深蓝学院
shenlanxueyuan.com

深度学习理论与实践

第7课：循环神经网络 RNN

主讲人 郑元春

中科院大数据挖掘与知识管理重点实验室
中科院虚拟经济与数据科学研究中心
从事机器学习与自然语言处理相关研究

RNN介绍

RNN推导

LSTM

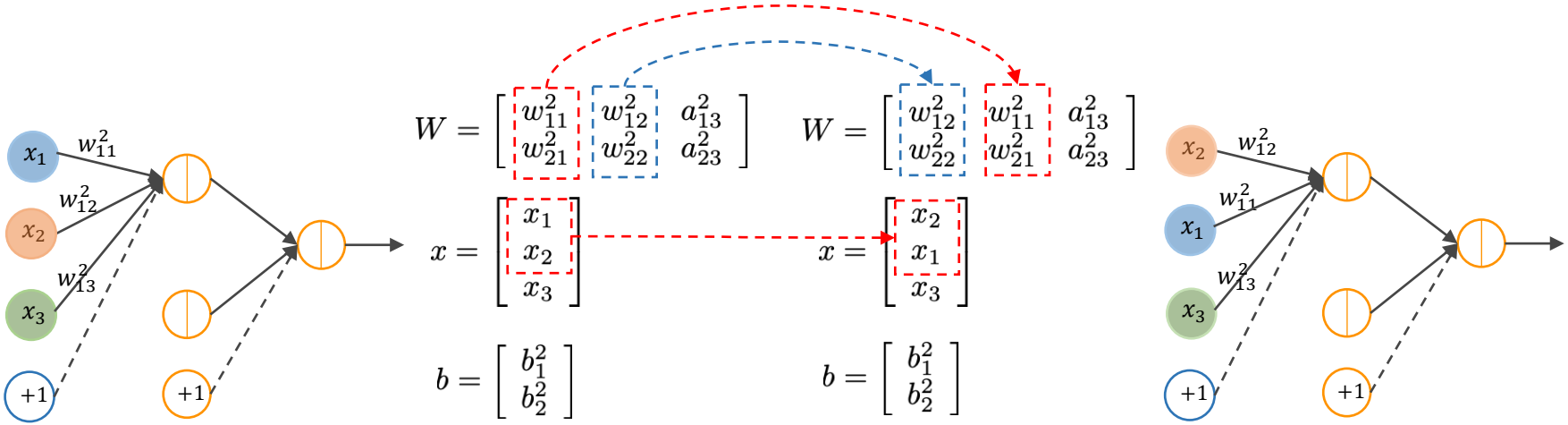
GRU

代码讲解

1. RNN介绍

知识回顾

在前几章节，我们介绍了前馈神经网络、卷积神经网络，并给出了误差反向传播的推导过程。但是这两种模型都将输入数据看作是静态的，当将输入节点的位置进行置换之后，对结果没有影响（如下图）。



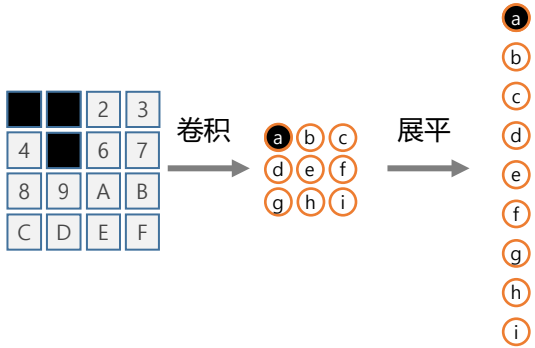
(a)原始网络

(b) x_1x_2交换

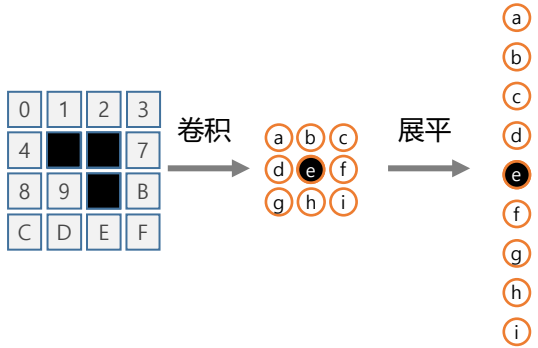
1. RNN介绍

网络的记忆能力

在前几章节，我们介绍了前馈神经网络、卷积神经网络，并给出了误差反向传播的推导过程。但是这两种模型都将输入数据看作是静态的，当将输入节点的位置进行置换之后，对结果没有影响（如下图）。



(c)卷积情况一



(d)卷积情况二

全连接神经网络和卷积神经网络是没有时序特征的

1. RNN介绍

网络的记忆能力

时间序列：是指将同一统计指标的数值按其发生的时间先后顺序排列而成的数列。时间序列分析的主要目的是根据已有的历史数据对未来进行预测。

在时间序列问题上，观察值具有时间先后的特征，历史数据可以影响未来数据的表达，因此需要网络具有记忆能力。

- **自回归模型(Autoregressive Model, AR)**

$$\boldsymbol{y}_t = w_0 + \sum_{k=1}^{K} w_k y_{t-k} + \epsilon_t \qquad \epsilon_t \sim N(0, \sigma^2)$$

- **有外部输入的非线性自回归模型(Nonlinear Autoregressive with Exogenous Input Model，NARX)**

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

非线形函数，可以是一个前馈网络

超参数

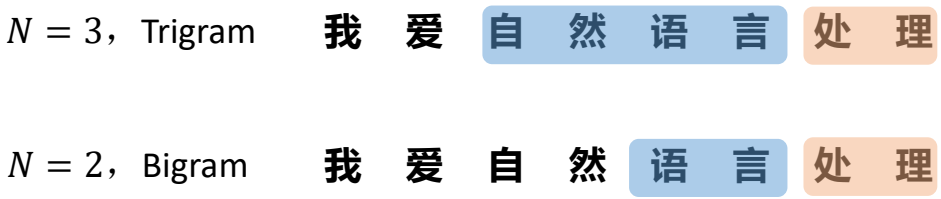
1. RNN介绍

网络的记忆能力

文本序列建模问题

在构造语句的时候，我们一般倾向于从左向右逐渐补全一个句子，那么在对文本进行建模的时候，就可以将句子的生成过程看作是一个时间序列问题，每一个单词都是一个时间点，从左向右逐渐生成，并且前面的句子会影响下一时刻单词的产生。

