理解LSTM (Understanding LSTM Networks翻译)

假言 关注他

Understanding LSTM Networks

原文章地址：colah.github.io/posts/2...

作者：Christopher *Olah（Google Brain, Research Scientist.）*

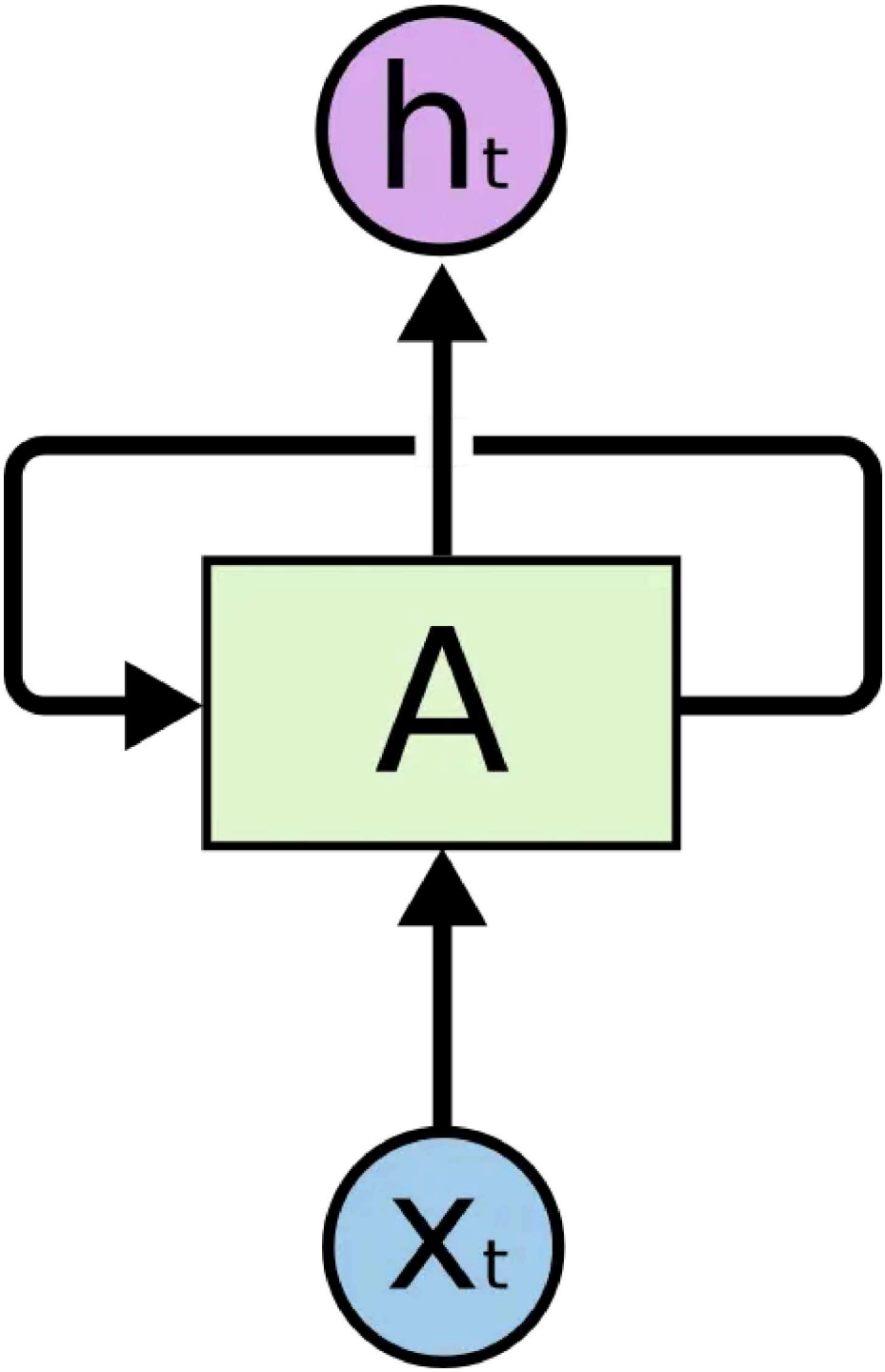
colah.github.io/about.h...

