长文详解Attention、Seq2Seq与交互式匹配

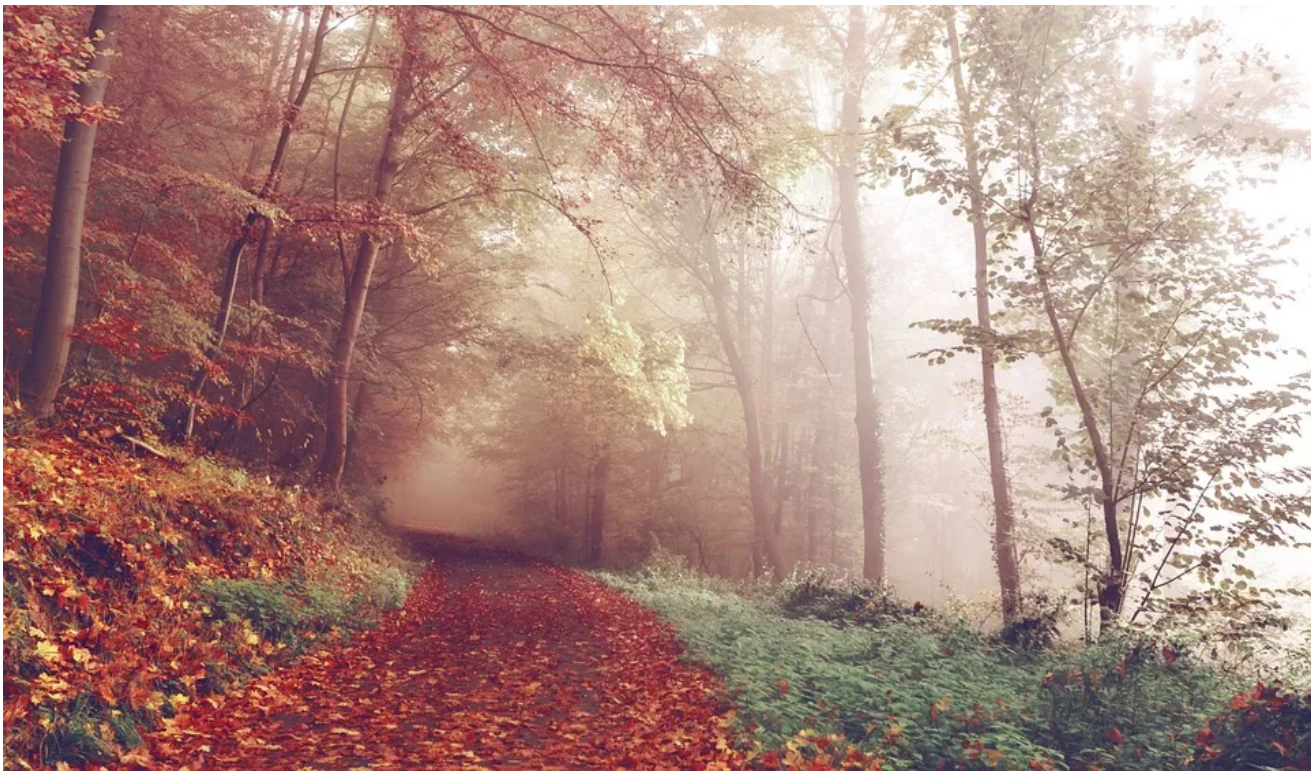
原创 谢铁 深度学习自然语言处理 2020-09-05

收录于话题

#seq2seq 1 #交互式匹配 1 #attention 1

点击上方，选择**星标**或**置顶**，每天给你送干货🤗！

> 阅读大概需要19分钟🤤
> 跟随小博主，每天进步一丢丢 😁



作者：谢铁

公司：苏宁金融

研究方向：智能问答

知乎链接：https://zhuanlan.zhihu.com/p/146760904

一、Attention的介绍

其实关于注意力机制（Attention）的概念，网上已经有一大把的介绍，这里就不针针对其概念做过多介绍。一句话说就是聚焦关键的地方，忽略不重要的地方。放在nlp里可能就是把某些关键性

单词的权重提升，其他意义不大的词权重降低。

在应用于nlp之前，Attention已经在图像领域大放光彩了，因此你可能会看见用人的视觉来解释attention的，非常自然合理也通俗易懂地解释了attention的机制。不过有个地方有些违反直觉，就是大脑中的注意力机制主要是为了减少计算量（忽略掉不重要的信息），然而在计算机里的attention机制却无一例外地需要增加计算量。这是因为，计算机并不知道一堆数据哪个会比较重要，所以它需要一个全局的扫描，从而加大了计算。正因如此，有人觉得注意力机制实际上更像是查询机制。因此本文会主要从查询机制的角度去理解attention。

二、Query角度的解释

上一节提到了Attention有点像查询机制，本文就是打算从这个角度去解释attention。一旦理解了以后，你会发现这种解释也同样通俗易懂。

实际上，Attention机制就是一种软查询。什么是软查询，为了理解这一点，我们需要知道什么是硬查询。

所谓硬查询，它的过程类似于python中的Dict。本质上是用一个q，在一个k-v对里查询v，如果q等于某个k，就可以把这个k对应的v取出来。硬查询的“硬”体现在，q必须等于某个k才能查到k所对应的v，如果不相等，则什么都查不到。

那么相对的，软查询，“软”体现在，不需要q=k，而是，q会与所有的k计算一下相似度，按照相似的程度，从不同的k中按等份取出对应的v。

举个例子理解下吧：
假设有个字典，{苹果:3, 香蕉2, 青椒:1.5， 面包:1}，它的key值对应一种食物，value，呃，whatever，当它是营养价值好了，在这个场景下它是一个可以查询营养价值小册子，如果想要查询苹果的营养价值，非常简单，找到苹果，拿出它对应的值3，如果想要查询红辣椒，不好意思，查无此物，这就是硬查询。

