NLP硬核入门-Seq2Seq和Attention机制

AINLP 2019-10-26

以下文章来源于数论遗珠 ， 作者阮智昊

数论遗珠
数学和机器学习的心得笔记

本文需要的前序知识储备是：循环神经网络RNN，词向量WordEmbedding，门控单元VanillaRNN/GRU/LSTM。

1 seq2seq

seq2seq是sequence to sequence的缩写。前一个sequence称为编码器encoder，用于接收源序列source sequence。后一个sequence称为解码器decoder，用于输出预测的目标序列target sequence。

seq2seq主要用于序列生成任务，例如：机器翻译、文本摘要、对话系统，等等。当然也可以用于文本分类等任务。

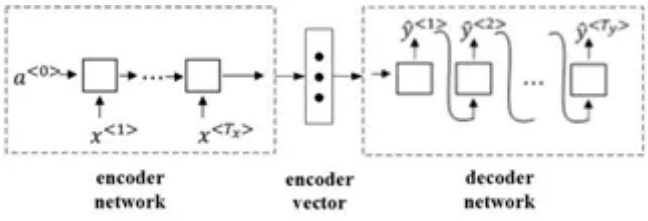

图1.1 seq2seq

最传统的seq2seq流程如图1.1所示：

（1）将源序列输入encoder网络。

（2）encoder将源序列的信息编码成一个定长的向量encoder vector。

（3）encoder vector被送入decoder网络。

（4）decoder根据输入的向量信息，输出预测的目标序列。

seq2seq在被提出后，马上受到了广泛的关注和应用，也暴露出一些问题。首先被关注到的，就是人们发现把一整个文本序列通过encoder压缩到区区一个向量里，很难通过decoder进行完美地没有信息缺失的解码。

此外，由于循环神经网络RNN的特性，源序列中越迟输入的文本，对encoder vector的影响也越大。换句话说，encoder vector里会包含更多的序列尾的文本信息，而忽略序列头的文本信息。所以在很多早期的论文中，会将文本序列进行倒序后再输入encoder，模型测评分数也会有一个显著地提高。

为了让decoder能够更好地提取源序列的信息，Bahdanau在2014年提出了注意力机制Attention Mechanism，Luong在2015年对Bahdanau Attention进行了改进。这是两个最经典的注意力机制模型。两个Attention模型的本质思路是一样的，下文均以Luong Attention模型作为范例。

2 Attention Mechanism

注意力机制的理解，可以参考CV领域的思想：我们在看到一幅画的时候，每个时刻总会有一个关注重点，比如说某个人、某个物品、某个动作。

所以，在NLP领域，我们在通过decoder预测输出目标序列的时候，也希望能够有一种机制，将目标序列当前step，和源序列某几个step的文本关联起来。

以翻译任务为例，将“我爱机器学习”翻译成“I love machine learning.”在decoder输出序列第一个step，我们希望关注输入序列中的“我”，并将“我”翻译成“I”；在第三个step，我们希望关注“机器”，并翻译成“machine”。

这个例子比较简单，我们就会产生一个初步的想法：是不是把源序列中的每个词语单独翻译成英文，再依次输出，就构成目标序列了呢？

但是，如果进一步思考下，我们就会发现两个问题：

（1）**一词多义**：源序列里的同一个词，在输出序列里，可能根据场景的不同，会有不同的输出。例如“我”可能被翻译成“I”，也有可能被翻译成“me”。这有点类似于中文的“一词多义”，在英文里估计是叫做“一词多态”吧，我们姑且将这类由一个词可以映射成多个词的现象，广义地统称为“一词多义”。解决“一词多义”问题的一个有效途径，就是参考源序列的语境信息，也就是上下文信息，来生成词向量。

（2）**序列顺序**：源序列和目标序列并不是顺序依次映射的，例如“你是谁？”翻译成“who are you?”，不同语言有不同的语法规则和顺序。这就需要在decoder输出的每一个step，确定当前step应该翻译源序列中的哪一个词。

