><font size='4px' color='purple'>\*\*最近阅读了[Bello, Irwan等(2016)的论文《Neural Combinatorial Optimization with Reinforcement Learning》](https://arxiv.org/pdf/1611.09940.pdf)，这篇论文算是开启了基于RNN和RL求解组合优化问题的新篇章。作者的博士研究课题是使用强化学习(RL)求解车间生产调度问题(job shop scheduling)，由于车间调度本质上就是一种组合优化问题，因此是不是可以借用这篇论文的思想并将其应用于一个新的领域呢？所以在此整理了RNN和LSTM相关知识，为后面的研究和实现打下理论基础！\*\*</font>

# 1. 循环神经网络

人类不是每一秒都能从头开始思考的。当你阅读这篇文章时，你会基于你对前面单词的理解来理解每个单词，而不会把所有东西都扔掉，重新开始思考。人们的思维是持续的。

传统的神经网络无法做到这一点，这似乎是一个很大的缺点。例如，假设你想对电影中每时每刻发生的事件进行分类，目前还不清楚传统的神经网络如何利用它对电影中先前事件的推理来告知后来的事件。

循环神经网络解决了这个问题。它们是具有循环的网络，可以让信息持续下去。

![循环神经网络有循环](https://img-blog.csdnimg.cn/20201205104234996.png#pic\_center =153x237 )

上面的图中，神经网络$A$，其输入是$x\_t$，输出一个值$h\_t$，环可以让信息从一步通向网络传递到下一步。

这些循环使得循环神经网络看起来有点神秘，但是往深处想想，其实RNN和一般的NN并没有多大的不同。一个RNN可以被认为是同一个网络的多个副本，每个副本都传递一个信息给下一个副本。考虑一下展开后的循环是什么样的：

![展开的循环神经网络](https://img-blog.csdnimg.cn/20201205105153907.png#pic\_center)

这个链式性质表明RNN是和序列和列表密切相关的。它们是用于处理此类数据的神经网络的自然结构。

将前一个节点的信息传递给下一个的做法是很有效的，最近几年，RNN在不同的领域上都取得了可喜的效果：语音识别、语言模型、机器翻译、图像字幕…。可以看看Andrej Karpathy讨论RNN令人惊喜的成果的博文[《The Unresonable Effectiveness of Recurrent Neural Network》](http://karpathy.github.io/2015/05/21/rnn-effectiveness/)。

取得这些成果的重点是，他们都用了一种特殊的RNN——LSTM，在很多应用上，它的效果都比标准的RNN好很多。这篇文章的重点也是讲LSTM的。

# 2. 长期依赖问题

RNN的一个吸引人之处是它可以将先前的信息与当前的任务联系起来，就像视频前面的帧可以输入到当前环节来预测或者理解当前帧。如果RNN能真正做到这样，那人它的确非常有用了，但是它真的能吗？实际上，这要视情况而定。

有时，我们只需要就近的一些信息就可以完成当前任务。例如，一个语言模型想要根据前面的词预测最后一个词，如要预测“the clouds are in the （\*sky\*），”的最后一个词，我们不需要更多的上下文语境就能知道最后一个词是sky。在这个例子中，相关信息和需要它的地方之间距离很近，RNN可以学会使用过去的信息。

![X0、X1与h3距离较近](https://img-blog.csdnimg.cn/20201205105834866.png#pic\_center)

但是，也有很多句子需要更多上下文信息，如“I grew up in France. … I speak fluent （\*French\*）.”。从临近的语境中可以知道，最后一个词应该是一种语言，但是要知道是哪种语言，必须要前面的“France”，要再往前推。需要的信息和当前点可能隔很远。

不幸的是，随着距离的增加，RNN并不能学会将这些信息联系起来。

![X0、X1可能与需要预测的位置ht+1距离很远](https://img-blog.csdnimg.cn/20201205105929705.png#pic\_center)

理论上，RNNs完全可以掌控“长期依赖”，我们可以人为地选择好的参数。但实践证明，RNN基本不可能学习很长的句子。[Hochreiter(1991德国)](http://people.idsia.ch/~juergen/SeppHochreiter1991ThesisAdvisorSchmidhuber.pdf)和[Begnio(1994)](http://www-dsi.ing.unifi.it/~paolo/ps/tnn-94-gradient.pdf)对此有很深的研究，并且找到了根本原因。