循环神经网络Recurrent Neural Networks

人不会每一秒钟都从头开始思考。当你读这篇文章的时候，你理解每一个单词都是基于你对之前单词的理解。你不会把所有的东西都扔掉，重新开始思考。你的思想有毅力。

传统的神经网络做不到这一点，这似乎是一个主要的缺点。例如，假设您想要对电影中每一点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其对电影中先前事件的推理来为后来的事件提供信息。

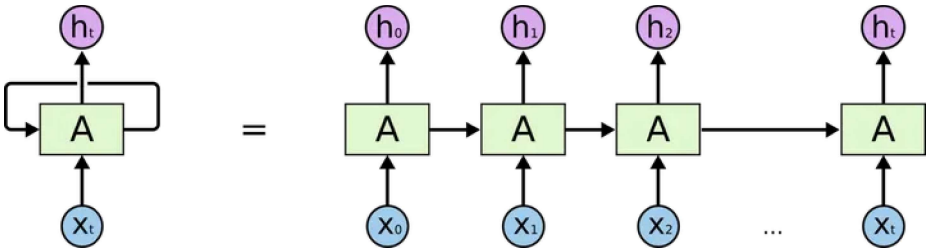
递归神经网络解决了这个问题。它们是包含循环的网络，允许信息持续存在。



Recurrent Neural Networks have loops.

在上面的图中，一个神经网络块 **A** ，观察某个输入**xt**并输出一个值**ht**。循环（loop）允许信息从网络的一个步骤传递到下一个步骤。

这些循环使得递归神经网络看起来有点神秘。然而，如果你多想一下，就会发现它们和普通的神经网络并没有太大的不同。递归神经网络可以看作是同一网络的多个副本，每个副本都向后继网络传递一条消息。考虑一下如果我们展开循环会发生什么:



An unrolled recurrent neural network.

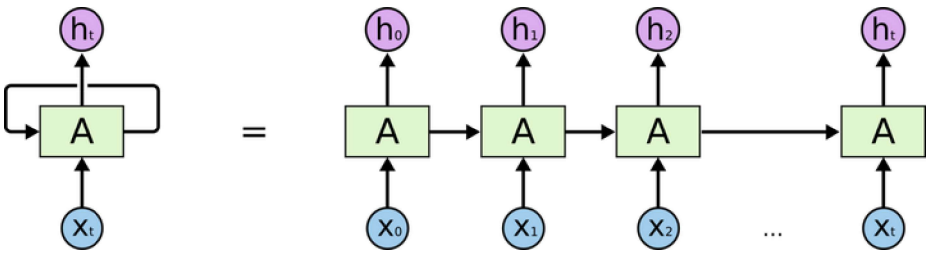
这种类似链的性质表明，递归神经网络与序列和列表密切相关。它们是神经网络用来处理这些数据的自然结构。

它们确实被大规模应用了，在过去的几年里，把RNNs应用到各种各样的问题上，并取得了令人难以置信的成功:语音识别、语言建模、翻译、图像字幕等等。我将把关于RNNs可以实现的惊人壮举的讨论留给Andrej Karpathy的博客文章 The Unreasonable Effectiveness of Recurrent Neural Networks。但它们真的很神奇。

这些成功的关键是使用“ LSTM”，这是一种非常特殊的递归神经网络，在很多任务上都能比标准的RNN好得多。 利用递归神经网络几乎可以实现所有令人兴奋的结果。 本文将探讨的是这些LSTM。

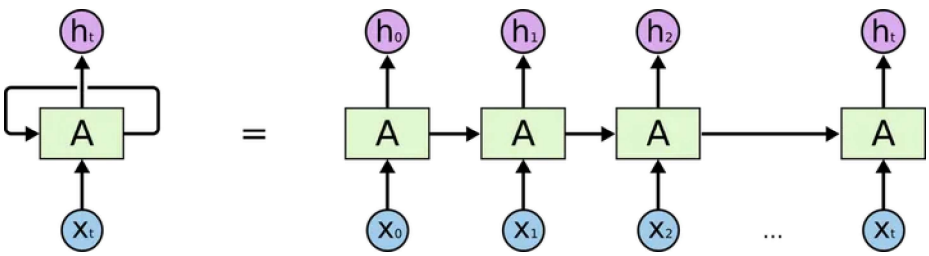
长期依赖的问题 (The Problem of Long-Term Dependencies)

RNNs的一个吸引人的地方是，他们可能能够将以前的信息与现在的任务联系起来，例如使用视频前面的几帧画面可能有助于理解现在这一帧的画面。如果RNNs能做到这一点，它们将非常有用。但它们能不能有效，这得视情况而定。