这两个问题也就是Attention机制的关注重点。

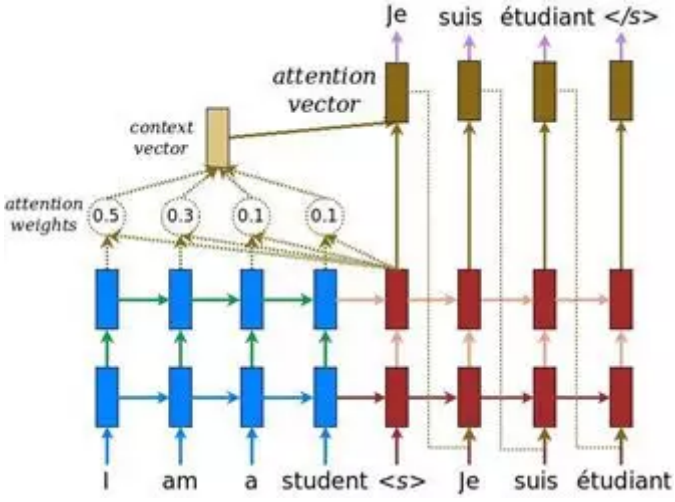

图2.1 LuongAttention

图2.1是一个Luong Attention的示意图，是作者其后续的论文里呈现的一张修订后的示意图。

还有个理解Attention的方式，就是参考残差网络ResNet。因为源序列太长了，导致早期的信息无法有效地被传递，所以需要一个额外的通道，把早期的信息送到解码器上。送的时候还不能只送一个词的信息，最好把上下文信息一起给送了。

下一节会用一个最简单的模型，介绍Attention机制的实现步骤，在此之前，先约定下参数符号：

h(output)：RNN的隐藏状态，主要用于将信息输出到RNN模型外。

s(state)：RNN的状态，主要用于RNN模型内部，模型将信息传递给下一个step。

a：对齐向量

c：上下文信息向量。

x：源序列。

y：目标序列。

下标s表示源序列，下标t表示目标序列。s1表示源序列第1个step，以此类推。

括号里的output和state，是为了方便读者将论文里的算法理论，和工业实践里tensorflow的tf.nn.dynamic_rnn函数联系起来，稍微有个印象就好。dynamic_rnn函数返回两个向量，第一个是output，也就是encoder所有step、网络最后一层输出的h；第二个是state，也就是encoder最后一个step、网络层所有层输出的s。

3 Attention五部曲

3.1 执行encoder

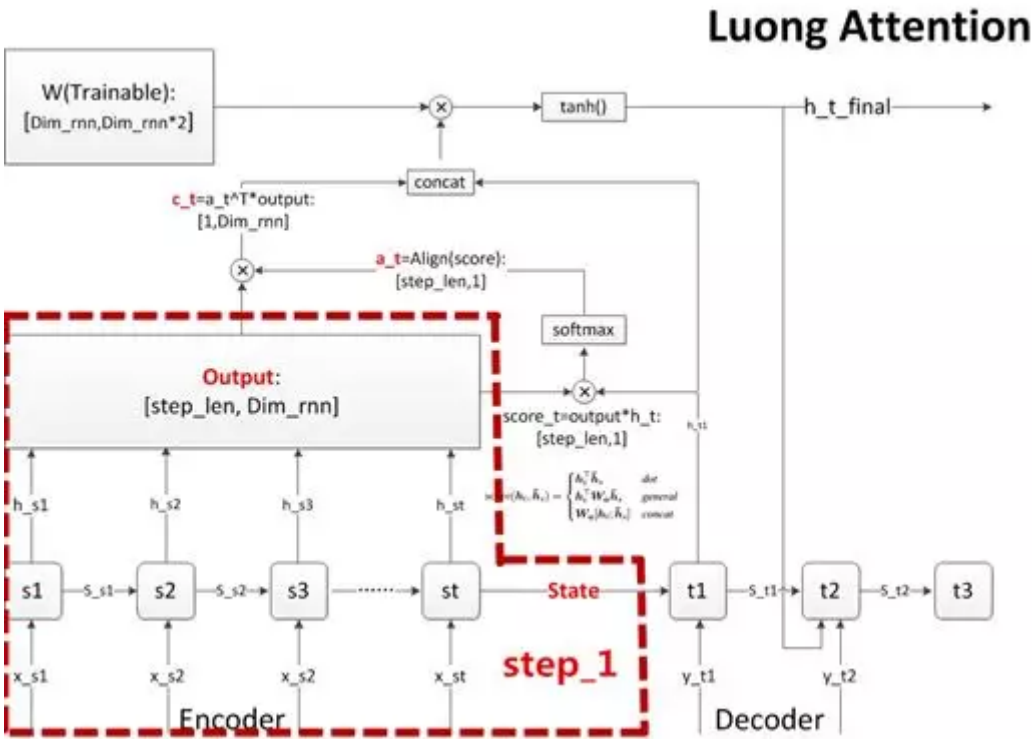

图3.1 步骤一：执行encoder

步骤一的行为是将源数据依次输入Encoder，执行Encoder。**目的在于将源序列的信息，编译成语义向量，供后续decoder使用。**

在每个step，endocer会输出一个表征当前step语义的output（h）向量和一个state（s）向量：

（1）我们收集每个step的output（h），构成output矩阵，矩阵的维度是[step_len,dim_rnn]，即源数据step长度，乘以rnn单元数量。

