

了解 LSTM 网络

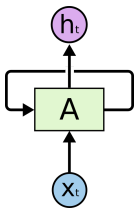
发表于 2015 年 8 月 27 日

循环神经网络

人类并不是每时每刻都从头开始思考。当你阅读这篇文章时，你会根据对前面单词的理解来理解每个单词。你不会扔掉所有东西并重新从头开始思考。你的思想有恒心。

传统的神经网络无法做到这一点，这似乎是一个主要缺点。例如，假设您想要对电影中每个点发生的事件类型进行分类。目前尚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。

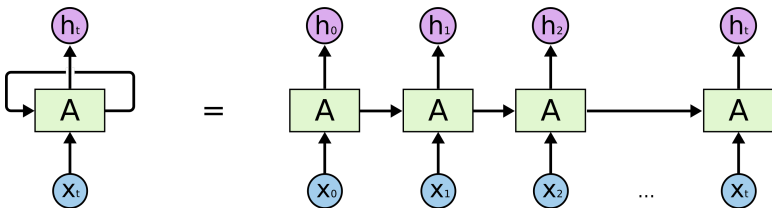
循环神经网络解决了这个问题。它们是带有循环的网络，允许信息持续存在。



循环神经网络有循环。

在上图中，一块神经网络，A，查看一些输入 x_t 并输出一个值 h_t 。循环允许信息从网络的一个步骤传递到下一步。

这些循环使循环神经网络显得有些神秘。然而，如果你多思考一下，就会发现它们与普通的神经网络并没有什么不同。循环神经网络可以被认为是同一网络的多个副本，每个副本将消息传递给后继网络。考虑一下如果我们展开循环会发生什么：



展开的循环神经网络。

这种链状性质揭示了循环神经网络与序列和列表密切相关。它们是用于此类数据的神经网络的自然架构。

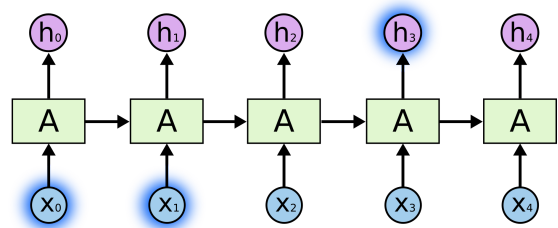
而且它们肯定被使用过！在过去的几年里，RNN 在解决各种问题上取得了令人难以置信的成功：语音识别、语言建模、翻译、图像字幕.....这样的例子不胜枚举。我将把关于 RNN 可以实现的惊人壮举的讨论留给 Andrej Karpathy 的优秀博客文章《循环神经网络的不合理有效性 (<http://karpathy.github.io/2015/05/21/rnn-effectiveness/>)》。但他们确实非常了不起。

这些成功的关键是使用“LSTM”，这是一种非常特殊的循环神经网络，对于许多任务来说，其效果比标准版本要好得多。几乎所有基于循环神经网络的令人兴奋的结果都是用它们实现的。本文将探讨的正是这些 LSTM。

长期依赖问题

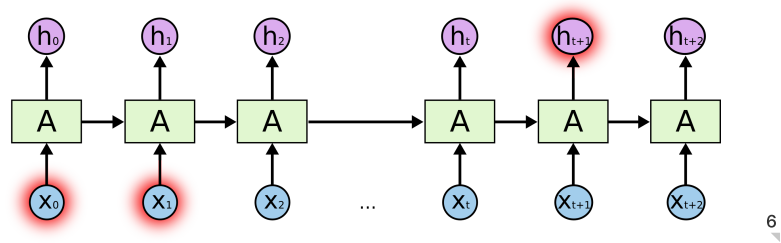
RNN 的吸引力之一是它们可能能够将先前的信息与当前的任务联系起来，例如使用先前的视频帧可能有助于理解当前的帧。如果 RNN 能够做到这一点，那么它们将非常有用。但他们可以吗？这取决于。

有时，我们只需要查看最近的信息来执行当前的任务。例如，考虑一个语言模型尝试根据之前的单词来预测下一个单词。如果我们试图预测“the clouds are in the *sky*”中的最后一个单词，我们不需要任何进一步的上下文——很明显下一个单词将是天空。在这种情况下，当相关信息与需要的信息之间的差距很小时，RNN 可以学习使用过去的信息。