但也有一些情况，我们需要更多的上下文。试着预测课文中的最后一个单词“我在法国长大……我说一口流利的法语。” “最近的信息显示，下一个单词很可能是一种语言的名字，但如果我们想缩小范围，我们需要更早的法语语境。”相关信息与需要它的点之间的差距完全有可能变得非常大。

不幸的是，随着这种差距的扩大，RNNs无法学会连接信息。



从理论上讲，RNN绝对有能力处理这种“长期依赖性”。人们可以为他们精心选择参数，以解决这种形式的问题。遗憾的是，在实践中，RNN似乎无法学习它们。 Hochreiter (1991) [German] 和 Bengio, et al. (1994)等人对此问题进行了深入探讨。 他们发现了一些RNN很难做到的根本原因。【ai.dinfo.unifi.it/paolo...

people.idsia.ch/~juerge...】

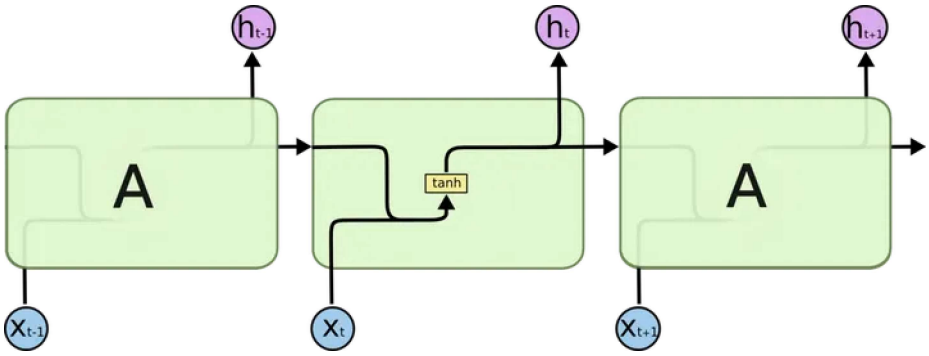
幸运的是，LSTM没有这个问题！

LSTM Networks

长期短期记忆网络通常被称为“LSTMs”，是一种特殊的RNN，能够学习长期依赖关系。它们由Hochreiter & Schmidhuber(1997)引入，并在随后的工作中被许多人提炼和推广。1．他们在各种各样的问题上都做得非常好，现在被广泛使用。

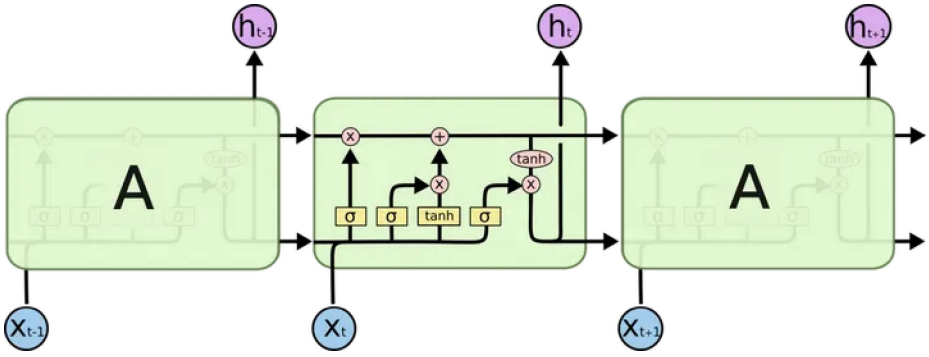
LSTMs的设计是为了避免长期依赖问题。长时间记住信息实际上是他们的默认行为，而不是他们努力学习的东西!

所有的递归神经网络都具有神经网络的重复模块链的形式。 在标准RNN中，此重复模块将具有非常简单的结构，例如单个tanh层。



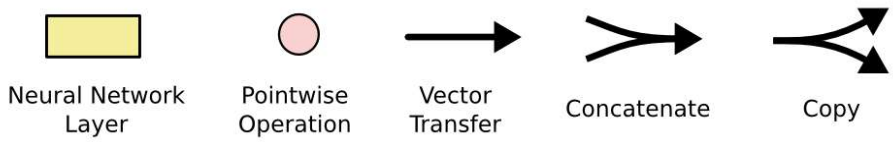
The repeating module in a standard RNN contains a single layer.

LSTM也具有这种链状结构，但是重复模块具有不同的结构。而不是只有一个神经网络层，而是有四个以非常特殊的方式进行交互的层。



The repeating module in an LSTM contains four interacting layers.

