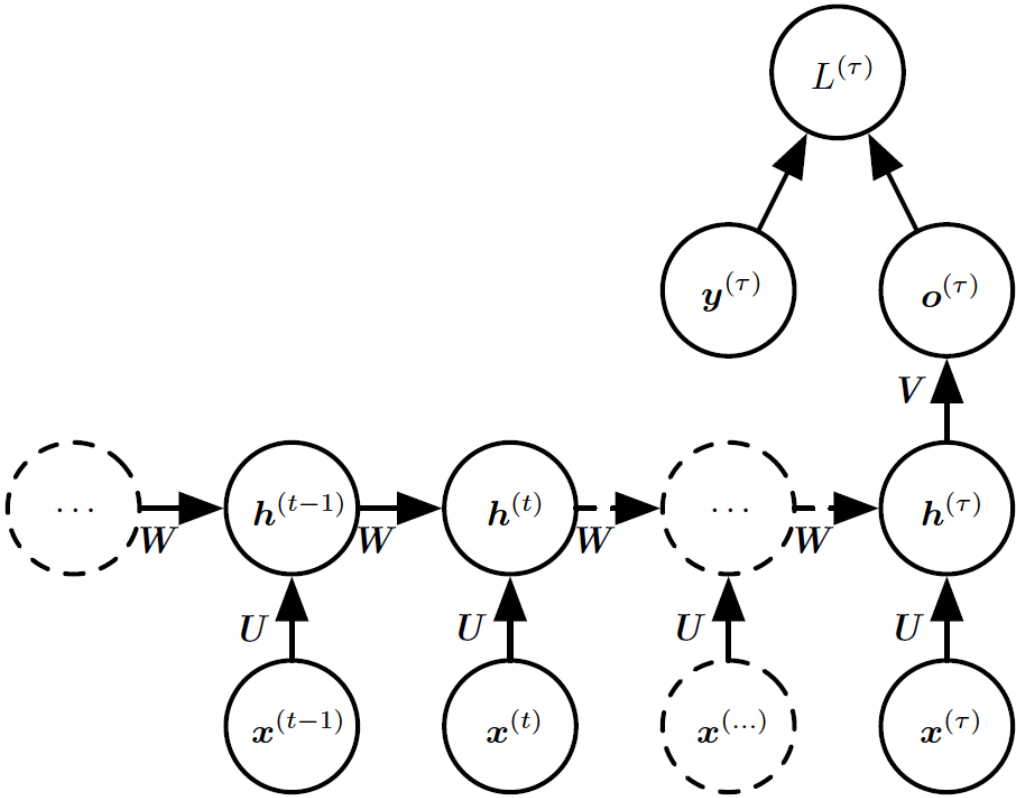
典型RNN设计模式2：考虑输出及损失

- 每个时间步都产生一个输出，只有当前时刻的输出到下个时刻的隐藏单元之间有循环连接的循环网络



典型RNN设计模式3：考虑输出及损失

- 隐藏单元之间存在循环连接，但读取整个序列后产生单个输出的循环网络



前向传播公式：以模式1为例

- 前向传播

$$\begin{aligned}
\boldsymbol{a}^{(t)} &= \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)}, \\
\boldsymbol{h}^{(t)} &= \tanh(\boldsymbol{a}^{(t)}), \\
\boldsymbol{o}^{(t)} &= \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)}, \\
\hat{\boldsymbol{y}}^{(t)} &= \mathrm{softmax}(\boldsymbol{o}^{(t)}),
\end{aligned}$$

- 损失函数

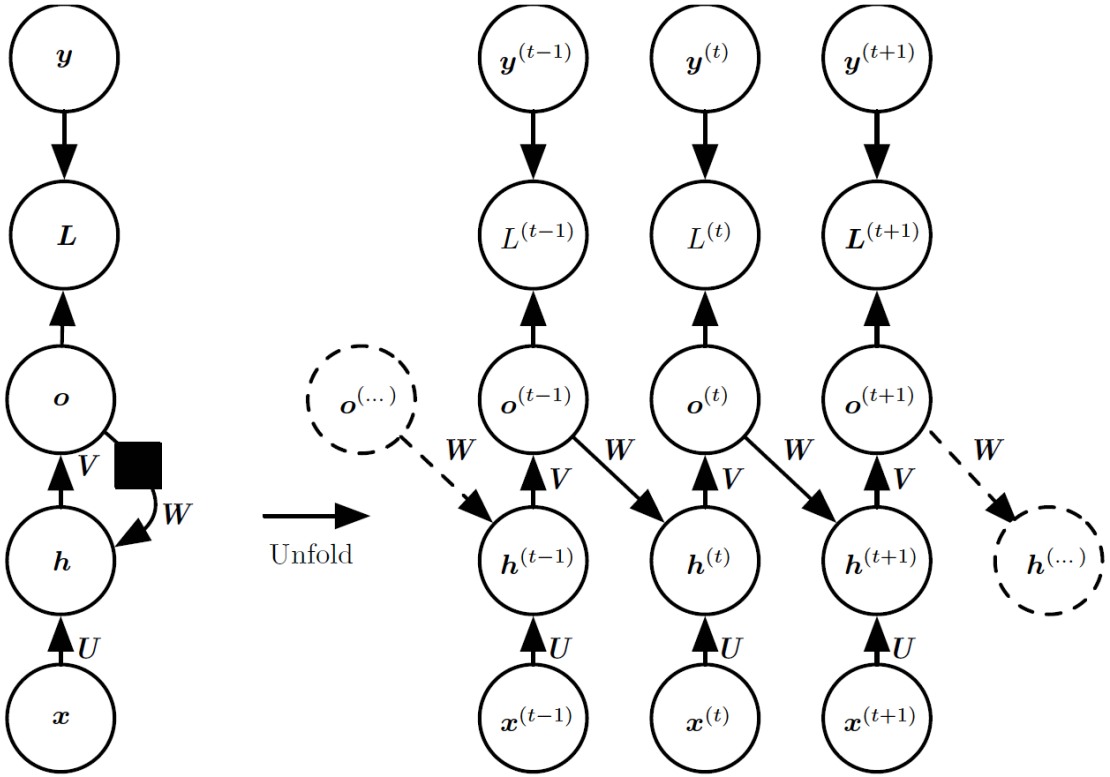
$$\begin{aligned}
&L\left(\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(\tau)}\}, \{\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(\tau)}\}\right) \\
&= \sum_t L^{(t)} \\
&= -\sum_t \log p_{\text{model}}\left(y^{(t)} \mid \{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(t)}\}\right),
\end{aligned}$$

