**RNNs and LSTMs**

**摘要：**

在自然语言处理（nlp）领域的发展中，为更好建模和体现语言的时间特性，循环神经网络（RNNs）和长短期记忆网络（LSTMs）等模型扮演着重要角色。本文综述了RNNs和LSTMs的发展历程、应用以及相关的关键概念。RNNs具有直接处理语言序列性质的机制，无需使用固定大小的窗口，其循环连接的表示方式允许模型的决策依赖于过去数百个单词的信息。LSTMs作为RNNs的变体，通过引入门控机制和记忆细胞解决了梯度消失等问题，能够更好地捕捉长序列之间的依赖关系。RNNs在nlp任务中的广泛应用（包括语言建模、序列标注、序列分类和文本生成等），其基础架构和注意力机制等关键技术的引入都是我们关注的问题。

1. 循环神经网络（RNNs）：

循环神经网络是一种经典的序列模型，能够处理不定长度的输入序列，并具有一定的记忆能力。它与前馈网络的主要区别在于循环连接，即某个单元的值直接或间接地依赖于其自身早期输出作为输入（当前输入的输入向量xt乘以一个权重矩阵，然后通过非线性激活函数计算一层隐藏单元的值，使用这个隐藏层来计算相应的输出yt）。这样的网络比较难进行推理和训练，但在应用于语言时非常有效。

在RNN中，各种权重矩阵在不同时刻共享，通过一个训练集、一个损失函数和反向传播来调整权重。传统方法训练时，采用两遍计算：在第一遍中进行前向推理，计算隐藏层和输出，并在每个时间步中累积损失，保存每个时间步的隐藏层值以在下一个时间步使用。第二遍时逆向处理序列，依旧计算每步的损失，通过时间进行反向传播。现在通常将RNN明确展开为前馈计算图以此消除“循环”，允许直接训练网络权重。只是对于长输入序列，需要先将输入展开为固定长度段，再将每个段视为一个独立的训练项输入网络训练。

1. RNNs在nlp任务中的应用：

RNNs在nlp领域有广泛的应用，包括但不限于：

（1）语言建模：对文本序列进行概率建模，预测下一个单词或标记。

与N-gram语言模型相比，RNNs对上下文没有固定长度的限制，前一个隐藏层中包含的上下文信息可以延伸到序列的开始，符合语言的时间性质。

在实际应用中，输入序列包含一系列单词，每个单词都表示为one-hot向量，输出y表示预测的整个词汇表上的概率分布。在每个步骤中，模型使用词嵌入矩阵来检索当前单词的嵌入，并将其与前一步的隐藏层组合起来计算一个新的隐藏层，进而生成一个输出层，最后通过softmax层生成整个词汇表上的概率分布。

训练时，采用自监督的方法：无需人为添加标签，只需将一个文本语料库作为训练材料，并在每个时间步t给模型正确的历史序列来预测下一个单词，最小化在训练序列中预测下一个单词时的误差（交叉熵损失），即模型分配给训练序列中下一个单词的负对数概率。同时，可以采取权重绑定策略来将词汇表的输入和输出嵌入关联起来，以此消除冗余、减小参数计算量。

（2）序列标注：

输入词嵌入，输出softmax层生成的给定标签集上的标签概率，即对序列中的每个元素进行标记，如词性标注、命名实体识别等。

（3）序列分类：

将整个文本序列分配到特定类别，如垃圾邮件检测、情感分析等。

一般来说，我们将待分类文本中的单词逐个通过RNN，每个时间步生成一个新的隐藏层，不需要计算中间输出的损失项，取文本的最后一个标记的隐藏层（或取它们逐元素的平均值或最大值）视为整个序列的压缩表示，通过可能类别上的softmax来选择一个类别。

（4）文本生成：

与N-gram语言模型类似，使用语言模型根据先前的选择反复抽样下一个单词逐步生成，即自回归生成（因果语言模型生成）：从句子起始标记开始，使用第一个单词的单词嵌入作为下一个时间步的网络输入，然后以同样的方式抽样下一个单词继续生成，直到抽样到句子结束标记或达到给定的生成长度限制。

