**学习进度**

本周学习任务：

学习吴恩达深度学习课程v5.72，关于循环序列模型的内容。

1. GRU

传统RNN模型不擅长捕获长期依赖效应，**门控循环单元（GRU）**通过引入门控组件的方式，保留了部分记忆并传递到下一层，有效缓解了长期记忆遗失问题。如图1.1所示，GRU主要由更新门及重置门组成，用于控制本层候选记忆传递到未来的程度，用于控制前层记忆与本层输入的结合程度。其中，重置门的计算公式为：，更新门的计算公式为：。

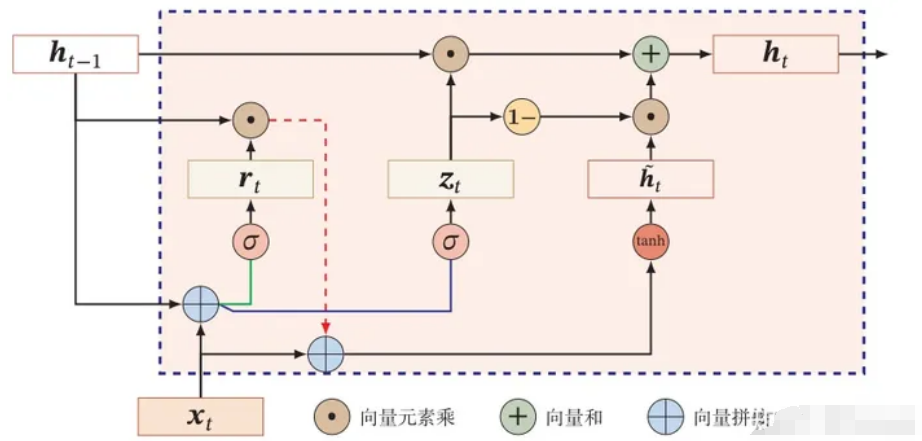


图1.1 GRU结构

工作流程：计算出某时间步的重置门、更新门后，需要更新出本层候选记忆，计算公式为：。然后，即可获得本层最终输出的记忆，计算公式为：。

1. LSTM

**长短期记忆（LSTM）**网络，同样通过在隐藏层引入更多门控组件的方式，动态筛选出相对重要的长期记忆，从而更有效地改善了长期记忆遗忘问题。如图2.1所示，LSTM主要通过遗忘门、输入门和输出门，控制t时间步的细胞状态。其中，遗忘门接收本层输入及前层隐藏状态，用于控制上层记忆的遗忘程度，计算公式为：。输入门用于控制本层输入的记忆程度，计算公式为：。输出门用于控制哪些记忆应作为隐藏状态传递到下一时间步中，计算公式为：。

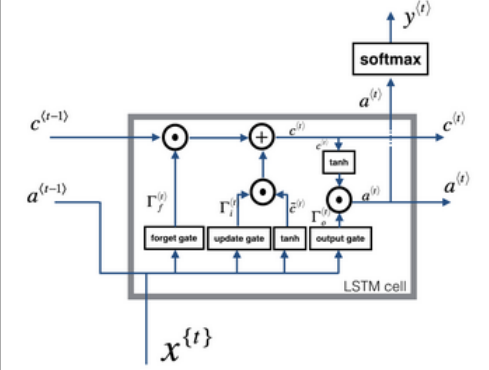


图2.1 LSTM结构

工作流程：计算出某时间步的遗忘门、输入门、输出门后，需要更新出本层候选记忆，计算公式为：。然后，即可获得本层记忆，计算公式为：。最后，计算出本层的隐藏状态，计算公式为：。

在循环序列模型架构中，可以使用LSTM单元替换传统的RNN结点，且替换前后的正向传播过程一致。然而，LSTM网络的反向传播过程相对复杂，这是由于LSTM单元中涉及更多的门控组件及其参数矩阵造成的。LSTM反向传播计算，如图2.2所示。