（2）我们收集最后一个step的state（s），作为传入decoder的向量。

encoder对于模型的贡献，在于提供了outputs矩阵和state向量。

注一：为了便于理解，我这里的encoder使用了单层网络，多层网络的outputs和state见上一节末尾的描述。

注二：很多论文的h和s的描述并没有一个统一的标准，经常混淆。因为早期论文的RNN单元，是用VanillaRNN或GRU实现的，这两个门控单元在同一个step，输出的h和s是一样的。但是，若通过LSTM实现，h和s是不同的，这个需要引起注意。

注三：早期的论文中，encoder的state是直接传递给decoder，作为initial state的。但是在工程应用中，也存在直接将0序列作为initial state传递给decoder的情况。另外，部分论文也有将state进行一些处理，添加一些额外的信息，再传递给decoder的算法。总之，encoder和decoder之间传递state的方式比较灵活，可以根据实际情况自行选择和改进。

注四：RNN的单元数量，即为encoder输出向量的维度。也就是用dim_rnn维度的向量，来表征源序列当前step文本的语义信息。对照同样表征语义信息的词向量的维度dim_word_embd，我建议两者的维度不宜相差过大，否则会造成浪费。

3.2 计算对齐系数a

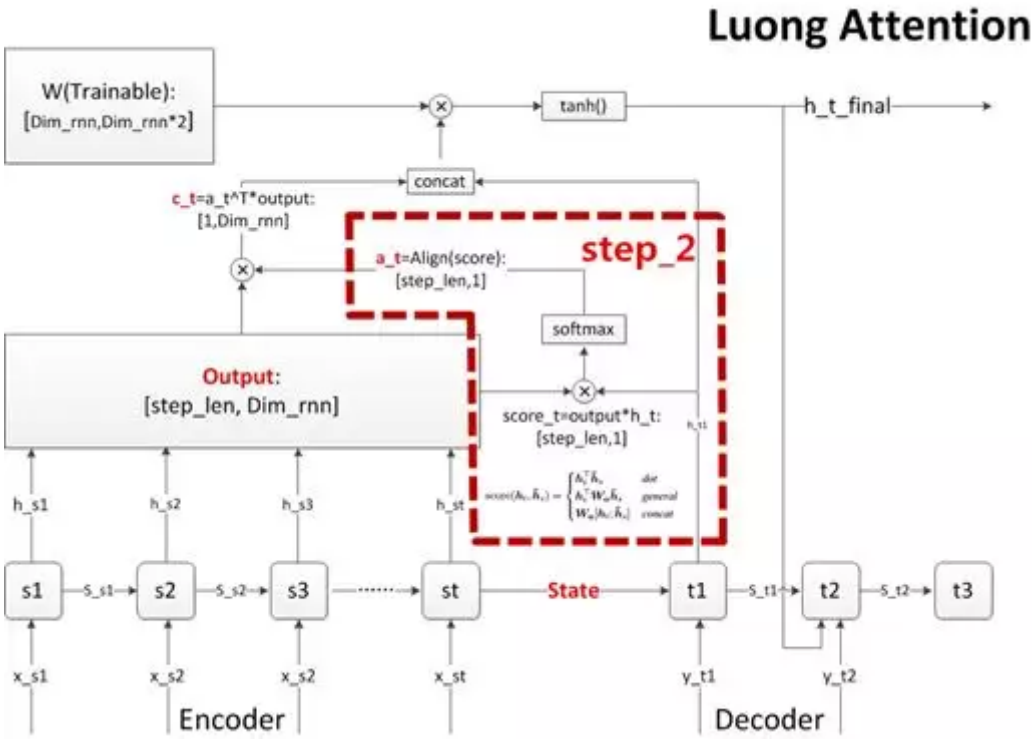

图3.2 步骤二：计算对齐系数a

步骤二解决的是第2节提出的“序列顺序”的问题。**在decoder的每个step，我们需要关注源序列的所有step和目标序列当前step的相关性大小，并输出相关（对齐）系数a。**

