


知乎

首发于[AI机动队](#)

切换模式

 写文章

登录/注册



Attention机制详解（一）——Seq2Seq中的Attention

[川陀学者](#)

为心理史学而奋斗!

1148 人赞同了该文章

Attention模型在机器学习领域越来越得到广泛的应用，准备写一个关于Attention模型的专题，主要分为三个部分：

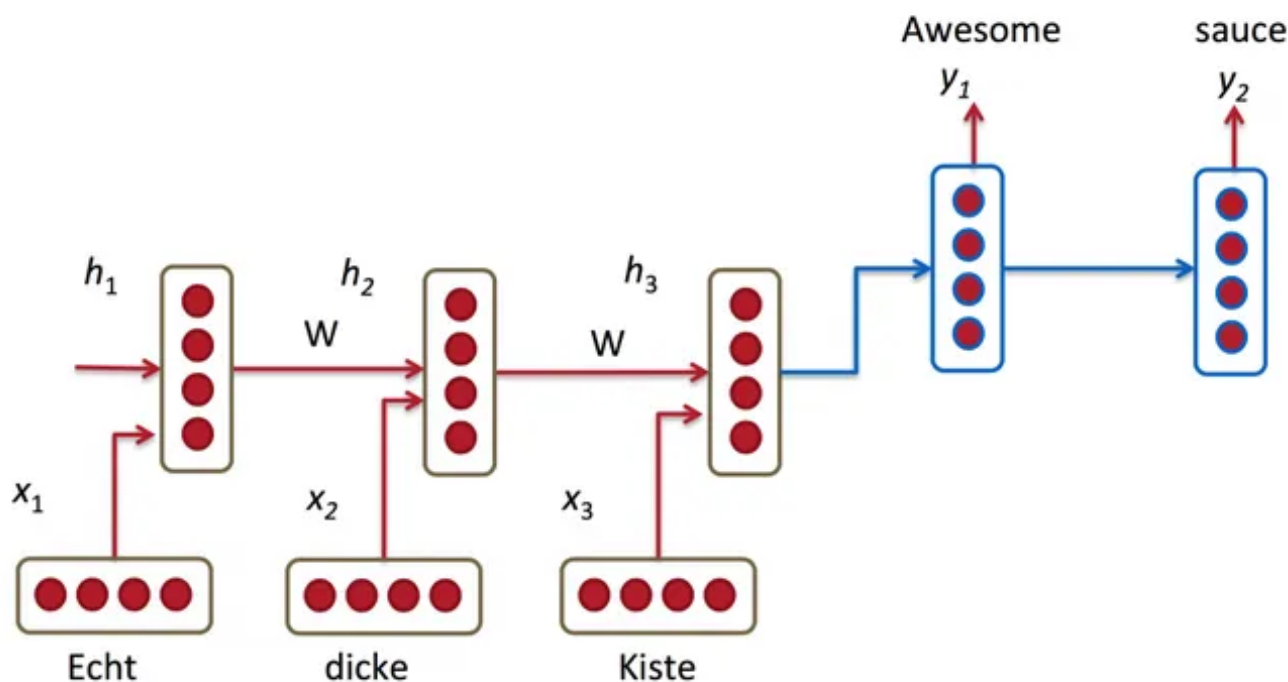
（一）在Seq2Seq问题中RNN与Attention的结合。（二）抛弃RNN的Self-Attention模型以及谷歌的Transformer架构。（三）Attention及Transformer在自然语言处理及图像处理等方面的应用。主要参考资料是Yoshua Bengio组的论文、谷歌研究组的论文及Tensor2Tensor的官方文档、斯坦福自然语言处理相关部分讲义等。

这一篇先来介绍早期的在Machine Translation(机器翻译)中Attention机制与RNN的结合。

RNN结构的局限

机器翻译解决的是输入是一串在某种语言中的一句话，输出是目标语言相对应的话的问题，如将德语中的一段话翻译成合适的英语。之前的Neural Machine Translation(以下简称NMT)模型中，通常的配置是encoder-decoder结构，即