如果用数学语言来说，我们可以把这个字典看成两个列表，KEY=[苹果，香蕉，青椒，面包]，VALUE=[3, 2, 1.5, 1]。假设查询条件是QUERY=苹果。那么硬查询就是让QUERY与KEY做and操作。对K的查询结果就是[1, 0, 0, 0]，接下来让这个结果与VALUE相乘，就得到了最终的查询结果“3”，即：

$$[1,0,0,0]\odot[3,2,1.5,1]=3\times1+0\times2+0\times1.5+0\times1=3$$

从上我们可以看到，所谓的硬查询的硬就是体现在元素的权重非0即1，除了与QUERY一模一样的KEY对应的元素之外，其他的元素不再提供任何信息，非常绝对没有变通。实际上有种Attention机制叫Hard Attention，这里的Hard的意义与这个是一致的。

那如果是软查询呢？软查询不需要QUERY与KEY完全相等，当我们的QUERY是红辣椒时，而字典里没有红辣椒，没关系，我们可以参考一下其他食物的营养价值以推测红辣椒的营养价值，现在把红辣椒与所有的食物一一比较一下，看有没有相似之处。先是红苹果，颜色一致也都是植物界，算30%相似吧。接下来香蕉，同属植物界，也许只有10%的相似。青椒呢？都是辣椒，他俩有60%相似，接下来面包，一点儿也不相似。接下来，越相似的食物提供的参考价值越大，我们把相似度当成权重把所有食物的营养价值加权相加就能大致推算出来红辣椒的营养价值了：

$$[0.3, 0.1, 0.6, 0] \odot [3, 2, 1.5, 1] = 0.3 \times 3 + 0.1 \times 2 + 0.6 \times 1.5 + 0 \times 1 = 2$$

上面就是一个软查询的过程，软查询的软体现在查询的过程，字典里所有的元素权重不再非0即1了，只要相似（而不是全等）就有参考价值。当然，我的例子举的有点不太符合生活直觉...意思到位就行了。现在还剩下一个问题是，这个相似程度怎么计算？其实如果更加形式化的表达软查询的过程就是 $sim(QK) \times V$ （这实际上是所有attention的通用框架）。这个公式本身没什么，但关键的地方在于 $sim()$ 函数是如何定义的。我这里罗列了这么几种：

（1）加性与concat方式：

$$sim(q, k) = v_a^T tanh(Wq + Uk)$$

这个方法实际上比较早期的了，大家可能在很多地方看见过这个公式，我也是抄过来的，注意这里的 v 并不是VALUE，而是一个可以训练的参数。

与这类似地还有把Q与K concat起来的，

$$sim(q, k) = v_a^T tanh(W[q; k])$$

这两个公式乍看起来很难直观理解它在做什么，但其实意义很简单：我们的目的是找到一个 靠谱的 $sim()$ 函数，既然神经网络能够拟合任意函数，那我何不让神经网络去干这件事呢？所以你看到的这个公式就是一个神经网络，目的是为了拟合 $sim()$ 函数。多乘了一个参数 v 主要是为了把结果从向量变成一个矩阵（为什么要变成一个矩阵下面会说）。这种方式计算量会比较大，用的人已经不多了。

（2）乘性

包括：

$$sim(q,k) = \frac{q^T k}{||q|| \cdot ||k||}$$

$$sim(q,k) = q^T k$$

$$sim(q,k) = \frac{q^T k}{\sqrt{d_k}}$$

$$sim(q,k) = q^T W k$$

上述几个公式的思想基本上和第一个公式一样，也就是cos相似度。第二个可以看成没有归一化的cos相似度，值得注意的是第三个，也就是attention all you need这篇论文里提到的方法，它除以了一个常数 $\sqrt{d_k}$ （ k 的维度开根号），其主要目的是：在维度较大的时候，向量点积会导致一个较大的结果（点积是两个向量对应元素相乘后累加，显然维度很大，累加项也就越多），容易引起梯度爆炸。因此这里除以了一个常量来缓解这一问题。至于为什么是 $\sqrt{d_k}$ ，可以假设q与k的元素服从均值为0，方差为1的正态分布，并且独立。点积后，其分布服从均值0，方差为 d_k （标准差正好是 $\sqrt{d_k}$ ），因此这里除以 $\sqrt{d_k}$ 是为了将分布转成标准正态分布，从而防止梯度过于庞大。第三种是早期的乘性方案，多了一个参数 W ，我觉得可以理解为对q与k做了词线性变换。

三、直观解释

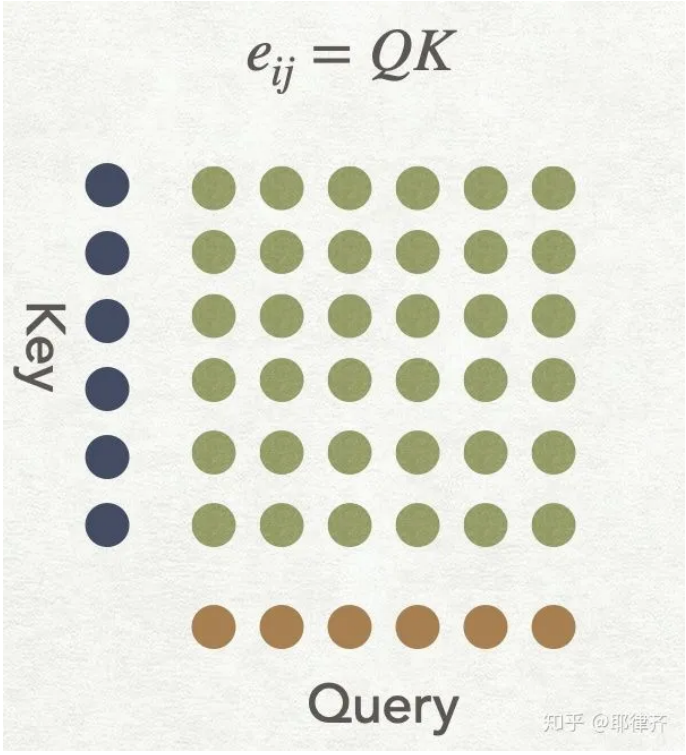
以上是从Query的角度对attention做了解释，下面会从更加细节更加直观的角度来阐明下，attention究竟做了什么事情。