- 梯度计算：通过时间反向传播（BPTT）
  - 计算复杂性： $\mathcal{O}(\tau)$
  - 存储复杂性： $\mathcal{O}(\tau)$

RNN训练

- 导师驱动过程
- 通过时间反向传播（BPTT）
- 作为有向图模型的循环网络
- 基于上下文的**RNN** 序列建模

RNN设计模式2



- 导师驱动过程：在时刻$t+1$接收真实值$y(t)$作为输入

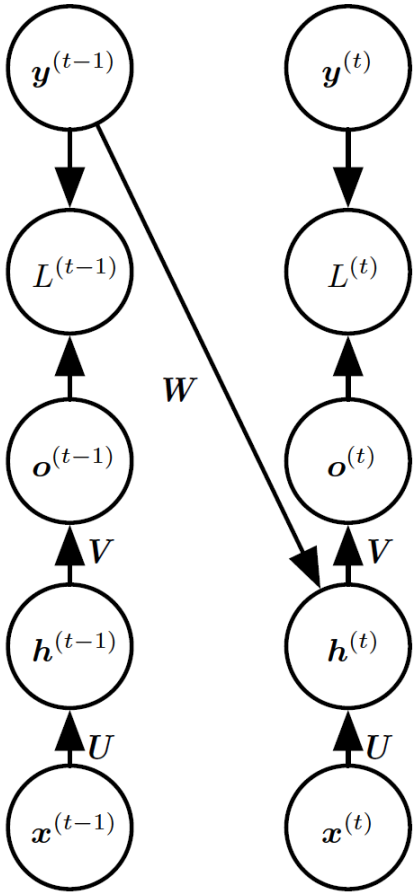
导师驱动过程

- 条件最大似然准则

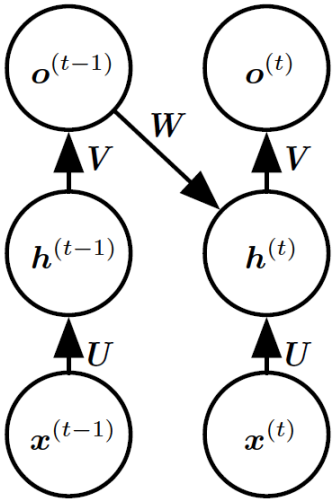
$$\log p(\boldsymbol{y}^{(1)}, \boldsymbol{y}^{(2)} \mid \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)})$$
$$= \log p(\boldsymbol{y}^{(2)} \mid \boldsymbol{y}^{(1)}, \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)}) + \log p(\boldsymbol{y}^{(1)} \mid \boldsymbol{x}^{(1)}, \boldsymbol{x}^{(2)})$$

