NLP中的Attention机制介绍

李世杰

内容

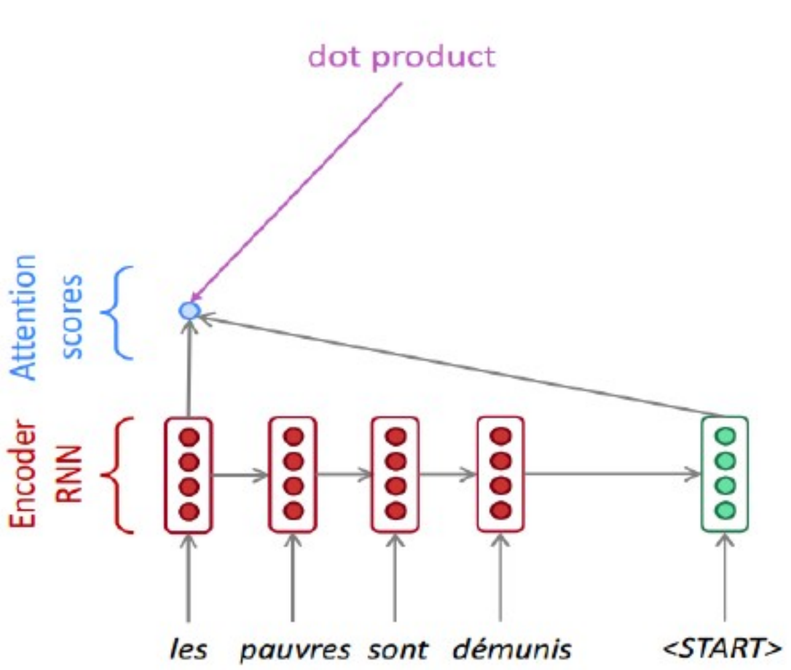
复习

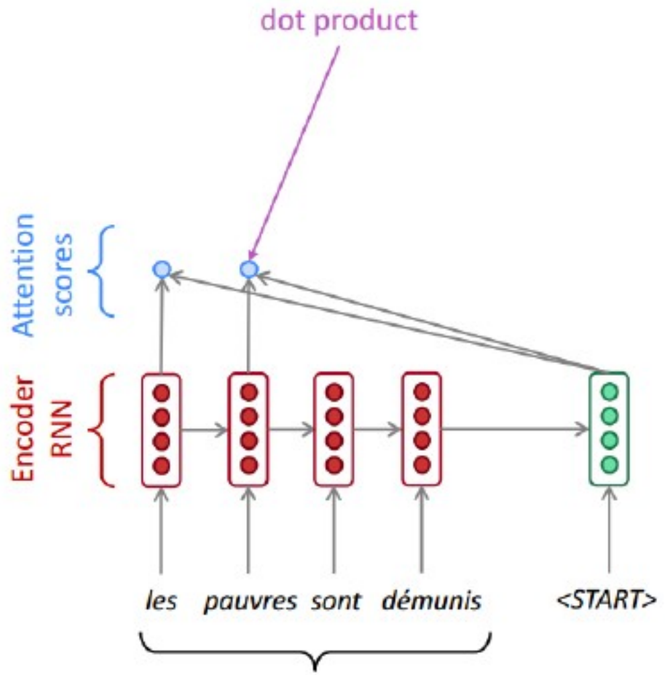
Attention机制通用定义

Attention score的计算变体

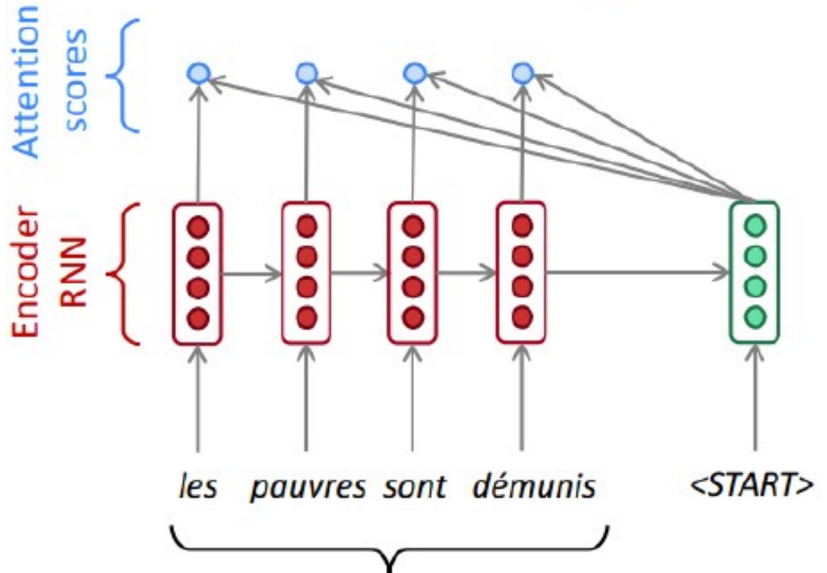
更多attention种类

总结



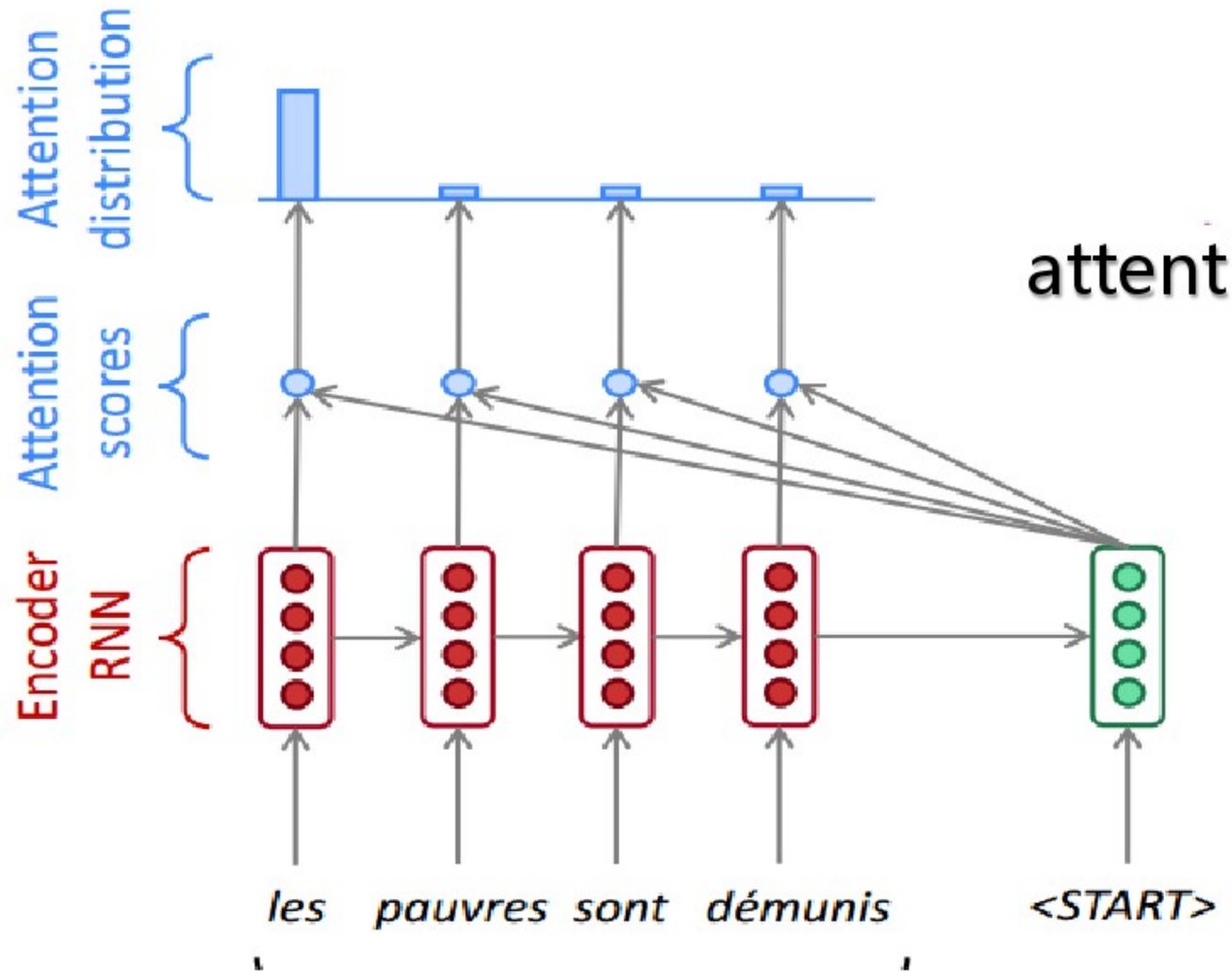


$$h_1, \ldots, h_N \in \mathbb{R}^h$$



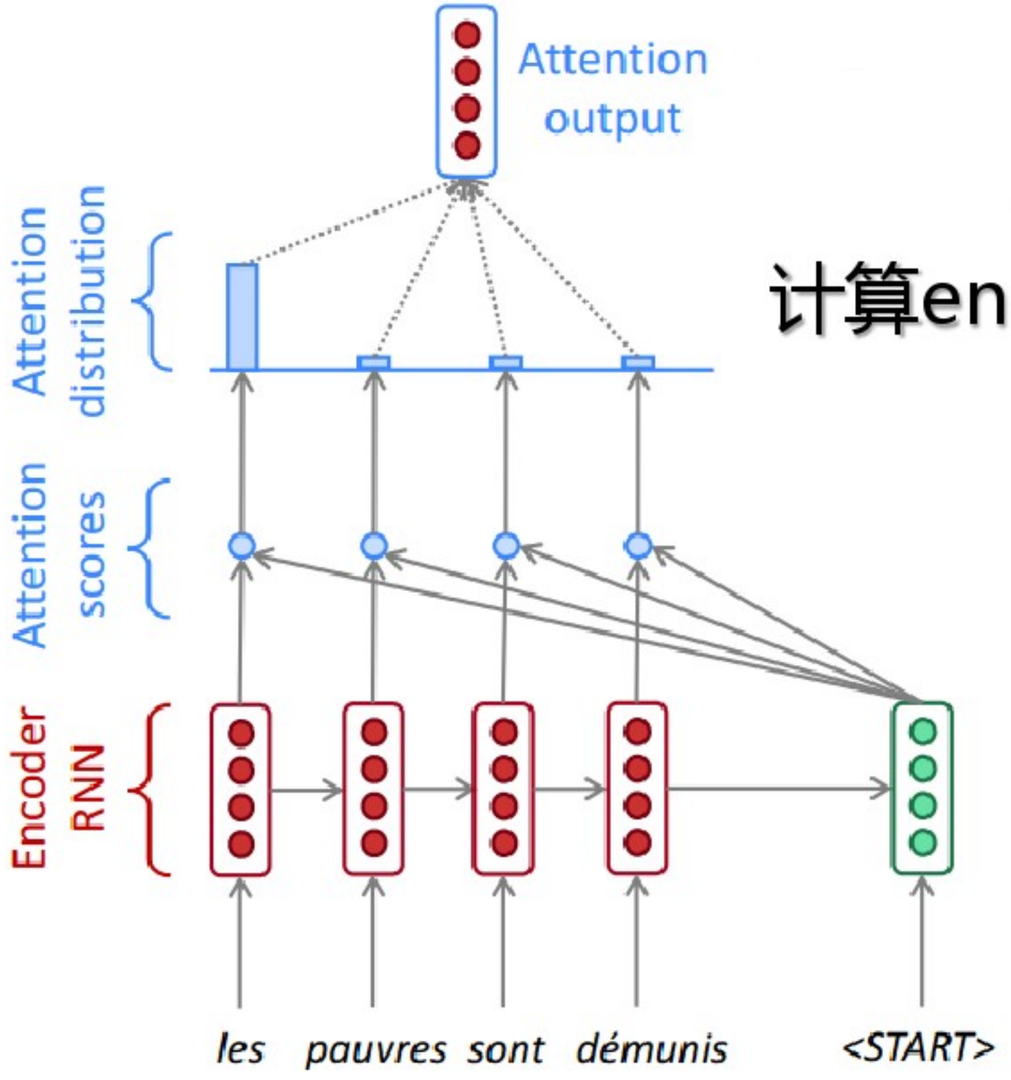
dot得到attention score

$$= [s_t^T h_1, \ldots, s_t^T h_N]$$



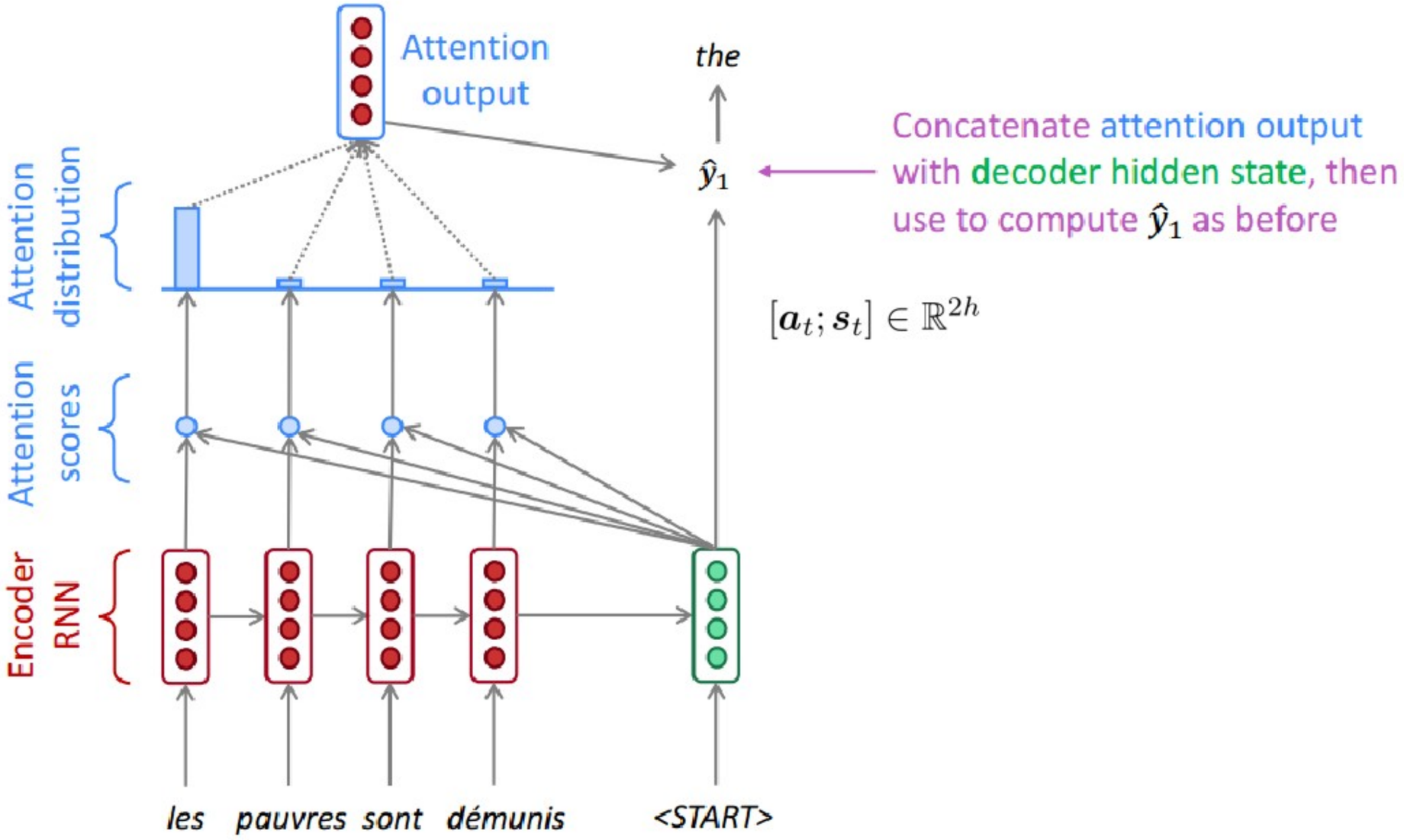
利用softmax函数：attention scores转化为概率分布

