10.1 RNN、LSTM、GRU常见问题★

1．什么是循环神经网络

RNN对具有序列特性的数据非常有效，它能挖掘数据中的时序信息以及语义信息，利用了RNN的这种能力，使深度学习模型在解决语音识别、语言模型、机器翻译以及时序分析等NLP领域的问题时有所突破。

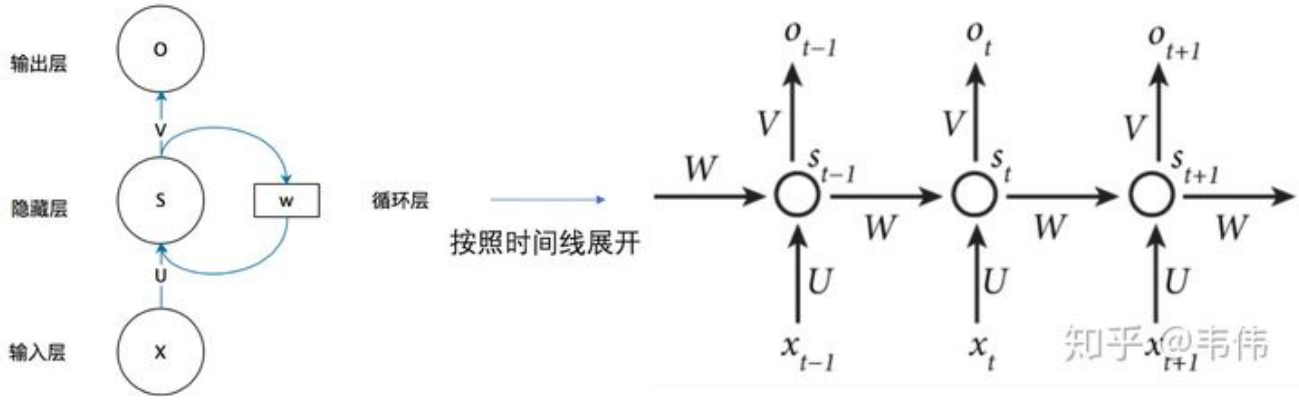
2．为什么要发明循环神经网络

第一句话：I like eating apple！（我喜欢吃苹果！）

第二句话：The Apple is a great company！（苹果真是一家很棒的公司！）

现在的任务是要给apple打Label，我们都知道第一个apple是一种水果，第二个apple是苹果公司，假设我们现在有大量的已经标记好的数据以供训练模型，当我们使用全连接的神经网络时，我们做法是把apple这个单词的特征向量输入到我们的模型中，在输出结果时，让我们的label里，正确的label概率最大，来训练模型，但我们的语料库中，有的apple的label是水果，有的label是公司，这将导致，模型在训练的过程中，预测的准确程度，取决于训练集中哪个label多一些，这样的模型对于我们来说完全没有作用。**问题就出在了我们没有结合上下文去训练模型，而是单独的在训练apple这个单词的label，这也是全连接神经网络模型所不能做到的，于是就有了我们的循环神经网络。**

3．RNN结构



等等，这又是什么？？别慌，很容易看，举个例子，有一句话是，I love you，那么在利用RNN做一些事情时，比如命名实体识别，上图中的 X_{t-1} 代表的就是I这个单词的向量， X 代表的是love这个单词的向量， X_{t+1} 代表的是you这个单词的向量，以此类推，我们注意到，上图展开后，W一直没有变，**W其实是每个时间点之间的权重矩阵**，我们注意到，RNN之所以可以解决序列问题，**是因为它可以记住每一时刻的信息，每一时刻的隐藏层不仅由该时刻的输入层决定，还由上一时刻的隐藏层决定**，公式如下，其中 O_t 代表t时刻的输出，S_t 代表t时刻的隐藏层的值：