- 避免通过时间反向传播

导师驱动过程



Train time



Test time

闭环模式训练到开环模式应用的不一致性

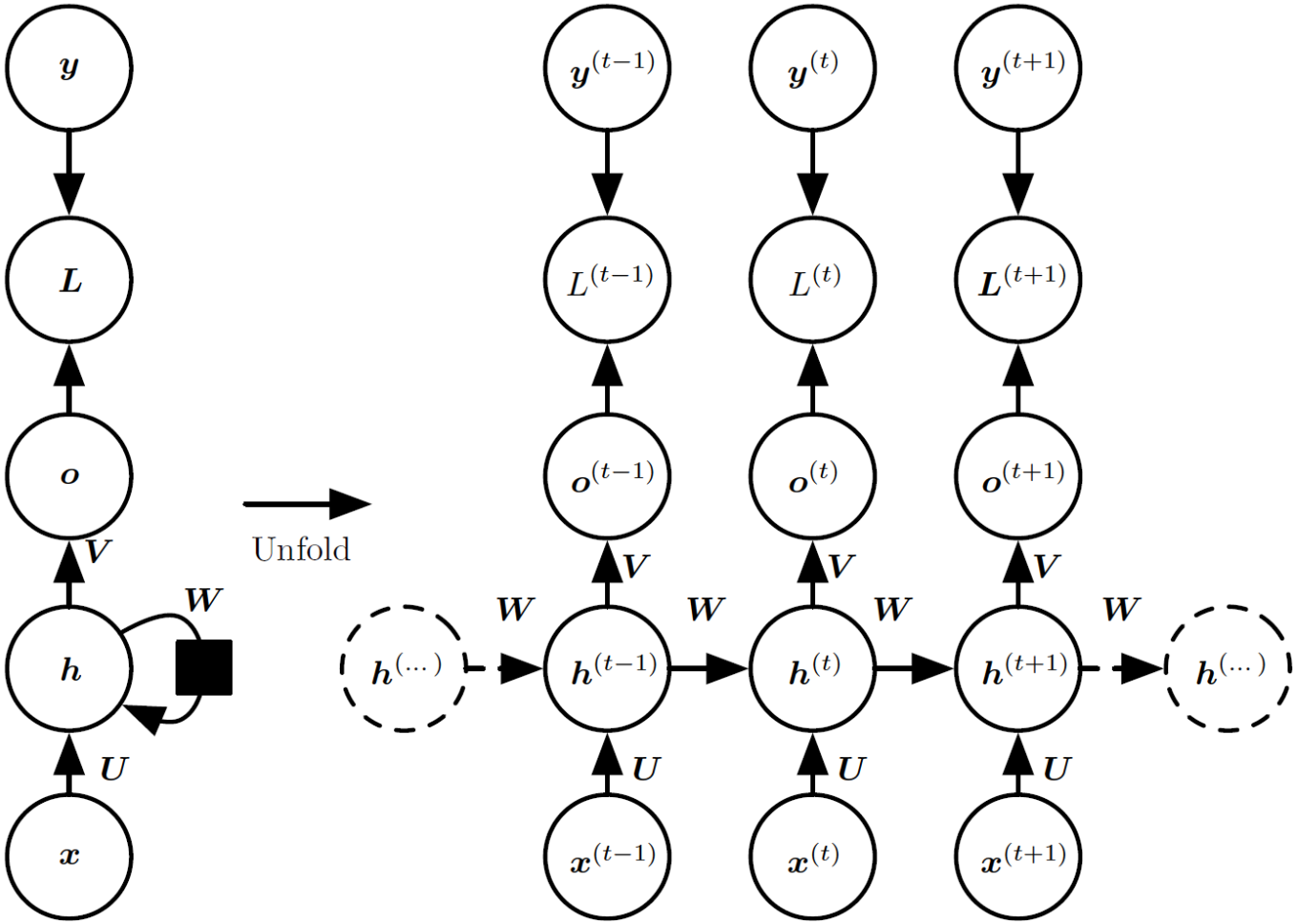
导师驱动过程：改进

- 同时使用导师驱动过程和自由运行的输入进行训练
- 随机选择生成值或真实的数据值作为输入
  - 结合课程学习：逐步增加使用生成值作为输入的概率

RNN训练

- 导师驱动过程
- 通过时间反向传播（BPTT）
- 作为有向图模型的循环网络
- 基于上下文的**RNN** 序列建模

BPTT：计算循环神经网络的梯度



BPTT：计算循环神经网络的梯度

$$
\begin{aligned}
&L\big(\{\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(\tau)}\},\{\boldsymbol{y}^{(1)},\ldots,\boldsymbol{y}^{(\tau)}\}\big)\\
&=\sum_t L^{(t)}\\
&=-\sum_t \log p_{\text{model}}\big(y^{(t)}\mid\{\boldsymbol{x}^{(1)},\ldots,\boldsymbol{x}^{(t)}\}\big)
\end{aligned}
$$

$$
\begin{aligned}
\boldsymbol{a}^{(t)} &= \boldsymbol{b}+\boldsymbol{W}\boldsymbol{h}^{(t-1)}+\boldsymbol{U}\boldsymbol{x}^{(t)},\\
\boldsymbol{h}^{(t)} &= \tanh(\boldsymbol{a}^{(t)}),\\
\boldsymbol{o}^{(t)} &= \boldsymbol{c}+\boldsymbol{V}\boldsymbol{h}^{(t)},\\
\hat{\boldsymbol{y}}^{(t)} &= \text{softmax}(\boldsymbol{o}^{(t)}),
\end{aligned}
$$

- 参数：$\boldsymbol{U}, \boldsymbol{V}, \boldsymbol{W}, \boldsymbol{b}$ 和 $\boldsymbol{c}$
- 节点序列：$\boldsymbol{x}^{(t)}, \boldsymbol{h}^{(t)}, \boldsymbol{o}^{(t)}$ 和 $L^{(t)}$

梯度计算: L -> o -> h -> x

- $\frac{\partial L}{\partial L^{(t)}} = 1$

- softmax 函数

$$(\nabla_{\boldsymbol{o}^{(t)}} L)_i = \frac{\partial L}{\partial o_i^{(t)}} = \frac{\partial L}{\partial L^{(t)}} \frac{\partial L^{(t)}}{\partial o_i^{(t)}} = \hat{y}_i^{(t)} - \mathbf{1}_{i,y^{(t)}}$$

- $\nabla_{\boldsymbol{h}^{(\tau)}} L = \boldsymbol{V}^\top \nabla_{\boldsymbol{o}^{(\tau)}} L$

$$\nabla_{\boldsymbol{h}^{(t)}} L = \left(\frac{\partial \boldsymbol{h}^{(t+1)}}{\partial \boldsymbol{h}^{(t)}}\right)^\top (\nabla_{\boldsymbol{h}^{(t+1)}} L) + \left(\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{h}^{(t)}}\right)^\top (\nabla_{\boldsymbol{o}^{(t)}} L)$$

$$= \boldsymbol{W}^\top (\nabla_{\boldsymbol{h}^{(t+1)}} L) \mathrm{diag}\left(1 - (\boldsymbol{h}^{(t+1)})^2\right) + \boldsymbol{V}^\top (\nabla_{\boldsymbol{o}^{(t)}} L),$$