$$\alpha^t = \text{softmax}(\boldsymbol{e}^t) \in \mathbb{R}^N$$



按照上一步概率分布：
计算encoder的hidden states的加权求和

$$\boldsymbol{a}_t = \sum_{i=1}^{N} \alpha_i^t \boldsymbol{h}_i \in \mathbb{R}^h$$


Attention output
the
$\hat{y}_1$
Concatenate attention output with decoder hidden state, then use to compute $\hat{y}_1$ as before
$[a_t; s_t] \in \mathbb{R}^{2h}$
Attention distribution
Attention scores
Encoder RNN
les
pauvres
sont
démunis
<START>

内容

复习

Attention机制通用定义

Attention score的计算变体

更多attention种类

总结

Attention机制的一个更加通用的定义（课程里面的定义）

- 给定一组向量集合*values*，以及一个向量*query*，attention机制是一种根据该*query*计算*values*的加权求和的机制。
- attention的重点就是这个集合*values*中的每个*value*的“权值”的计算方法。
- 有时候也把这种attention的机制叫做query的输出关注了（或者说叫考虑到了）原文的不同部分。（Query attends to the values）

举例：seq2seq中，哪个是query，哪个是values？

从定义来看Attention的感性认识

- The weighted sum is a selective summary of the information contained in the values, where the query determines which values to focus on.