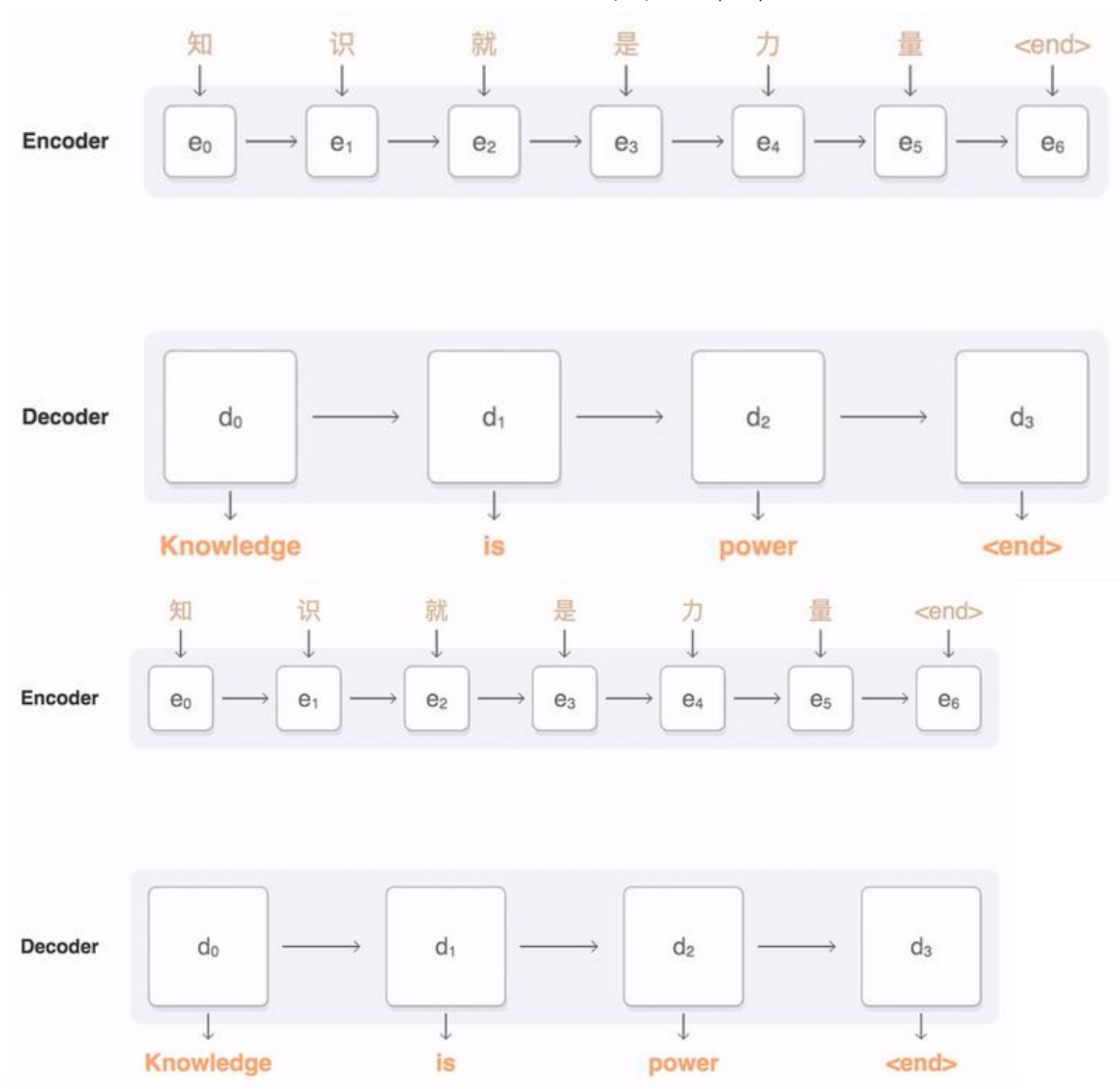
encoder读取输入的句子将其转换为定长的一个向量，然后decoder再将这个向量翻译成对应的目标语言的文字。通常encoder及decoder均采用RNN结构如LSTM或GRU等（RNN基础知识可参考[循环神经网络RNN——深度学习第十章](#)），如下图所示，我们利用encoder RNN将输入语句信息总结到最后一个hidden vector中，并将其作为decoder初始的hidden vector，利用decoder解码成对应的其他语言中的文字。