1. RNNs变体——长短时记忆网络LSTMs

在传统的RNNs中，隐藏层及其确定隐藏层中值的权重被要求同时执行两个任务：为当前决策提供有用的信息，以及更新并传递信息以供将来的决策使用。同时在反向传播时可能出现梯度趋于0并累积越来越小的“消失”和梯度爆炸等问题，导致在处理长序列时性能下降。

为解决这些问题，使网络学到更多的远程信息，引入LSTMs三门机制：输入门（决定哪些来自当前输入的信息应该存储在细胞状态中）、输出门（控制应该使用哪些细胞状态信息来计算输出和新的隐藏状态）、遗忘门（确定应该丢弃或遗忘前一个细胞状态中的哪些信息）。通过门控单元和记忆细胞，LSTMs能够更好地捕捉长序列之间的依赖关系，因此在语言建模、机器翻译、文本生成等任务中表现出色。

虽然结构更加复杂，但LSTMs具有封装性，与简单的RNNs一样，可以展开成深度前馈网络，并以常规方式使用反向传播进行训练。

1. RNNs常见网络架构与关键技术：
2. 堆叠RNNs

由多个网络组成，其中一个层的输出作为后续层的输入，网络在不同层次上诱导了不同抽象级别的表示。一般来说，堆叠数量越多，训练成本越高，而最佳的堆叠RNN数量因应用任务和训练集不同而不同。

1. 双向RNNs

运行两个单独的循环神经网络，一个从左到右，一个从右到左，将网络计算的两个表示连接成一个单一的向量，该向量在每个时间点捕获输入的左右上下文，这样的网络既能很好地反映前文又能联系后文，在序列分类等任务上能方便地解决上下文关联等整体特征问题。

（3）编码器-解码器模型与注意力机制的引入

编码器-解码器将输入序列翻译成与输入不同长度的输出序列，适用于机器翻译等任务。在该模型中，编码器负责将源文本编码成上下文向量，而解码器则利用这个上下文向量生成目标文本。

然而，原始的编码器-解码器模型存在瓶颈问题，即只能利用编码器的最后一个隐藏状态作为上下文，这导致了信息丢失。为了解决上下文信息瓶颈的问题，引入注意力机制，允许解码器动态地从编码器的所有隐藏状态中获取信息，使用加权和归一化来保留重要信息，而不仅仅是最后一个隐藏状态。此外，还有一些更复杂的注意力机制，如双线性注意力、多头注意力等。这一机制大大提高了模型对输入序列的理解能力，并在机器翻译等任务中取得了重大突破。

**结论：**

RNNs及其变体LSTMs以及编码器-解码器模型是自然语言处理中的重要工具，它们在语言建模、机器翻译等任务中发挥着关键作用，并且随着注意力机制的引入，模型的性能和应用范围已经取得进一步的提升，但仍然面临着一些挑战，如梯度消失、训练速度慢等问题。未来的研究方向可以集中在进一步改进模型的性能和效率、探索更有效的训练方法和结构设计上。同时，深入研究注意力机制的原理和应用，以及与其他技术的结合，将有助于进一步提升深度学习模型在nlp领域的表现，推动该领域的进步和发展。

**问题：**

1. 数据集一定的情况下，LSTM用于文本生成任务的时候，在生成文本多样性方面是否会弱于他的变体GRU？
2. 多层LSTM堆叠是否更难训练？是否会导致梯度爆炸等问题？
3. RNN解码相较transformer快主要是因为多层注意力的原因吗？

**参考文献：**

1. Dan Jurafsky, James H. Martin. Speech and Language Processing (3rd ed. draft)
2. Kostiantyn Onyshchenko,Yana Daniiel.USING LONG SHORT-TERM MEMORY NETWORKS FOR NATURAL LANGUAGE PROCESSING[J].ВІСНИК НАЦІОНАЛЬНОГО ТЕХНІЧНОГО УНІВЕРСИТЕТУ "ХПÌ": СИСТЕМНИЙ АНАЛІЗ,2023(1):89-96.
3. Ashish Vaswani,Noam Shazeer,Niki Parmar,et al.Attention Is All You Need[Z].arxiv,2017.
4. Zhe Chen.Attention Is Not All You Need Anymore[Z].arxiv,2023.