所以，在decoder输出一个预测值前，都会针对encoder的所有step，计算一个score。这个score表示当前的decoder工作，需要从encoder的哪些step里抽取信息，以及抽取的权重大小。然后将score汇总向量化后，每个decoder step能获得一个维度为[step_len,1]的score向量。

这个score的计算方式有很多种，图3.2中列举了Luong Attention提及的3种的传统计算方式。我画的流程图中采用的是第1种，就是将源序列所有step的output（h）和目标序列当前step的output（h）逐个相乘，得到的值即为score。有些论文就是在score的计算方式上进行创新的。

计算出score后，很自然地按惯例使用softmax进行归一化，得到对齐向量a，维度也是[step_len,1]。

注一：很多论文的各种参数的缩写符号都不一样，有一个理清模型流程顺序的小技巧：就是去找softmax函数，以softmax为锚点。Attention常见的使用softmax的地方有两个，一个是步骤二的对齐系数a，另一个在步骤五将会提到，在输出预测词之前，要对概率分数进行softmax归一化处理。

注二：对齐系数a虽然只是一个过程数据，但是却蕴含很重要的信息，可用于PointerNet和CopyNet。

3.3 计算上下文语义向量c

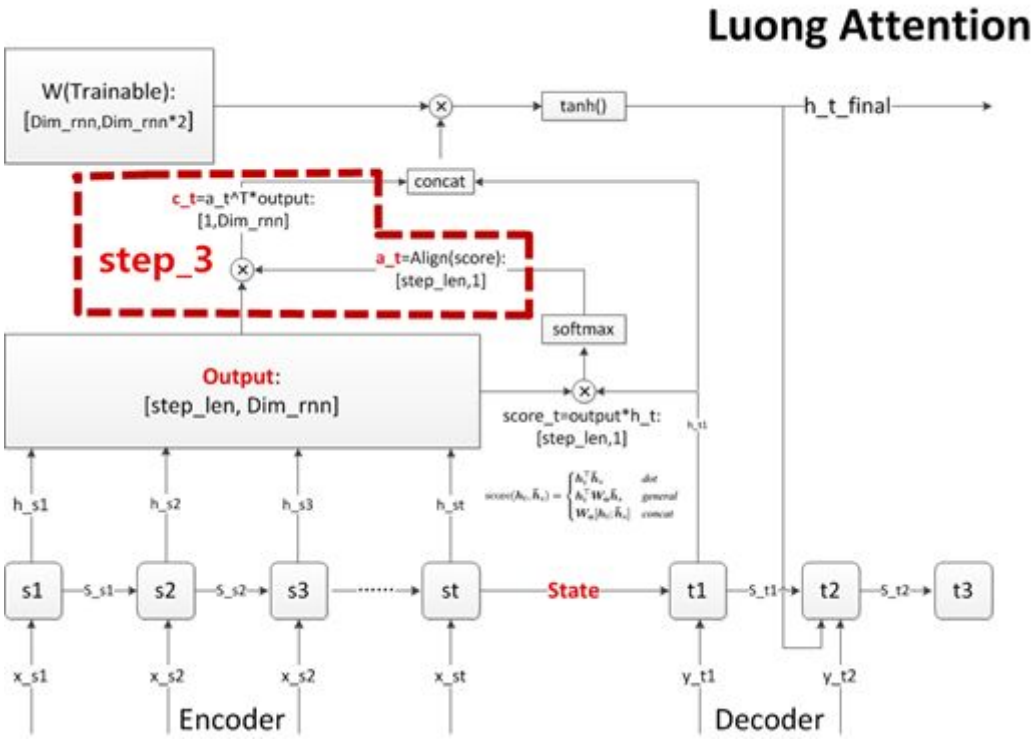

图3.3 步骤三：计算上下文语义向量c

在描述这个步骤前，我们先回顾下词向量的CBOW模型。在CBOW模型收敛后，每个词的词向量，等于它周围若干个词的词向量的均值。这其中蕴含的意思是：**表征一个词的，不是这个词本身，而是这个词的上下文（语境）。**

CBOW模型是比较简单粗暴地将上下文的词向量求平均。实际上，如果能够以一个加权平均的方式获取词向量，那么这个词向量一定能够更准确地表达这个词在当前语境里的语义。

举个例子：“孔夫子经历了好几个春秋寒暑，终于修订完成了春秋麟史。”在这里，第一个“春秋”表示“一年”，“经历”、“寒暑”显然和它关系更密切，利用加权上下文构成词向量时，应该赋予更高的权重。第二个“春秋”表示儒家六经之一，“修订”、“麟史”关系和它更密切，同样应该赋予高权重。