但是这个结构有些问题，尤其是RNN机制实际中存在长程梯度消失的问题，对于较长的句子，我们很难寄希望于将输入的序列转化为定长的向量而保存所有的有效信息，所以随着所需翻译句子的长度的增加，这种结构的效果会显著下降。

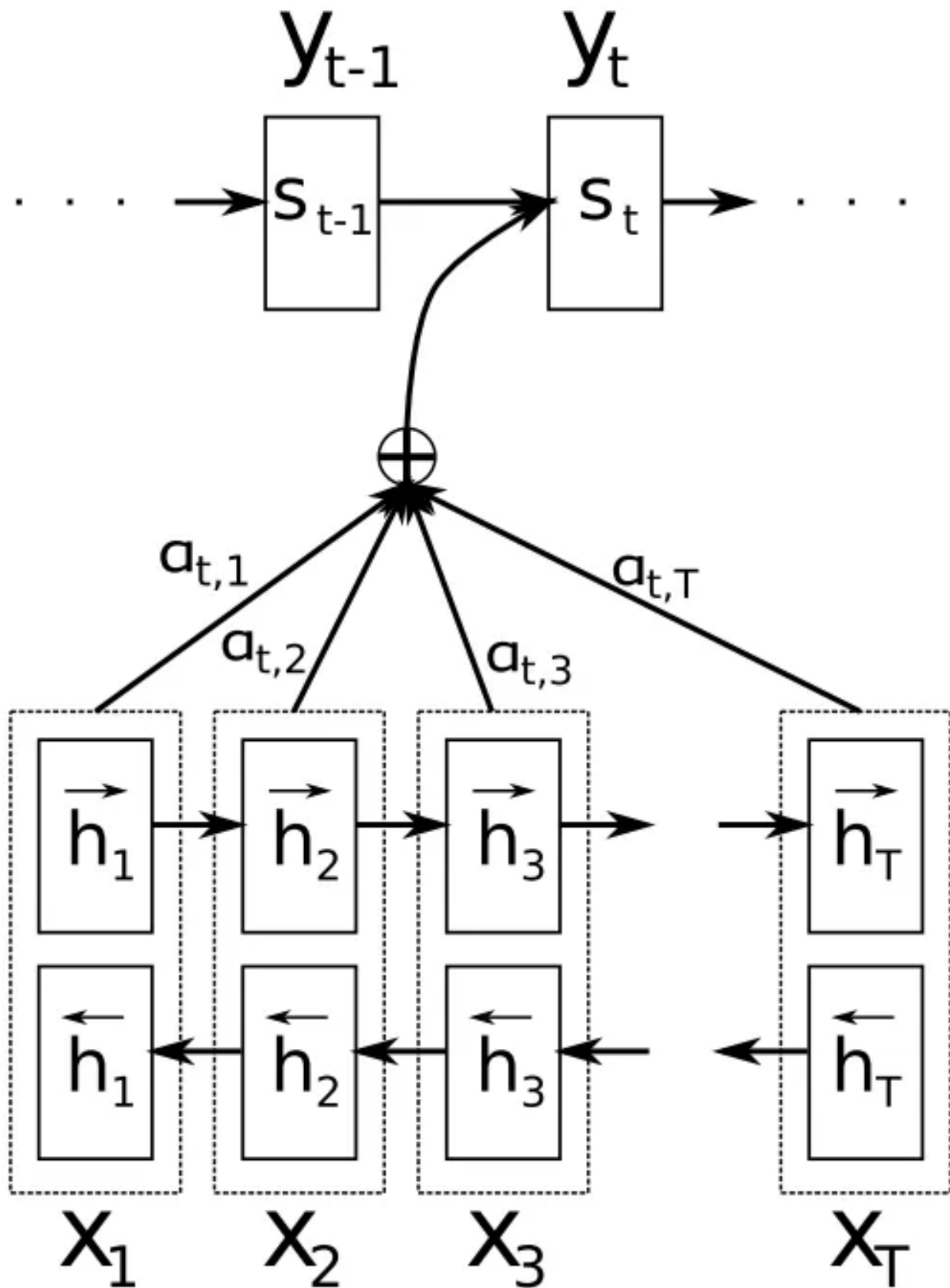
Attention机制的引入

为了解决这一由长序列到定长向量转化而造成的信息损失的瓶颈，Attention注意力机制被引入了。Attention机制跟人类翻译文章时候的思路有些类似，即将注意力关注于我们翻译部分对应的上下文。