N-gram 语言模型：使用固定宽度的窗口

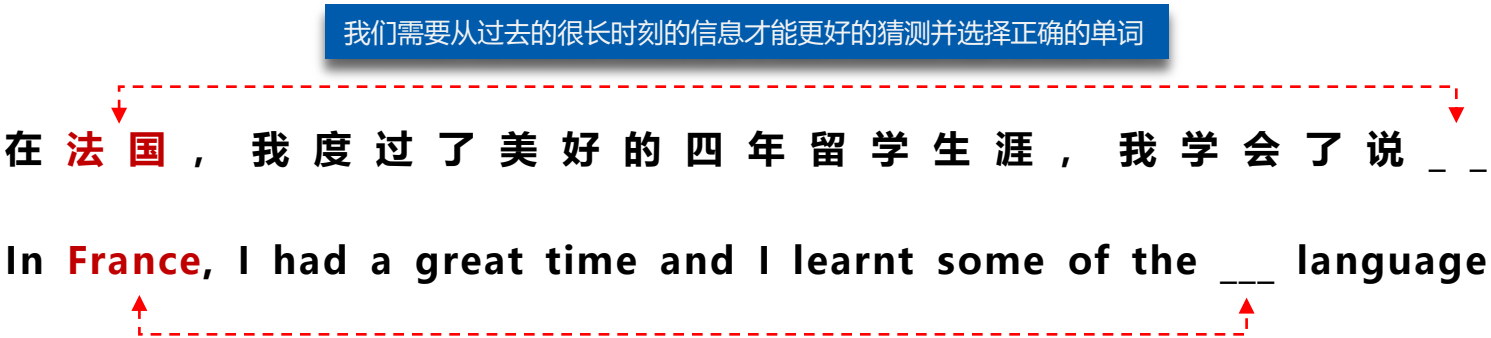


缺点：受限于窗口的宽度，N并不能取很大的值，会出现长期依赖缺失的问题。

1. RNN介绍

网络的记忆能力

N-Gram语言模型—长期依赖问题



1. RNN介绍

网络的记忆能力

➢ 使用全连接网络

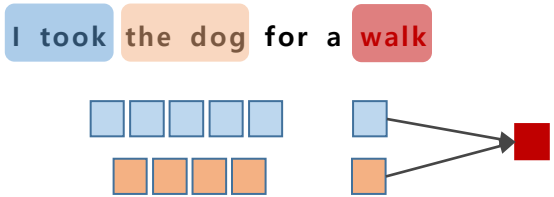
This morning, I took the dog for a walk

I took the dog for a walk this morning

不同的输入样本，但是同一个短语出现在不同的位置，当输入到神经网络里面的时候，由于不同位置的权重连结是相互独立的，所以会造成参数也是独立的。带来的问题就是相同的词组不能用同一份权重去训练。

对于参数共享问题，上节课我们学习过的卷积神经网络不就是做参数共享的么？

➢ 使用卷积网络



词序信息依旧不完整

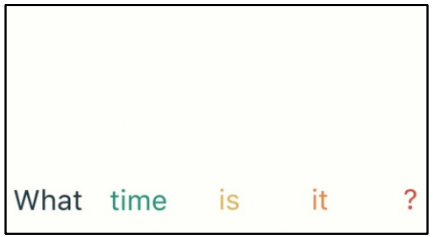
我们在对时间序列进行建模的时候，其时序信息一定要表示完整

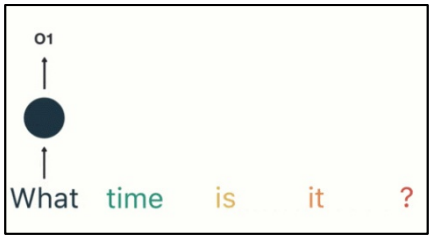
1. RNN介绍

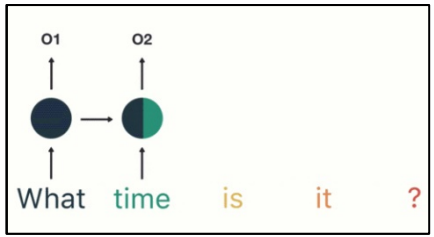
RNN与语言模型

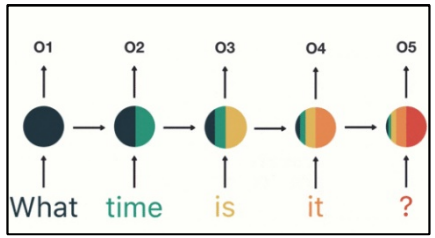
语言模型：给定句子的部分(历史句子，上下文)，预测出一个单词(下一个单词，上下文单词，目标词)。

$$p(s) = p\left(w_1^N\right) \approx p(w_1) \cdot p(w_2|w_1) \cdot p(w_3|w_1^2) \cdots p\left(w_N\middle|w_1^{N-1}\right)$$









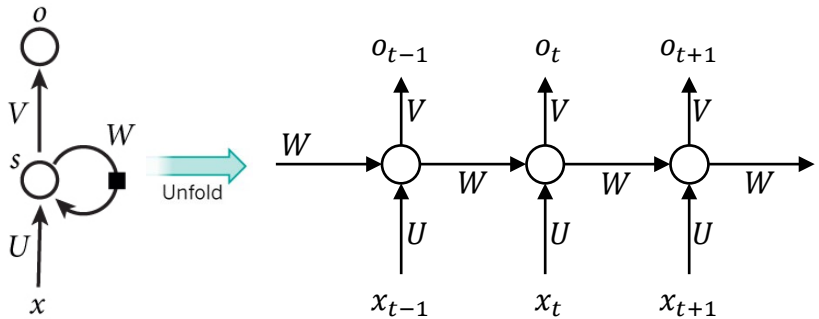
1. RNN介绍

RNN与语言模型

语言模型：给定句子的部分(历史句子，上下文)，预测出一个单词(下一个单词，上下文单词，目标词)。

$$p(s) = p\left(w_1^N\right) \approx p(w_1) \cdot p(w_2|w_1) \cdot p(w_3|w_1^2) \cdots p\left(w_N\middle|w_1^{N-1}\right)$$

概括RNN模型：将传统的神经网络加入**时间**上的特征。



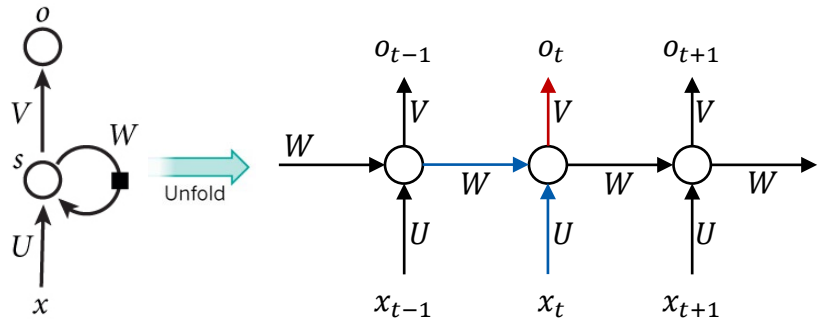
CNN是空间上的深度神经网络；RNN是时间上的深度神经网络。

1. RNN介绍

RNN与语言模型

$\{\boldsymbol{x}_t, \boldsymbol{o}_t\}_{t=1}^{T}$：训练数据集样本对；

$\hat{\boldsymbol{o}}_t$：RNN网络对应输入$\boldsymbol{x}_t$的输出；



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

$$\boldsymbol{y}_t = \boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{c}$$

