

## Recurrent Neural Network ( RNN )

### 核心思想

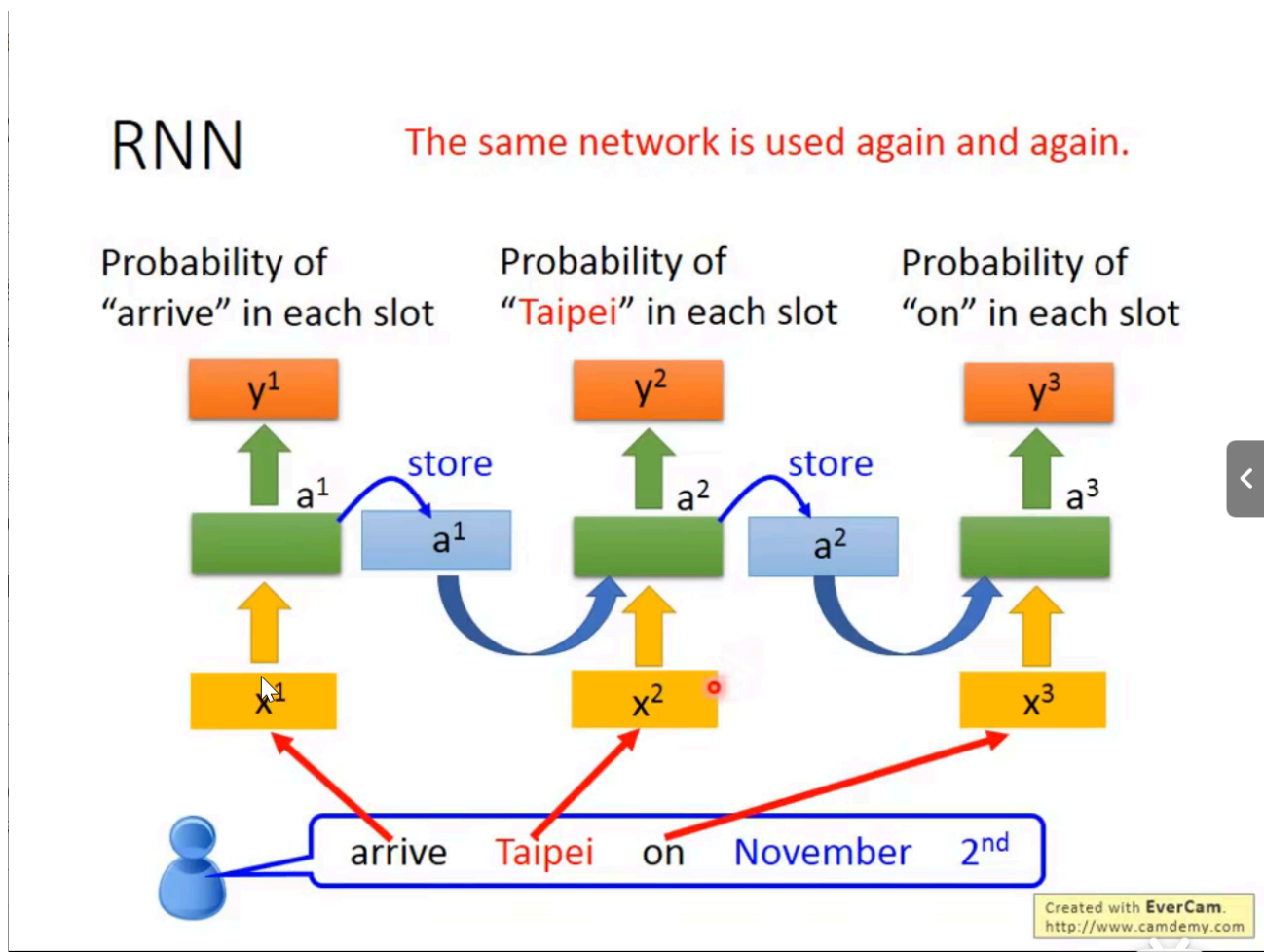
RNN的核心思想是在处理序列时，网络不仅考虑**当前输入**，还利用**之前的信息**

### 工作原理

具体来说：

1. 隐藏层接收**当前输入** $x_t$ 和前一时间步的隐藏状态 $h_{t-1}$
2. 隐藏层计算得到新的**隐藏状态** $h_t$ （这就是隐藏层的输出）
3. 这个隐藏层的输出 $h_t$ 会：
  - 被存储为记忆并传递到下一时间步
  - 作为当前时间步的状态表示