- 换句话说，attention机制也是一种根据一些其他向量表达（query）从向量表达集合（values）中获得特定向量表达（attention）的方法

内容

复习

Attention机制通用定义

Attention score的计算变体

更多attention种类

总结

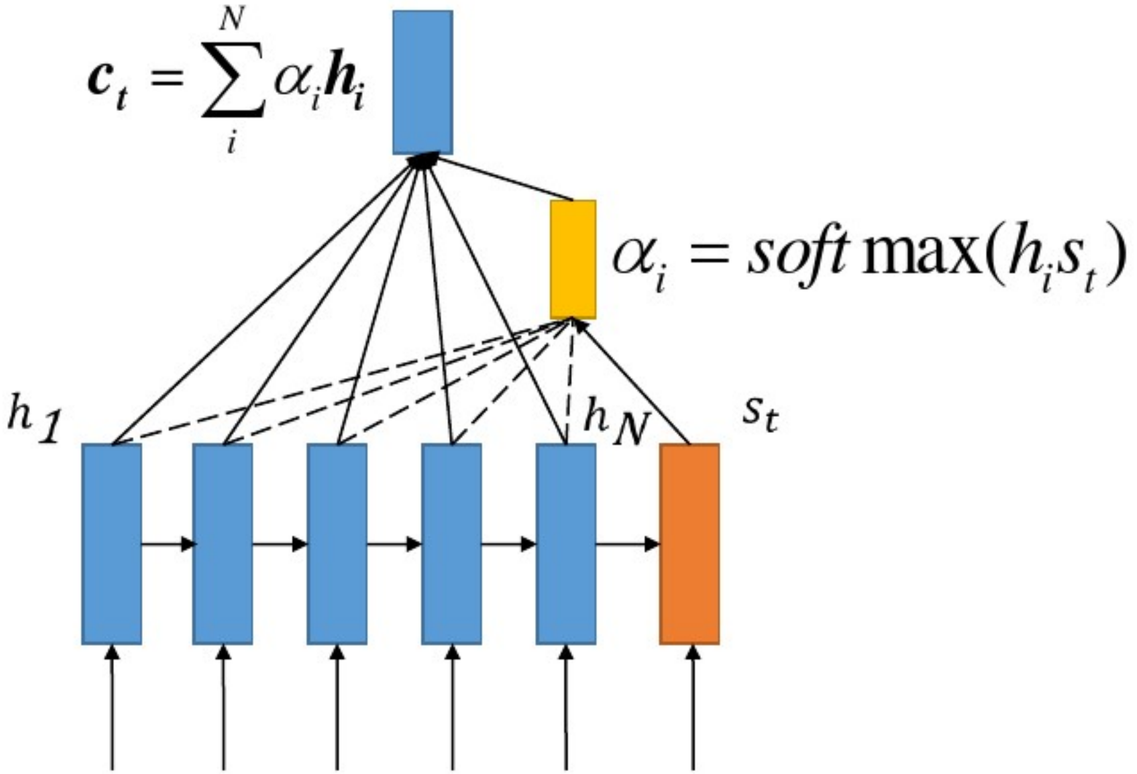
针对attention向量计算方式变体

- Soft attention
- Hard attention
- “半软半硬”的attention（local attention）

- 动态attention
- 静态attention
- 强制前向attention

Soft attention

Soft attention就是我们上面讲过的那种最常见的attention，是在求注意力分配概率分布的时候，对于输入句子X中任意一个单词都给出个概率，是个概率分布

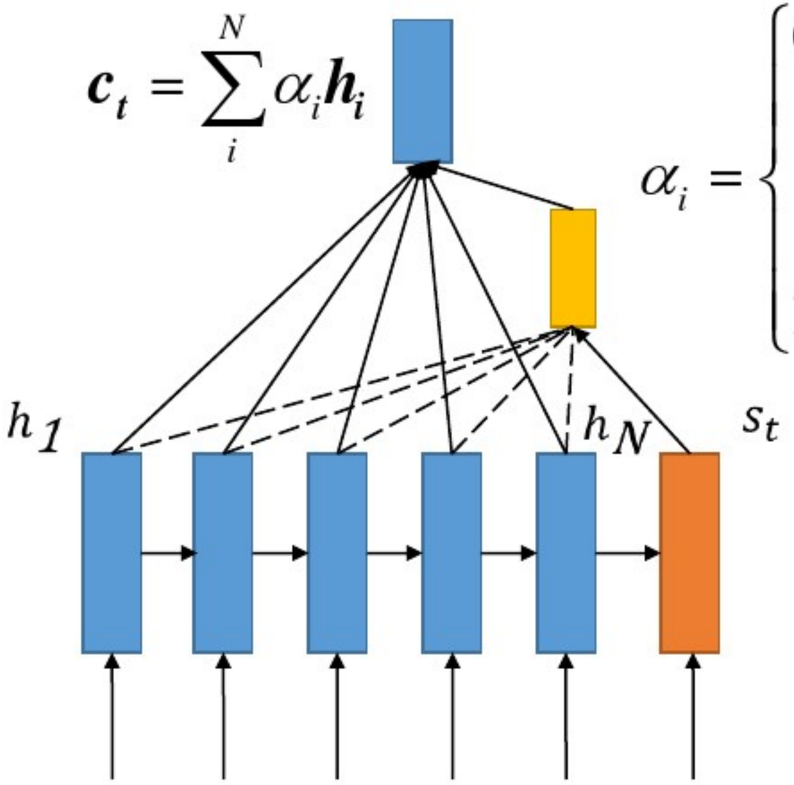


Hard attention

