第 14 章　NLP中的注意力机制

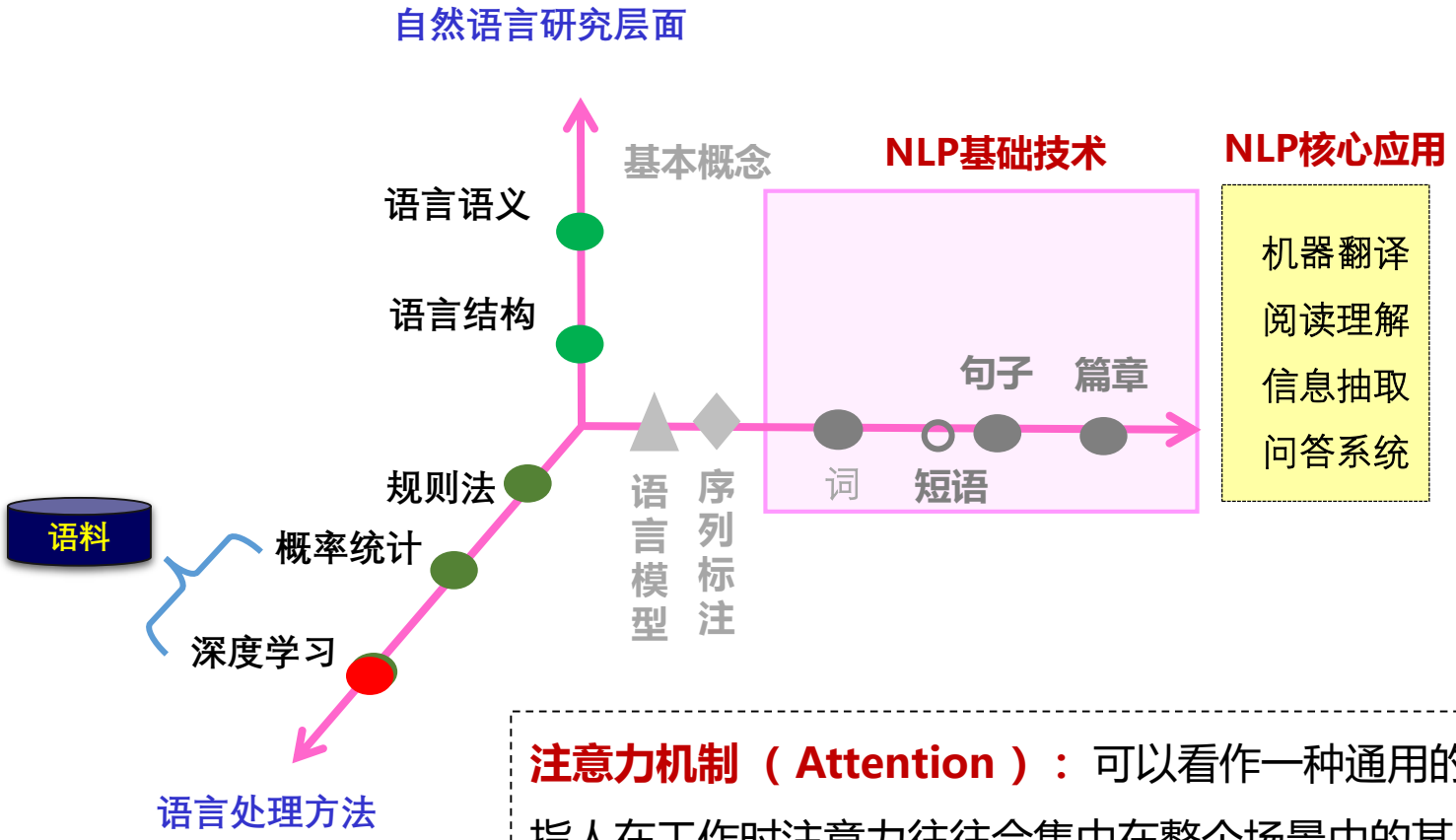
中科院信息工程研究所第二研究室

胡玥

huyue@iie.ac.cn

自然语言处理课程内容及安排

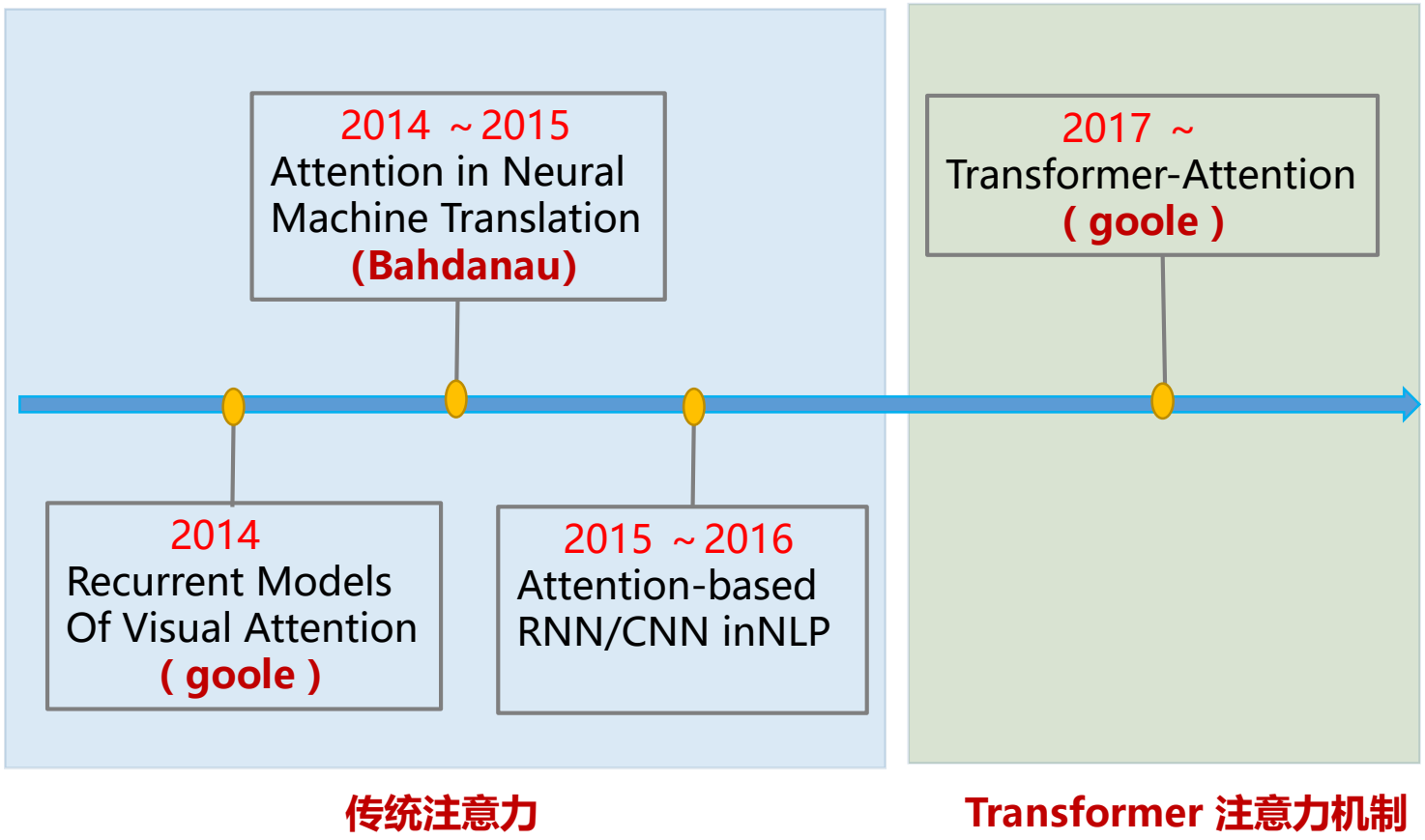
✧ **课程内容：**



注意力机制（Attention）： 可以看作一种通用的思想，指人在工作时注意力往往会集中在整个场景中的某些焦点上

概述

注意力机制发展历史：



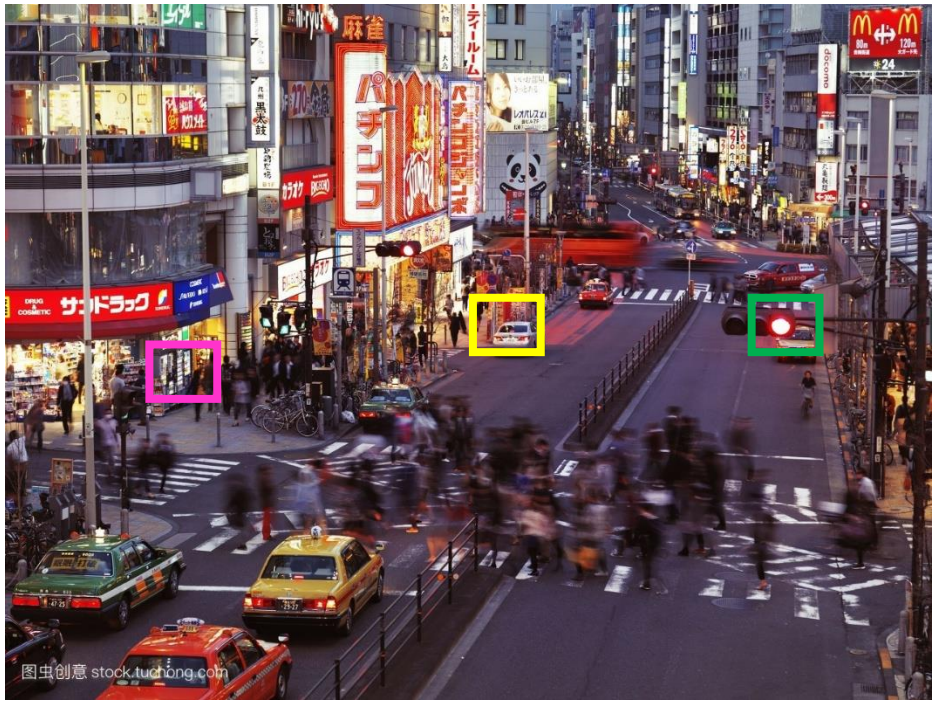
内 容 提 要

14.1 传统注意力机制

14.2 Transformer 注意力机制

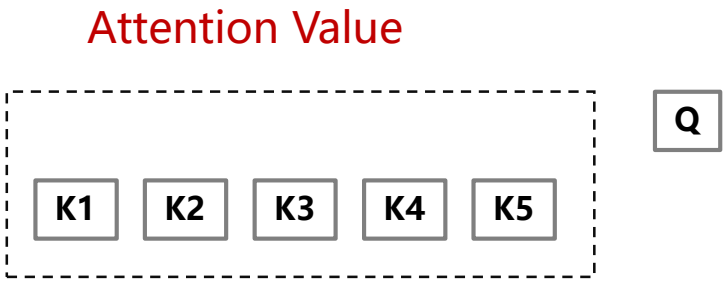
14.1 传统注意力机制

注意力机制（Attention）的本质来自于人类视觉注意力机制。人们视觉在感知东西的时候一般不会是Attention一个场景从到头看到尾每次全部都看，而往往是根据需求观察注意特定的一部分。而且当人们发现一个场景经常在某部分出现自己想观察的东西时，人们会进行学习在将来再出现类似场景时把注意力放到该部分上。