先不必关心发生的细节。 稍后，我们将逐步介绍LSTM图。 现在，让我们先尝试一下将要使用的符号。

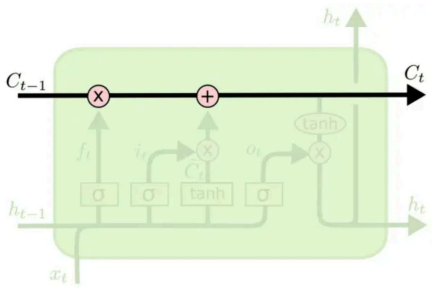


在上图中，每条线都承载着整个矢量，从一个节点的输出到另一节点的输入。 粉色圆圈表示逐点操作，例如矢量加法，而黄色框表示学习的神经网络层。 合并的行表示串联，而分叉的行表示要复制的内容，并且副本将到达不同的位置。

LSTM背后的核心思想 （The Core Idea Behind LSTMs）

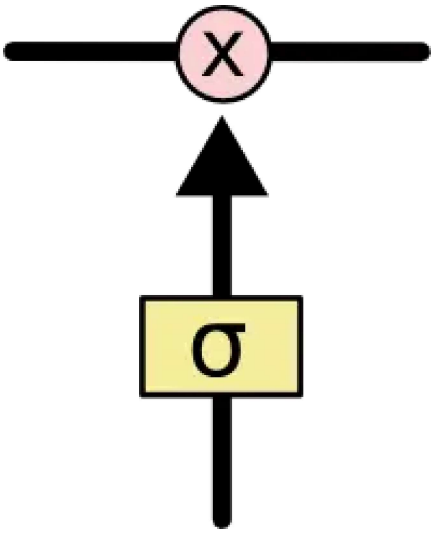
LSTM的关键是单元状态（cell state），水平线贯穿图的顶部。

单元状态有点像传送带。 它沿整个链条一直沿直线延伸，只有一些较小的线性相互作用。 信息不加改变地流动非常容易。



LSTM确实具有删除或向单元状态添加信息的能力，这些功能由称为门的结构（structures called gates）精心调节。

门是一种选择性地让信息通过的方式。 它们由一个 sigmoid neural net layer和一个 pointwise multiplication operation组成。



sigmoid lay

值为 0 表示“不让任何内容通过”，

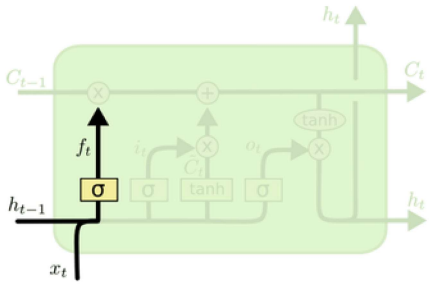
值为 1 表示“让所有内容通过！”

LSTM具有三个这样的门，以保护和控制单元状态。

LSTM分步指南（Step-by-Step LSTM Walk Through）

LSTM的第一步是决定要从单元状态中丢弃哪些信息。该决定由称为“遗忘门”（forget gate layer）的神经网络层sigmoid layer决定。它查看ht-1和xt，并为单元状态Ct-1中的每个数字输出介于0和1之间的数字。1代表“完全保留此条件”，0代表“完全保留此条件”。

让我们回到语言模型的示例，该模型试图根据所有先前的单词来预测下一个单词。在这样的问题中，细胞状态可能包括当前受试者的性别，从而可以使用正确的代词。当我们看到一个新主题时，我们想忘记旧主题的性别。

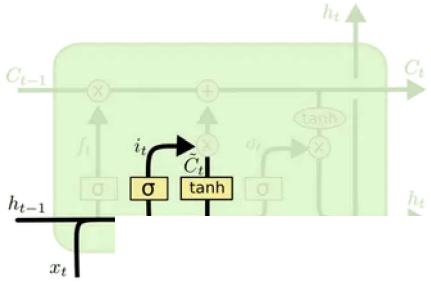


$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] \; + \; b_f\right)$$

下一步是确定要在单元状态下存储哪些新信息。 这包括两个部分。 首先，称为“input gate layer”的sigmoid layer 决定了我们将更新哪些值。 接下来，tanh层创建一个新候选值