而幸运的是，LSTM不会存在这个问题。

# 3. LSTM 网络

LSTM是一种特殊的RNN，可以解决长期依赖问题。它是由[Hochreiter和Schnidhuber(1997)](http://www.bioinf.jku.at/publications/older/2604.pdf)最先提出的，后来很多人用它解决了很多问题，现在被广泛地应用。

LSTM是专门设计出来解决长期依赖问题的。记住长周期有用信息是它的基本功能，而不用去努力学习得到。

所有递归神经网络都具有神经网络模块重复链的形式。在标准的RNNs中，这个重复模块将有一个非常简单的结构，比如一个单一的tanh层。

![仅有一个tanh层的RNN](https://img-blog.csdnimg.cn/20201205110118273.png#pic\_center)

LSTM也是这种链式的结构，只是重复模块的内部结构不一样，它不是单独的NN层，而是4个相互影响的NN。

![LSTM中的重复模块包含四个交互层](https://img-blog.csdnimg.cn/2020120511023147.png#pic\_center)

在上图中，每一行都将一个完整的向量从一个节点的输出传递到其他节点的输入。粉色的圆圈代表点态操作，比如向量加法，而黄色方框是学习到的神经网络层。线合并表示连接，线分叉表示复制的内容和复制到不同位置。

其中各个符号的意义如下：

![LSTM中各符号的意义](https://img-blog.csdnimg.cn/20201205110315476.png#pic\_center)

# 4. LSTM的核心思想

LSTM的重点是单元状态（cell state），即下面这条从图中最上面穿过的水平线。

cell state有点像传送带，它沿着整个链往下延伸，只有微弱的线性交互作用，很容易让信息保持不变地沿着其流动。

![单元状态流动](https://img-blog.csdnimg.cn/20201205110430662.png#pic\_center)

依靠称为门（gates）的结构的精细管理，LSTM确实有能力从单元状态删除或者向单元状态增加信息。

门是一种选择性地让信息通过的方式。门由sigmoid神经网络层和一个点乘运算组成。

![门的组成：sigmoid层和点乘](https://img-blog.csdnimg.cn/20201205110516567.png#pic\_center)

Sigmoid层的输出介于0和1之间，描述了可以允许每个组件有多少可以通过。输出为0表示信息完全不通过，输出为1表示信息全部通过。

一个LSTM有3个这样的门来保护和控制单元状态。

# 5. 逐步看LSTM的工作原理

LSTM的第一步是决定从单元状态中丢弃哪些信息，这个决策是由包含sigmoid层的\*遗忘门（forget gate layer）制定的\*。遗忘门以$h\_{t-1}$和$x\_t$，针对单元状态$C\_{t-1}$中的每个值都输出一个介于0和1的值，1表示完全保留，0表示完全丢掉。

让我们回到基于前面的所有词预测下一个词的语言模型示例中，单元状态可能包含现在主语的性别，这样就可以使用正确的代词。而当看到一个新的主语时，我们就希望忘记旧主语的性别。

![遗忘门](https://img-blog.csdnimg.cn/20201205110637912.png#pic\_center)

下一步就是决定存储哪些新的信息到单元状态中。这由两部分构成，一是由sigmoid层构成的输入门（input gate layer），它用来决定要更新哪些值。另一个是由tanh层创建的新候选值向量$\tilde{C}\_{t}$，该向量会被添加到单元状态里。

例如语言模型中，希望将新主语的性别添加到单元状态中，以替换我们忘记的旧的性别。

![输入门](https://img-blog.csdnimg.cn/20201205110732679.png#pic\_center)

现在是时候将旧单元状态$C\_{t-1}$更新到新单元状态$C\_t$里了。前面已经决定了要做什么，这里只需要一次完成就好。

将旧状态与$ft$，从而忘记先前决定要遗忘的信息，然后加上$i\_{t} \* \tilde{C}\_{t}$，该值为根据更新每个状态值的程度缩放后的新的候选值。

在语言模型的示例中，这是我们实际删除旧主语的性别信息并添加新信息的地方，正如在前面的步骤中所决定的那样。

![单元状态更新](https://img-blog.csdnimg.cn/2020120511082270.png#pic\_center)

最后决定要输出什么。这个输出基于单元状态，但会有一个滤波过程。首先，一个sigmoid层决定输出哪些单元状态，然后将单元状态通过tanh层（将值压缩在-1到1之间），再与sigmoid门的输出相乘，得到最后的输出。