Soft是给每个单词都赋予一个单词match概率，那么如果不这样做，直接从输入句子里面找到某个特定的单词，然后把目标句子单词和这个单词对齐，而其它输入句子中的单词硬性地认为对齐概率为0，这就是Hard Attention Model的思想。

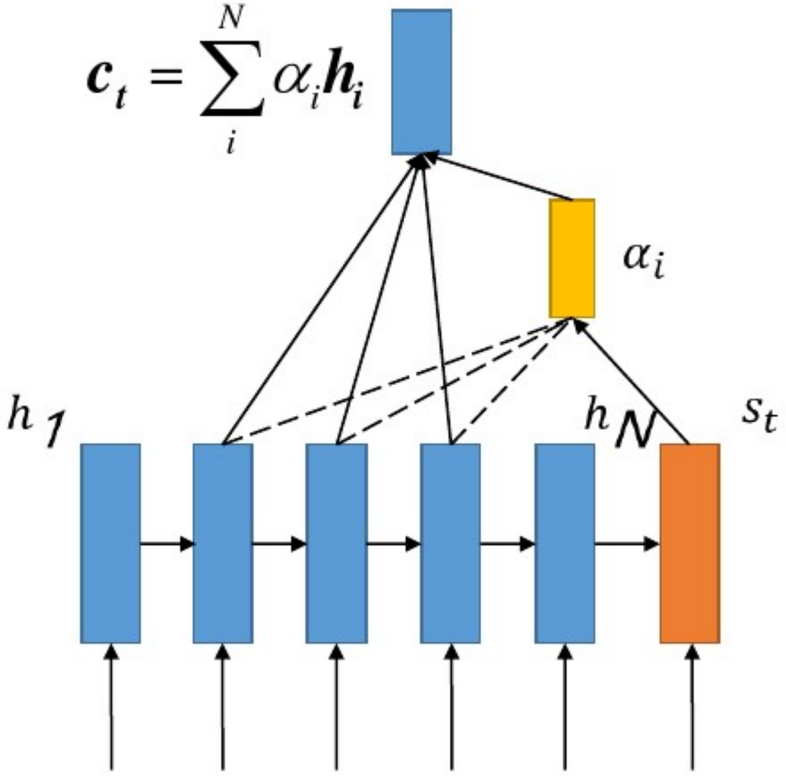
$$\boldsymbol{c}_t = \sum_i^N \alpha_i \boldsymbol{h}_i$$

$$\alpha_i = \begin{cases} 0 & \text{if} \quad \text{i} = pt \\ 1 & else \end{cases}$$



local attention（半软半硬attention）

在这个模型中，对于是时刻t的每一个目标词汇，模型首先产生一个对齐的位置 pt（aligned position），context vector 由编码器中一个集合的隐藏层状态计算得到，编码器中的隐藏层包含在窗口[pt-D,pt+D]中，D的大小通过经验选择。