但也有一些情况我们需要更多背景信息。考虑尝试预测文本中的最后一个单词“我在法国长大.....我说流利的法语”。最近的信息表明，下一个单词可能是一种语言的名称，但如果我们想缩小哪种语言的范围，我们需要更早的法语的上下文。相关信息与需要的信息点之间的差距完全有可能变得非常大。

不幸的是，随着差距的扩大，RNN 变得无法学习连接信息。



理论上，RNN 绝对有能力处理这种“长期依赖关系”。人类可以仔细地选择参数来解决这种形式的玩具问题。遗憾的是，在实践中，RNN 似乎无法学习它们。Hochreiter (1991) [德语] (<http://people.idsia.ch/~juergen/SeppHochreiter1991ThesisAdvisorSchmidhuber.pdf>) 和 Bengio 等人 (<http://www-dsi.ing.unifi.it/~paolo/ps/tnn-94-gradient.pdf>) 深入探讨了这个问题。(1994) (<http://www-dsi.ing.unifi.it/~paolo/ps/tnn-94-gradient.pdf>)，他发现了一些非常根本的原因来解释为什么这可能很困难。

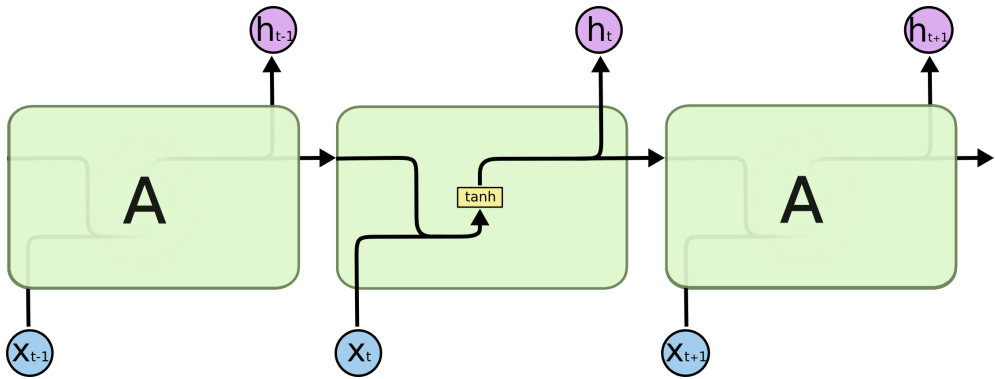
值得庆幸的是，LSTM 没有这个问题！

LSTM 网络

长短期记忆网络（通常简称为“LSTM”）是一种特殊的 RNN，能够学习长期依赖性。它们由Hochreiter & Schmidhuber (1997) (<http://www.bioinf.jku.at/publications/older/2604.pdf>)提出，并在后续工作中被许多人完善和推广。¹它们在解决各种各样的问题上都表现得非常好，并且现在被广泛使用。

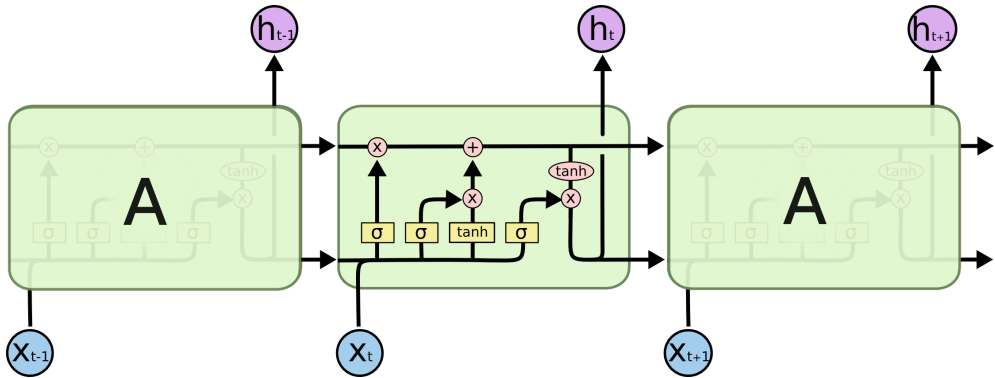
LSTM 的设计明确是为了避免长期依赖问题。长时间记住信息实际上是他们的默认行为，而不是他们努力学习的東西！