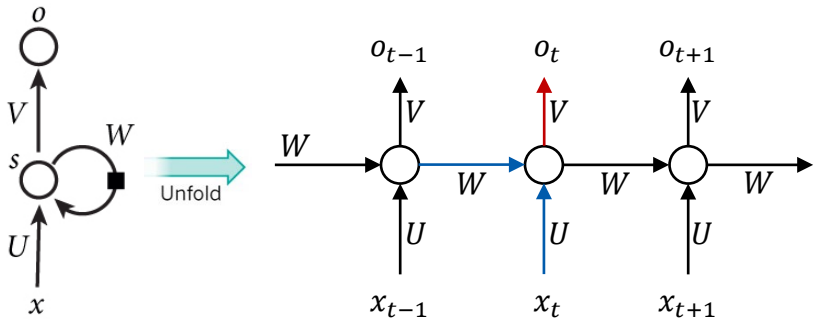
$$\hat{\boldsymbol{o}}_t = g\left(\boldsymbol{y}_t\right)$$

其中，g为输出层激活函数，f为隐藏层激活函数。为了便于讲解，我们将其统一记为f。

1. RNN介绍

RNN与语言模型

将输入看成是一个时间序列，每一次输出o都对应时间序列上的一个时刻，后面的时刻需要递归调用前面时刻的结果。



$$
\begin{aligned}
\boldsymbol{o}_t &= g(\boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{c}) \\
\boldsymbol{h}_t &= f(\boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\boldsymbol{h}_{t-1} + \boldsymbol{b}) \\
&= f(\boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}f(\boldsymbol{U}\boldsymbol{x}_{t-1} + \boldsymbol{W}\boldsymbol{h}_{t-2} + \boldsymbol{b}) + \boldsymbol{b}) \\
&= f(\boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}f(\boldsymbol{U}\boldsymbol{x}_{t-1} + \boldsymbol{W}f(\boldsymbol{U}\boldsymbol{x}_{t-2} + \boldsymbol{W}\boldsymbol{h}_{t-3} + \boldsymbol{b}) + \boldsymbol{b}) + \boldsymbol{b}) \\
&= f(\boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}f(\boldsymbol{U}\boldsymbol{x}_{t-1} + \boldsymbol{W}f(\boldsymbol{U}\boldsymbol{x}_{t-2} + \boldsymbol{W}f(\boldsymbol{U}\boldsymbol{x}_{t-3} + \ldots) + \boldsymbol{b}) + \boldsymbol{b}) + \boldsymbol{b})
\end{aligned}
$$

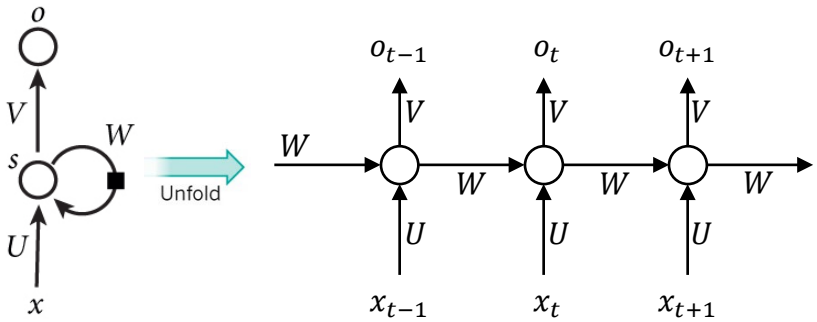
1. RNN介绍

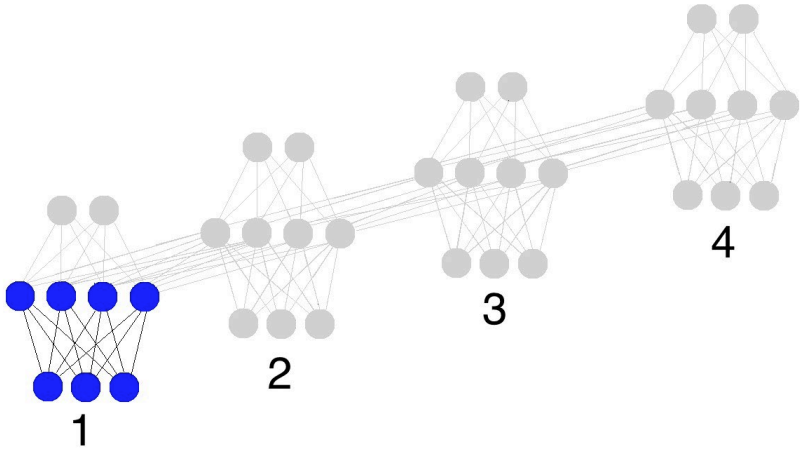
RNN与语言模型

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

$$\boldsymbol{y}_t = \boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{c}$$

$$\hat{\boldsymbol{o}}_t = g\left(\boldsymbol{y}_t\right)$$





1. RNN介绍

双向RNN模型

对于部分序列模型，只看前面的词是不够的，比如下面这句话：

我的手机坏了，我打算

____

填在空格处的可选词包括(买，修等)，现在我们把这句话的后半部分给出来。这样相当于增加了很多的信息，以此进一步确定。

我的手机坏了，我打算____一部新手机。

这样后面的句子提供了额外的信息，更加接近我们的思考方式。这个时候在单向的RNN基础之上，我们需要的是一个能从前向后，同时也能从后向前工作的RNN。

语言从侧面反映了思考方式和对世界的认知抽象

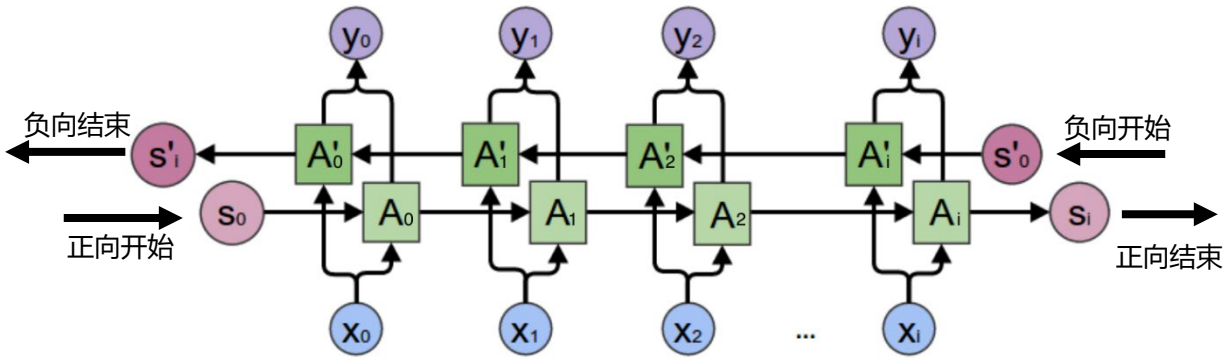
1. RNN介绍

双向RNN模型

能从前向后，同时也能从后向前工作的RNN。搭建两套RNN，不同方向，然后组合。

隐藏层要保存两个值，一个A参与正向计算，另一个值A'参与反向计算。最终的输出层的值y取决于A和A'。

正向计算和反向计算不共享权重，也就是说U和U'、W和W'、V和V'都是不同的权重矩阵。



1. RNN介绍

深度双向RNN模型

前面介绍的**循环神经网络**只有一个隐藏层，我们当然也可以堆叠两个以上的隐藏层，这样就得到了**深度循环神经网络。**

我们把第i个隐藏层的输出值表示为$\boldsymbol{h}_t^{(i)}$、$\boldsymbol{h}_{t'}^{(i)}$，为了展示方便，忽略偏置，则:

$$\hat{\boldsymbol{o}}_t = f\left(\boldsymbol{V}^{(i)}\boldsymbol{h}_t^{(i)} + \boldsymbol{V}'^{(i)}\boldsymbol{h}_{t'}^{(i)}\right)$$