attention的过程一共分为三步：**一、计算相似度得分，二、归一化，三、加权**

（1）计算相似度得分：

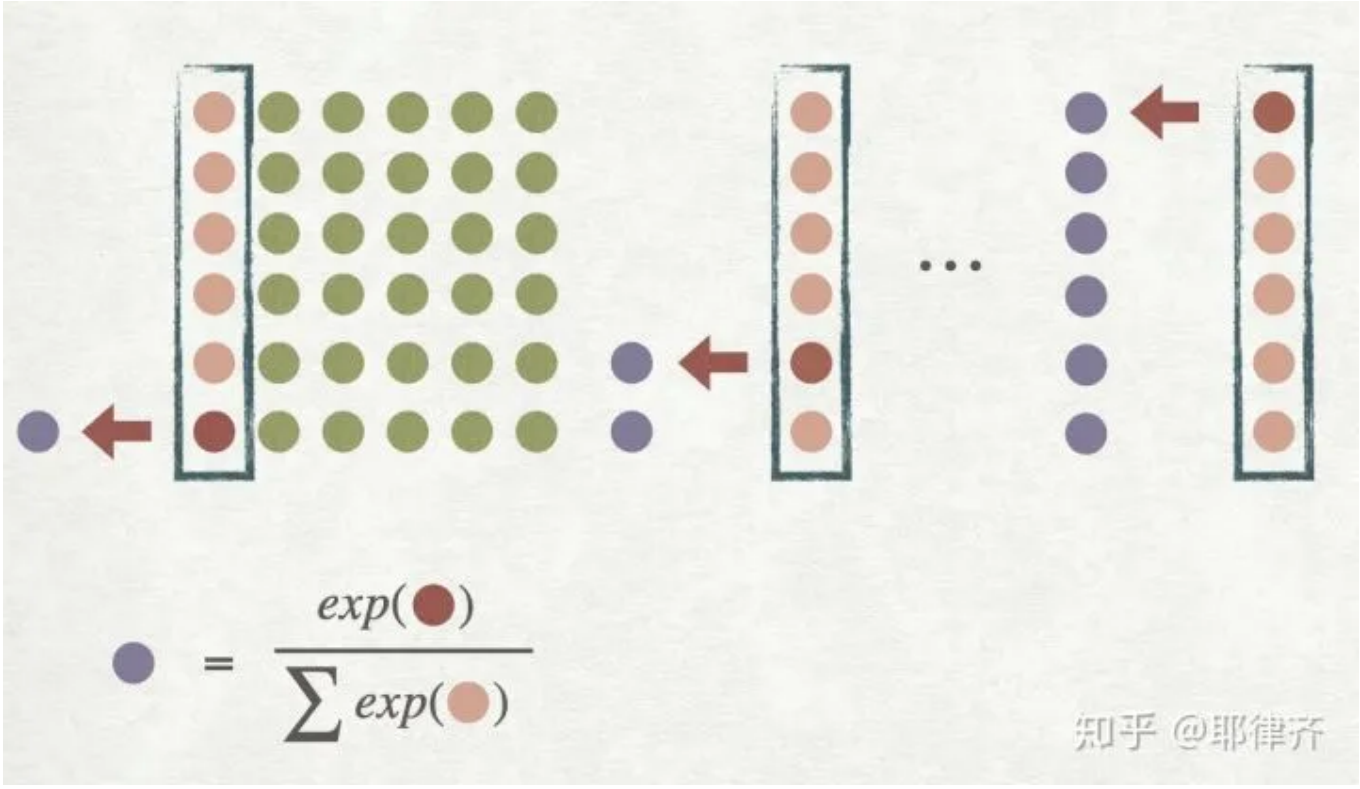
这一步就是上面提到的 $sim(QK)$ ，公式上面介绍了不少，现在来看一下，这个函数具体做了什么事情。这里用一张我之前用于分享交互式匹配的图来辅助理解一下。

首先两个向量分别代表Q和K（浅蓝色和浅黄色表示），$sim(QK)$做了什么事呢？就是让Q和K这两个向量里的元素俩俩比较一下，得出一个相似度矩阵。相似度矩阵每一个元素e_{ij}表示Q中第i个元素与K中第j个元素的相似度。下图是相似度计算的一个形象化的表示（但是要注意的是，真实情况下，图里的圆圈通常是向量而不是标量，换句话说这里应该是一个3维张量，但为了更清晰的展示过程，简化了这一点）。



（2）归一化

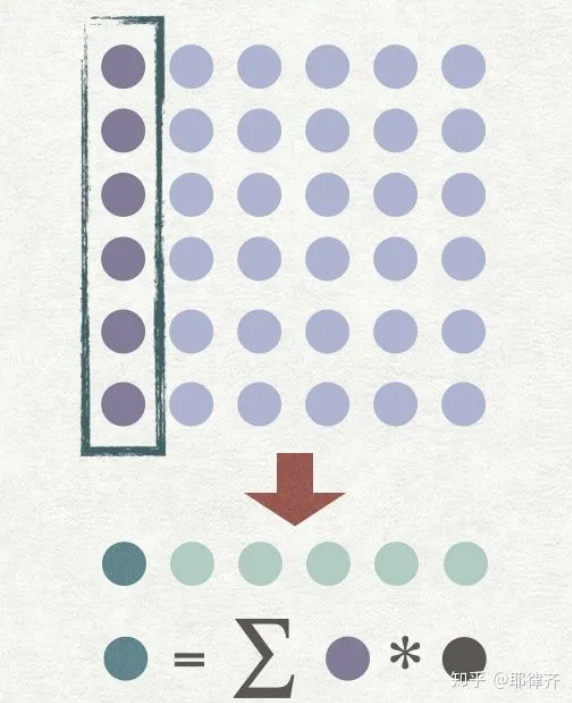
上图中，绿色圆圈所表示的e_{ij}是未归一化的相似度，attention里通常会对这个值进行归一化。归一化的方式很简单，就是套一个softmax函数。但为了更深刻的理解，下图展示了这一过程。