梯度计算: 参数

$$\boldsymbol{a}^{(t)} = \boldsymbol{b} + \boldsymbol{W}\boldsymbol{h}^{(t-1)} + \boldsymbol{U}\boldsymbol{x}^{(t)},$$
$$\boldsymbol{h}^{(t)} = \tanh(\boldsymbol{a}^{(t)}),$$
$$\boldsymbol{o}^{(t)} = \boldsymbol{c} + \boldsymbol{V}\boldsymbol{h}^{(t)},$$
$$\hat{\boldsymbol{y}}^{(t)} = \mathrm{softmax}(\boldsymbol{o}^{(t)}),$$

$$\nabla_{\boldsymbol{c}} L = \sum_t \left(\frac{\partial \boldsymbol{o}^{(t)}}{\partial \boldsymbol{c}}\right)^\top \nabla_{\boldsymbol{o}^{(t)}} L = \sum_t \nabla_{\boldsymbol{o}^{(t)}} L,$$

$$\nabla_{\boldsymbol{b}} L = \sum_t \left(\frac{\partial \boldsymbol{h}^{(t)}}{\partial \boldsymbol{b}^{(t)}}\right)^\top \nabla_{\boldsymbol{h}^{(t)}} L = \sum_t \mathrm{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right) \nabla_{\boldsymbol{h}^{(t)}} L,$$

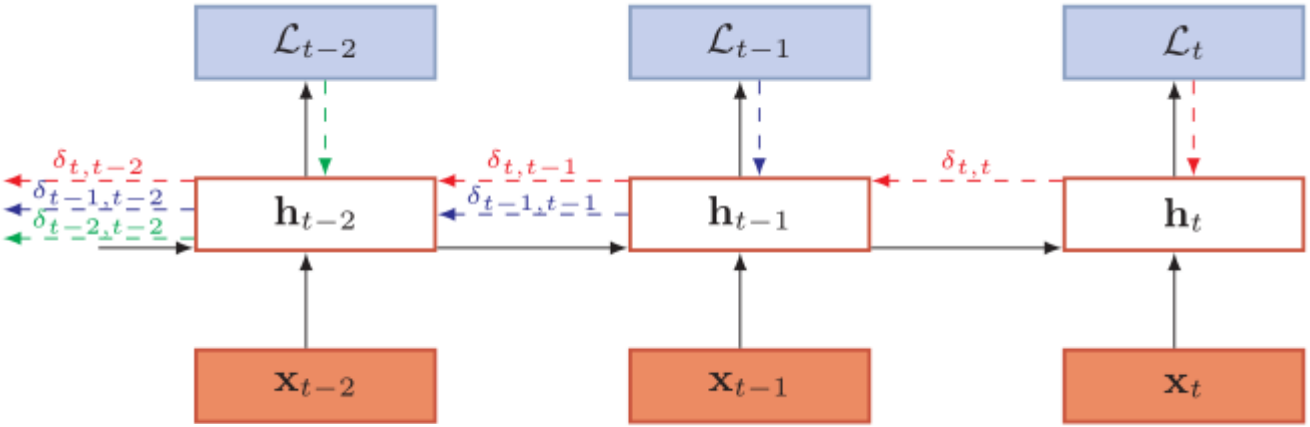
$$\nabla_{\boldsymbol{V}} L = \sum_t \sum_i \left(\frac{\partial L}{\partial o_i^{(t)}}\right) \nabla_{\boldsymbol{V}} o_i^{(t)} = \sum_t (\nabla_{\boldsymbol{o}^{(t)}} L) \boldsymbol{h}^{(t)\top},$$

$$\nabla_{\boldsymbol{W}} L = \sum_t \sum_i \left(\frac{\partial L}{\partial h_i^{(t)}}\right) \nabla_{\boldsymbol{W}^{(t)}} h_i^{(t)}$$
$$= \sum_t \mathrm{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right) (\nabla_{\boldsymbol{h}^{(t)}} L) \boldsymbol{h}^{(t-1)\top},$$

$$\nabla_{\boldsymbol{U}} L = \sum_t \sum_i \left(\frac{\partial L}{\partial h_i^{(t)}}\right) \nabla_{\boldsymbol{U}^{(t)}} h_i^{(t)}$$
$$= \sum_t \mathrm{diag}\left(1 - \left(\boldsymbol{h}^{(t)}\right)^2\right) (\nabla_{\boldsymbol{h}^{(t)}} L) \boldsymbol{x}^{(t)\top},$$

BPTT

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



$$\frac{\partial \mathcal{L}}{\partial U} = \sum_{t=1}^{T}\sum_{k=1}^{t} \delta_{t,k}\mathbf{h}_{k-1}^{\mathrm{T}}$$

$$\delta_{t,k} = \prod_{\tau=k}^{t-1}\left(\mathrm{diag}(f'(\mathbf{z}_\tau))U^{\mathrm{T}}\right)\delta_{t,t}$$

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

RNN训练

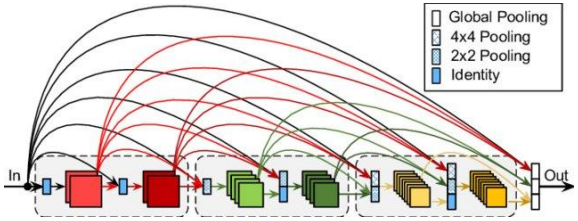
- 导师驱动过程
- 通过时间反向传播（BPTT）
- 作为有向图模型的循环网络
- 基于上下文的**RNN** 序列建模