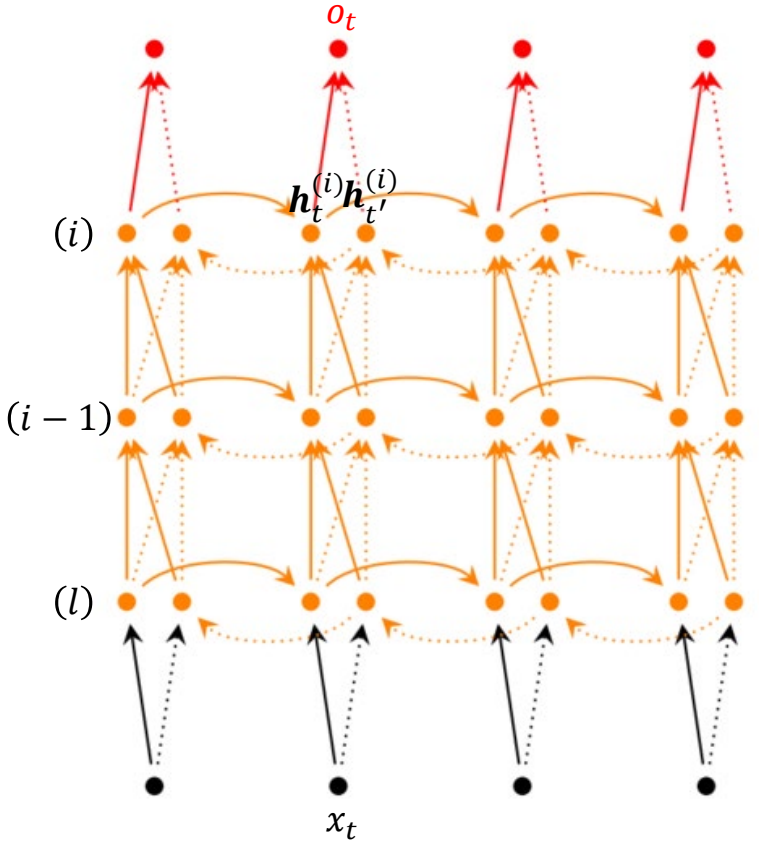
$$\boldsymbol{h}_t^{(i)} = f\left(\boldsymbol{U}^{(i)}\boldsymbol{h}_t^{(i-1)} + \boldsymbol{W}^{(i)}\boldsymbol{h}_{t-1}^{(i)}\right)$$

$$\boldsymbol{h}_{t'}^{(i)} = f\left(\boldsymbol{U}'^{(i)}\boldsymbol{h}_{t'}^{(i-1)} + \boldsymbol{W}'^{(i)}\boldsymbol{h}_{t+1'}^{(i)}\right)$$

……

$$\boldsymbol{h}_t^{(l)} = f\left(\boldsymbol{U}^{(l)}x_t + \boldsymbol{W}^{(l)}\boldsymbol{h}_{t-1}^{(l)}\right)$$

$$\boldsymbol{h}_{t'}^{(l)} = f\left(\boldsymbol{U}'^{(l)}x_t + \boldsymbol{W}'^{(l)}\boldsymbol{h}_{t+1'}^{(l)}\right)$$