首先看最右下角的元素归一化的过程，实际上就是该元素的指数（深红色表示）除以这一列所有元素指数的加和（包括它自己）。然后依次对第一列的每一行做同样的操作。接着是第二列、第三列，直到每一个元素都进行了归一化。

（3）加权

最后一步就更加直接了当了，就是加权相加。如下图：



新的向量中，第一个位置的元素值（深绿色圆圈表示）等于第一列中相似度值（attention权重）与Value中值（黑色圆圈表示）乘积的加和。

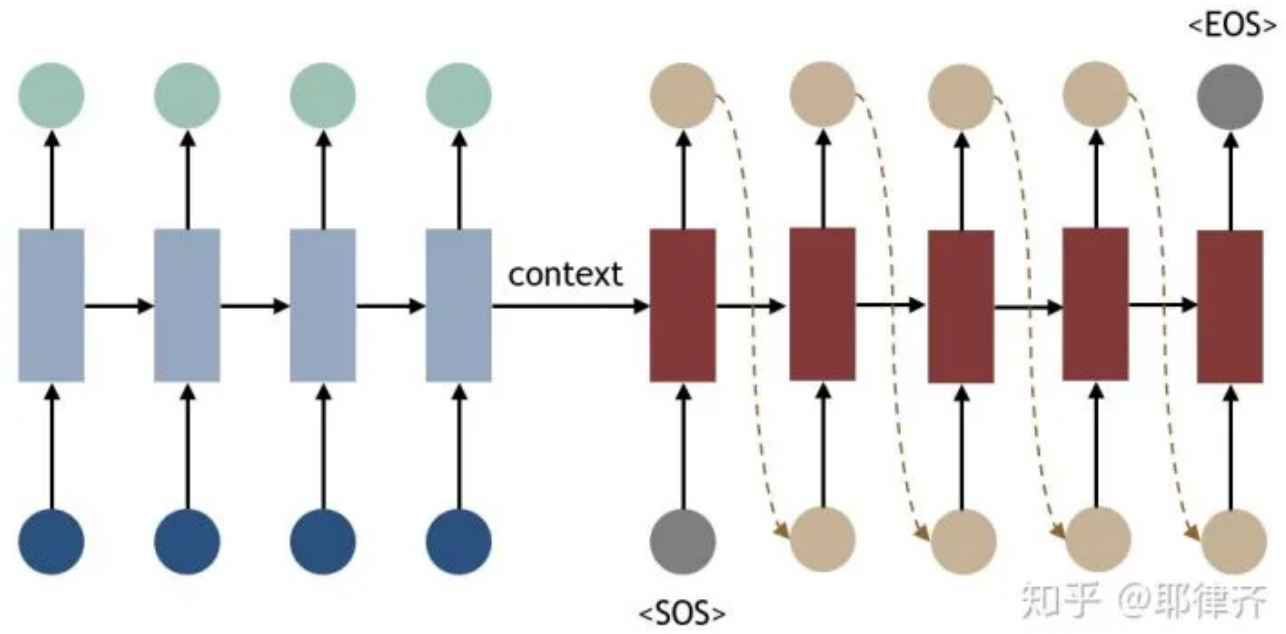
至此，attention也解释完了，流程是不是清楚了一些呢，其实本身也不是什么特别难以理解的东西。其实最后还剩下一些问题，比如，在不同的场景下，QKV分别对应什么呢，这是理解一个具体场景中attention重要的一步。在后面的文章我会从具体的attention应用中去介绍attention的用法以及一些我自己的感悟。

上面主要介绍了attention的原理，下面会将会着重介绍attention在各个场景中的应用。实际上，为了理解attention发挥了什么样的作用，搞清楚attention的QKV分别对应的是什么角色是非常重要的，文本也是从这个角度去分析与attention有关的几个算法。

四、翻译模型中的attention

至少在我的了解中，attention最开始是在翻译模型中引入到NLP领域的。这其实也是一个非常容易理解的场景。先来看看，如果没有attention，翻译模型是如何做的？

一般来说翻译模型包含一个encoder和一个decoder。encoder的作用是把源语言的句子编码为一个向量。decoder的作用正好相反，是把encoder编码出的向量解码为一个个字。为了方便理解，我这里用LSTM做介绍。