所有循环神经网络都具有神经网络重复模块链的形式。在标准 RNN 中，这个重复模块将具有非常简单的结构，例如单个 tanh 层。



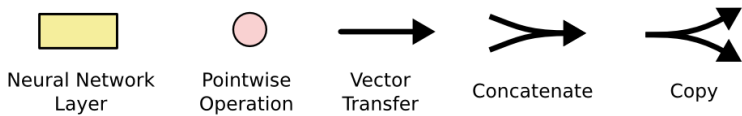
标准 RNN 中的重复模块包含单层。

LSTM 也具有这种链式结构，但重复模块具有不同的结构。神经网络层不是单一的，而是四个，以非常特殊的方式相互作用。



LSTM 中的重复模块包含四个相互作用的层。

不要担心正在发生的事情的细节。稍后我们将逐步介绍 LSTM 图。现在，让我们尝试熟悉我们将使用的符号。



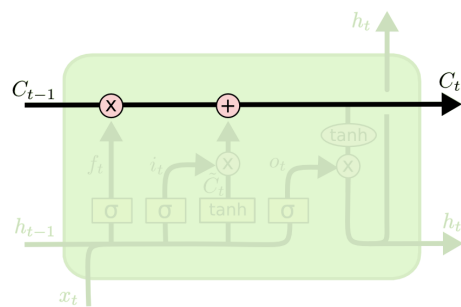
在上图中，每一行都承载一个完整的向量，从一个节点的输出到其他节点的输入。粉色圆圈代表逐点运算，例如向量加法，而黄色框是学习的神经网络层。行合并表示串联，而行分叉表示其内容被复制并且副本前往不同的位置。

LSTM 背后的核心思想

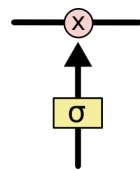
44C

LSTM 的关键是单元状态，即穿过图顶部的水平线。

细胞状态有点像传送带。它直接沿着整个链条运行，只有一些较小的线性相互作用。信息很容易不改变地流动。



LSTM 确实有能力删除或添加信息到细胞状态，并由称为门的结构仔细调节。
门是一种选择性地让信息通过的方式。它们由 sigmoid 神经网络层和逐点乘法运算组成。

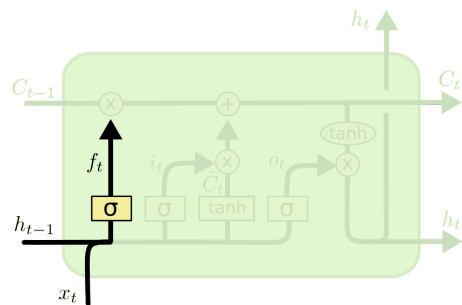


sigmoid 层输出 0 到 1 之间的数字，描述每个组件应该有多少通过。值为零意味着“不让任何东西通过”，而值1意味着“让所有东西通过！”

LSTM 具有三个这样的门，用于保护和控制单元状态。

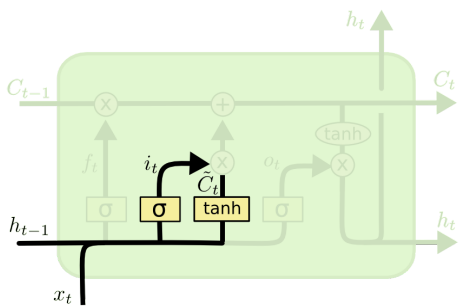
LSTM 逐步演练

LSTM 的第一步是决定我们要从细胞状态中丢弃哪些信息。这个决定是由称为“忘记门层”的 sigmoid 层做出的。它看着 h_{t-1} 和 x_t ，并输出一个介于0和1对于细胞状态中的每个数字 C_{t-1} 。A1代表“完全保留这个”，而0代表“彻底摆脱这个”。
让我们回到我们的语言模型示例，该模型尝试根据所有先前的单词来预测下一个单词。在这样的领域中，细胞状态可能包括当前主题的性别，以便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。



