Report of Deep Learning for Natural Langauge Processing

刘新宇
LXYbhu@buaa.edu.cn

Abstract

Seq2seq（Sequence-to-Sequence）和LSTM（Long Short-Term Memory）都是深度学习中常用的模型。Seq2seq是一种用于处理序列到序列（sequence-to-sequence）问题的模型架构，LSTM是一种特殊类型的循环神经网络（Recurrent Neural Network，RNN）。传统的RNN在处理长序列时容易出现梯度消失或梯度爆炸的问题，而LSTM通过引入门控机制来解决这个问题。通常Seq2seq模型中的编码器和解码器可以使用LSTM作为基本的循环神经网络结构，以便更好地捕捉序列数据中的上下文和语义信息。本次作用使用LSTM来实现文本生成的模型，输入可以为一段已知的金庸小说段落，来生成新的段落并做分析。

Methodology

LSTM是一种特殊的递归神经网络 。这种网络与一般的前馈神经网络不同，LSTM可以利用时间序列对输入进行分析；简而言之，当使用前馈神经网络时，神经网络会认为我们t时刻输入的内容与t+1时刻输入的内容完全无关，对于许多情况，例如图片分类识别，这是毫无问题的，可是对于一些情景，例如自然语言处理或者我们需要分析类似于连拍照片这样的数据时，合理运用t或之前的输入来处理t+n时刻显然可以更加合理的运用输入的信息。为了运用到时间维度上信息，人们设计了递归神经网络，一个简单的递归神经网络可以用图1这种方式表示，在图中，x_t是在t时刻的输入信息，h_t是在t时刻的输入信息，我们可以看大到神经元A会递归的调用自身并且将t−1时刻的信息传递给t时刻。所示的简单

递归神经网络存在一个“硬伤“，长期依赖问题：递归神经网络只能处理我们需要较接近的上下文的情况。

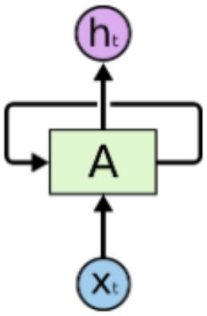

图1 简单的递归神经网络

LSTM是SimpleRNN的变体，它解决了梯度消失的问题。LSTM增加了一个可以相隔多个timesteps来传递信息的方法。想想有一个传送带在你处理sequences时一起运转。每个时间节点的信息都可以放到传送带上，或者从传送带上拿下来，当然你也可以更新传送带上的信息。这样就保存了很久之前的信息，防止了信息的丢失。

LSTM的关键是单元状态，LSTM能够从RNN中脱颖而出的关键就在于上图中从单元中贯穿而过的线 ——神经元的隐藏态（单元状态），我们可以将神经元的隐藏态简单的理解成递归神经网络对于输入数据的“记忆”，用 C_t 表示神经元在t时刻过后的“记忆”，这个向量涵盖了在t+1时刻前神经网络对于所有输入信息的“概括总结”。

LSTM遗忘门：对于上一时刻LSTM中的单元状态来说，一些“信息”可能会随着时间的流逝而“过时”。为了不让过多记忆影响神经网络对现在输入的处理，我们应该选择性遗忘一些在之前单元状态中的分量——这个工作就交给了“遗忘门”。下面的公式可以用来描述遗忘门的计算，其中f_t就是sigmoid神经层的输出向量：

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

LSTM记忆门：记忆门是用来控制是否将在t时刻（现在）的数据并入单元状态中的控制单位。首先，用tanh函数层将现在的向量中的有效信息提取出来，然后使用sigmoid函数来控制这些记忆要放“多少”进入单元状态。

LSTM输出门:就是LSTM单元用于计算当前时刻的输出值的神经层。输出层会先将当前输入值与上一时刻输出值整合后的向量（也就是公式中的$[h_{t-1}, x_t]$）用sigmoid函数提取其中的信息，接着，会将当前的单元状态通过tanh函数压缩映射到区间(-1, 1)中。

Experimental Studies

在具体的实验部分，我们使用 Keras 来实现字符级的 LSTM 文本生成

在数据准备阶段，我们需要大量的文本数据(语料，corpus)来训练语言模型。我们可以去找足够大的一个或多个文本文件：维基百科、各种书籍等都可。这里我们选择用一些金庸的作品，这样我们学习出来的语言模型将是有金庸的写作风格和主题的。

接下来，我们要把文本做成数据 (向量化)：从 text 里提取长度为 maxlen 的序列(序列之间存在部分重叠)，进行 one-hot 编码，然后打包成 (sequences, maxlen, unique_characters) 形状的。同时，还需要准备数组 y，包含对应的目标，即在每一个所提取的序列之后出现的字符(也是 one-hot 编码的)。

在构建网络时，我们使用一个 LSTM 层 + 一个 softmax 激活的 Dense 层（其实并不一定要用 LSTM，用一维卷积层也是可以生成序列）。

然后训练语言模型并从中采样。给定一个语言模型和一个种子文本片段，就可以通过重复以下操作来生成新的文本：

1. 给定目前已有文本，从模型中得到下一个字符的概率分布；
2. 根据某个温度对分布进行重新加权；
3. 根据重新加权后的分布对下一个字符进行随机采样；
4. 将新字符添加到文本末尾。

最后，再来训练并生成文本。我们在每轮完成后都使用一系列不同的温度值来生成文本，这样就可以看到，随着模型收敛，生成的文本如何变化，以及温度对采样策略的影响。

实验结果分析：

我用一些金庸的文章去训练，得到的结果大概是这样的：

几个短衣人物也和他同坐在一处这马车立刻走动了，贝壳天气还早，赊了两碗酒没有吃过人的孩子，米要钱买这一点粗浅事情都不知道……天气还早不妨事么他……全屋子都很静这时红鼻子老拱的小曲，而且又破费了二十千的赏钱，他一定须在夜里的十二点钟才回家。

我们发现，利用更多的数据训练更大的模型，训练时间更长，生成的样本会更连贯、更真实。但是，这样生成的文本并没有任何意义。机器所做的仅仅是从统计模型中对数据进行采样，它并没有理解人类的语言，也不知道自己在说什么。

Conclusion

本次实验通过构建LSTM网络，生成了一些金庸风格的段落。但其计算量较大，需要较长的时间来学习，训练和推理速度相对较慢。

LSTM 的文本生成效果取决于数据质量、数据规模、序列长度、网络结构和超参数调节，需要根据具体任务和数据来进行调节和优化。

通过这次作业，加深了我对自然语言处理用途的理解，但本次任务仍存在不足之处，没有进行一些对比实验，会在以后的学习工作中补足。