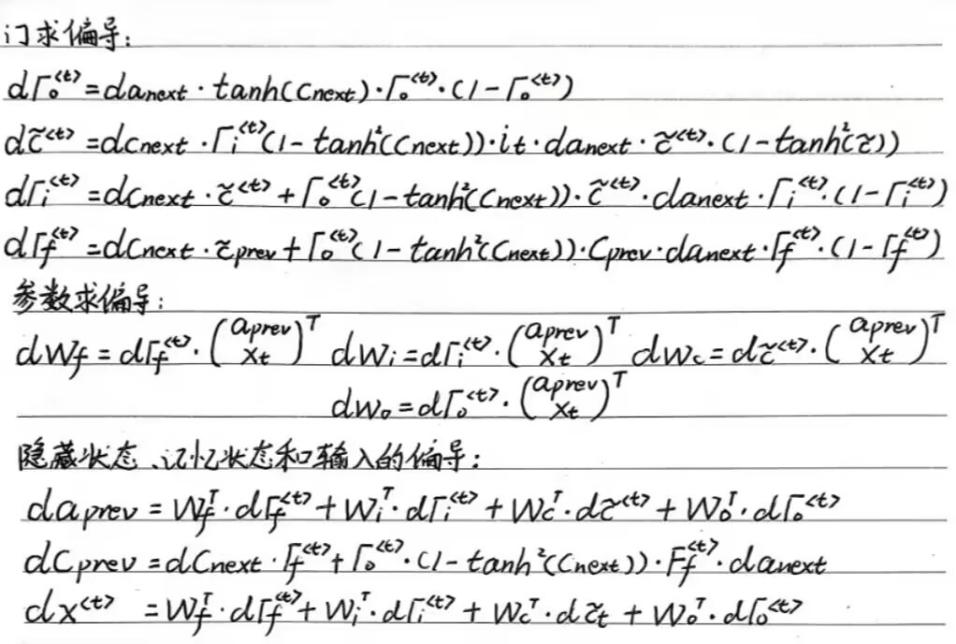


图2.2 LSTM反向传播计算

值得注意的是，GRU作为LSTM的简化变体，通过更新门及重置门，一定程度上缓解了长期记忆遗忘问题。相比于LSTM，GRU由于组件参数更少，计算速度更快，可以扩大模型的规模。而LSTM由于存在三个门控组件，模型更加强大灵活，能更好地缓解长期记忆遗忘问题，是大部分情况下的默认选择。

1. Bidirectional RNN

在传统RNN模型架构中，每个时间步只能考虑到前文信息。而**双向循环神经网络（Bidirectional RNN）**通过引入逆向激活值的方式，使得每个时间步都能同时考虑到上下文信息。如图3.1所示，Bidirectional RNN首先需要计算出所有的正向激活值（图中紫色部分），然后反向计算出所有逆向激活值（图中绿色部分），最后同时依据正、逆向激活值分时间步计算出对应的预测值。显而易见的是，Bidirectional RNN也可以使用RNN、GRU或LSTM等基本单元来构建模型。

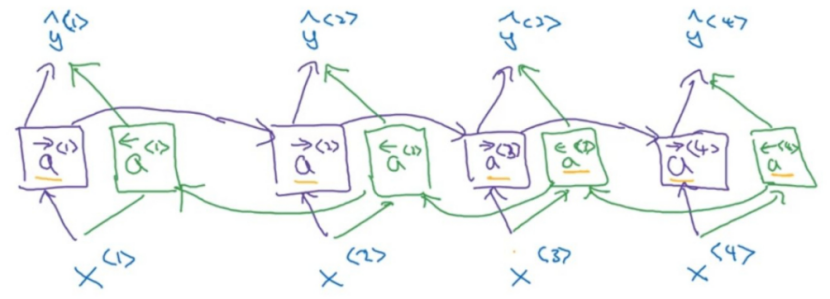


图3.1 Bidirectional RNN网络架构

值得注意的是，Bi-RNN需要完整的输入序列，才能展开预测工作。例如，基于Bi-RNN搭建语音识别系统，需要收录到完整的语音表达后，才能开始处理这段语音。

1. Deep RNNs

为了学习更复杂的函数，**深层循环神经网络（Deep RNNs）**通过堆叠多层RNN的方式，构建出了一个更深的模型架构。如图4.1所示，第三个时间步中第二层的激活值，是根据同时间步中的浅层激活值及同层的前一时间步激活值计算而来。