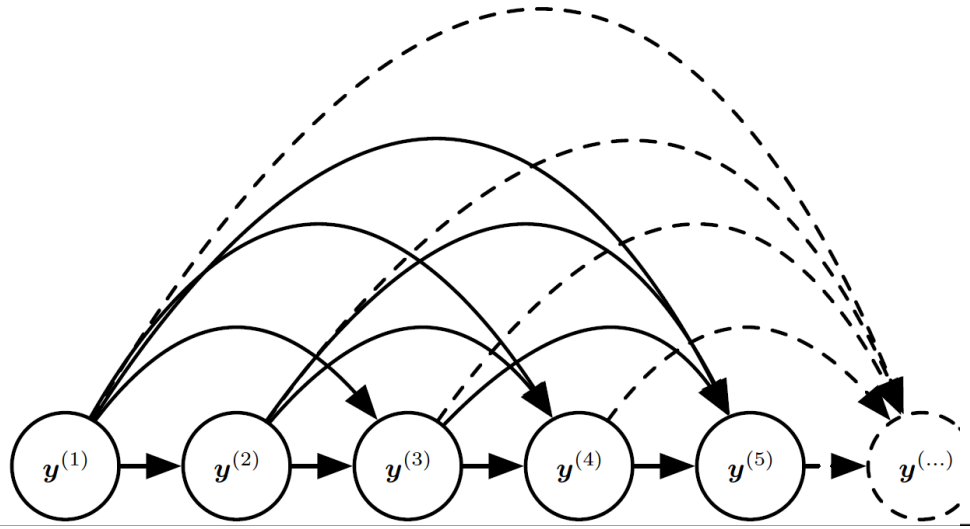
作为有向图模型的循环网络

$$\log p(\boldsymbol{y}^{(t)} \mid \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(t)})$$

$$\log p(\boldsymbol{y}^{(t)} \mid \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(t)}, \boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(t-1)})$$

- 有向图模型包含所有从过去$\boldsymbol{y}^{(i)}$ 到当前$\boldsymbol{y}^{(t)}$ 的边。





作为有向图模型的循环网络

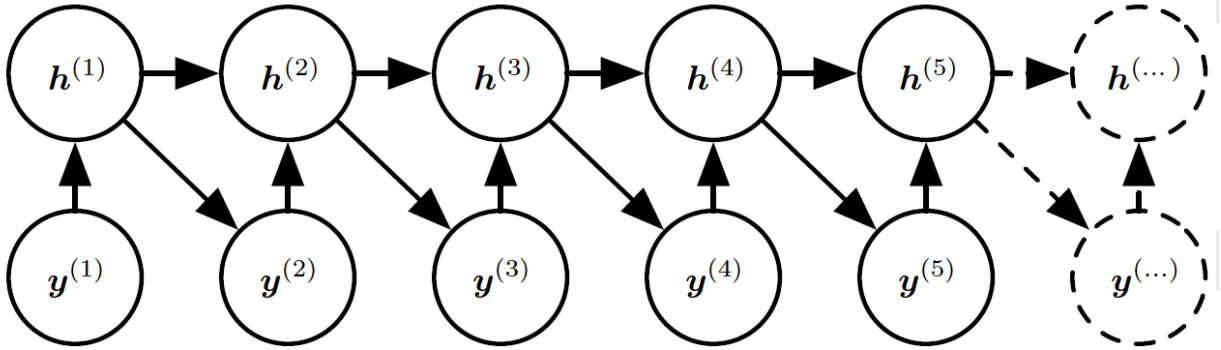
- 没有额外的输入**x**

$$P(\mathbb{Y}) = P(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(\tau)}) = \prod_{t=1}^{\tau} P(\mathbf{y}^{(t)} \mid \mathbf{y}^{(t-1)}, \mathbf{y}^{(t-2)}, \ldots, \mathbf{y}^{(1)})$$

$$L = \sum_t L^{(t)}$$

$$L^{(t)} = -\log P(\mathrm{y}^{(t)} = y^{(t)} \mid y^{(t-1)}, y^{(t-2)}, \ldots, y^{(1)})$$