14.1 传统注意力机制

如：



如要形成Q对所有K的 Attention Value （Att-V）：

方法1：

Att-V=K1+K2+K3+K4+K5　不足：各个K权重一样， 不能很好刻画问题

方法2：

Att-V= $a1$ x K1+$a2$ x K2+$a3$ x K3+$a4$ x K4+$a5$ x K5　（a_i表示K_i对Q的权重）

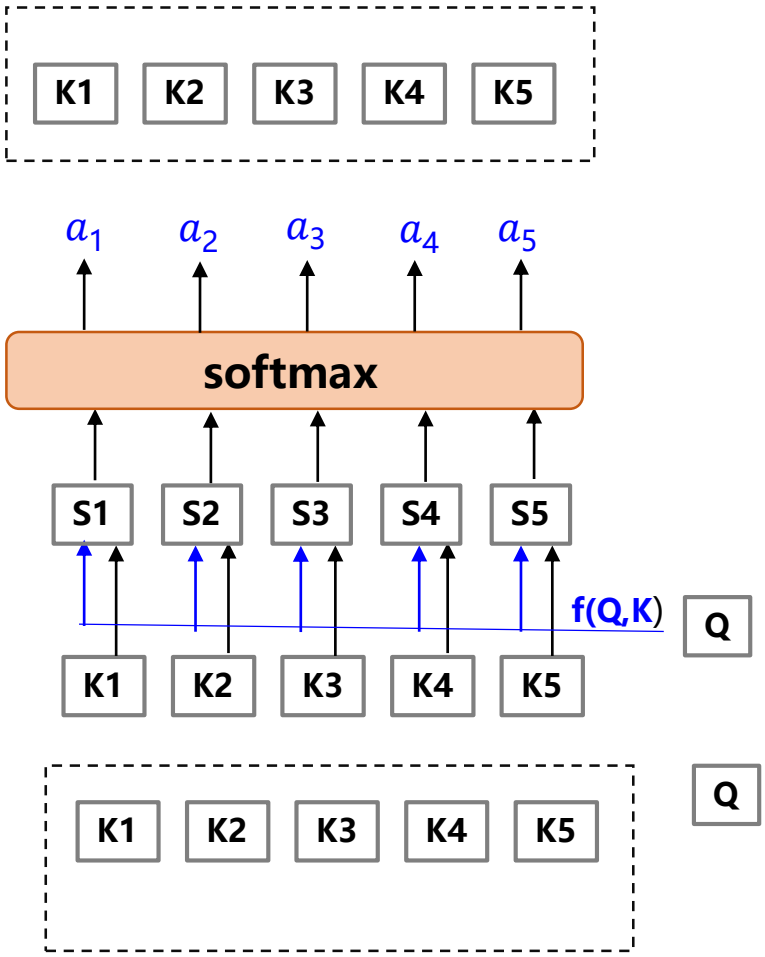
能更准确刻画问题

问题：如何求权重 a_i ？

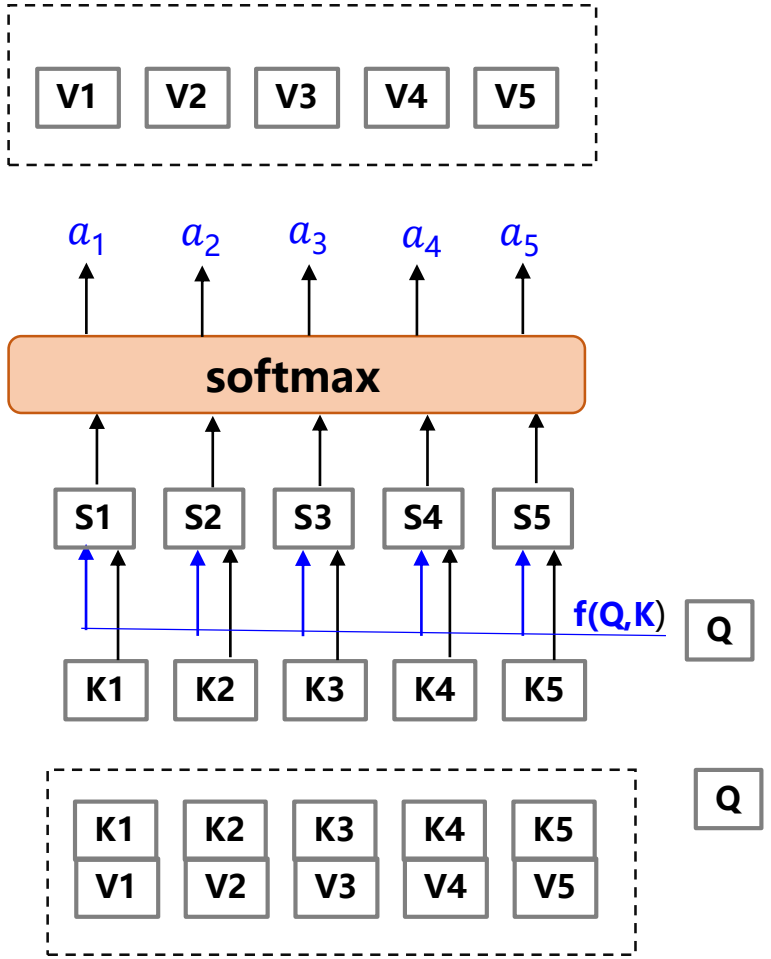
14.1 传统注意力机制

$$\text{Attention}(Q,K,V)=\sum_i ai Vi$$