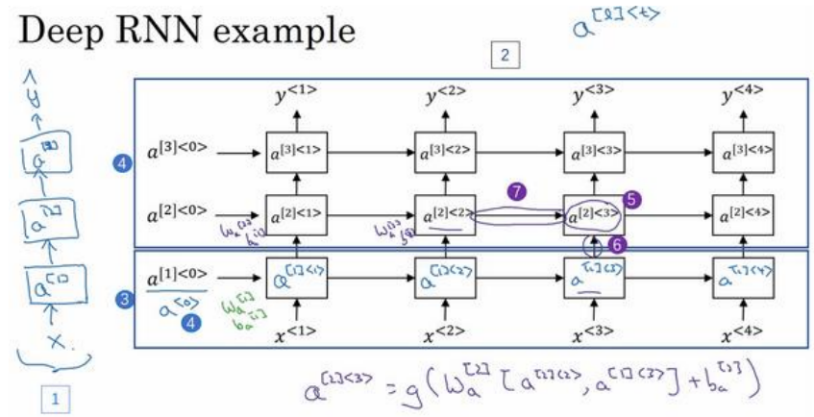


图4.1 Deep RNNs模型架构

值得注意的是，考虑到模型的复杂程度，一般很少堆叠太多有水平连接的RNN层。如图4.2所示，更常见的做法是在每个时间步上延深无水平连接的纵向单向网络。

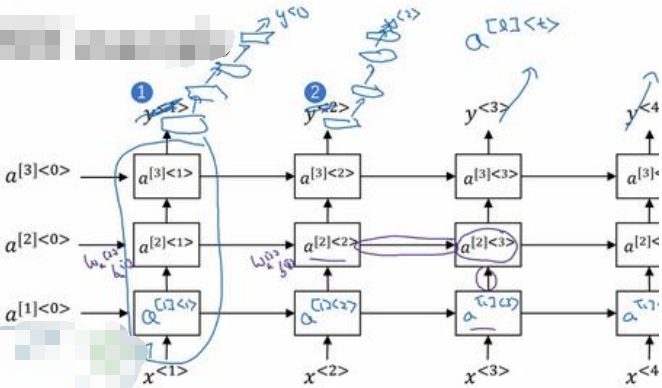


图4.2 纵向延深的Deep RNNs模型架构

1. 词嵌入（Word Embedding）

在自然语言处理领域，需要考虑如何将文本表示为计算机能够处理的数值型数据。传统文本表示方法有：**独热编码**、**词袋模型**、**TF-IDF**及**N-gram**方法。若用独热编码、词袋模型及TF-IDF模型表示句子，则词向量维度等于文档集中所有非重词汇的数量。若用N-gram方法表示句子，则词向量维度等于n-gram对象的数量，向量的值表示对应索引位置上n-gram对象在句子中出现的频次。显而易见的是，基于词频信息的传统文本表示方法，大多存在数据稀疏性、维度灾难、忽略语义及词序信息、定长文本表示及无法消除歧义等问题。

**词嵌入（Word Embedding）**方法通过将字词映射到低维连续向量空间的方式，更好地捕捉了字词间的语义相似性，学习了海量文本的语义规律。在连续向量空间中，每个维度都对应了特定的抽象语义，并且语义相近的字词在向量空间中通常彼此靠近。

问题是：如何使用词嵌入方法完成文本表示工作？常见做法是：首先选择适宜的词嵌入模型，例如Word2Vec、GloVe、FastText及BERT等；接着训练或直接加载预训练过的词嵌入模型；然后输入经过预处理的句子，从而获得对应的词嵌入向量；最后将表示对应字词的词嵌入向量组合起来即可。