同一个Network在3个不同的时间点，被使用了3次



# Long Short-term Memory(LSTM)

## 核心理念

引入了门控机制和细胞状态，使网络能够更有效地学习长距离依赖关系

## 核心组件

1. 细胞状态 (Cell State):
  - 存储的是系统的长期记忆信息
  - 表示为  $C_t$
2. 三种门控机制:
  - \*\*遗忘门 (Forget Gate)\*\*: 决定丢弃细胞状态中的哪些信息
  - \*\*输入门 (Input Gate)\*\*: 决定更新哪些新信息到细胞状态
  - \*\*输出门 (Output Gate)\*\*: 决定基于细胞状态输出哪些信息
3. 隐藏状态 (Hidden State):
  - 对外输出，反映当前上下文，也是下一个时间步的输入
  - 由细胞状态经过输出门得出
  - 表示为  $h_t$

## 工作流程

⊙为逐元素相乘，这是因为门控机制的存在。

门的输出会赋予输入的每个维度一个权重，因此要用到⊙

### 1. 遗忘门计算

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

其中：

- $W_f$ 是遗忘门的权重矩阵， $b_f$ 是偏置
- $f_t$ 是遗忘概率，决定上一时刻的细胞状态  $C_t$  保留多少

### 2. 输入门计算

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \quad \tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

其中：

- $W_i$ 、 $W_C$ 是输入门和候选记忆的权重
- $\tilde{C}_t$ 为候选记忆

- $i_t$  主要用于控制候选记忆  $\tilde{C}_t$  对记忆细胞  $C_t$  的贡献

### 3. 更新记忆细胞

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

- $f_t$  决定了上一时间步的细胞状态应该保留多少

### 4. 输出门计算

$$o_t = \sigma(W_o \odot [h_{t-1}, x_t] + b_o)$$

其中：

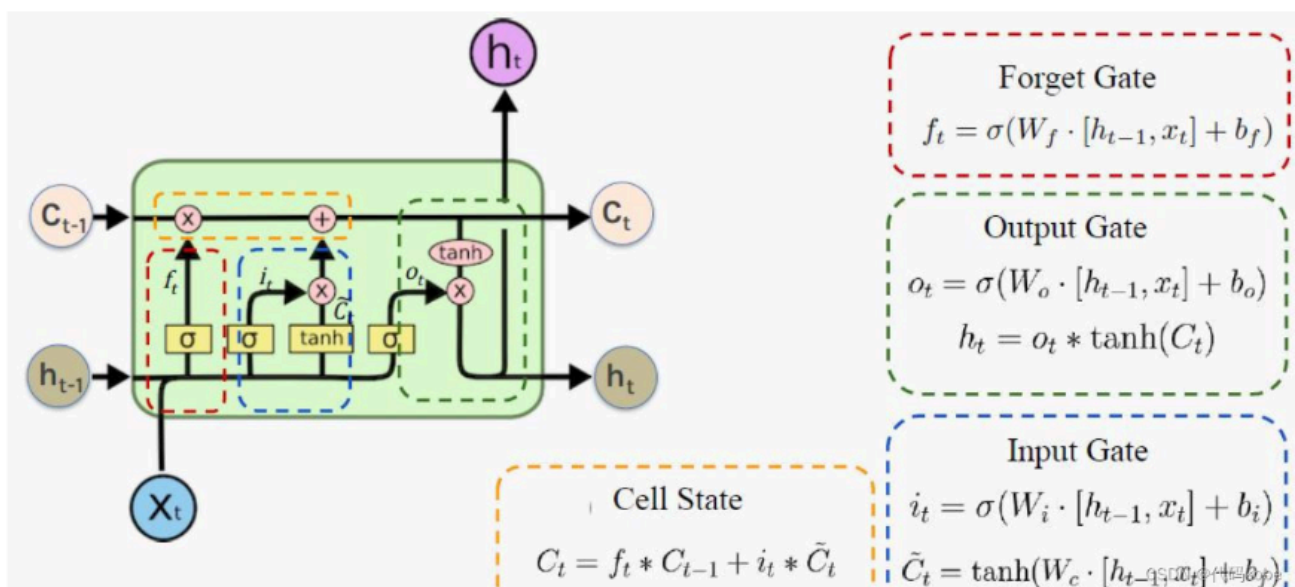
- $W_o$  是输出门的权重

### 5. 计算隐藏状态并传递到下一时刻

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

其中：

- $o_t$  为输出门的输出， $C_t$  为细胞状态



## 优势

**LSTM (长的短期记忆网络)** 被专门设计用来缓解RNN中的**梯度消失**问题。LSTM通过几个关键机制有效地处理了梯度消失：

1. **单元状态 (Cell State)** - LSTM引入了一条贯穿整个序列的"高速公路"，让信息可以直接从早期时间步传递到后期时间步，而不必经过多次非线性变换。这条路径上只有简单的