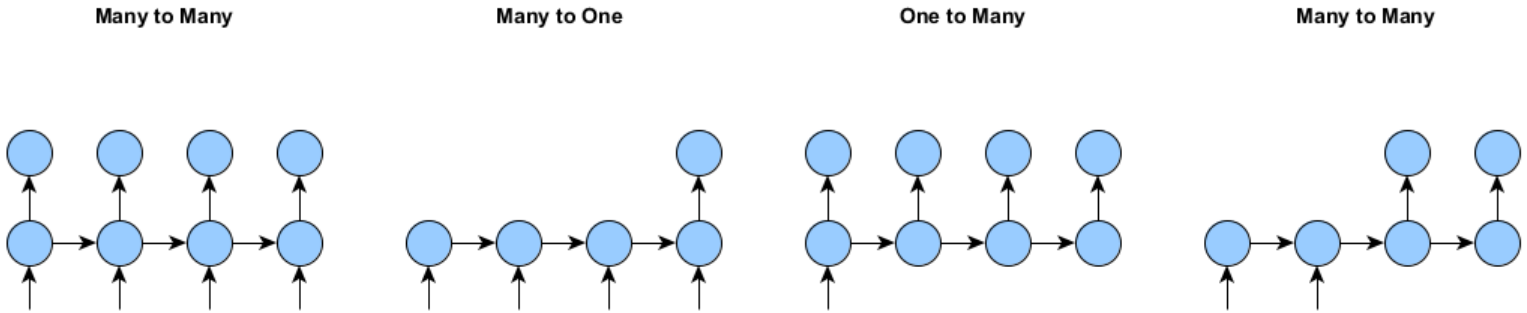


1. RNN介绍

RNN类型

根据RNN输出的个数，可以将RNN分为多对多型，多对一型，一对多型。

各个类型的RNN在结构上没有很大的区别，只是在对待时刻与是否需要中间输出时区别对待。



1. RNN介绍

RNN模型目标

将语句看作是序列模型，想要对其建模需要解决以下问题：

- 句子长度变长问题
- 保留序列的顺序
- 长期依赖问题
- 在整个序列长度上进行参数共享

RNN

RNN推导

LSTM

GRU

代码讲解

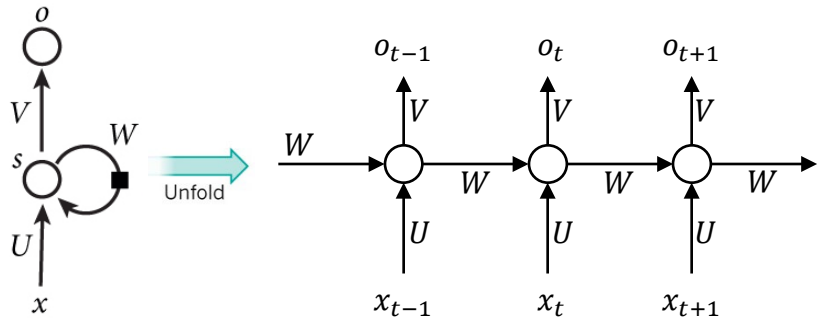
2. RNN训练

BPTT算法（Back Propagation Through Time）

BPTT算法是针对循环层的训练算法，它的基本原理和BP算法一样，也包含同样的3个步骤：

- 前向计算每个神经元的输出值；
- 反向计算每个神经元的**误差项**值δ；
- 将此误差按照不同的层反向传递；
- 计算每个权重的梯度。

最后，再用**随机梯度下降**算法更新权重。

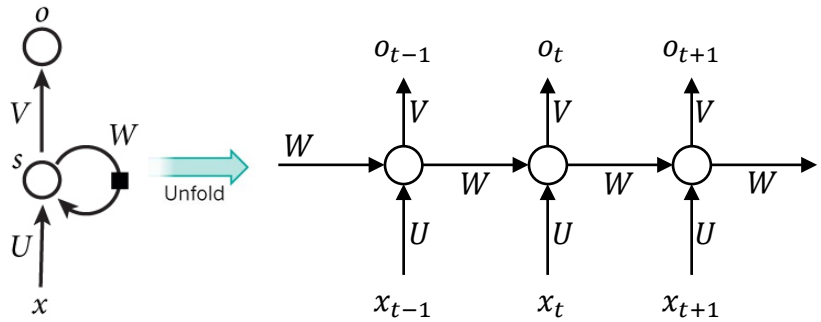


不同的是，网络在时间上是权值共享的，误差反传时需要将时间考虑进去，这也是为什么训练算法叫做**BPTT**。

2. RNN训练-单隐藏层

前向计算过程

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



我们假设输入向量$\boldsymbol{x}$的维度是m，隐藏层向量$\boldsymbol{h}$的维度是n，则矩阵U的维度是$n \times m$，矩阵W的维度是$n \times n$。下面是上式展开成矩阵的样子，看起来更直观一些：

$$\begin{bmatrix} h_t^1 \\ h_t^2 \\ . \\ . \\ h_t^n \end{bmatrix} = f\left(\begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1m} \\ u_{21} & u_{22} & \cdots & u_{2m} \\ & \cdots & \cdots & \\ u_{n1} & u_{n2} & \cdots & u_{nm} \end{bmatrix} \begin{bmatrix} x_t^1 \\ x_t^2 \\ . \\ . \\ x_t^m \end{bmatrix} + \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1n} \\ w_{21} & w_{22} & \cdots & w_{2n} \\ & \cdots & \cdots & \\ w_{n1} & w_{n2} & \cdots & w_{nn} \end{bmatrix} \begin{bmatrix} h_{t-1}^1 \\ h_{t-1}^2 \\ . \\ . \\ h_{t-1}^n \end{bmatrix} \right)$$

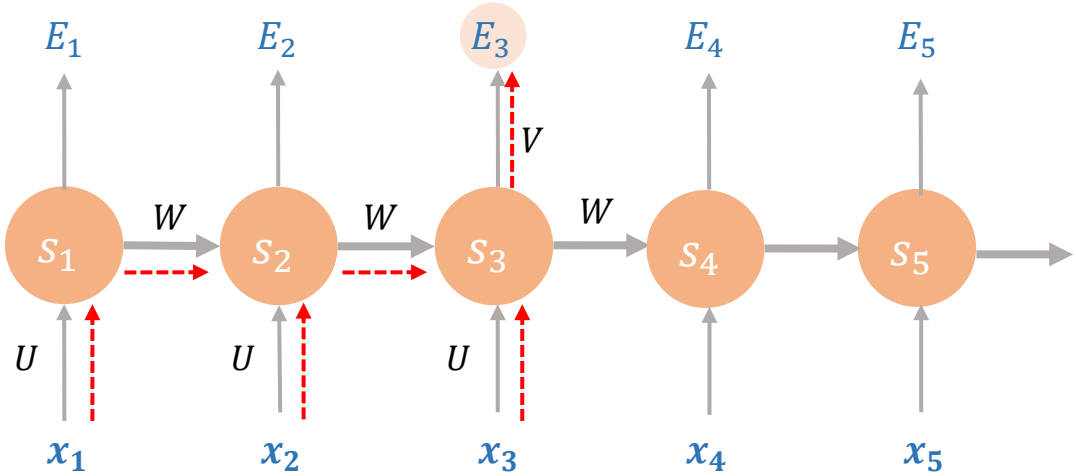
$\boldsymbol{h}_t \in \mathbb{R}^{n \times 1}$　　$\boldsymbol{x}_t \in \mathbb{R}^{m \times 1}$　　$\boldsymbol{h}_{t-1} \in \mathbb{R}^{n \times 1}$

2. RNN训练-BPTT

前向计算过程

计算第t_3时刻的输出$\hat{o}_3$

$$\boldsymbol{h}_1 = f(\boldsymbol{U}\boldsymbol{x}_1 + \boldsymbol{b})$$
$$\boldsymbol{h}_2 = f(\boldsymbol{U}\boldsymbol{x}_2 + \boldsymbol{W}\boldsymbol{h}_1 + \boldsymbol{b})$$
$$\boldsymbol{h}_3 = f(\boldsymbol{U}\boldsymbol{x}_3 + \boldsymbol{W}\boldsymbol{h}_2 + \boldsymbol{b})$$
$$\hat{\boldsymbol{o}}_3 = g(\boldsymbol{V}\boldsymbol{h}_3 + \boldsymbol{c})$$

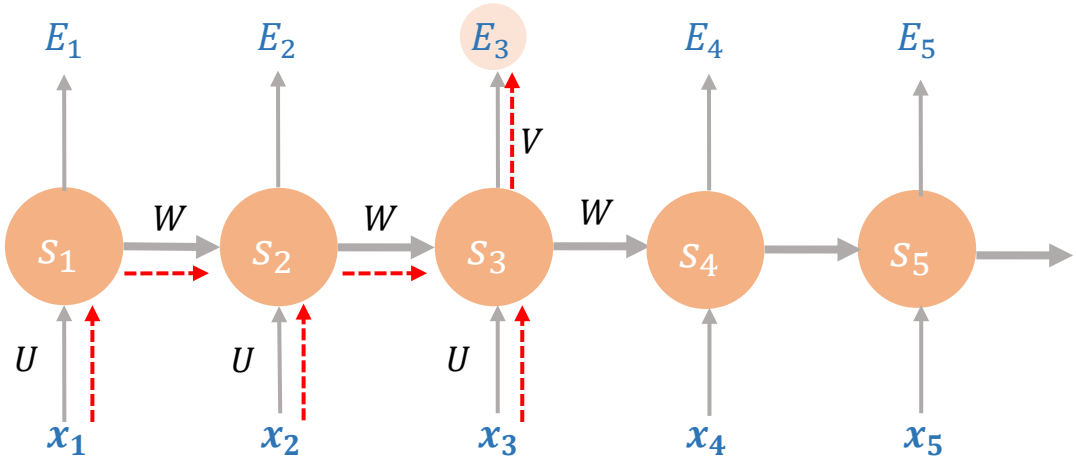


2. RNN训练-BPTT

误差

L_t：对应输入$\boldsymbol{x}_t$的损失函数，用于量化$\hat{\boldsymbol{o}}_t$与$\boldsymbol{o}_t$的差距；

$L = \sum_{t=1}^{T} L_t$：RNN网络对于整个训练数据集的损失函数。



误差形式有多种，比如交叉熵、均方误差等。

$$L = \sum_{t=1}^{T} L_t = \sum_{t=1}^{T} -\boldsymbol{o}_t log\hat{\boldsymbol{o}}_t$$

$$L = \sum_{t=1}^{T} L_t = \sum_{t=1}^{T} \|\hat{\boldsymbol{o}}_t - \boldsymbol{o}_t\|^2$$