Att-V = $a1$ x K1 + $a2$ x K2 + $a3$ x K3 + $a4$ x K4 + $a5$ x K5

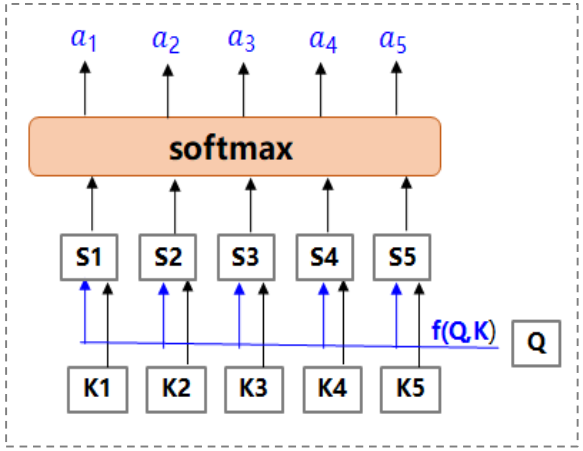


Att-V = $a1$ x V1 + $a2$ x V2 + $a3$ x V3 + $a4$ x V4 + $a5$ x V5



14.1 传统注意力机制

注意力打分函数 f(Q,K)

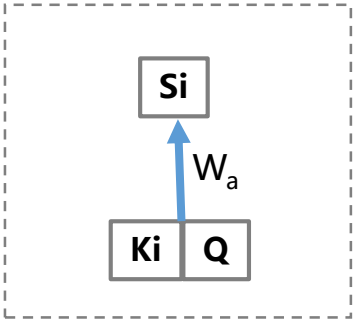


$$f(Q,K)=\begin{cases} Q^T K_i & dot \\ Q^T W_a K_i & general \\ W_a[Q,K_i] & concat \\ V^T_a \tanh(W_a Q+U_a K_i) & perceptron \end{cases}$$