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

下一步是决定我们要在单元状态中存储哪些新信息。这有两个部分。首先，称为“输入门层”的 sigmoid 层决定我们要更新哪些值。接下来，tanh 层创建新候选值的向量， C_t ，可以添加到状态中。在下一步中，我们将结合这两者来创建状态更新。
在我们的语言模型的示例中，我们希望将新主题的性别添加到单元状态中，以替换我们忘记的旧主题。

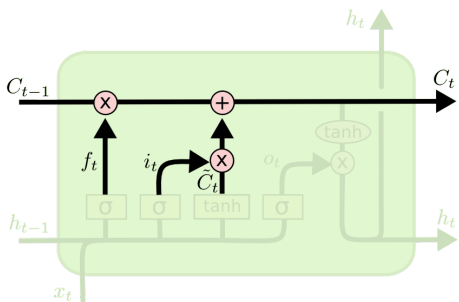


$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i)$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C)$$

现在是时候更新旧的单元格状态了， C_{t-1} ，进入新的细胞状态 C_t 。前面的步骤已经决定了要做什么，我们只需要实际去做就可以了。

我们将旧状态乘以 f_t ，忘记我们之前决定忘记的事情。然后我们添加我 $i_t \cdot \tilde{C}_t$ 。这是新的候选值，根据我们决定更新每个状态值的量进行缩放。

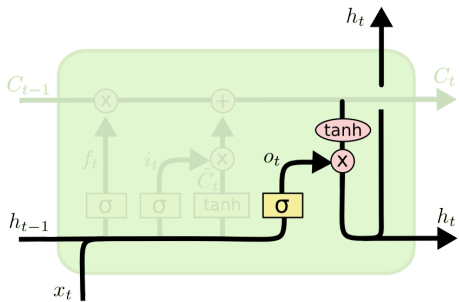
在语言模型的情况下，我们实际上会在此处删除有关旧主题性别的信息并添加新信息，正如我们在前面的步骤中决定的那样。



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

最后，我们需要决定要输出什么。该输出将基于我们的细胞状态，但将是过滤版本。首先，我们运行一个 sigmoid 层，它决定我们要输出细胞状态的哪些部分。然后，我们将细胞状态通过 tanh（将值推至介于-1和1）并将其乘以 sigmoid 门的输出，这样我们就只输出我们决定输出的部分。

对于语言模型示例，由于它刚刚看到一个主语，因此它可能想要输出与动词相关的信息，以防接下来出现这种情况。例如，它可能会输出主语是单数还是复数，以便我们知道接下来的动词应该变位成什么形式。