$$O_t = g(V \cdot S_t)$$

$$S_t = f(U \cdot X_t + W \cdot S_{t-1})$$

St的值不仅仅取决于Xt，还取决于St-1

值得注意的一点是，在整个训练过程中，每一时刻所用的都是同样的W。

4．RNN的优缺点

优点：

- 处理任意长度的输入
- 模型形状不随输入长度增加
- 计算考虑了历史信息
- 权重随时间共享

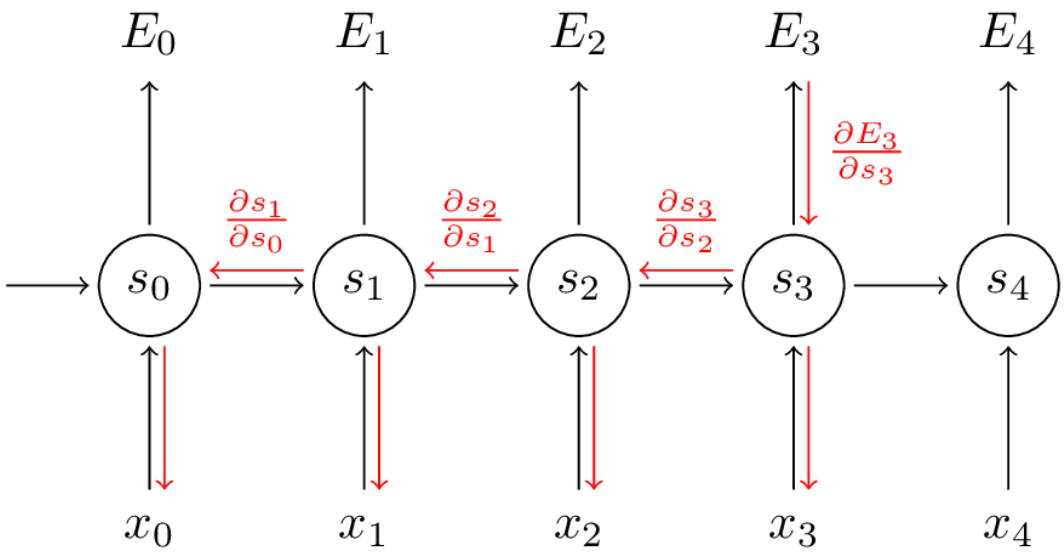
缺点：

- 计算速度慢
- 难以获取很久以前的信息
- 无法考虑当前状态的任何未来输入

5．RNN的训练

由于RNN在所有的时间步中共享参数（U,V,W）。由于RNN模型如果需要实现长期记忆的话需要将当前的隐含态的计算与前n次的计算挂钩，那样的话计算量会呈指数式增长，导致模型训练的时间大幅增加，因此RNN模型一般不直接用来进行长期记忆计算。使用BPTT训练的普通RNNs由于消失/爆炸梯度问题而难以学习长期依赖关系(例如，步骤之间相距很远的依赖关系)。

6．RNN的梯度消失与梯度爆炸

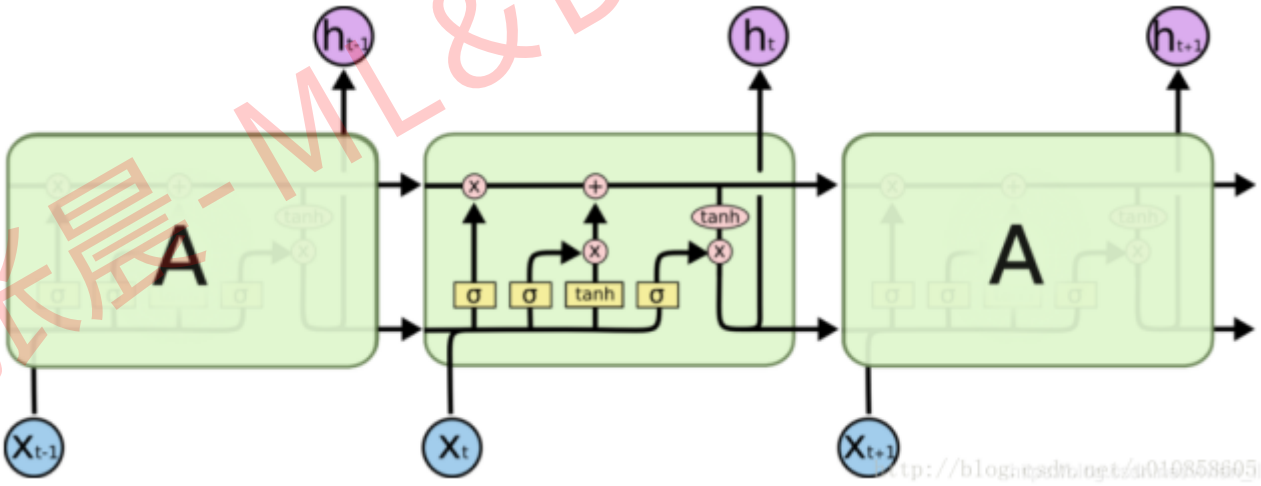


梯度消失和爆炸的根本原因在于$\prod_{j=k+1}^{t} \frac{\partial S_j}{\partial S_{j-1}}$ 这里。

要消除这种情况就需要把这一坨在求偏导的过程中去掉，至于怎么去掉，一种办法就是使 $\frac{\partial S_j}{\partial S_{j-1}} \approx 1$ 另一种办法就是使 $\frac{\partial S_j}{\partial S_{j-1}} \approx 0$ 。