2. RNN训练-BPTT

误差反向传播 — 先导知识

求解误差对$\boldsymbol{V}, \boldsymbol{c}, \boldsymbol{W}, \boldsymbol{b}$的梯度，涉及到标量$\boldsymbol{L}$ 对向量$\boldsymbol{c}$ /矩阵$\boldsymbol{V}$ 的求导，无法直接使用链式求导法则。

向量对向量的链式求导

假设$\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}$分别是$m, n, p$维向量，且3个向量存在$\boldsymbol{x} \to \boldsymbol{y} \to \boldsymbol{z}$的依赖关系，那么，

$$\underset{p \times m}{\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{x}}} = \underset{p \times n}{\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{y}}} \underset{n \times m}{\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}}$$

注意：此处链式求导法则成立的前提是3个都是向量，其中任意一个都不能是矩阵。

标量对向量的链式求导

假设$\boldsymbol{x}, \boldsymbol{y}$分别是$m, n$维向量，$z$为标量，且3个量存在$\boldsymbol{x} \to \boldsymbol{y} \to z$的依赖关系，那么，

$$\underset{m \times 1}{\frac{\partial z}{\partial \boldsymbol{x}}} = \underset{m \times n}{\left(\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}\right)^T} \underset{n \times 1}{\frac{\partial z}{\partial \boldsymbol{y}}}$$

2. RNN训练-BPTT

误差反向传播 — 先导知识

求解误差对$\boldsymbol{V}, \boldsymbol{c}, \boldsymbol{W}, \boldsymbol{b}$的梯度，涉及到标量$\boldsymbol{L}$ 对向量$\boldsymbol{c}$ /矩阵$\boldsymbol{V}$ 的求导，无法直接使用链式求导法则。

标量对向量的链式求导

假设$\boldsymbol{x}, \boldsymbol{y}$分别是$m, n$维向量，$z$为标量，且3个量存在$\boldsymbol{x} \to \boldsymbol{y} \to z$的依赖关系，那么，

$$\underset{m\times 1}{\frac{\partial z}{\partial \boldsymbol{x}}} = \underset{m\times n}{\left(\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{x}}\right)^T} \underset{n\times 1}{\frac{\partial z}{\partial \boldsymbol{y}}}$$

标量对矩阵的链式求导

假设$\boldsymbol{X}, \boldsymbol{Y}$为矩阵，$z$为标量，且3个量存在$\boldsymbol{x} \to \boldsymbol{y} \to z$的依赖关系，那么，

$z = f(\boldsymbol{Y})$ $\boldsymbol{Y} = \boldsymbol{AX} + \boldsymbol{B}$	$\frac{\partial z}{\partial \boldsymbol{X}} = \boldsymbol{A}^T \frac{\partial z}{\partial \boldsymbol{Y}}$
$z = f(\boldsymbol{Y})$ $\boldsymbol{Y} = \boldsymbol{XA} + \boldsymbol{B}$	$\frac{\partial z}{\partial \boldsymbol{X}} = \frac{\partial z}{\partial \boldsymbol{Y}} \boldsymbol{A}^T$

注意：这里的结论并非通过链式求导法则求得，具体可参考下一页课件。

2. RNN训练-BPTT

误差反向传播 — 先导知识

标量对矩阵的链式求导

假设$\boldsymbol{X}, \boldsymbol{Y}$为矩阵，$z$为标量，且3个量存在$\boldsymbol{x} \rightarrow \boldsymbol{y} \rightarrow z$的依赖关系

我们来看这个常见问题：A, X, B, Y都是矩阵，z是标量，其中$z = f(Y), Y = AX + B$,我们要求出$\frac{\partial z}{\partial X}$,这个问题在机器学习中是很常见的。此时，我们并不能直接整体使用矩阵的链式求导法则，因为矩阵对矩阵的求导结果不好处理。

这里我们回归初心，使用定义法试一试,先使用上面的标量链式求导公式：

$$\frac{\partial z}{\partial x_{ij}} = \sum_{k,l} \frac{\partial z}{\partial Y_{kl}} \frac{\partial Y_{kl}}{\partial X_{ij}}$$

我们再来看看后半部分的导数：

$$\frac{\partial Y_{kl}}{\partial X_{ij}} = \frac{\partial \sum_s (A_{ks} X_{sl})}{\partial X_{ij}} = \frac{\partial A_{ki} X_{il}}{\partial X_{ij}} = A_{ki} \delta_{lj}$$

其中δ_{lj}在$l = j$时为1，否则为0.

那么最终的标签链式求导公式转化为：

$$\frac{\partial z}{\partial x_{ij}} = \sum_{k,l} \frac{\partial z}{\partial Y_{kl}} A_{ki} \delta_{lj} = \sum_{k} \frac{\partial z}{\partial Y_{kj}} A_{ki}$$

即矩阵A^T的第i行和$\frac{\partial z}{\partial Y}$的第j列的内积。排列成矩阵即为：

$$\frac{\partial z}{\partial X} = A^T \frac{\partial z}{\partial Y}$$

总结下就是：

$$z = f(Y), Y = AX + B \rightarrow \frac{\partial z}{\partial X} = A^T \frac{\partial z}{\partial Y}$$

这结论在$\mathbf{x}$是一个向量的时候也成立，即：

$$z = f(\mathbf{y}), \mathbf{y} = A\mathbf{x} + \mathbf{b} \rightarrow \frac{\partial z}{\partial \mathbf{x}} = A^T \frac{\partial z}{\partial \mathbf{y}}$$

如果要求导的自变量在左边，线性变换在右边，也有类似稍有不同的结论如下，证明方法是类似的，这里直接给出结论：

$$z = f(Y), Y = XA + B \rightarrow \frac{\partial z}{\partial X} = \frac{\partial z}{\partial Y} A^T$$

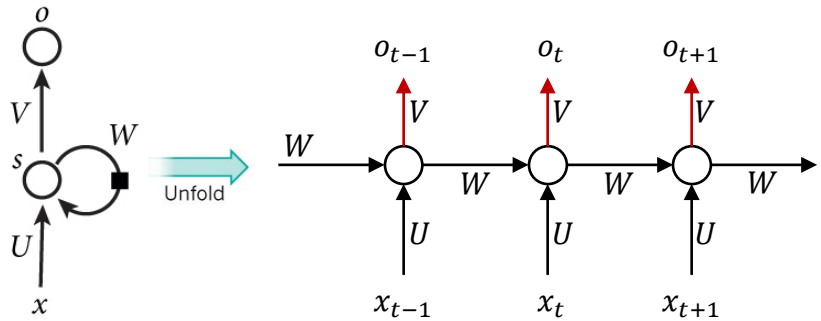
2. RNN训练-BPTT

误差反向传播V, c

V, c 连接隐藏层和输出层，其联系为：

$$y_t = Vh_t + c$$

$$\hat{o}_t = g(y_t)$$



因此，误差对V, c的梯度为：

$$\frac{\partial L}{\partial V} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial V} \quad \overline{\qquad V \to y_t \to L_t \qquad} \quad \sum_{t=1}^{T} \frac{\partial L_t}{\partial y_t} (h_t)^T$$