寻找pt并计算alpha的方式又大致分为两种：

Local – m：假设对齐位置就是pt = t （线性对齐）

然后计算窗口内的softmax，窗口外的alpha可以取0

$$\alpha_i = \frac{exp(score(\tilde{h}_i, s_t))}{\sum_i exp(score(\tilde{h}_i, s_t))}$$

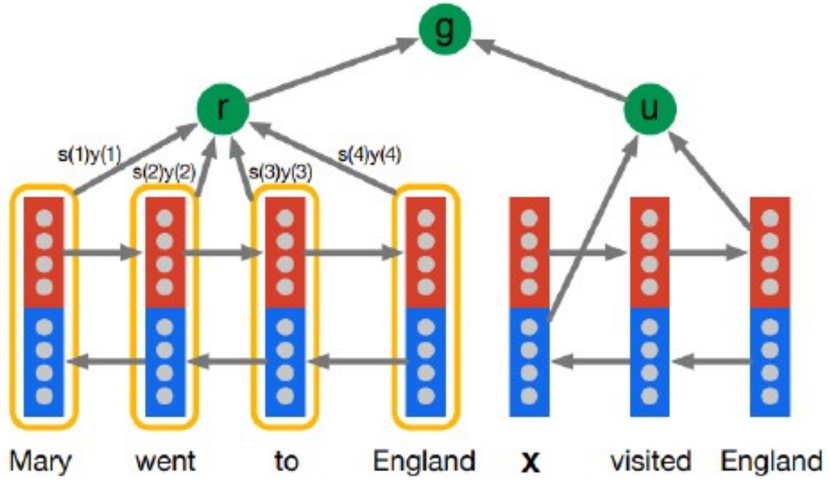
Local – p：先通过一个函数预测pt在[0,S]之间，然后取一个类高斯分布乘以softmax。

$$pt = S \cdot sigmoid(v_p^T tanh(W_p s_t))$$

$$\alpha_t = \frac{exp(score(\tilde{h}_i, s_t))}{\sum_i exp(score(\tilde{h}_i, s_t))} exp(-\frac{(s - pt)}{2\sigma^2})$$

动态attention、静态attention、强制前向attention

- 动态attention：就是softmax attention
- 静态attention：对输出句子共用一个St的attention就够了，一般用在Bilstm的首位hidden state输出拼接起来作为St（如图所示中的u）
- 强制前向attention：要求在生成目标句子单词时，如果某个输入句子单词已经和输出单词对齐了，那么后面基本不太考虑再用它了



$$e'_{ti} = \begin{cases} exp(e_{ti}) & \text{if } t = 1 \\ \frac{exp(e_{ti})}{\sum_{j=1}^{t-1} \exp(e_{ji})} & \text{otherwise} \end{cases}$$

针对Attention score的计算方式变体

- 已有 $\boldsymbol{h}_1, \ldots, \boldsymbol{h}_N \in \mathbb{R}^{d_1}$ 的情况下，计算query $\boldsymbol{s} \in \mathbb{R}^{d_2}$ 的attention向量a（很多时候也称作上下文向量，context vector）使用的公式为：

$$\alpha = \text{softmax}(\boldsymbol{e}) \in \mathbb{R}^N \qquad \text{(take softmax)}$$

$$\boldsymbol{a} = \sum_{i=1}^{N} \alpha_i \boldsymbol{h}_i \in \mathbb{R}^{d_1} \qquad \text{(take weighted sum)}$$

- 点积 attention score
（Basic dot-product attention）：

这个就是我们常见的attention score计算方式

$$\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{h}_i \in \mathbb{R}$$

- 乘法 attention score
（Multiplicative attention）：

$$\boldsymbol{e}_i = \boldsymbol{s}^T \boldsymbol{W} \boldsymbol{h}_i \in \mathbb{R}$$

- 加法 attention score

（Additive attention:

$$\boldsymbol{e}_i = \boldsymbol{v}^T \tanh(\boldsymbol{W}_1 \boldsymbol{h}_i + \boldsymbol{W}_2 \boldsymbol{s}) \in \mathbb{R}$$

内容

复习

Attention机制通用定义

Attention score的计算变体

更多attention种类

总结

特殊的attention score计算方式

- Self attention
- 思想：Self attention也叫做intra-attention在没有任何额外信息的情况下，我们仍然可以通过允许句子使用self attention机制来处理自己，从句子中提取关注信息。
- 它在很多任务上都有十分出色的表现，比如阅读理解 (Cheng et al., 2016)、文本继承 (textual entailment/Parikh et al., 2016)、自动文本摘要 (Paulus et al., 2017)。