在步骤三里，**我们将对齐系数a作为权重，对encoder每个step的output向量进行加权求和（对齐向量a点乘outputs矩阵），得到decoder当前step的上下文语义向量c。**

注一：BERT也有用到对齐系数的思想，而且更为直观漂亮。

3.4 更新decoder状态

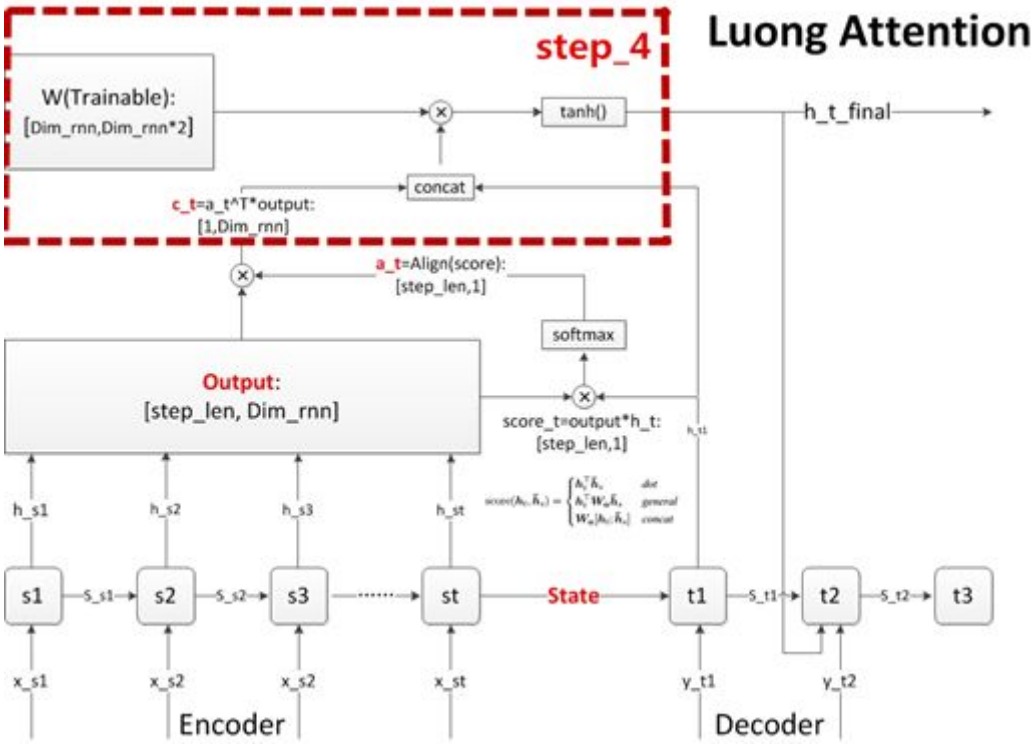

图3.4 步骤四：更新decoder状态

在步骤四里，需要**更新decoder状态，这个状态可以是h，也可以是s。**能够用于更新h和s的信息数据，可以是：前step的s，现step的h，现step的上下文向量c，以及其它一些包含有效信息的数据。

BahdanauAttention和Luong Attention最大的区别就在于这个步骤，前者更新的是s，后者更新的是h。不过由于Bahdanau用的是前step的s，Luong用的是先step的h，所以后者在工程化实现上会简单点。

具体的更新公式的细节，在这里不作详细描述，因为不同模型可能会采用不同的更新公式，很多论文也是围绕更新公式进行创新点研究的。

需要注意的是，在这个环节，训练模式和预测模式略有差别：decoder每个step都要输入一个数据，在训练模式，输入的数据是目标序列当前step的真实值，而不使用前step的h；在预测模式，输入的数据是前step的h，而不使用输出序列的真实值。虽然在图3.4中，我画了两条输入，但是要根据模型当前处于训练模式还是预测模式，选择其中的一条进行输入。