- 隐藏单元作为解耦单元

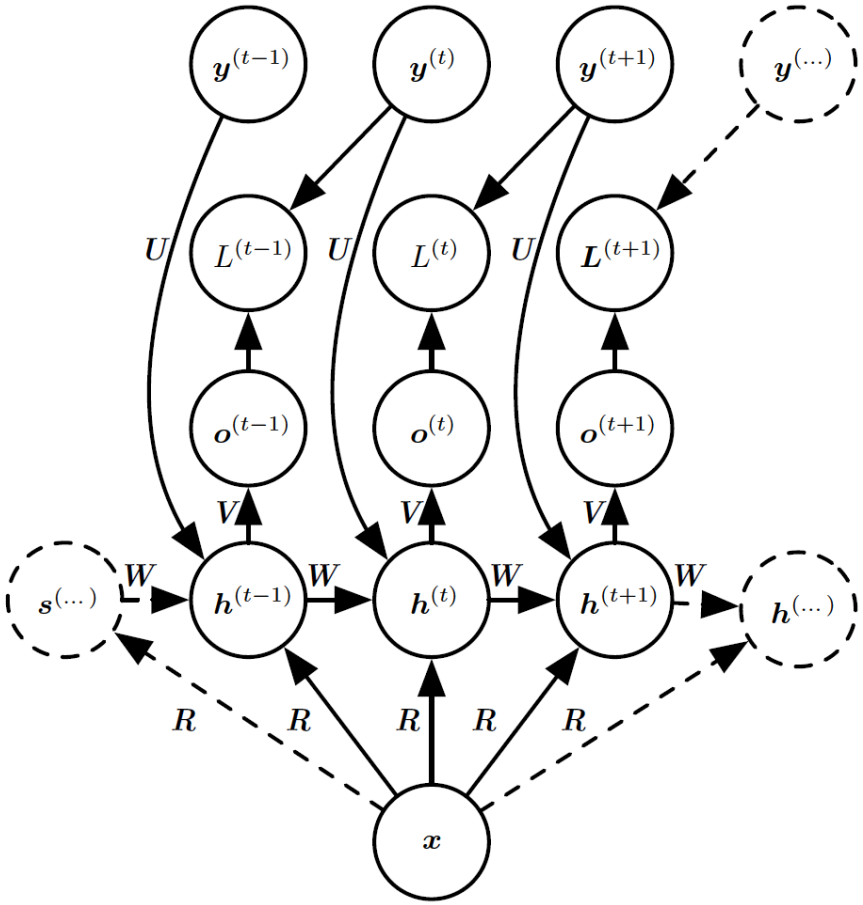


RNN训练

- 导师驱动过程
- 通过时间反向传播（BPTT）
- 作为有向图模型的循环网络
- 基于上下文的**RNN** 序列建模

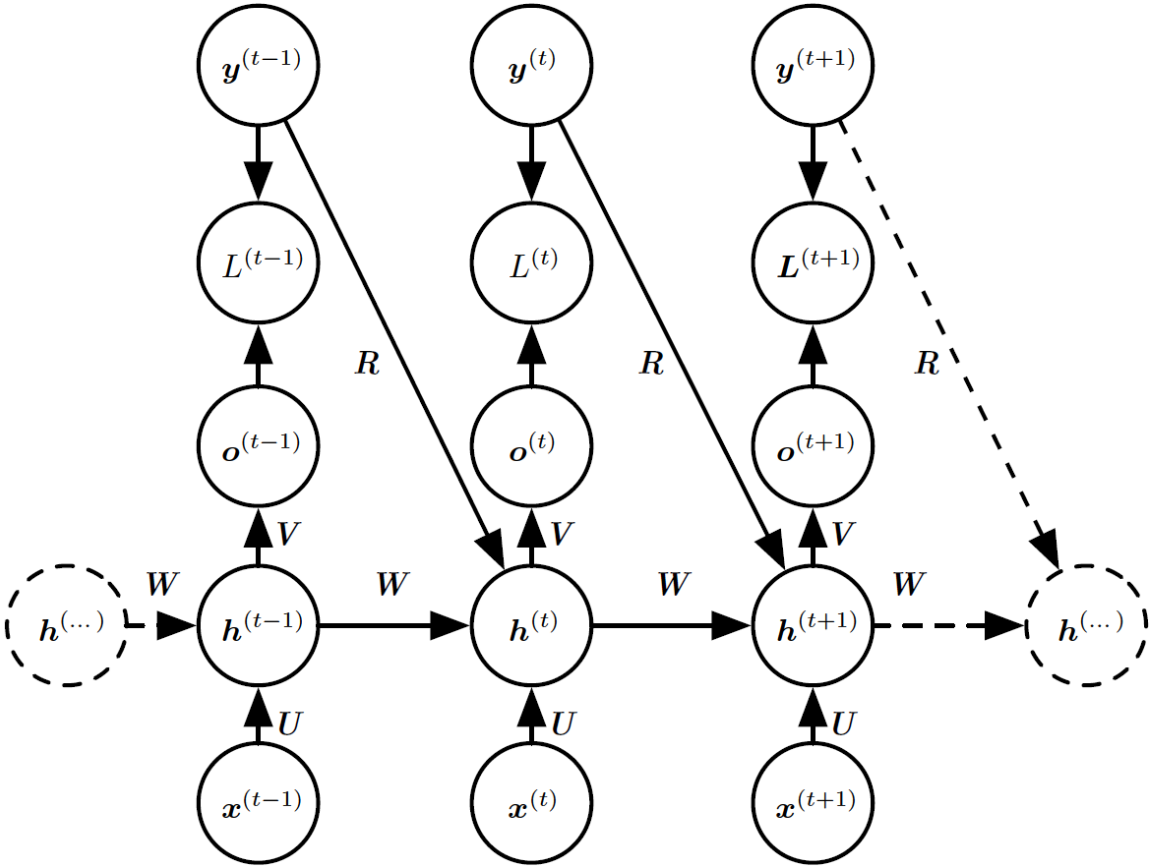
基于上下文的RNN 序列建模

- 只使用单个向量$\boldsymbol{x}$ 作为输入



基于上下文的RNN 序列建模

- 接收向量序列$\boldsymbol{x}^{(t)}$



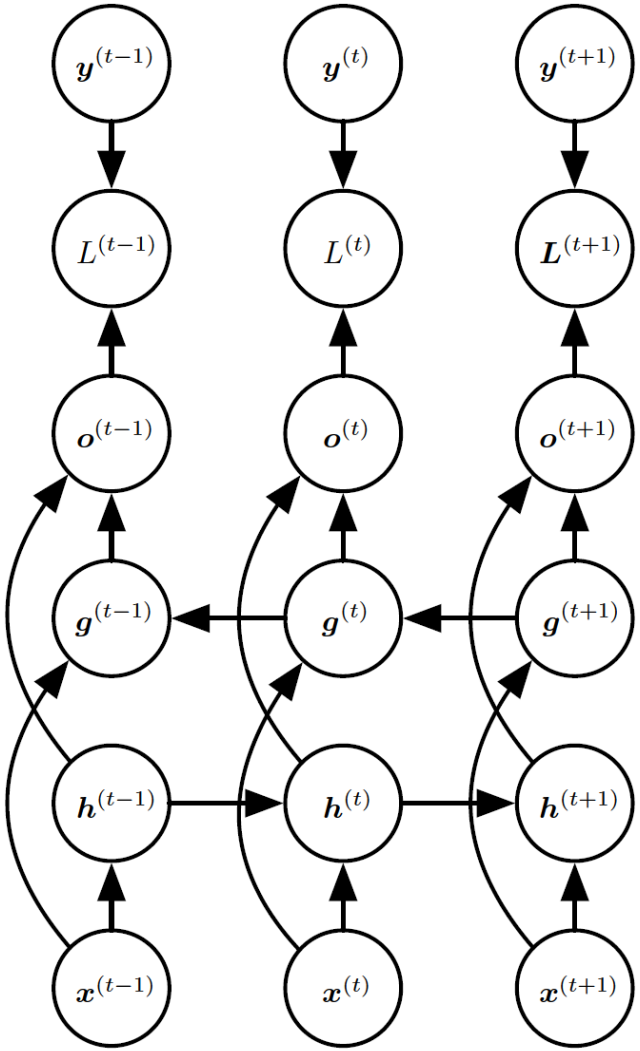
循环神经网络

- 循环神经网络（Recurrent NN）
- 双向RNN
- 序列到序列模型
- 长短期记忆（LSTM）、GRU

双向RNN

- 起因：要输出的$\boldsymbol{y}^{(t)}$ 的预测可能依赖于整个输入序列
- 如：语音识别、图像自然语言描述、手写识别
- 双向RNN：结合时间上从序列起点开始移动的RNN 和另一个时间上从序列末尾开始移动的RNN