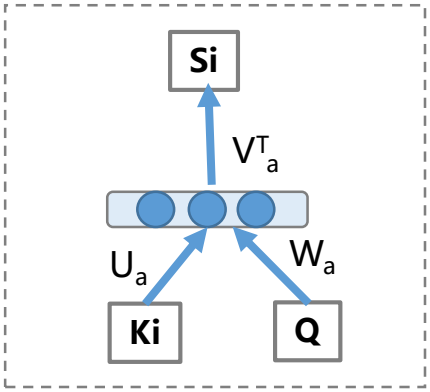
乘法模型

加法模型

$W_a[Q,K_i]$



$V^T_a \tanh(W_a Q+U_a K_i)$



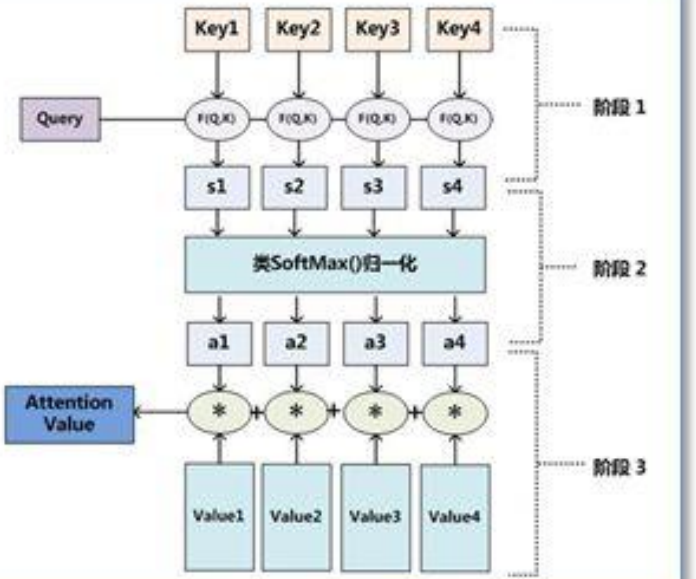
14.1 传统注意力机制

Attention 模型

$$f(Q,K_i)=\begin{cases} Q^T K_i & dot \\ Q^T W_a K_i & general \\ W_a[Q;K_i] & concat \\ v_a^T \tanh(W_a Q + U_a K_i) & perceptron \end{cases}$$

$$a_i = soft\max(f(Q,K_i)) = \frac{\exp(f(Q,K_i))}{\sum_j \exp(f(Q,K_j))}$$

$$\text{Attention}(Q,K,V) = \sum_i a_i V_i$$

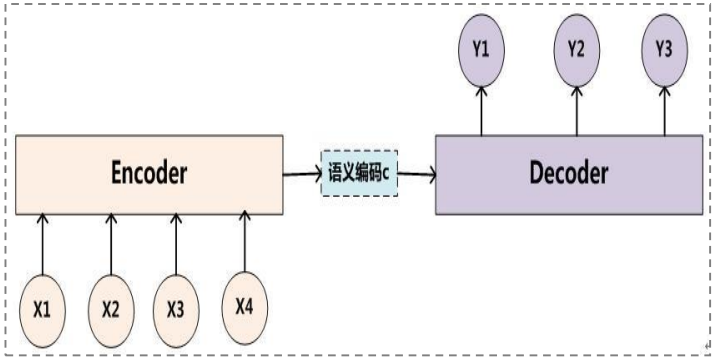


14.1 传统注意力机制

传统 Attention 应用

Encoder-Decoder 框架 RNN回顾:

Encoder和Decoder一般采用RNN模型，对于句子比较长的情形，LSTMGRU模型效果要明显优于RNN模型。但当句子长度超过 30 以后，LSTM模型的效果会急剧下降，一般此时会引入Attention模型。