线性操作（加法和乘法），使梯度能够更容易地反向流动。

2. **门控机制**- LSTM有三个门：输入门、遗忘门和输出门。这些门控制信息的流动：

- 遗忘门决定丢弃哪些信息
- 输入门决定存储哪些新信息
- 输出门决定输出哪些信息

3. **加法操作** - 与标准RNN中的乘法操作不同，LSTM使用加法来更新单元状态。由于加法操作的导数是1，不会导致梯度缩放，从而避免了梯度消失问题。

4. **选择性记忆机制** - LSTM可以学会长期保存重要信息，同时有选择地更新或丢弃不再需要的信息。

从数学角度看，标准RNN中梯度会依赖于权重的连乘，而LSTM中的单元状态更新**主要涉及加法操作（Memory和Input的值是相加的）**，反向传播时梯度不会因为连乘而衰减。

## GRU（门控循环单元）

简化版的LSTM，是LSTM的变体，与LSTM相比结构更加简单，训练速度更加快，无单独的记忆单元

### 核心组件

1. **更新门（update gate）**

用于融合当前信息和过去信息

决定有多少过去的信息需要保留到当前时刻，以及有多少当前的输入信息需要被整合到新的隐藏状态中  $z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$

- $W_z$ 是更新门的权重矩阵
- $z_t$ 接近1时，表示更多地保留过去的隐藏状态
- $z_t$ 接近0时，表示更多地使用当前的输入来更新隐藏状态

2. **重置门（reset gate）**

用于控制要遗忘多少过去的信息  $r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$

- $W_r$ 是重置门的权重矩阵
- $r_t$ 接近0时，表示过去的隐藏状态  $h_{t-1}$  被**更多的遗忘**，模型更容易捕捉到新的输入信息

### 工作流程

1. **计算更新门**

更新门控制多少旧的记忆  $h_{t-1}$  保留到当前  $h_t$  中  $z_t = \sigma(W_z \cdot [h_{t-1}, x_t])$

## 2. 计算重置门

重置门控制遗忘多少旧信息，用于生成新的候选状态  $r_t = \sigma(W_r \cdot [h_{t-1}, x_t])$

## 3. 计算候选状态

基于当前输入  $x_t$  和部分历史信息  $h_{t-1}$  以及重置门  $r_t$ ，生成一个临时状态  $\tilde{h}_t = \tanh(W \cdot [r_t \cdot h_{t-1}, x_t])$

## 4. 计算新的隐藏状态

结合更新门  $z_t$ ，将旧状态  $h_{t-1}$  和候选状态  $\tilde{h}_t$  融合  $h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \tilde{h}_t$

# Slot Filling (槽位填充)

## 工作原理

- 槽位定义**：根据特定领域或任务预先定义一组槽位（如餐厅预订可能包括"菜系"、"人数"、"日期"、"时间"等槽位）
- 序列标注**：通常将槽位填充建模为序列标注问题
  - 对输入句子中的每个词进行标注，指出它是否属于某个槽位

## 缺陷

### 梯度爆炸原因

- 重复矩阵乘法**：在反向传播过程中，梯度会通过时间步骤反向传递，这涉及到重复的权重矩阵乘法。如果权重矩阵的特征值大于1，那么随着时间步的增加，梯度值会呈指数增长。
- 长序列累积效应**：当处理长序列数据时，梯度需要通过多个时间步传播，每通过一步就会与权重矩阵相乘，如果权重较大，梯度会迅速累积膨胀。
- 激活函数的导数**：如果使用的激活函数（如ReLU）在某些区域导数大于1，也会放大梯度。

### 梯度消失原因

- 重复矩阵乘法**：与梯度爆炸类似，但当权重矩阵的特征值小于1时，重复相乘会导致梯度值逐渐变得非常小。
- 饱和激活函数**：传统RNN常用的sigmoid和tanh激活函数，它们的导数在输入绝对值较大时会接近于0，进一步促进了梯度消失。

3. **长距离依赖问题**：对于长序列，早期时间步的信息需要经过多次传递才能影响当前输出，但梯度消失使得网络难以捕获这些长期依赖关系。

## Connectionist Temporal Classification (连接时序分类)

CTC是一个用于**序列建模和标注对齐**的损失函数

### 作用

在语音识别中，输入是一个很长的声学特征序列，而输出是一段文本（如“hello”）。传统方法需要知道每个字符对应声学特征的起止时间，即对齐信息。CTC 的关键就在于：

- **不需要对齐标注**，它自动学习从输入序列中找出最可能的输出序列。
- 允许重复字符和空白符 `blank`，通过规则对多个路径进行归一合并，从而输出最终的目标序列。

### 学习

CTC Loss 给每一条“合法路径”分配一个概率，最后对所有合法路径求和得到一个总概率。模型学习的目标就是：让这些合法路径的概率之和最大化（也就是 loss 越小越好）。