同样的，Attention模型中，当我们翻译当前词语时，我们会寻找源语句中相对应的几个词语，并结合之前的已经翻译的部分作出相应的翻译，如下图所示，当我们翻译“knowledge”时，只需将注意力放在源句中“知识”的部分，当翻译“power”时，只需将注意力集中在“力量”。这样，当我们decoder预测目标翻译的时候就可以看到encoder的所有信息，而不仅局限于原来模型中定长的隐藏向量，并且不会丧失长程的信息。



GIF

以上是直观理解，我们来详细的解释一下数学上对应哪些运算。



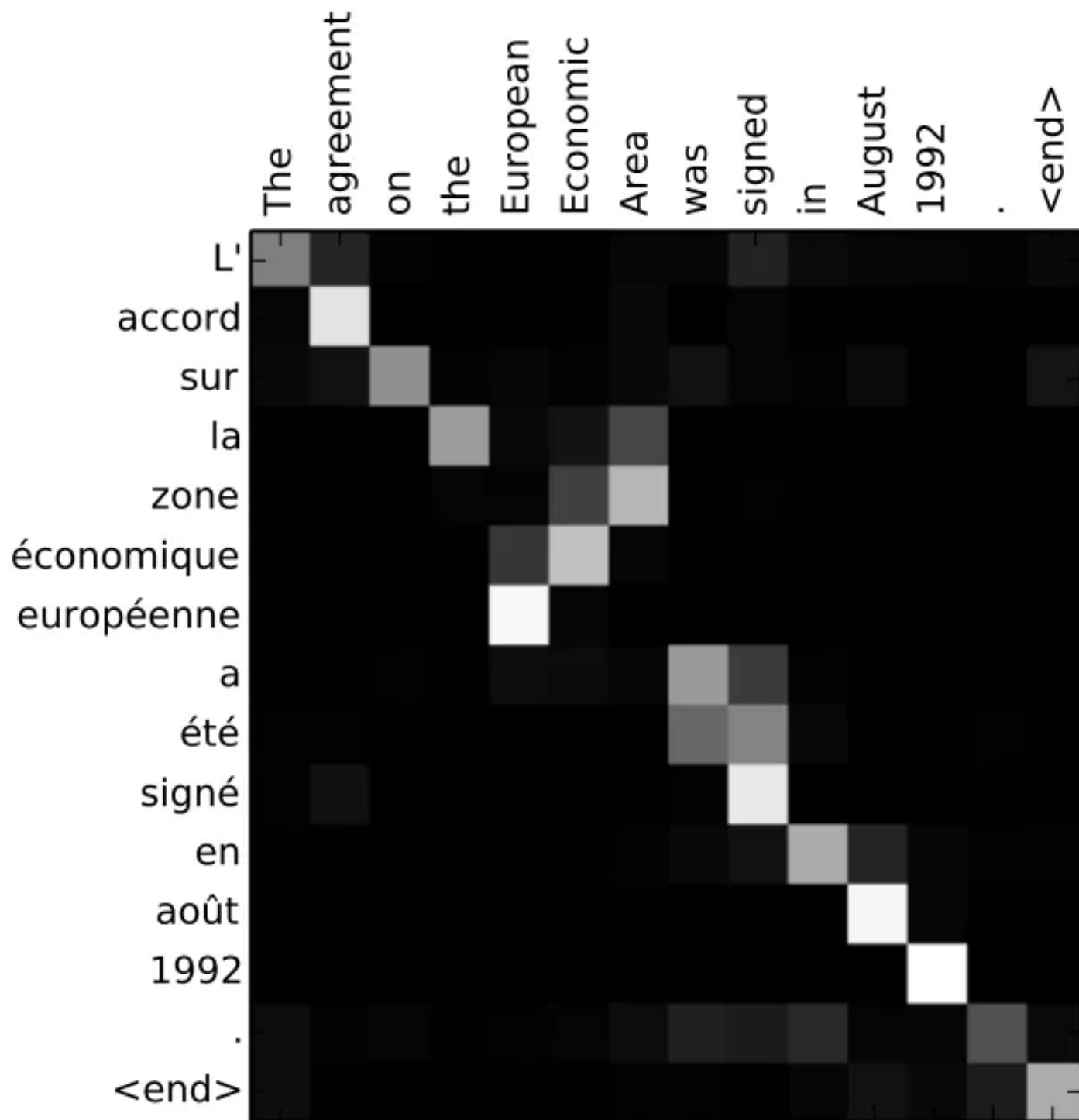
- 首先我们利用RNN结构得到encoder中的hidden state (h_1, h_2, \dots, h_T) ,