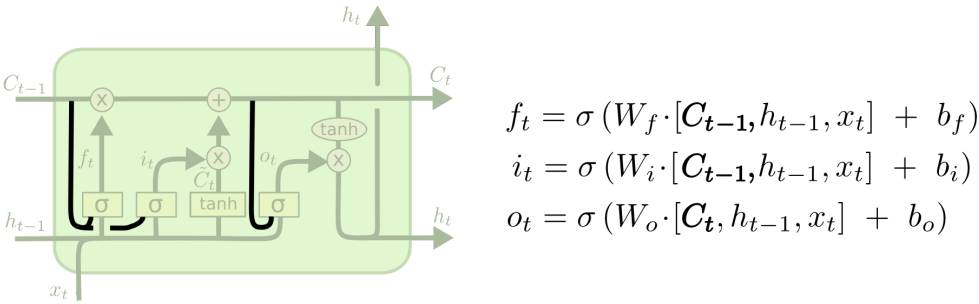


$$o_t = \sigma(W_o [h_{t-1}, x_t] + b_o)$$
$$h_t = o_t * \tanh(C_t)$$

长短期记忆的变体

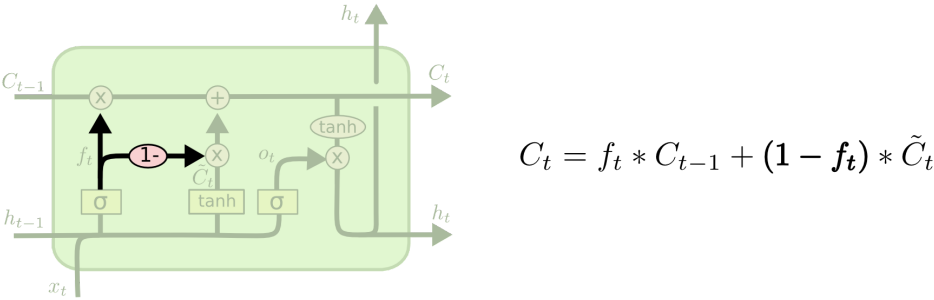
到目前为止我所描述的是一个非常普通的 LSTM。但并非所有的 LSTM 都与上述相同。事实上，似乎几乎每一篇涉及 LSTM 的论文都使用略有不同的版本。这些差异很小，但值得一提的是其中一些差异。

Gers 和 Schmidhuber (2000) (<ftp://ftp.idsia.ch/pub/juergen/TimeCount-LJCNN2000.pdf>)提出的一种流行的 LSTM 变体是添加“窥视孔连接”。这意味着我们让门层查看单元状态。

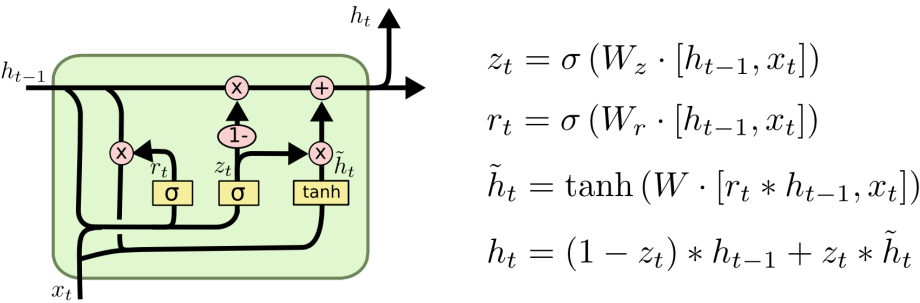


上图为所有门添加了窥视孔，但许多论文都会给出一些窥视孔，而不是其他的。

另一种变化是使用耦合的遗忘门和输入门。我们不是单独决定忘记什么以及应该添加新信息，而是一起做出这些决定。我们只会忘记何时要在其位置输入某些内容。只有当我们忘记旧的东西时，我们才会向状态输入新值。



LSTM 的一个稍微戏剧性的变化是由 Cho 等人提出的门控循环单元 (GRU)。(2014) (<http://arxiv.org/pdf/1406.1078v3.pdf>)。它将遗忘门和输入门组合成一个“更新门”。它还合并了单元状态和隐藏状态，并进行了一些其他更改。由此产生的模型比标准 LSTM 模型更简单，并且越来越受欢迎。



这些只是最著名的 LSTM 变体中的几个。还有很多其他的，例如 Yao 等人提出的深度门控 RNN。(2015) (<http://arxiv.org/pdf/1508.03790v2.pdf>)。还有一些完全不同的方法来解决长期依赖性，例如 Koutnik 等人的 Clockwork RNN。(2014) (<http://arxiv.org/pdf/1402.3511v1.pdf>)。

这些变体中哪一个最好？这些差异重要吗？格雷夫等人。(2015) (<http://arxiv.org/pdf/1503.04069.pdf>)对流行的变体进行了很好的比较，发现它们都是相同的。约泽福维奇等人。(2015) (<http://jmlr.org/proceedings/papers/v37/jozefowicz15.pdf>)测试了超过一万个 RNN 架构，发现一些在某些任务上比 LSTM 表现更好。

结论

之前，我提到过人们使用 RNN 取得了显著的成果。本质上所有这些都是在 LSTM 实现的。它们确实可以更好地完成大多数任务！

LSTM 写成一组方程，看起来相当吓人。希望本文中逐步介绍它们能让它们更容易理解。

LSTM 是我们使用 RNN 实现目标的一大进步。人们很自然地想知道：还有另一大步骤吗？研究人员的普遍观点是：“是的！下一步就是关注！”这个想法是让 RNN 的每一步都从更大的信息集中挑选信息来查看。例如，如果您使用 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其输出的每个单词。事实上，徐等人。(2015) (<http://arxiv.org/pdf/1502.03044v2.pdf>)就这样做——如果你想探索注意力，这可能是一个有趣的起点！使用注意力已经取得了许多令人兴奋的结果，而且似乎还有更多的结果即将到来.....