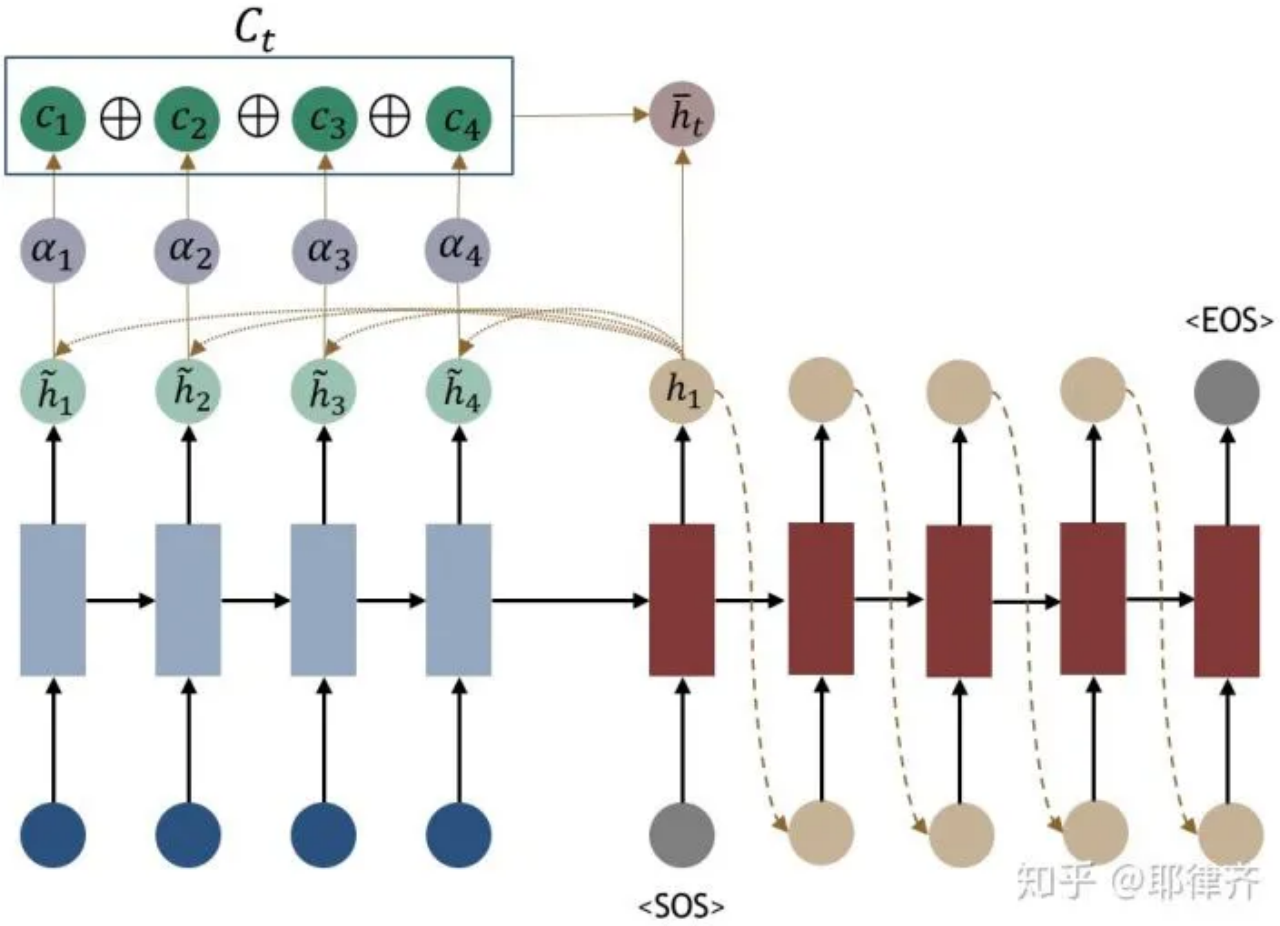


encoder-decoder结构

首先，我们知道LSTM有两个输入，一个就是输入值 x_t ，另外一个是上一轮的输出值 h_{t-1} 与状态值 c_{t-1} 。encoder这边的LSTM很容易理解，和一般的LSTM并无二致。再看decoder这边，既然是需要解码源语言，那么源语言的信息肯定得有。最初的想法比较简单，把源语言的信息编码（也称context信息）传给decoder中的lstm作为初始状态，这个contex信息随着decoder中间那个通道往后传。当然这个做法问题很多，由于context信息在整个流程中只用了一次（也就是decoder第一个step中用了），很显然，context的信息会随着decoder中step增加，信息衰减越来越严重。

PS：一个小补充，如果是训练过程，由于目标语言的句子中每个单词模型是能看见的，因此训练阶段的单词的编码就是其embeding，而如果是预测阶段，目标语言中的字是看不见的，因此预测阶段会拿上一个时间步的输出作为输入（这就是图中虚线存在的意义）。另外第一个字之前一般会设置一个起始符<SOS>，作为初始输入。

为了避免信息衰减，一个非常简单明了的改进思路是，在decoder的每个step都用一下这个原始的context信息，这样的话context就不会衰减了。但这会产生另外一个问题：在翻译每个字的时候，用的都是同样的信息，这是违反直觉的。因为翻译不同的字我们在源语言上关注的点是不一样的，比如，Cat sit on the mat， 翻译猫的时候，我们应该更加关注Cat而不是其他单词。所以，attention就可以派上用场了。