对于语言模型示例，由于只看到了一个主语，因此它可能希望输出与动词相关的信息。例如，它可以输出主语是单数还是复数，这样我们就可以知道如果一个动词接在后面，它应该变成什么形式。比如刚看到一个人称代词he或they（单元状态已经存储），而下一个词可能是一个动词，那么我们从人称代词（cell状态）就可以看出下一个动词的形式，比如(makes, make)，he对应makes，they对应make。

![输出门](https://img-blog.csdnimg.cn/20201205110853649.png#pic\_center)

# 6. LSTM的变体

上面描述的是最常见的标准LSTM，但是不是所有LSTM都是这样子的。事实上，每篇致力于LSTMs的paper所用的版本都有微小的差别，这些差别真的很小，但是其中一些是值得一提的。

其中一种流行的变体由[Gers和Schmidhuber(2000)](ftp://ftp.idsia.ch/pub/juergen/TimeCount-IJCNN2000.pdf)提出，加入了窥视孔连接（peephole connections）。这意味着，单元状态也输入到所有门中来判别哪些丢，哪些加。

![所有门都输入单元状态](https://img-blog.csdnimg.cn/2020120511094924.png#pic\_center)

上图在所有的门中都加入了单元状态，但有很多paper只部分加入。

另一种变体是使用耦合的遗忘门和输入门，不去分开决定遗忘什么输入什么，而是一起做决定，只有在代入信息时才会忘记，也只有在忘记旧值时才向状态输入新值。

![耦合的遗忘门和输入门](https://img-blog.csdnimg.cn/20201205111031220.png#pic\_center)

还有一种稍微不同的变体是GRU（Gated Recurrent Unit），由[cho(2014)](http://arxiv.org/pdf/1406.1078v3.pdf)提出。它将遗忘门和输入门合并为一个更新门（update gate），同时还合并了单元状态和隐层状态。该模型比LSTM简单，并且越来越流行。

![GRU](https://img-blog.csdnimg.cn/20201205111108521.png#pic\_center)

以上只是一小部分的LSTM变体，还有很多其他可用的，如Yao（2015）的[Depth Gated RNNs](http://arxiv.org/pdf/1508.03790v2.pdf)，还有一些完全不同的消除长期依赖的，如koutaik（2014）提出的[ Clockwork RNNs](http://arxiv.org/pdf/1402.3511v1.pdf)。

这些变体中哪一个是最好的呢？这些差异重要吗？[Greff等人(2015)](http://arxiv.org/pdf/1503.04069.pdf)对流行的变体做了一个很好的比较，发现它们都差不多。[Jozefowicz等人(2015)](http://jmlr.org/proceedings/papers/v37/jozefowicz15.pdf)测试了超过一万种RNN架构，发现有些架构在某些任务上比LSTMs工作得更好。

# 7. 结论

前面我提到了人们用RNNs取得的显著成果。实际上所有这些都是通过LSTMs实现的。在大多数任务中，它们确实工作得更好！

将LSTMs写成一组方程，看起来相当吓人。希望在这篇文章中，通过一步一步地讲解，可以更容易地理解其原理。

LSTMs是我们可以通过RNNs完成的一大步。人们自然会想：是否还有更大的进步？研究人员普遍认为：是的！还有下一步，那就是注意力机制！其思想是让RNN的每一步都从一些更大的信息集合中挑选信息。例如，如果使用RNN创建描述图像的标题，它可能会选择图像的一部分来查看输出的每个单词。事实上，[Xu等人(2015)](http://arxiv.org/pdf/1502.03044v2.pdf)正是这么做的，如果你想探究注意力，这可能是一个有趣的起点！使用注意力已经有了许多令人兴奋的结果，而且似乎还有更多的结果即将出现……

注意力并不是RNN研究中唯一令人兴奋的线索。例如，[Kalchbrenner等(2015)](http://arxiv.org/pdf/1507.01526v1.pdf)的网格(Grid) LSTMs似乎非常有前途。在生成模型中使用RNNs，如[Gregor, et al. (2015)](http://arxiv.org/pdf/1502.04623.pdf)， [Chung, et al.(2015)](http://arxiv.org/pdf/1506.02216v3.pdf)，或[Bayer &amp; Osendorfer(2015)](http://arxiv.org/pdf/1411.7610v3.pdf)也很有趣。过去的几年对于循环神经网络来说是一个激动人心的时期，而接下来的几年更将是有过之而无不及！