双向RNN



- 思考：二维
  - MRF/CRF

循环神经网络

- 循环神经网络（Recurrent NN）
- 双向RNN
- 序列到序列模型
- 长短期记忆（LSTM）、GRU

序列建模问题：机器翻译

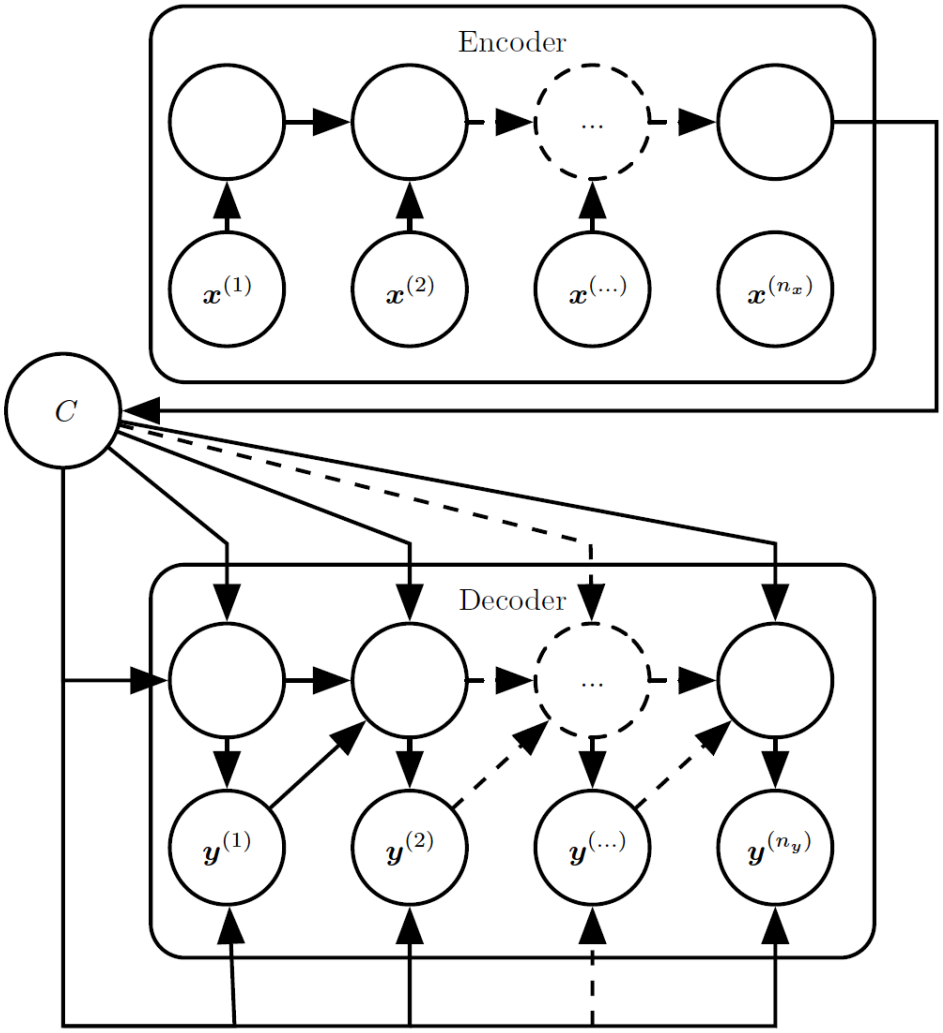
Input	Two field measurements for atmospheric fine particles were conducted in Baoan district of Shenzhen during the summer and winter in 2004.
Google	大气细颗粒两个现场测量在深圳市宝安区2004夏季和冬季期间进行。
Baidu	2004宝安区深圳夏季和冬季大气细颗粒物的两场测量。
Youdao	两个大气细粒子进行了实地测量在深圳宝安区2004年夏季和冬季。