3.5 计算输出预测词

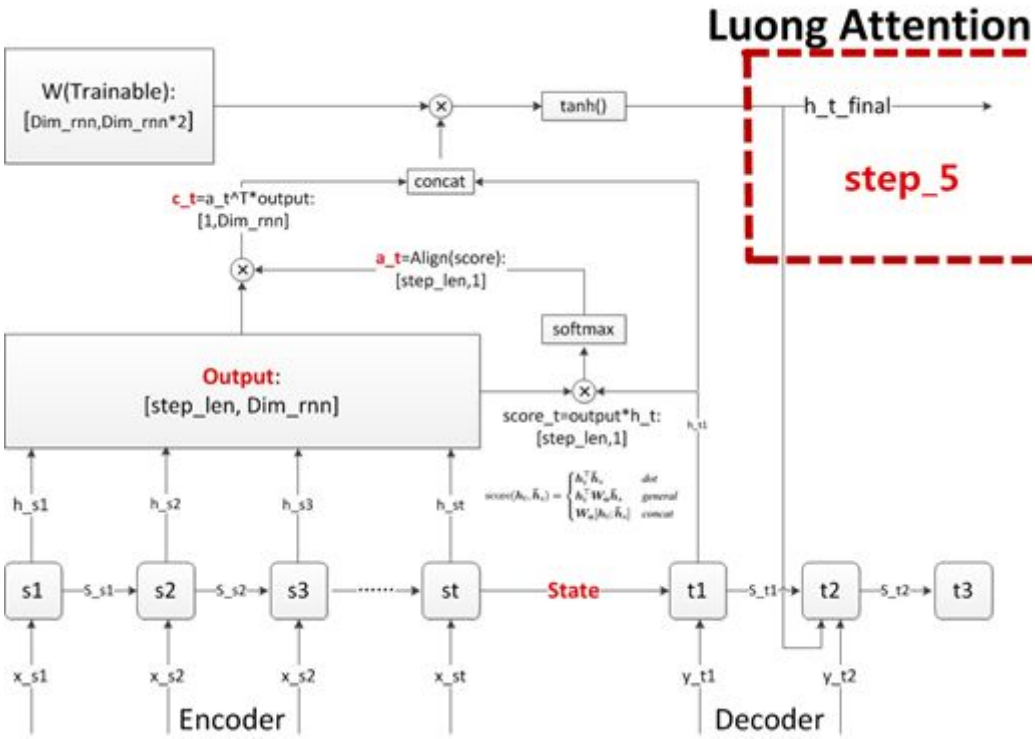

图3.5 步骤五：计算输出预测词

这个步骤我在图里没有画全，其实很简单，同CBOW模型/Skip-Gram模型的隐藏层到输出层的那部分一样，做一个语义向量到目标词表的映射（如果attention用于分类模型，那就是做一个到各个分类的映射），然后再进行softmax就可以了。

4 其它

4.1 Local Attention和Global Attention

前文所提及的Attention都是Global Attention，还有一个Local Attention，将在这个小节作一个简单的说明。

Global Attention就是针对源序列的所有step，求对齐系数a。而LocalAttention只针对源序列的部分step，求对齐系数a，这个部分step的长度是超参数，需要凭经验人为配置。

Local Attention所截取的部分step的中心点的选取（对齐）方式，是另一个需要关注的点。论文中提及了两个对齐方式：

（1）Monotonicalignment (local-m)：简单粗暴的，直接按源序列和目标序列的step绝对值对齐。

（2）Predictivealignment (local-p)：通过模型，学习计算出截断step的对齐中心。

Luong的论文里有提及，LocalAttention的效果优于Global Attention。

注：CV领域有个Soft-Attention和Hard-Attention，和这里NLP领域的两个Attention优点类似。

4.2 常见的可以替换改进的模块

1.用于生成对齐向量a的分值score的计算方式。

2.h和s的更新公式。

3.基本RNN的结构，包括替换门控单元、更改RNN层数、单向改双向等。

参考资料

[1] Bahdanau D ,Cho K , Bengio Y . Neural Machine Translation by Jointly Learning to Align andTranslate[J]. Computer Science, 2014.

[2] Luong M T ,Pham H , Manning C D . Effective Approaches to Attention-based Neural MachineTranslation[J]. Computer Science, 2015.

[3] Andrew Ng RecurrentNeural Networks

由于微信文章有修改字数的限制，故附上知乎文章的链接：https://zhuanlan.zhihu.com/p/73589030

后续有更新或纠错，会在知乎文章上呈现。

本文转载自公众号：数论遗珠，作者阮智昊

推荐阅读

神经网络硬核入门-反向传播(BP)算法

赛尔笔记 | Attention！注意力机制可解释吗?