为了解决这种问题，有三种方式：（1）合适地初始化W，即采用正则化；（2）将tanh或sigmoid替换为ReLU；（3）使用LSTM或者GRU。

7．LSTM结构



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

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] \; + \; b_i\right)$$

$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] \; + \; b_C)$$

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

$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

8．LSTM三个门的作用

- 遗忘门：用来控制之前的信息有多少可以流传至当前时刻（做取舍）
- 输入门：用来控制为细胞状态增加多少新信息（做取舍）
- 输出门：控制细胞状态中有多少信息可以作为当前时刻的输出（做取舍）

9．LSTM解决了RNN的什么问题

长时依赖问题（梯度消失）：RNN所谓梯度消失的真正含义是，梯度被近距离梯度主导，远距离梯度很小，导致模型难以学到远距离的信息。

LSTM和普通RNN正是贵族和乞丐，RNN什么信息它都存下来，因为它没有挑选的能力，而LSTM不一样，它会选择性的存储信息，因为它能力强，它有门控装置，它可以尽情的选择。普通RNN只有中间的Memory Cell用来存所有的信息，而LSTM多了三个Gate，也就是三个门，什么意思呢？在现实生活中，门就是用来控制进出的，门关上了，你就进不去房子了，门打开你就能进去，同理，这里的门是用来控制每一时刻信息记忆与遗忘的。

RNN梯度消失的原因是，随着梯度的传导，梯度被近距离梯度主导，模型难以学习到远距离的信息。具体原因也就是 $\prod_{k=t+1}^{T} \frac{\partial h^{(k)}}{\partial h^{(k-1)}}$ 部分，在迭代过程中，每一步 $\frac{\partial h^{(k)}}{\partial h^{(k-1)}}$ **始终在[0,1]之间或者始终大于1。**

而对于LSTM模型而言，针对 $\frac{\partial C^{(k)}}{\partial C^{(k-1)}}$ 求得，

$$\frac{\partial C^{(k)}}{\partial C^{(k-1)}} = \frac{\partial C^{(k)}}{\partial f^{(k)}}\frac{\partial f^{(k)}}{\partial h^{(k-1)}}\frac{\partial h^{(k-1)}}{\partial C^{(k-1)}} + \frac{\partial C^{(k)}}{\partial i^{(k)}}\frac{\partial i^{(k)}}{\partial h^{(k-1)}}\frac{\partial h^{(k-1)}}{\partial C^{(k-1)}} + \frac{\partial C^{(k)}}{\partial a^{(k)}}\frac{\partial a^{(k)}}{\partial h^{(k-1)}}\frac{\partial h^{(k-1)}}{\partial C^{(k-1)}} + \frac{\partial C^{(k)}}{\partial C^{(k-1)}}$$

具体计算后得到，

$$\begin{aligned}\frac{\partial C^{(k)}}{\partial C^{(k-1)}} &= C^{(k-1)}\sigma^{'}(\cdot)W_f o^{(k-1)}tanh^{'}(C^{(k-1)}) \\ &+ a^{(k)}\sigma^{'}(\cdot)W_i o^{(k-1)}tanh^{'}(C^{(k-1)}) \\ &+ i^{(k)}tanh^{'}(\cdot)W_c * o^{(k-1)}tanh^{'}(C^{(k-1)}) \\ &+ f^{(t)}\end{aligned}$$

$$\prod_{k=t+1}^{T}\frac{\partial C^{(k)}}{\partial C^{(k-1)}} = (f^{(k)}f^{(k+1)}\dots f^{(T)}) + other$$

在LSTM迭代过程中，针对 $\prod_{k=t+1}^{T}\frac{\partial C^{(k)}}{\partial C^{(k-1)}}$ 而言，每一步 $\frac{\partial C^{(k)}}{\partial C^{(k-1)}}$ **可以自主的选择在[0,1]之间，或者大于1**，因为 $f^{(k)}$ 是可训练学习的。那么整体 $\prod_{k=t+1}^{T}\frac{\partial C^{(k)}}{\partial C^{(k-1)}}$ 也就不会一直减小，

LSTM遗忘门值可以选择在[0,1]之间，让LSTM来改善梯度消失的情况。也可以选择接近1，让遗忘门饱和，此时远距离信息梯度不消失。也可以选择接近0，此时模型是故意阻断梯度流，遗忘之前信息。