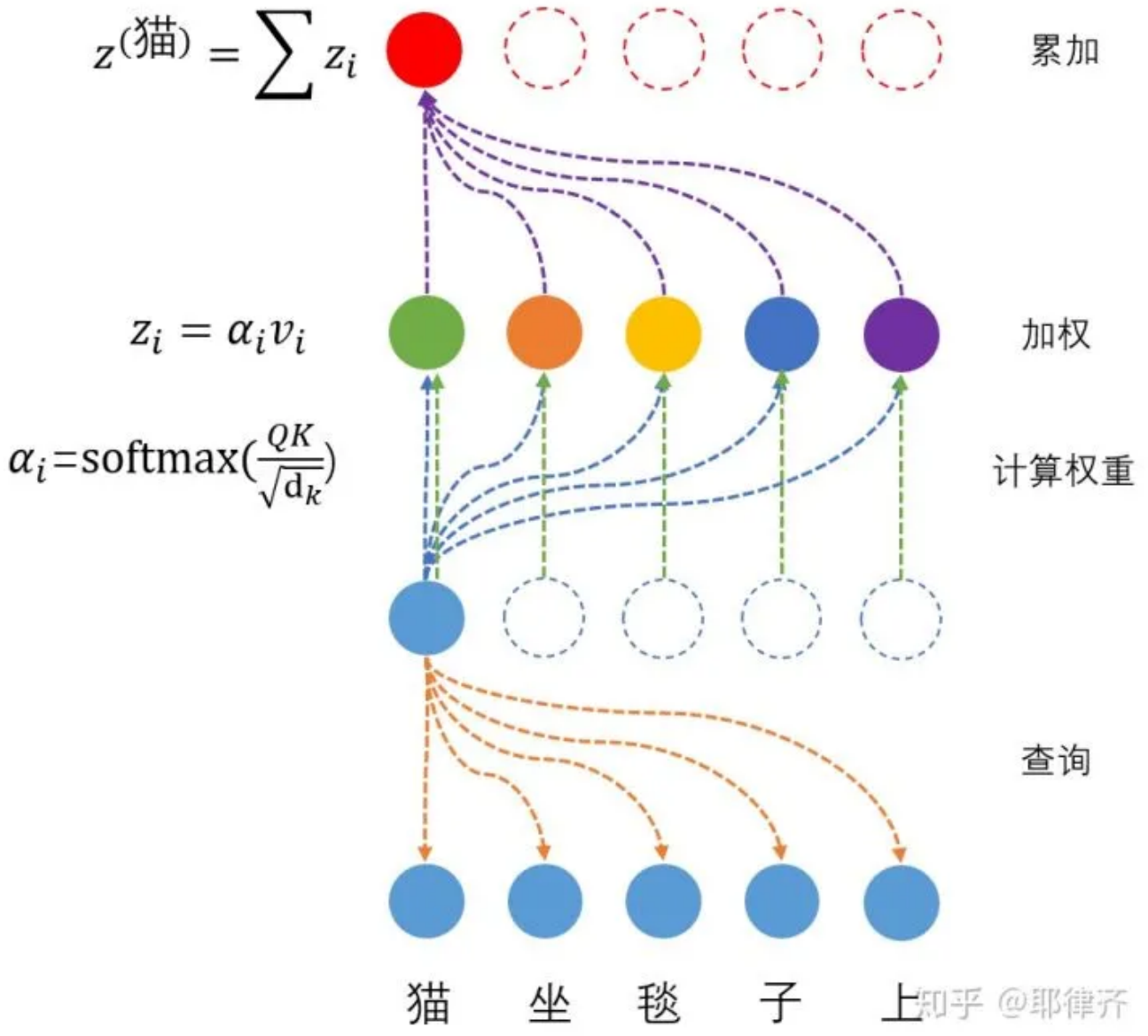


还记得attention就是一个软查询过程吗，我们假设decoder中在第t个step下，lstm的输出是 h_t 。我们可以让 h_t 在encoder中的每一个 $\tilde{h}$ 上做一次查询，看看源语言中哪一个节点的输出跟自己关系最大，这样，我们就能够计算出attention权重 α 了。得到attention权重，剩下的事情就是加权累加了 $c_t = \sum_i^T \alpha_i \tilde{h}_i$ 。

不难理解，用QKV模型来解释的话，query就是 h_t ，key就是 $\tilde{h}$ ，value还是 $\tilde{h}$ 。

五、tansformer中的attention

tansformer中的attention主要是self-attention。可能由于transformer比较火对其研究的比较透彻，在我看来self-attention是一个比较容易理解的结构。从名字也能看出来，这个attention是自己跟自己做attention。也就是QKV来源都是自己（input embedding），只不过input做了三次不同的线性变换，具体过程可以看下图（注意为了简化模型，下图我省略了线性变换的步骤）。



首先是查询，黄色虚线所示，其含义为，某字符与其他字符（包括它自己）进行对比（即查询，实际上查询与计算权重应该是同一步，为了方便理解，图中我把这个步骤拆开来看了）。对比之后就开始计算权重，蓝色虚线所示，计算权重里面的细节可以参考（一）中的内容，其含义是看看当前字符与同句其他字符相似程度。最后就是在每个字符的原始值上加权了，权重就是当前字符与其他字符的相似度，图中绿色虚线与蓝色虚线所示，这个加权后值的一个物理含义是，每个字符通过相似度（attention）把自己与当前字符（猫）相关的那一部分信息拿出来。显然，猫自己提供的信息是最多的，但除此之外，其他字符也会根据相关程度提供一部分信息。最后把再把这些信息累加起来作为新的字符向量，这样新的字符向量实际上是包含了句子的全局信息的。上图演示了第一个字符的流程的selfa-atttention流程，以此类推，每个字符都这么操作一遍，就完成了整个self-attention的计算。再次提醒，在实际的代码中，并非一个个字符这样操作的，而是把$(q_1, q_2 \cdots q_3)$当成一个矩阵去计算的（不理解的可以看下这篇文章）。

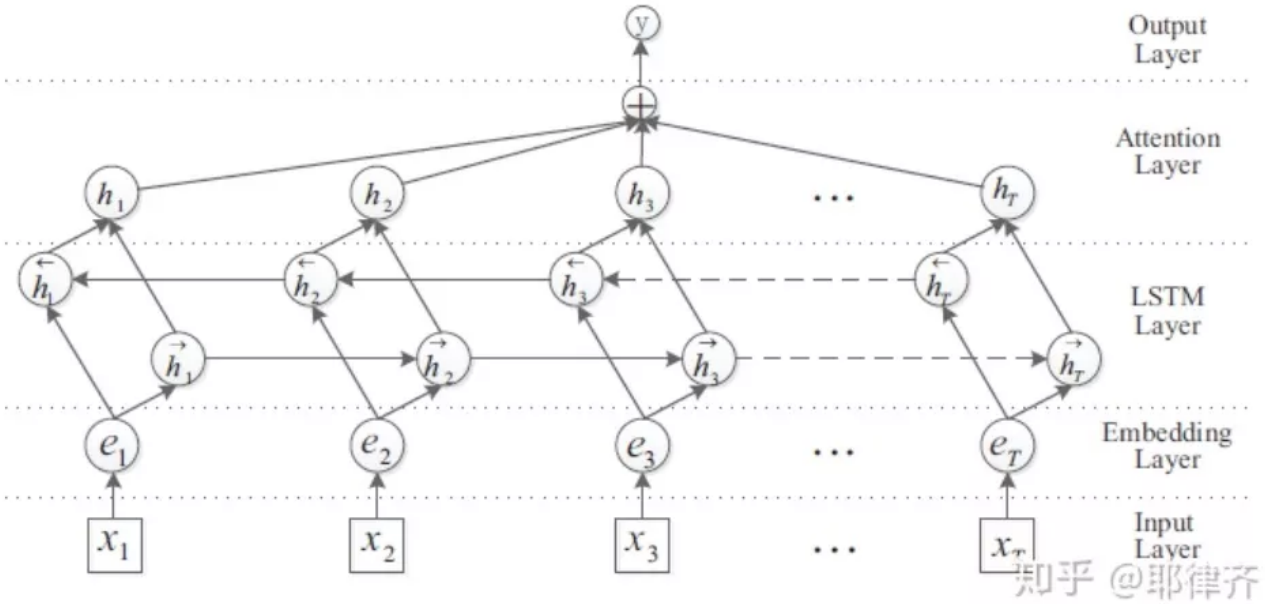
扩展：有没有想过一个问题，为什么采用transformer的BERT被称为mask language model，但是此前的language model却从不进行mask呢？正式transformer中的self-attention利用了