序列到序列模型

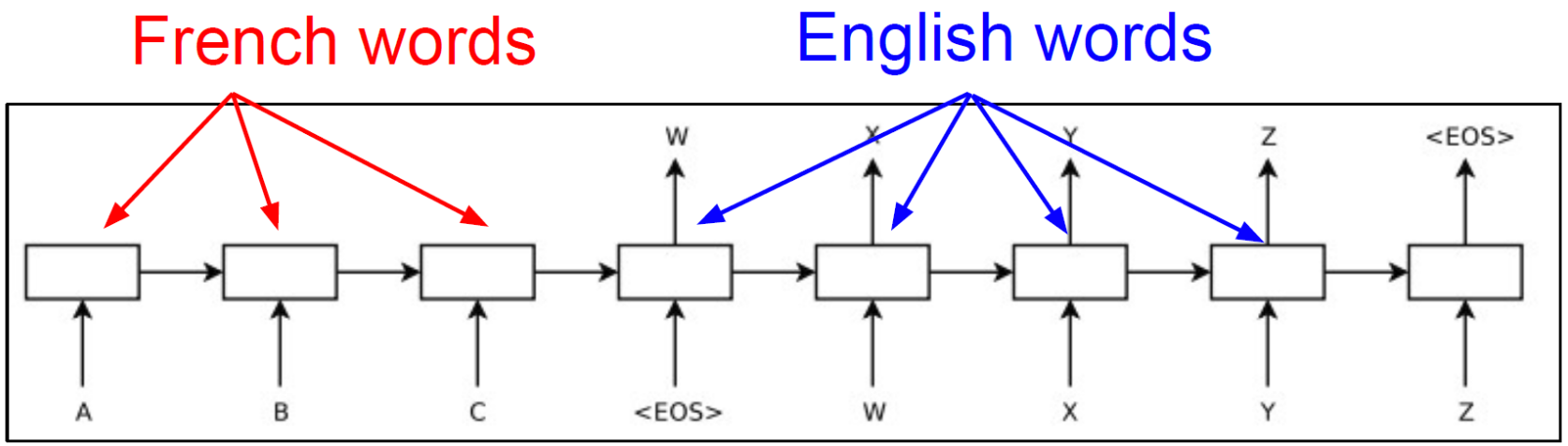
- 编码-解码或序列到序列架构
  - 编码器（encoder）或读取器(reader) 或输入(input) RNN 处理输入序列。编码器输出上下文 C（通常是最终隐藏状态的简单函数）。
  - 解码器（decoder）或写入器(writer) 或输出(output) RNN 则以固定长度的向量为条件产生输出序列$\boldsymbol{Y} = (\boldsymbol{y}^{(1)}; : : : ; \boldsymbol{y}^{(ny)})$。
  - 共同训练以最大化

$$\log P(\boldsymbol{y}^{(1)}, \ldots, \boldsymbol{y}^{(n_y)} \mid \boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(n_x)})$$

序列到序列模型


Encoder
...
$\boldsymbol{x}^{(1)}$
$\boldsymbol{x}^{(2)}$
$\boldsymbol{x}^{(\dots)}$
$\boldsymbol{x}^{(n_x)}$
C
Decoder
...
$\boldsymbol{y}^{(1)}$
$\boldsymbol{y}^{(2)}$
$\boldsymbol{y}^{(\dots)}$
$\boldsymbol{y}^{(n_y)}$

序列到序列模型：机器翻译

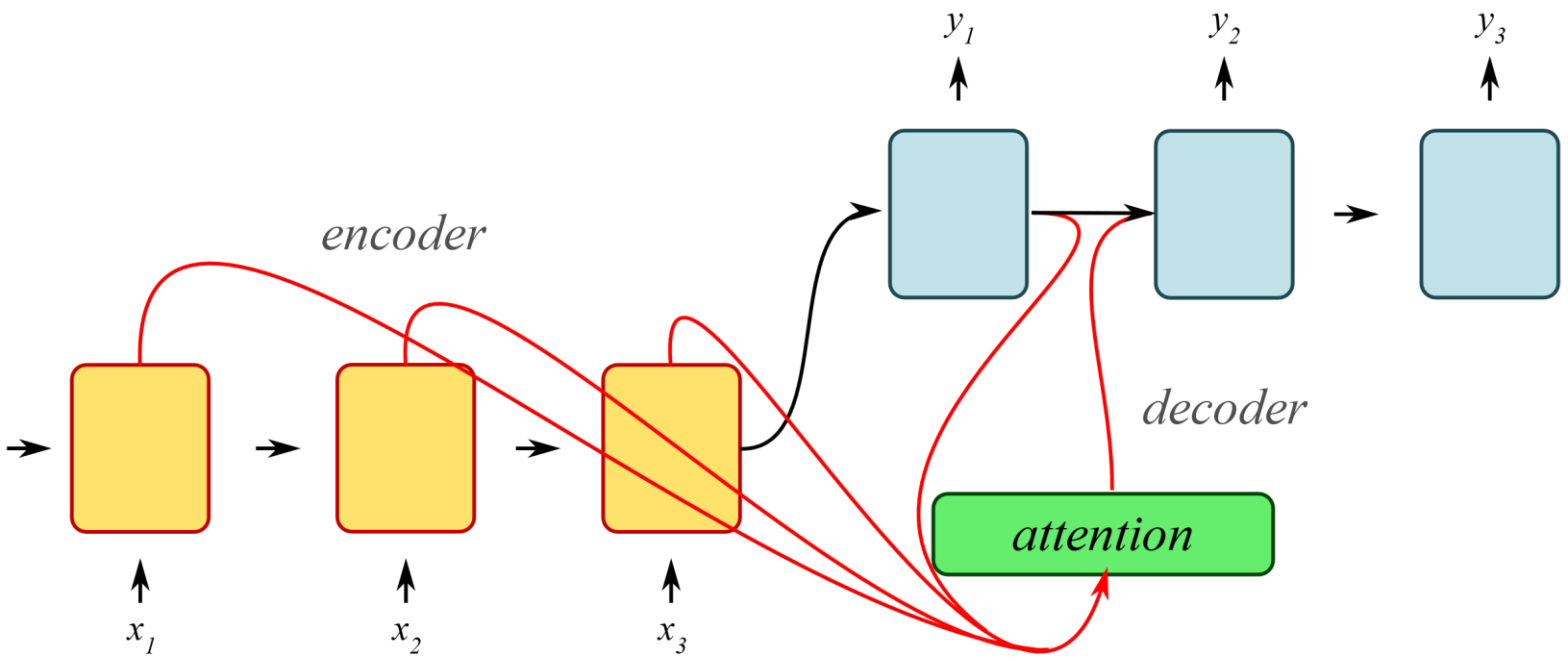


注意力机制：图像

Brushing teeth

Cutting trees

序列到序列模型：注意力机制



循环神经网络

- 循环神经网络（Recurrent NN）
- 双向RNN
- 序列到序列模型
- 长短期记忆（LSTM）、GRU