注意力并不是 RNN 研究中唯一令人兴奋的主题。例如，Kalchbrenner 等人的网络 LSTM。（2015）(<http://arxiv.org/pdf/1507.01526v1.pdf>)看起来非常有前途。在生成模型中使用 RNN 进行工作——例如Gregor等人。（2015）(<http://arxiv.org/pdf/1502.04623.pdf>)，钟，等人。(2015) (<http://arxiv.org/pdf/1506.02216v3.pdf>)或Bayer & Osendorfer (2015) (<http://arxiv.org/pdf/1411.7610v3.pdf>) – 似乎也很有趣。过去几年对于循环神经网络来说是一个激动人心的时期，而未来的几年只会更加如此！

致谢

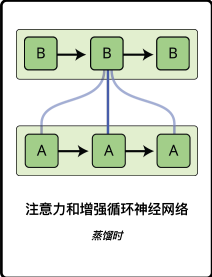
我感谢许多人帮助我更好地理解 LSTM、对可视化进行评论以及对本文提供反馈。

我非常感谢 Google 的同事提供的有用反馈，尤其是 Oriol Vinyals (<http://research.google.com/pubs/OriolVinyals.html>)、Greg Corrado (<http://research.google.com/pubs/GregCorrado.html>)、Jon Shlens (<http://research.google.com/pubs/JonathonShlens.html>)、Luke Vilnis (<http://people.cs.umass.edu/~luke/>) 和 Ilya Sutskever (<http://www.cs.toronto.edu/~ilya/>)。我还感谢许多其他朋友和同事花时间帮助我，包括达里奥·阿莫代 (Dario Amodei) (<https://www.linkedin.com/pub/dario-amodei/4/493/393>) 和雅各布·斯坦哈特 (Jacob Steinhardt) (<http://cs.stanford.edu/~jsteinhardt/>)。我特别感谢Kyunghyun Cho (<http://www.kyunghyuncho.me/>)对我的图表进行了极其深思熟虑的通信。

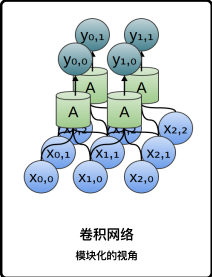
在写这篇文章之前，我在教授神经网络的两个研讨会系列中练习解释 LSTM。感谢所有参与这些活动的人对我的耐心和反馈。

1. 除了原作者之外，还有很多人现代 LSTM 做出了贡献。不完整的名单包括：Felix Gers、Fred Cummins、Santiago Fernandez、Justin Bayer、Daan Wierstra、Julian Togelius、Faustino Gomez、Matteo Gagliolo 和Alex Graves (<https://scholar.google.com/citations?user=DaFHynwAAAAJ&hl=en>)。↩

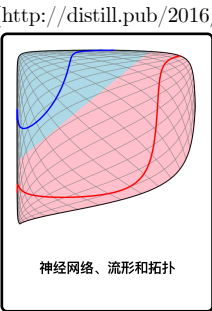
更多帖子



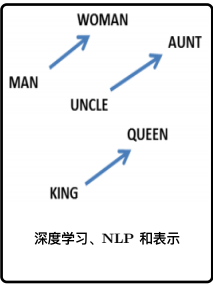
注意力和增强循环神经网络
蒸馏时



卷积网络
模块化的视角



神经网络、流形和拓扑



深度学习、NLP 和表示

(<http://distill.pub/2016/augmented-rnns/>) ([../posts/2014-07-Conv-Nets-Modular/](http://colah.github.io/posts/2014-07-Conv-Nets-Modular/)) ([../posts/2014-03-NN-Manifolds-Topology/](http://colah.github.io/posts/2014-03-NN-Manifolds-Topology/)) ([../posts/2014-07-NLP-RNNs-Representations/](http://colah.github.io/posts/2014-07-NLP-RNNs-Representations/))

75 条评论 ([/posts/2015-08-Understanding-LSTMs/#disqus_thread](http://colah.github.io/posts/2015-08-Understanding-LSTMs/#disqus_thread))