的向量，可以将其添加到状态中。 在下一步中，我们将两者结合起来以创建该状态的更新。

在我们的语言模型示例中，我们希望将新主题的性别添加到单元格状态，以替换我们已经遗忘的旧主题。



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

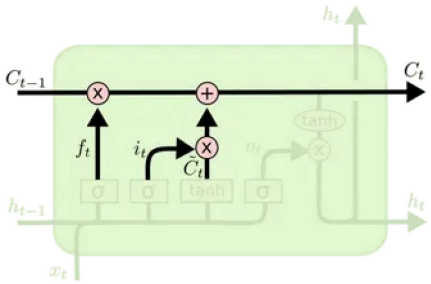
现在是时候将旧单元状态Ct-1更新为新单元状态Ct。 前面的步骤已经确定了要做什么，我们只需要实际进行即可。

我们将旧状态乘以ft，忘记了我们早先决定忘记的事情。 然后加

$$i_t * \tilde{C}_t$$

这是新的候选值，根据我们决定更新每个状态值的大小进行缩放。

对于语言模型，这是我们实际删除旧主题的性别的信息并添加新信息的地方，正如我们在前面的步骤中所确定的那样。

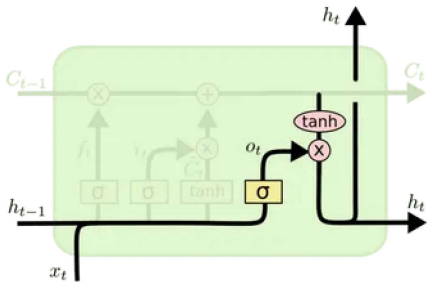


$$C_t = f_t * C_{t-1} + i_t * \tilde{C}_t$$

最后，我们需要决定要输出什么。 此输出将基于我们的单元状态，但将是过滤后的版本。 首先，我们运行一个 sigmoid layer ，确定要输出的单元状态的哪些部分。

然后，我们通过tanh放置单元状态（将值推到-1和1之间），然后将其乘以sigmoid gate的输出，以便仅输出确定的部分。

对于语言模型示例，由于它只是看到了一个主题，所以它可能希望输出与动词相关的信息，以防万一。例如，它可以输出主语是单数还是复数，这样我们就知道如果主语是单数或复数，那么动词应该变成什么形式。

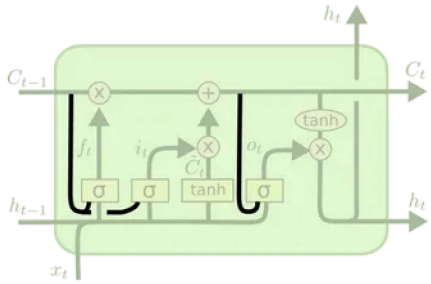


$$o_t = \sigma\left(W_o\left[h_{t-1}, x_t\right] + b_o\right)$$
$$h_t = o_t * \tanh\left(C_t\right)$$

长短时记忆的变异 （Variants on Long Short Term Memory）

到目前为止，我所描述的是一个非常普通的LSTM。 但是，并非所有的LSTM都与上述相同。 实际上，似乎几乎所有涉及LSTM的论文都使用略有不同的版本。差异不大，但其中一些值得一提

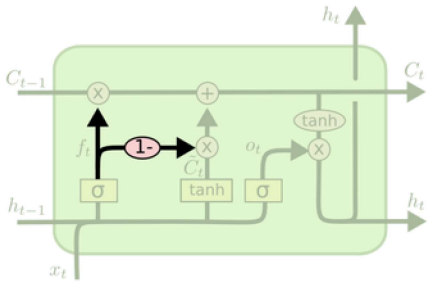
Gers＆Schmidhuber（2000）提出的一种流行的LSTM变体是添加“窥孔连接”。这意味着我们让栅极层查看单元状态。



$$f_t = \sigma\left(W_f \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_f\right)$$
$$i_t = \sigma\left(W_i \cdot [\boldsymbol{C_{t-1}}, h_{t-1}, x_t] \; + \; b_i\right)$$
$$o_t = \sigma\left(W_o \cdot [\boldsymbol{C_t}, h_{t-1}, x_t] \; + \; b_o\right)$$

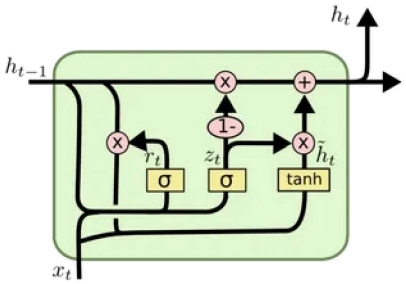
上图为所有门都添加了窥孔，但是许多论文都会给出一些窥孔，而没有其他。

另一个变化是使用耦合的忘记门和输入门。 我们可以一起做出这些决定，而不是分别决定要忘记什么以及应该向其中添加什么新信息。 我们只会在需要输入某些内容时才忘记它。 我们只会在忘记较旧的内容时才向状态输入新值。