$z = f(Y)$ $Y = XA + B$	$\frac{\partial z}{\partial X} = \frac{\partial z}{\partial Y} A^T$

$$\frac{\partial L}{\partial c} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial c} \quad \overline{\qquad c \to y_t \to L_t \qquad} \quad \sum_{t=1}^{T} \frac{\partial L_t}{\partial y_t}$$

注意：

（1）这里涉及到标量L 对向量c /矩阵V 的求导，无法直接使用链式求导法则。可参考前几页课件的先导知识。

（2）损失函数的形式确定后，$\frac{\partial L_t}{\partial y_t}$相应的也确定了。

2. RNN训练-BPTT

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

$$\boldsymbol{y}_t = \boldsymbol{V}\boldsymbol{h}_t + \boldsymbol{c}$$

$$\hat{\boldsymbol{o}}_t = g\left(\boldsymbol{y}_t\right)$$

误差反向传播$\boldsymbol{W}, \boldsymbol{U}, \boldsymbol{b}$

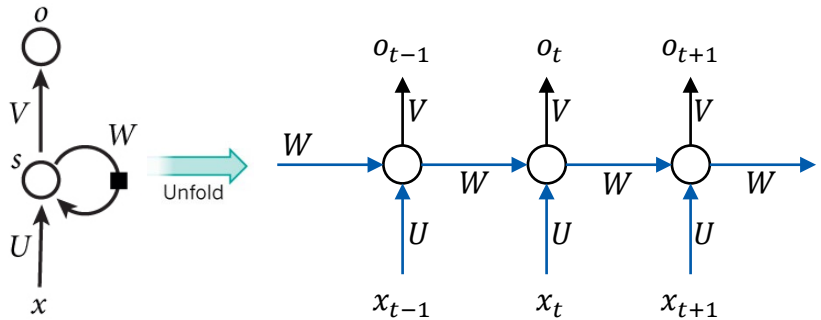
从RNN的模型可以看出，在反向传播时，在某一时刻t隐藏层的梯度损失由当前时刻的输出$\hat{\boldsymbol{o}}_t$对应的梯度损失和$t+1$时刻的反传回来的梯度损失两部分共同决定。

令$\boldsymbol{net}_t = \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\boldsymbol{h}_{t-1} + \boldsymbol{b}$，$t$时刻隐藏层误差项值$\boldsymbol{\delta}_t$为：

$$\begin{aligned}
\boldsymbol{\delta}_t &= \frac{\partial L}{\partial \boldsymbol{h}_t} \\
&= \left(\frac{\partial \boldsymbol{y}_t}{\partial \boldsymbol{h}_t}\right)^T \frac{\partial L}{\partial \boldsymbol{y}_t} + \left(\frac{\partial \boldsymbol{net}_{t+1}}{\partial \boldsymbol{h}_t}\right)^T \frac{\partial L}{\partial \boldsymbol{net}_{t+1}} \\
&= \boldsymbol{V}^T \frac{\partial L}{\partial \boldsymbol{y}_t} + \boldsymbol{W}^T \frac{\partial L}{\partial \boldsymbol{net}_{t+1}} \\
&= \boldsymbol{V}^T \frac{\partial L}{\partial \boldsymbol{y}_t} + \boldsymbol{W}^T \left(\frac{\partial \boldsymbol{h}_{t+1}}{\partial \boldsymbol{net}_{t+1}}\right)^T \frac{\partial L}{\partial \boldsymbol{h}_{t+1}} \\
&= \boldsymbol{V}^T \frac{\partial L}{\partial \boldsymbol{y}_t} + \boldsymbol{W}^T \left(f'(\boldsymbol{net}_{t+1})\right)^T \boldsymbol{\delta}_{t+1}
\end{aligned}$$

$\boldsymbol{T}$时刻隐藏层误差项值$\boldsymbol{\delta}_T$为：

$$\boldsymbol{\delta}_T = \frac{\partial L}{\partial \boldsymbol{h}_T} = \left(\frac{\partial \boldsymbol{y}_T}{\partial \boldsymbol{h}_T}\right)^T \frac{\partial L}{\partial \boldsymbol{y}_T} = \boldsymbol{V}^T \frac{\partial L}{\partial \boldsymbol{y}_T}$$

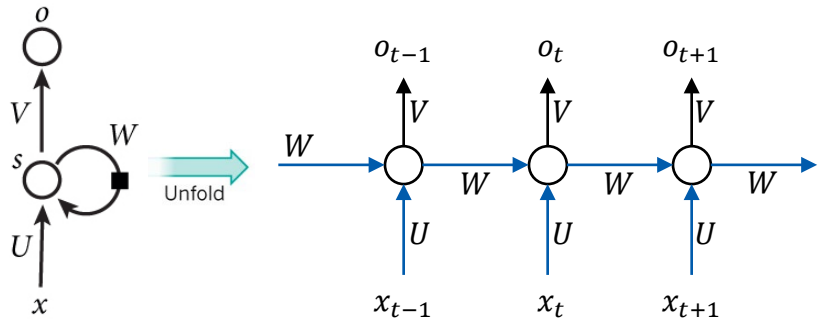


2. RNN训练-BPTT

误差反向传播W, U, b

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

$\boldsymbol{W}$的梯度$\frac{\partial L}{\partial \boldsymbol{W}}$，属于标量对矩阵求导。



为了借用先导知识的结论，

令$\boldsymbol{net}_t = \boldsymbol{U}\boldsymbol{x}_t + \boldsymbol{W}\boldsymbol{h}_{t-1} + \boldsymbol{b}$，

$z = f(\boldsymbol{Y})$ $\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{A} + \boldsymbol{B}$	$\frac{\partial z}{\partial \boldsymbol{X}} = \frac{\partial z}{\partial \boldsymbol{Y}}\boldsymbol{A}^T$

$$\frac{\partial L}{\partial \boldsymbol{W}} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial \boldsymbol{net}_t} (\boldsymbol{h}_{t-1})^T$$

$$= \sum_{t=1}^{T} \left(\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{net}_t}\right)^T \frac{\partial L_t}{\partial \boldsymbol{h}_t} (\boldsymbol{h}_{t-1})^T = \sum_{t=1}^{T} \left(\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{net}_t}\right)^T \boldsymbol{\delta}_t (\boldsymbol{h}_{t-1})^T$$

$$= \sum_{t=1}^{T} \left(f'(\boldsymbol{net}_t)\right)^T \boldsymbol{\delta}_t (\boldsymbol{h}_{t-1})^T$$