上下文信息对单词进行编码，也就说，如果不对目标字符进行mask，每一个字符上都包含目标字符的信息，这样直接预测会导致信息泄露。另外，如果了解过GPT这个算法的话，你会发现它也用了transformer，但是却没有进行mask操作，可以思考下为什么。

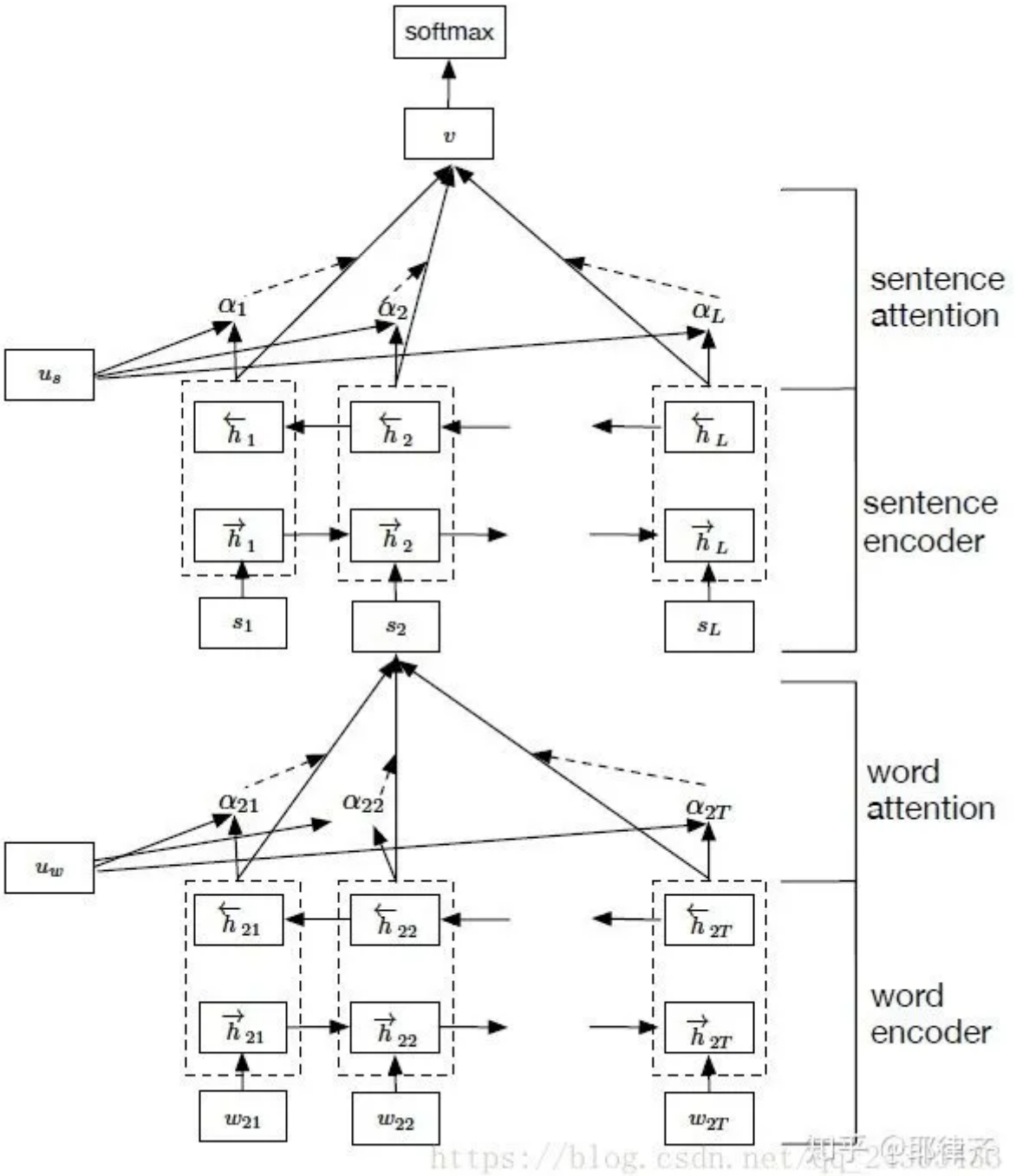
总结，self-attention的self指明它的QKV都来自于输入值（但会分别经过三种不同的线性变换）。

六、LSTM中的attention

在LSTM里用attention，我能找到最早的资料是Attention-Based Bidirectional Long Short-Term Memory Networks for Relation Classification，总之如果你看过这篇论文的话，你应该见过这幅图：



但我认为这副图对于attention的细节展示的不够，不过attention类似用法在Hierarchical Attention Networks for Document Classification里也出现过，只不过后者用了两个attention，一个是单词级的，一个是句子级的，但实质上attention的机制都是一回事。这篇论文的图展示的细节更多：



这两个图的差别在于后者的图在attention的层体现了新参数 u_w （词级别） u_s （句级别）（两者算法上是一致的，只是图画的有差别）。这个新的参数实际上扮演的是 K的角色。在lstm中的attention中，Q和V的角色还是输入值（这里的输入值是bi-lstm的输出：lstm正反向结果的concat），计算权重的过程和上文都是一样的，我这就不重新画图了，唯一的区别就是K，在self-attention里K也是来自于输入值（经过了线性变换），物理意义上一段也解释了，比较明确。但是这里的QK相乘（论文中是： $softmax(u_{it}^{\top}u_w)$ ）是 什么意义呢？之前讨论过两个向量相乘某种意义上是一种相似度，那么这里到底又是计算一种什么样的相似度呢？论文中把K（图中的 u_w 或 u_s ）称为context vector，所以不难猜测它实际上记录了句子的全局信息。所以Q与K的计算相似度的含义也就是在查看哪个step的输出对于整句话更加重要，并把这个值作为attention。看起来也是挺合理的，但为什么 u_w 是个context vector？我觉得是这里的K对每个时间步上的输出都做了一次“交互”（与lstm的输出相乘），又因为它是一个可以训练的参数，虽然我们不知道这个