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

LSTM的一个稍微显著的变化是由Cho等人(2014)引入的门控复发单元，或GRU。它将遗忘和输入门组合成一个“更新门”。“它还融合了细胞状态和隐藏状态，并做出一些其他的改变。得到的模型比标准LSTM模型更简单，并且越来越受欢迎。



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

这些只是最著名的LSTM变体中的少数。 还有许多其他方法，例如Depth Gated RNNs by Yao, et al. (2015).。arxiv.org/pdf/1508.0379...

还有一些完全不同的方法来处理长期依赖关系，比如 Koutnik, et al. (2014).。arxiv.org/pdf/1402.3511...

以下哪个变体最好？ 差异重要吗？Greff, et al. (2015)对流行的变体进行了很好的比较，发现它们几乎相同。

Jozefowicz, et al. (2015)测试了超过一万种RNN架构，发现其中某些在某些任务上比LSTM更好。arxiv.org/pdf/1402.3511...

结论Conclusion

之前，我提到了人们使用RNN所取得的非凡成就。基本上所有这些都是使用LSTM实现的。对于大多数任务，它们的确工作得更好！

写成一组方程式，LSTM看起来很吓人。希望本文逐步介绍它们，使他们变得更加平易近人。

LSTM是我们可以使用RNN完成的重要一步。很自然地想：还有另外一个重大步骤吗？研究人员普遍认为：“是的！想法是让RNN的每一步都从更大的信息集中挑选信息。例如，如果您使用RNN创建描述图像的标题，则它可能会选择图像的一部分以查看其输出的每个单词。实际上，徐等人。（2015）正是这样做的-如果您想引起关注，这可能是一个有趣的起点！注意力吸引了许多令人振奋的结果，而且似乎还有很多其他事情……

注意力机制不是RNN研究中唯一令人兴奋的话题。例如，Kalchbrenner, et al. (2015)）似乎非常有前途。在生成模型中使用RNN的工作-例如Gregor等。 （
"http://arxiv.org/pdf/1502.04623.pdf">Gregor, et al. (2015),
="arxiv.org/pdf/1506.0221...">Chung, et al. (2015), or Bayer & Osendorfer (2015) –似乎也很有趣。最近几年对于递归神经网络来说是一个令人振奋的时刻，而即将到来的几年更是如此！

致谢Acknowledgments

感谢许多人帮助我更好地了解LSTM，评论了可视化效果并提供了有关此帖子的反馈。

非常感谢Google同事的宝贵反馈，尤其是Oriol Vinyals，Greg Corrado，Jon Shlens，Luke Vilnis和Ilya Sutskever。 我也感谢其他许多朋友和同事花时间帮助我，包括Dario Amodei和Jacob Steinhardt。 我特别感谢Cho Kyunghyun Cho关于我的图表的极其周到的通信。

在写这篇文章之前，我在两个关于神经网络的研讨会系列中练习解释LSTM。 感谢参与其中的每个人对我的耐心和反馈。

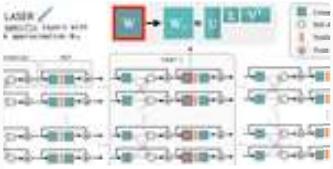
发布于 2019-09-26 19:56

深度学习（Deep Learning）

欢迎参与讨论

还没有评论，发表第一个评论吧

推荐阅读



麻省理工-2023：通过层选择性降级加速LLM大模型的推理...

SmartMindAI

ICLR'24 Agent论文合集：RL-based、LLM-based 前沿研...

lafmdp

论文分享：Rethinking Goal-Conditioned Supervised...

这是一篇ICLR 2022的工作。论文传送门： Rethinking Goal-Conditioned Supervised Learning and Its Connection to Offline RLGCSL方法使用自监督学习解决具有稀疏奖励的目标条件...

强化学习实... 发表于顶会论文分...

[ICLR 2024 (Spotligh里的Transformer还可

宣传一下最近的新工作，
是读博以来做得最难最累
感也最大的一个项目。它
个很简单的问题——自LL
来，我们见到了很多把LL
Vision Backbone后面的

庞子奇