Self attention计算方式

1.以当前的隐藏状态去计算和前面的隐藏状态的得分

$$e_{h_i} = h_t^T W h_i$$

2.以当前状态本身去计算得分，这种方式更常见，也更简单

$$e_{h_i} = v_a^T \tanh(W_a h_i)$$

$$e_{h_i} = \tanh(w^T h_i + \mathrm{b})$$

Self attention计算方式

- 针对第二种计算方式，其又有矩阵的变形，令矩阵
- H=[h1，h2，h3，h4……hn]∈nx2u表示句子的隐藏状态矩阵，每个隐藏状态为2u维。
- 那么上面的公式矩阵化之后就是：

$$A=softmax(V_a \tanh(W_a H^T))$$

$$C = AH$$

另一种特殊的attention score计算方式

- Key-value attention
- Key-value attention 是将hi拆分成了两部分[key_i;value_i]，然后使用的时候只针对key部分计算attention权重，然后加权求和的时候只使用value部分进行加权求和。公式如下，权重计算：

$$\begin{bmatrix} \boldsymbol{k}_t \\ \boldsymbol{v}_t \end{bmatrix} = \boldsymbol{h}_t \qquad \in \mathbb{R}^{2k}$$

$$\boldsymbol{M}_t = \tanh(\boldsymbol{W}^Y [\boldsymbol{k}_{t-L} \ \cdots \ \boldsymbol{k}_{t-1}] + (\boldsymbol{W}^h \boldsymbol{k}_t)\mathbf{1}^T) \qquad \in \mathbb{R}^{k \times L}$$

$$\boldsymbol{\alpha}_t = \mathrm{softmax}(\boldsymbol{w}^T \boldsymbol{M}_t) \qquad \in \mathbb{R}^{1 \times L}$$

$$\boldsymbol{r}_t = [\boldsymbol{v}_{t-L} \ \cdots \ \boldsymbol{v}_{t-1}]\boldsymbol{\alpha}^T \qquad \in \mathbb{R}^{k}$$

$$\boldsymbol{h}_t^* = \tanh(\boldsymbol{W}^r \boldsymbol{r}_t + \boldsymbol{W}^x \boldsymbol{v}_t) \qquad \in \mathbb{R}^{k}$$

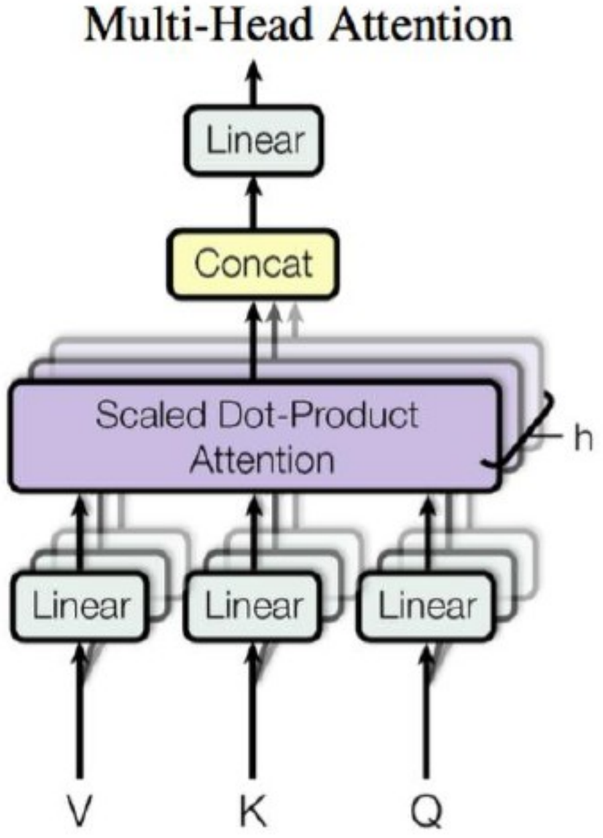
Attention is all you need 中的multi-head attention部分

Scaled dot-product attention

$$Attention(\boldsymbol{Q},\boldsymbol{K},\boldsymbol{V}) = softmax\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d_k}}\right)\boldsymbol{V}$$