问题： $\boldsymbol{h}_{t-1}$也与$\boldsymbol{W}$有关，为什么上式求导时，没有考虑呢？

2. RNN训练-BPTT

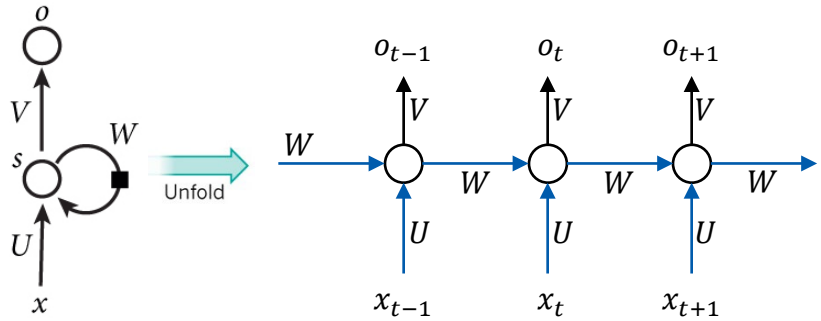
误差反向传播W, U, b

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

同理，$\boldsymbol{U}, \boldsymbol{b}$的梯度$\frac{\partial L}{\partial \boldsymbol{U}}, \frac{\partial L}{\partial \boldsymbol{b}}$分别为：

$$\frac{\partial L}{\partial \boldsymbol{U}} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial \boldsymbol{net}_t} (\boldsymbol{x}_t)^T = \sum_{t=1}^{T} \left(\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{net}_t}\right)^T \frac{\partial L_t}{\partial \boldsymbol{h}_t} (\boldsymbol{x}_t)^T = \sum_{t=1}^{T} \left(f'(\boldsymbol{net}_t)\right)^T \boldsymbol{\delta}_t (\boldsymbol{x}_t)^T$$

$$\frac{\partial L}{\partial \boldsymbol{b}} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial \boldsymbol{net}_t} = \sum_{t=1}^{T} \left(\frac{\partial \boldsymbol{h}_t}{\partial \boldsymbol{net}_t}\right)^T \frac{\partial L_t}{\partial \boldsymbol{h}_t} = \sum_{t=1}^{T} \left(f'(\boldsymbol{net}_t)\right)^T \boldsymbol{\delta}_t$$



2. RNN训练-BPTT

误差反向传播V, c, W, U, b

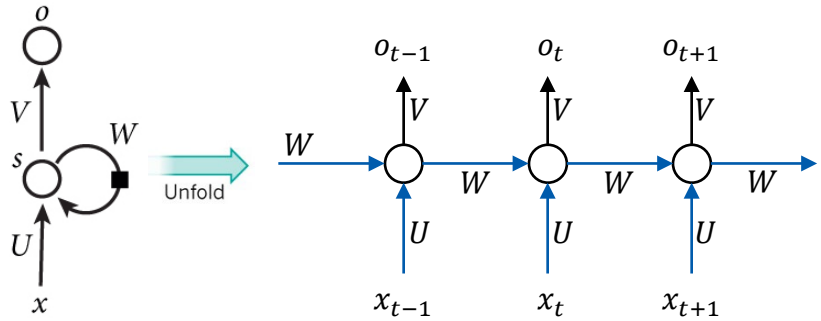
$$\frac{\partial L}{\partial V} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial y_t} (h_t)^T$$

$$\frac{\partial L}{\partial c} = \sum_{t=1}^{T} \frac{\partial L_t}{\partial y_t}$$

$$\frac{\partial L}{\partial W} = \sum_{t=1}^{T} \left(f'(net_t)\right)^T \delta_t (h_{t-1})^T$$

$$\frac{\partial L}{\partial U} = \sum_{t=1}^{T} \left(f'(net_t)\right)^T \delta_t (x_t)^T$$

$$\frac{\partial L}{\partial b} = \sum_{t=1}^{T} \left(f'(net_t)\right)^T \delta_t$$



T时刻隐藏层误差项值δ_T为：

$$\delta_T = \frac{\partial L}{\partial h_T} = \left(\frac{\partial y_T}{\partial h_T}\right)^T \frac{\partial L}{\partial y_T} = V^T \frac{\partial L}{\partial y_T}$$

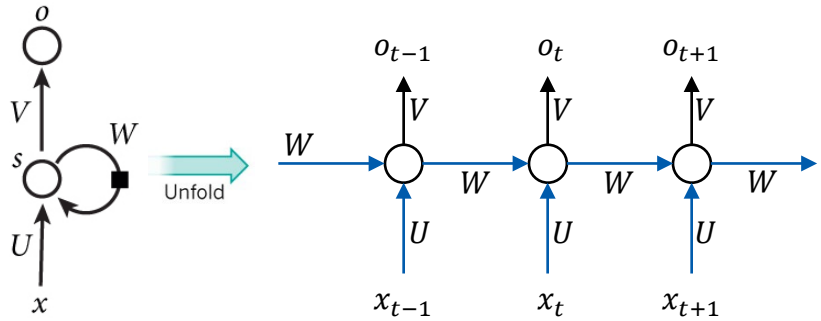
$$\delta_t = V^T \frac{\partial L}{\partial y_t} + W^T \left(f'(net_{t+1})\right)^T \delta_{t+1}$$

2. RNN训练-BPTT

梯度问题

误差在沿着时间线向前反传

$$\boldsymbol{\delta}_t = \boldsymbol{V}^T \frac{\partial L}{\partial \boldsymbol{y}_t} + \boldsymbol{W}^T \left(f'(\boldsymbol{net}_{t+1})\right)^T \boldsymbol{\delta}_{t+1}$$



$\boldsymbol{W}$: 初始值为standard normal distribution(大多数都是小于1)

f'：tanh 或是sigmoid 函数 ($f' < 1$)

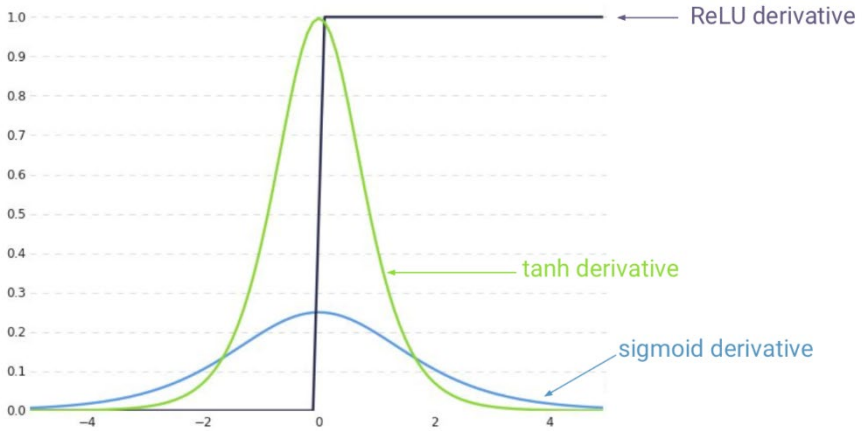
因此，随着误差反传， $\boldsymbol{\delta}_{t+1}$这一项累乘值越来越小， $\boldsymbol{\delta}_{t+1}$对$\boldsymbol{\delta}_t$的贡献越来越小。

2. RNN训练-BPTT

梯度问题—解决方法

$$\boldsymbol{\delta}_t = \boldsymbol{V}^T \frac{\partial L}{\partial \boldsymbol{y}_t} + \boldsymbol{W}^T \left(f'(\boldsymbol{net}_{t+1})\right)^T \boldsymbol{\delta}_{t+1}$$

- 选用ReLU激活函数
- W初始化时候用单位矩阵
- 改变RNN的结构，例如使用LSTM和GRU



RNN

RNN推导

LSTM

GRU

代码讲解