值得注意的是，训练出性能优秀的词嵌入模型，往往需要投入海量符合任务情景的文本。更常见的做法是，直接使用或微调解决类似情景任务的预训练词嵌入模型即可。此外，来自网络的评论文本往往存在大量主观偏见，如果需要自行训练词嵌入模型，还需要额外考虑词嵌入除偏问题。

1. 集束搜索（Beam Search）

以机器翻译任务为例，翻译模型能够生成许多预测结果，关键在于如何找到相对最好的结果？常见的想法是，使用**贪心搜索（Greedy Search）**寻求相对最优翻译结果，即：每生成一个词的分布后，就挑选出最可能的词，然后继续生成下一个词的分布并挑选。然而，这种依次挑选出最优词的方式，更容易挑选出常用词组，这并不一定是最好的翻译结果。

如图6.1所示，**集束搜索（Beam Search）**通过seq2seq模型，在分时间步预测结果序列的过程中，每次都保留概率最大的B个结果，然后在下一时间步中将保留的B个结果分别和词典中的所有非重字词组合，同样只保留概率最大的B个拼接而成的新字词组合。

graphic

图6.1 集束搜索的工作流程

由此可见，集束搜索本质上是一个求最大化概率的过程。然而，多个区间在0-1之间的概率值累乘，小数部分的位数逐渐增多，将造成数值下溢现象。为此，集束搜索一般记录概率的对数和。此外，由于目标函数更倾向于短序列（概率越乘越小），集束搜索可通过除以结果序列长度的方式，削减长序列结果概率值遭受到的惩罚。至此，集束搜索的目标函数为：。

问题是：如果通过集束搜索，真的获得了多个同样好的序列结果，又该如何评估机器翻译系统？针对此类情况，需要额外引入：**Bleu得分（Bilingual Evaluation Understudy Score）**。

值得注意的是，集束搜索通过设置集束宽参数B，控制每次只保留几个概率最高的拼接词组。对于B=1的集束搜索而言，实际上等同于贪心搜索。

1. 注意力机制

为了更灵活地处理变长输入序列、捕捉长距离依赖关系及改善梯度传播，**注意力机制（Attention Mechanism）**通过对序列不同部分赋予不同注意力权重的方式，动态地选择性关注相对重要的内容。注意力机制的主要类型有：自注意力、多头注意力、通道注意力及空间注意力。

其中，**自注意力机制**在处理序列数据时，每个时间步的输入值都与原始输入序列的所有分量相关联。具体而言，通过计算所在时间步对应序列分量与其它序列分量的相似度，获得当前时间步下不同序列分量的注意力权重，然后将每个序列分量与对应的注意力权重进行加权求和，从而获得当前时间步的输入值。

以计算第一个时间步的输入分量为例，自注意力机制的实现方式是：

1. 将原始输入序列分量进行Embedding处理，从而获得词嵌入向量，计算公式为：；
2. 依次计算所有序列分量与的相似度，公式为：、、、，其中q用来和其它序列分量进行匹配、k作为其它序列分量的关键信息和q匹配、v表示对应序列分量的特征信息、d等于q和k的矩阵维度；
3. 通过softmax操作，将所有序列分量与的相似度，归一化为注意力权重，公式为：；
4. 如图7.1所示，若原始输入序列的长度为3，则。

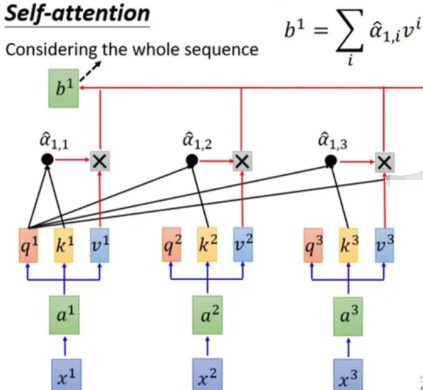


图7.1 自注意力机制的计算过程

**多头注意力机制**在自注意力的基础上，引入了多个独立自注意力头，通过分别计算独立自注意力头的注意力权重，并将所有注意力权重拼接或加权求和的方式，获得了更丰富的表达。如图7.2所示，在多头自注意力机制中，每个输入序列分量都可与多个q、k、v对应。然而，多头注意力机制的计算方式，本质上同自注意力机制一致。