- 假设当前decoder的hidden state 是 s_{t-1} ，我们可以计算每一个输入位置j与当前输出位置的关联性，
 $e_{tj} = a(s_{t-1}, h_j)$ ，写成相应的向量形式即为 $\vec{e}_t = (a(s_{t-1}, h_1), \dots, a(s_{t-1}, h_T))$ ，其中 a 是一种相关性的算符，例如常见的有点乘形式 $\vec{e}_t = \vec{s}_{t-1}^T \vec{h}$ ，加权点乘 $\vec{e}_t = \vec{s}_{t-1}^T W \vec{h}$ ，加和 $\vec{e}_t = \vec{v}^T \tanh(W_1 \vec{h} + W_2 \vec{s}_{t-1})$ 等等。
- 对于 \vec{e}_t 进行softmax操作将其normalize得到attention的分布， $\vec{\alpha}_t = \text{softmax}(\vec{e}_t)$ ，展开形式为

$$\alpha_{tj} = \frac{\exp(e_{tj})}{\sum_{k=1}^T \exp(e_{tk})}$$
- 利用 $\vec{\alpha}_t$ 我们可以进行加权求和得到相应的context vector $\vec{c}_t = \sum_{j=1}^T \alpha_{tj} h_j$
- 由此，我们可以计算decoder的下一个hidden state $s_t = f(s_{t-1}, y_{t-1}, c_t)$ 以及该位置的输出 $p(y_t | y_1, \dots, y_{t-1}, \vec{x}) = g(y_{t-1}, s_t, c_t)$ 。

这里关键的操作是计算encoder与decoder state之间的关联性的权重，得到Attention分布，从而对于当前输出位置得到比较重要的输入位置的权重，在预测输出时相应的会占较大的比重。

通过Attention机制的引入，我们打破了只能利用encoder最终单一向量结果的限制，从而使模型可以集中在所有对于下一个目标单词重要的输入信息上，使模型效果得到极大的改善。还有一个优点是，我们通过观察attention 权重矩阵的变化，可以更好地知道哪部分翻译对应哪部分源文字，有助于更好的理解模型工作机制，如下图所示。



当然，一个自然的疑问是，Attention机制如此有效，那么我们可不可以去掉模型中的RNN部分，仅仅利用Attention呢？下一篇会详细解释谷歌在Attention is All you need中提出的self-attention机制及Transformer架构

川陀学者：Attention机制详解 (二) —— Self-Attention与Transformer

2437 赞同 · 102 评论 文章



参考资料：

[1] [CS224n: Natural Language Processing with Deep Learning](#) 斯坦福自然语言处理教程。

[2][Neural Machine Translation by jointly learning to align and translate](#). D Bahdanau, etc.