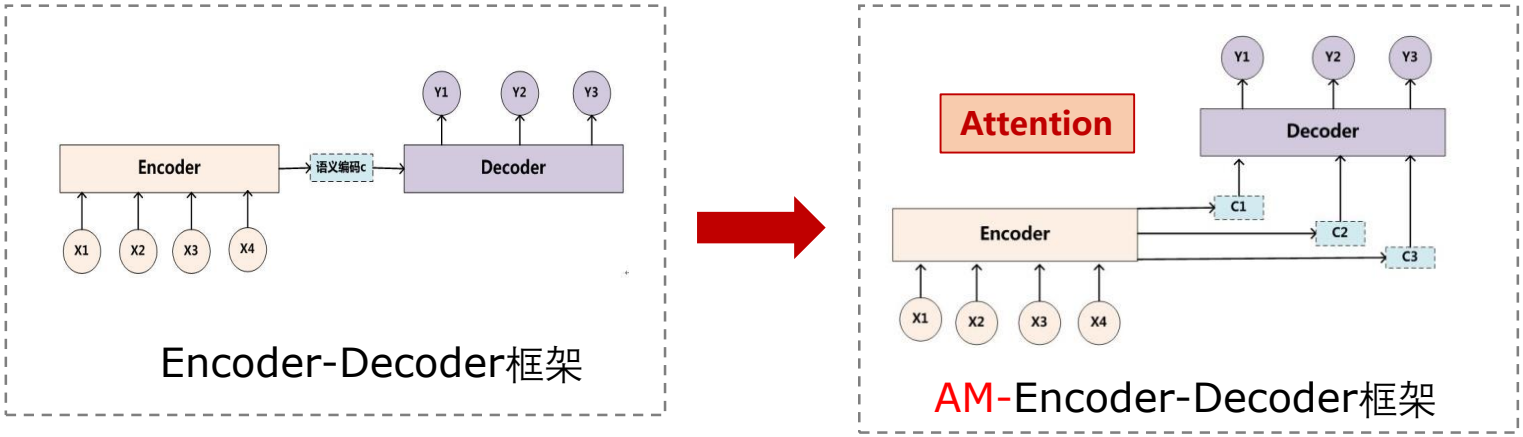


Encoder-Decoder框架

14.1 传统注意力机制

如何用?

一般用在Encoder-Decoder框架中的 Encoder 端和 Decoder 端之间关系计算中



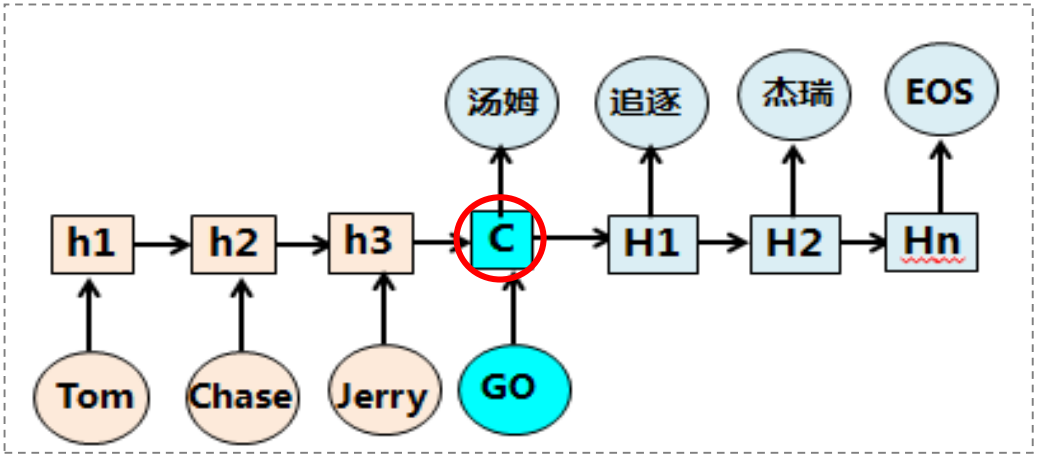
作用：让任务处理系统更专注于找到输入数据中显著的与当前输出相关的有用信息，从而提高输出的质量。

优势：不需要监督信号，可推理多种不同模态数据之间的难以解释、隐蔽性强、复杂映射关系，对于先验认知少的问题，极为有效。

14.1 传统注意力机制

机器翻译例:

Encoder-Decoder RNN



$$X = \langle x_1, x_2 \dots x_m \rangle$$

$$Y = \langle y_1, y_2 \dots y_n \rangle$$

$$C = \mathcal{F}(x_1, x_2 \dots x_m)$$

$$y_1 = f(C)$$
$$y_2 = f(C, y_1)$$
$$y_3 = f(C, y_1, y_2)$$
$$y_i = \mathcal{G}(C, y_1, y_2 \dots y_{i-1})$$