远距离梯度不至于完全消失，也就能够缓解RNN中存在的梯度消失问题。LSTM虽然能够缓解梯度消失问题，但并不能够缓解梯度爆炸问题，仍有可能发生梯度爆炸。但是，由于LSTM众多门控结构，和普通RNN相比，LSTM发生梯度爆炸的频率要低很多。梯度爆炸可通过梯度裁剪解决。

10．LSTM各模块分别使用什么激活函数？可以使用别的激活函数吗？

- 遗忘门：用来控制之前的信息有多少可以流传至当前时刻（做取舍），因此激活函数的取值应该在[0,1]范围之内，取sigmoid作为激活函数。
- 输入门：用来控制为细胞状态增加多少新信息（做取舍），激活函数的取值也应该在[0,1]范围之内，取sigmoid作为激活函数。通过tanh得到细胞状态的新信息（无需做取

舍），tanh函数的输出在[-1,1]范围之内，这与大多数场景下特征分布是0中心吻合。 此外，tanh函数在输入为0附近相比sigmod有更大的梯度，模型收敛更快。tanh用其他激活函数代替应该也可以。

- 输出门：控制细胞状态中有多少信息可以作为当前时刻的输出（做取舍），激活函数的取值也应该在[0,1]范围之内，取sigmoid作为激活函数。将细胞状态信息Ct经过tanh层转化为[-1,1]范围之间的值（无需做取舍），tanh用其他激活函数代替应该也可以。

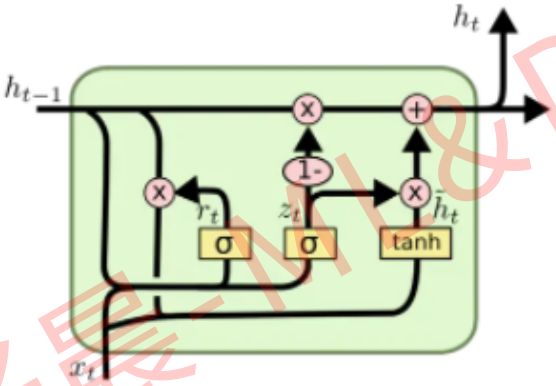
11．LSTM如何实现长 短期记忆 功能的?

- 当当前输入的序列中没有重要的信息时，LSTM的遗忘门的数值接近于1，以保留之前的信息；更新门的数据接近于0，即当前的信息不会大部分传入细胞状态中，此时过去的记忆会被保存，从而实现了长期的记忆功能；
- 当当前输入的序列中出现了重要的信息时，LSTM应该把其存入记忆时，此时输入门的数值将接近于1，即当前的信息大部分被保留下来；遗忘门的数值接近于0，这样旧的记忆大部分被遗忘，新的重要的信息被记忆。经过这样的设计，整个网络更容易学习到序列之间的长期依赖。

12．LSTM的hidden state 及cell state分别保存了什么信息

cell state是存储了长期记忆，借助forget gate，hidden state存储了短期记忆。

13．GRU结构



$$z_t = \sigma\left(W_z \cdot [h_{t-1}, x_t]\right)$$
$$r_t = \sigma\left(W_r \cdot [h_{t-1}, x_t]\right)$$
$$\tilde{h}_t = \tanh\left(W \cdot [r_t * h_{t-1}, x_t]\right)$$
$$h_t = (1 - z_t) * h_{t-1} + z_t * \tilde{h}_t$$

14．LSTM与GRU的比较

- 在LSTM中，遗忘门确定要保留先前单元状态的哪一部分，而输入门确定要添加的新内存的数量。这两个门相互独立，这意味着通过输入门添加的新信息量完全独立于通过忘记门保留的信息。对于GRU，更新门负责确定要保留前一存储器中的哪些信息，还负责控制要添加的新存储器。这意味着在GRU中保留先前的内存以及向该内存中添加新信息并不是独立的。
- 与LSTM相比，GRU的训练速度更快，这是因为训练期间需要更新的权重和参数数量较少。这可以归因于与LSTM的三个门相比，GRU单元中的门数量较少（两个门）。

15．怎么选择LSTM，GRU

通常情况下LSTM和GRU两者效果相差不大，GRU训练更快，所以一般会先选择使用GRU进行训练和调参，当无法再继续优化时可以把GRU替换成LSTM来看看是否有提高。

- [ ] https://zhuanlan.zhihu.com/p/123211148 <https://zhuanlan.zhihu.com/p/123211148>
- [ ] https://blog.csdn.net/vivian_ll/article/details/88780661 <https://blog.csdn.net/vivian_ll/article/details/88780661>
- [ ] https://www.jianshu.com/p/53e457937557 <https://www.jianshu.com/p/53e457937557>
- [ ] http://www.elecfans.com/d/1607848.html <http://www.elecfans.com/d/1607848.html>