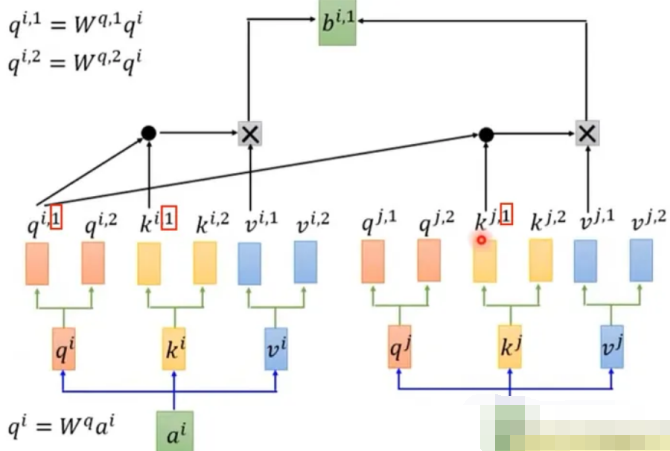


图7.2 多头注意力机制的计算过程

如图7.3所示，CBAM模型同时引入了通道注意力机制及空间注意力机制模块，前者旨在衡量不同通道的重要性程度，后者旨在衡量不同特征值的重要性程度。

graphic

图7.3 CBAM模型示意图

如图7.4所示，**通道注意力机制**模块运用全局平均池化方法，将每个通道的特征图压缩为一个标量值，然后通过激活函数进行归一化处理，最后将带有颜色的1\*1\*C矩阵和原始特征图C点乘，从而获得加权后的特征图。

graphic

图7.4 通道注意力机制模块的应用过程

如图7.5所示，**空间注意力机制**模块在每个特征点的通道方向上分别进行最大及平均池化处理，然后通过一个卷积层和sigmoid函数来学习每个空间位置上的权重，最后将权重应用到原始特征图的每个空间位置上，以获得凸显部分特征重要性的特征图。值得注意的是，经空间注意力机制处理过后的特征图，仅在high\*weight面上凸显了部分特征，不同通道切片上的注意力方式是一致的。只有经通道注意力机制处理过后的特征图，不同通道切片的重要程度才不同。

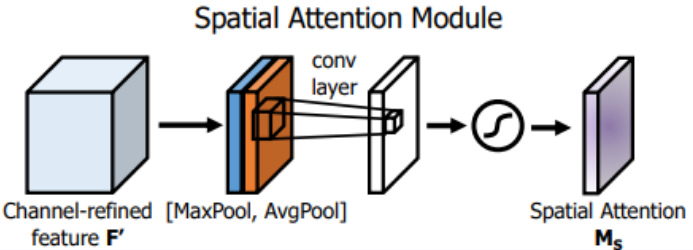


图7.5 空间注意力机制模块的应用过程

问题是：如何使用注意力机制？

传统做法是：将注意力机制同经典模型相结合。比如，RNN在处理繁琐的特征信息时，往往会因为层数过深而造成长期记忆遗失问题，通过引入注意力机制关注重点信息的方式，有效缓解了长期记忆遗失问题。此外，引入注意力机制的encoder-decoder模型，在处理机器翻译任务时，可以弥补不擅长处理不定长输入序列的缺陷。

常规做法是：基于注意力机制的算法。比如，基于自注意力机制的**Transformer算法**依赖注意力机制捕捉输入序列中不同位置之间的依赖关系，并通过并行计算提高了序列建模能力及计算效率。此外，**Reformer算法**在Transformer的基础上进行了如下改进：将点积注意力替换为局部位置敏感哈希注意力、用可逆残差取代标准残差层、将比较厚的层进行分块预处理，从而达到了保证性能却又占用更少内存的目的。值得注意的是，部分文献提及了**Hopfield算法**，指出transformer是Hopfield网络在连续状态下的一种特殊情况。