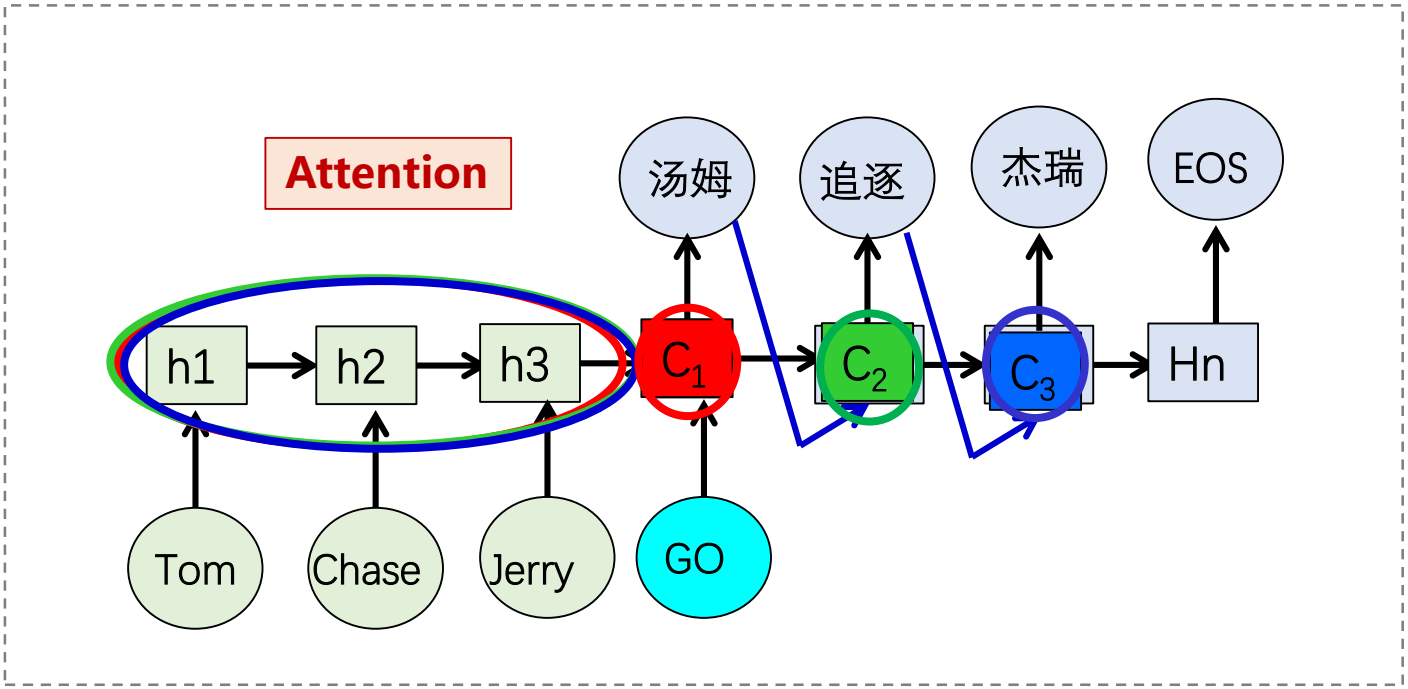
问题: **对不同的输出 Y_i 中间语义表示C 相同**

实际应该：在翻译“杰瑞”的时候，体现出英文单词对于翻译当前中文单词不同的影响程度，比如（Tom,0.3）(Chase,0.2) (Jerry,0.5)

加入 Attention 模型

14.1 传统注意力机制

Encoder-Decoder +Attention



$\alpha_{ij} = f(H_i, h_1^{Tx})$ $c_i = \sum_{j=1}^{T_x} a_{ij} h_j$

$y_1 = f1(C_1)$
$y_2 = f1(C_2, y_1)$
$y_3 = f1(C_3, y_2)$

14.1 传统注意力机制

Encoder (BiLSTM)-Decoder +Attention

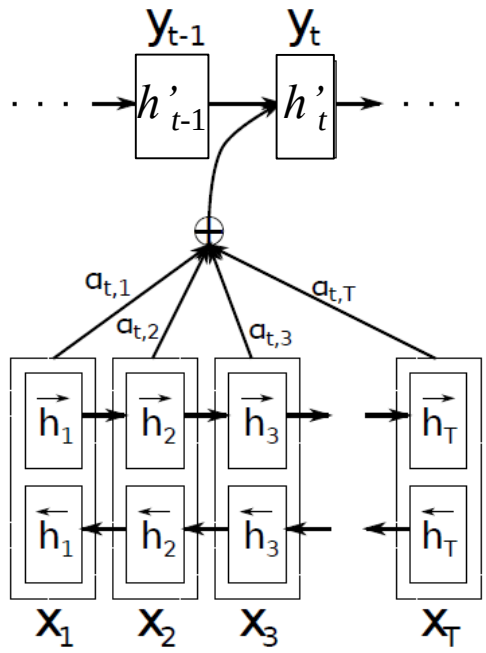


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.