参数他在优化过程中做了什么调整，但上述attention能够work最好的一种理论值正是u_w中记录了context信息，我认为随着训练的进行它就会做出相应的调整。（当然这种解释是用结论反推的，不知道还有什么更好的解释，同时我也不清楚是否正确，欢迎评论区补充）。

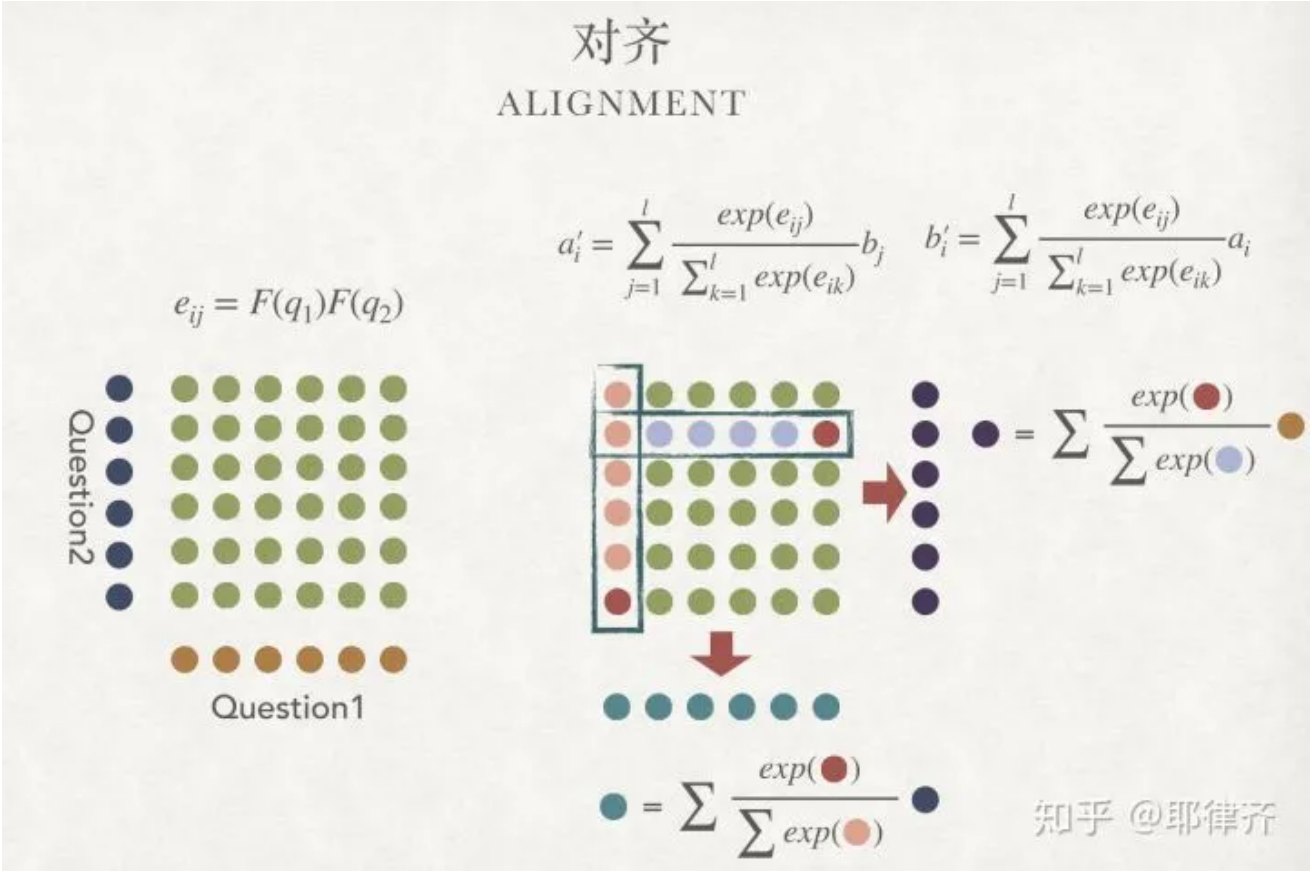
总之lstm的attention，Q与V仍然来自于输入值，而K来自于一个新的可训练参数。

七、文本匹配中的attention

说起深度学习的文本匹配，主要有Siamese Net为代表的特征提取式的匹配，这个就是用encoder分别对两个句子提取特征，然后算个距离。这个方法与本话题无关，不作讨论。另一种是ESIM（Enhanced LSTM for Natural Language Inference）为代表（至少我以它为代表）的交互式匹配。这个交互式，说的其实就是attention。

交互式匹配就是在比较A、B两个句子的时候，提取两个句子的交互特征。所谓交互特征就是说，当提取A句子中 a_1 （表示第一个字）的特征的时候，拿w_1 的词向量去和B中的每个单词的词向量做一次对比，很明显，这里就是在计算attention（相关性）。拿这个attention与B中每个单词的词向量相乘，我们得到结果可以理解为按照与a_1的相关性（attention）我们给B的每个词向量加权，而把这个加权后B中词向量的累加作为a_1与B的交互特征，这个交互特征表明了B句子的语义与a_1最相关的那个部分。

这里给出一个图示，其实过程与我第一篇很类似，细节就不再赘述了。



很明显，交互式匹配中的attention，Q与K分别来自于两个句子，而V呢，在提取A的交互特征的时候，V来自于句子B，提取B的交互特征时则反之。

下载一：中文版！学习TensorFlow、PyTorch、机器学习、深度学习和数据结构五件套！

后台回复【**五件套**】

下载二：南大模式识别PPT