不妨从一个向量出发

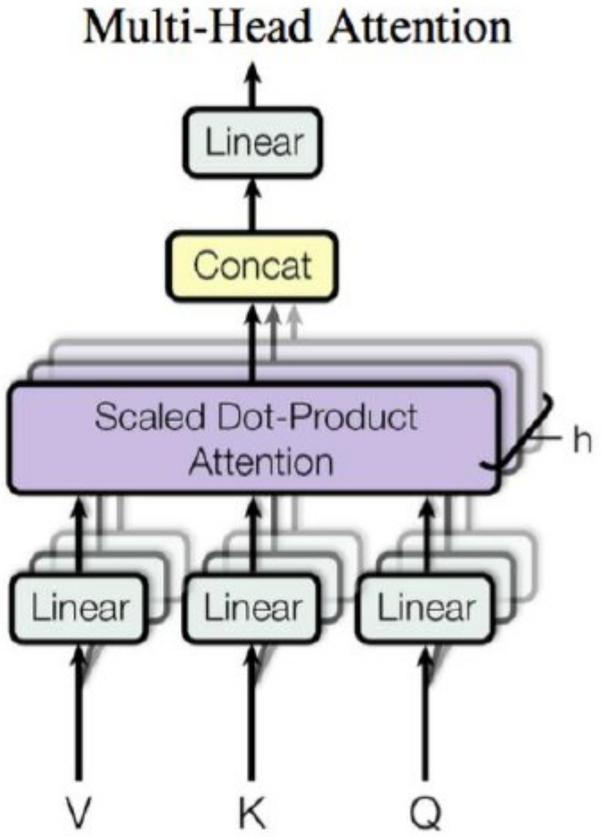
$$Attention(\boldsymbol{q}_t,\boldsymbol{K},\boldsymbol{V}) = \sum_{s-1}^{m}\frac{1}{Z}\exp\left(\frac{\langle \boldsymbol{q}_t,\boldsymbol{k}_s\rangle}{\sqrt{d_k}}\right)\boldsymbol{v}_s$$



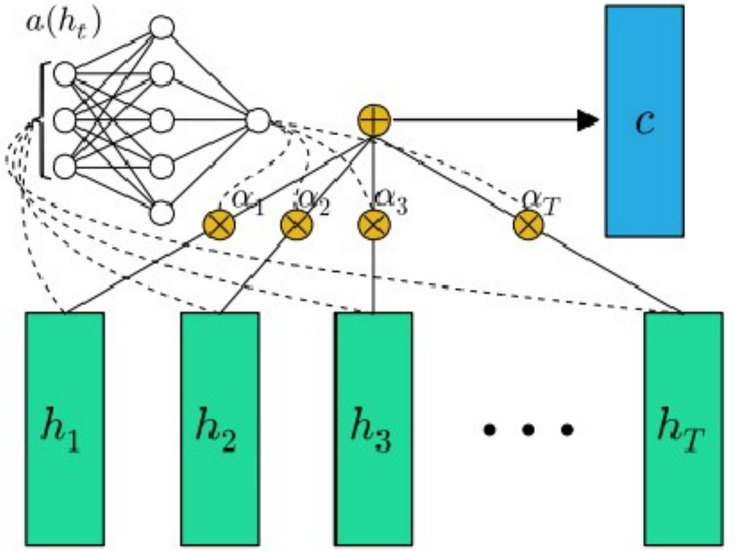
Attention is all you need 中的multi-head attention部分

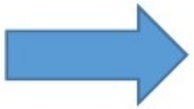
$$head_i = Attention(\boldsymbol{Q}\boldsymbol{W}_i^Q, \boldsymbol{K}\boldsymbol{W}_i^K, \boldsymbol{V}\boldsymbol{W}_i^V)$$

$$MultiHead(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = Concat(head_1, \ldots, head_h)$$



代码实现





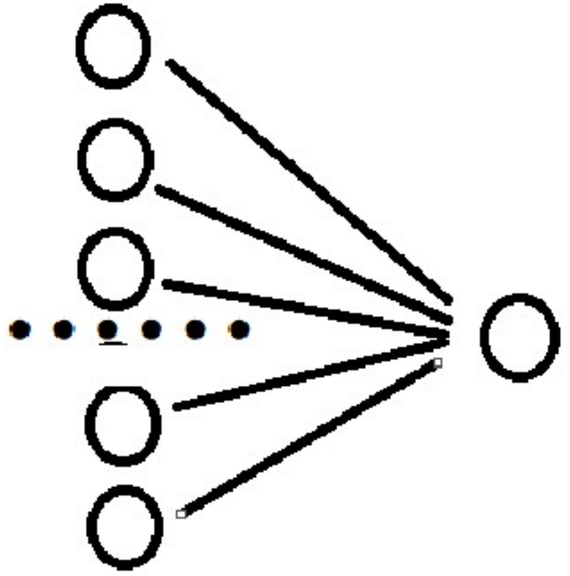


Figure 1: Schematic of our proposed "feed-forward" attention mechanism (cf. (Cho, 2015) Figure 1). Vectors in the hidden state sequence h_t are fed into the learnable function $a(h_t)$ to produce a probability vector α. The vector c is computed as a weighted average of h_t, with weighting given by α.

1.2 FEED-FORWARD ATTENTION

A straightforward simplification to the attention mechanism described above which would allow it to be used to produce a single vector c from an entire sequence could be formulated as follows:

$$e_t = a(h_t), \alpha_t = \frac{\exp(e_t)}{\sum_{k=1}^{T} \exp(e_k)}, c = \sum_{t=1}^{T} \alpha_t h_t \quad (1)$$

总结

- Attention机制就是一个加权求和机制
- Attention的权重由当前的hidden state和需要计算的hidden state通过一定的方式先判断出score得分然后再softmax得到。
- Attention可以用来干什么：主要是关注到长序列中的关键信息。