[3][Visualizing A Neural Machine Translation Model \(Mechanics of Seq2seq Models With Attention\)](#) 可视化 NMT的博客。

[4][Overview - seq2seq](#) TF seq2seq文档。

[5][Attention and Memory in Deep Learning and NLP](#)。

编辑于 2019-04-18 02:33

机器学习 自然语言处理 Attention-based Model

▲赞同 1148

▼

●65 条评论

🔗分享

♥喜欢

★收藏

📄申请转载

...

▲

赞同 1148

🔗

分享

写下你的评论...

65 条评论

默认
最新



[begone](#)

唯一一篇看懂的,谢谢大佬

就像这样直接放公式不好么,csdn的垃圾教程全是类比类比然后显然,我上去就是给他一拳你比你妈呢

2019-10-22

●回复

♥119



[饶锦蒙](#)



暴躁老哥

2021-05-21

●回复

♥5



企鹅宝儿

CSDN 适合本科生，研究生就别看了，里面的内容过于拉垮
2021-10-03

● 回复

♥ 2

查看全部 8 条回复 >



CSdragon

很多人讲attention机制的时候首先弄上公式，其实简单点说attention机制就是把前面的输出连接到后面的输出，通过对两个输出进行一定的计算得到新的输出。先把这点通俗易懂的讲清楚，然后在套公式。这里的公式其实只是解释attention机制行得通的原因，到底神经网络是不是真是按照这些公式来运行还要进一步研究。

2020-11-12

● 回复

♥ 10



徐晓龙

对这里的解码看的不懂

2018-11-18

● 回复

♥ 3



风过天晴



公式的符号含义很多都没解释，不容易看懂。

2021-04-24

●回复 ●❤1



[liuyufengdelaogong](#)

st和yt-1有关系吗，在动图上好像没有体现

2020-02-17

●回复 ●❤3



[Marvel小狗子](#)

上一步的输出yt-1作为t时刻一部分输入求得St

2022-11-21

●回复 ●❤喜欢



[乌鸦](#)



您好，请问那个vt是什么向量呢？

2020-06-10

●回复 ●❤1



[吃酸奶只舔盖](#)

▶

[我会做数列](#)

v_t是一个需要网络学习得到的向量

2021-10-12

●回复 ●❤喜欢



[我会做数列](#)

老哥你现在知道了吗？

2021-10-03

●回复 ●❤喜欢



[Mingzhe](#)



Decode的部分还没有理解

2018-12-11

● 回复 ● 2



[川陀学者](#)

作者

能说一下具体哪里没有理解吗，如果直观理解的话可以看第一个动图。

2018-12-11

● 回复 ● 喜欢



[知乎用户t5UA49](#)

关键的操作是计算encoder与decoder state之间的关联性的权重。这一步不明白是什么意思，要达到什么目的，为什么这样做

2019-09-28

● 回复 ● 1



[小呆瓜](#)

我觉得应该是计算自相关矩阵 $a(h_i, h_j)$ ，你觉得呢

2021-04-09

● 回复 ● 喜欢



[Anticoder](#)

现在用的多的应该是 $e_{tj} = a(s_t, h_j)$ ，这种吧

2019-04-01

● 回复 ● 1



[SKY](#)



你好 讲得非常好 我有两个问题：s和h这样的话纬度必须是一致的吗？还有就是这样attention需要训练的weight是什么呢？我的理解是我们需要先将s和h通过不同的w映射到相同纬度分别为query和key 之后做相似度计算 你觉得我的理解是否正确？谢谢

2018-12-25

● 回复 ● 1



[ZUZU](#)

s和h不一定要一样维度的，只要能表示出s和h的相似度就可以。

只不过一般经常把s和h放到同一个维度下，方便运算

2020-11-09

●回复

♥喜欢



[akaace](#)

►
[SCLVBW](#)

weight1和weight2本身就在整个模型里，最后根据loss进行反向传播，梯度下降即可。

2020-11-01

●回复

♥喜欢

展开其他 1 条回复 ►



[Joshua](#)

点赞！想知道怎么讲权重可视化...

2018-12-18

●回复

♥1



[海东青](#)

►
[UncleRudy](#)

有讲权重的可视化吗

2023-01-30

●回复

♥喜欢



[UncleRudy](#)

权重的理解可以参考李宏毅讲的attention