■ **模型结构**

编码器采用双向RNN，解码器采用单向RNN

输入：X（源语句子）

输出：Y（目标语句子）

$$p(y_i \mid y_1, \ldots, y_{i-1}, x) = g(y_{i-1}, h_i, c_i)$$

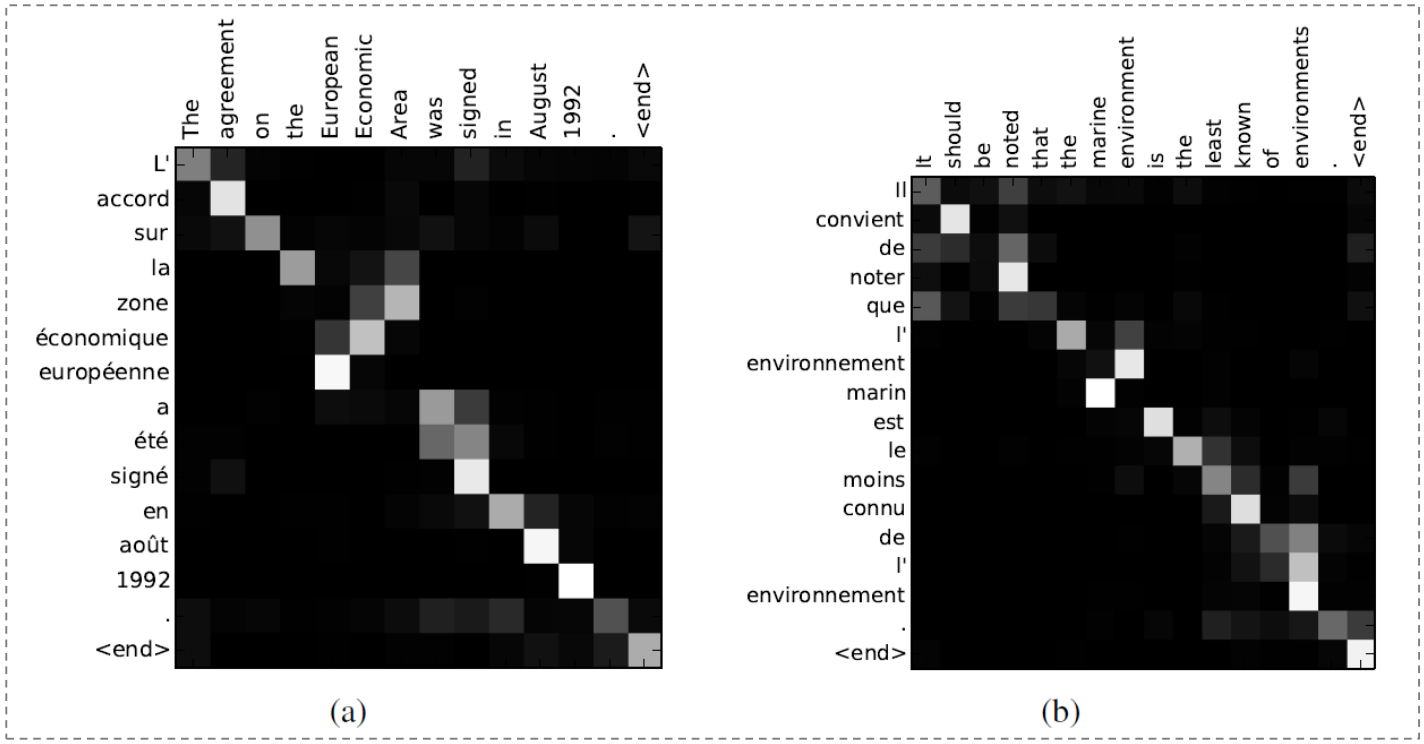
$$h_i^{'} = f(h_{i-1}^{'}, y_{i-1}, c_i)$$

$$c_i = \sum_{j=1}^{T_x} a_{ij} h_j$$

Neural machine translation by jointly learning to align and translate. ICLR 2015

14.1 传统注意力机制

注意力机制可视化效果

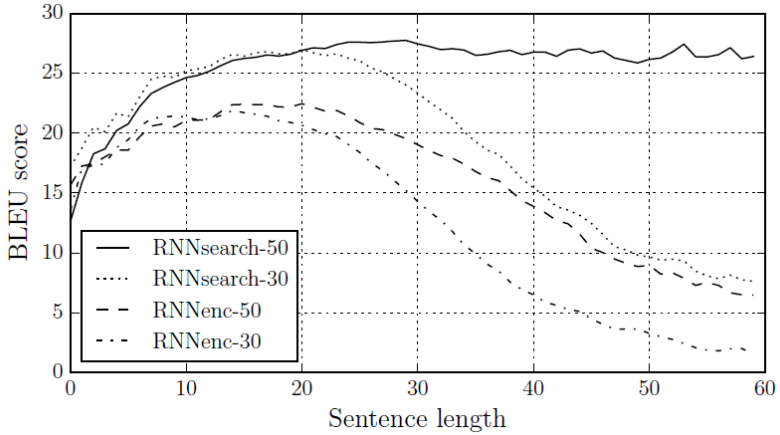


注意力机制的双语对齐（英语→法语）效果

Neural machine translation by jointly learning to align and translate. ICLR 2015

14.1 传统注意力机制

注意力机制实验结果



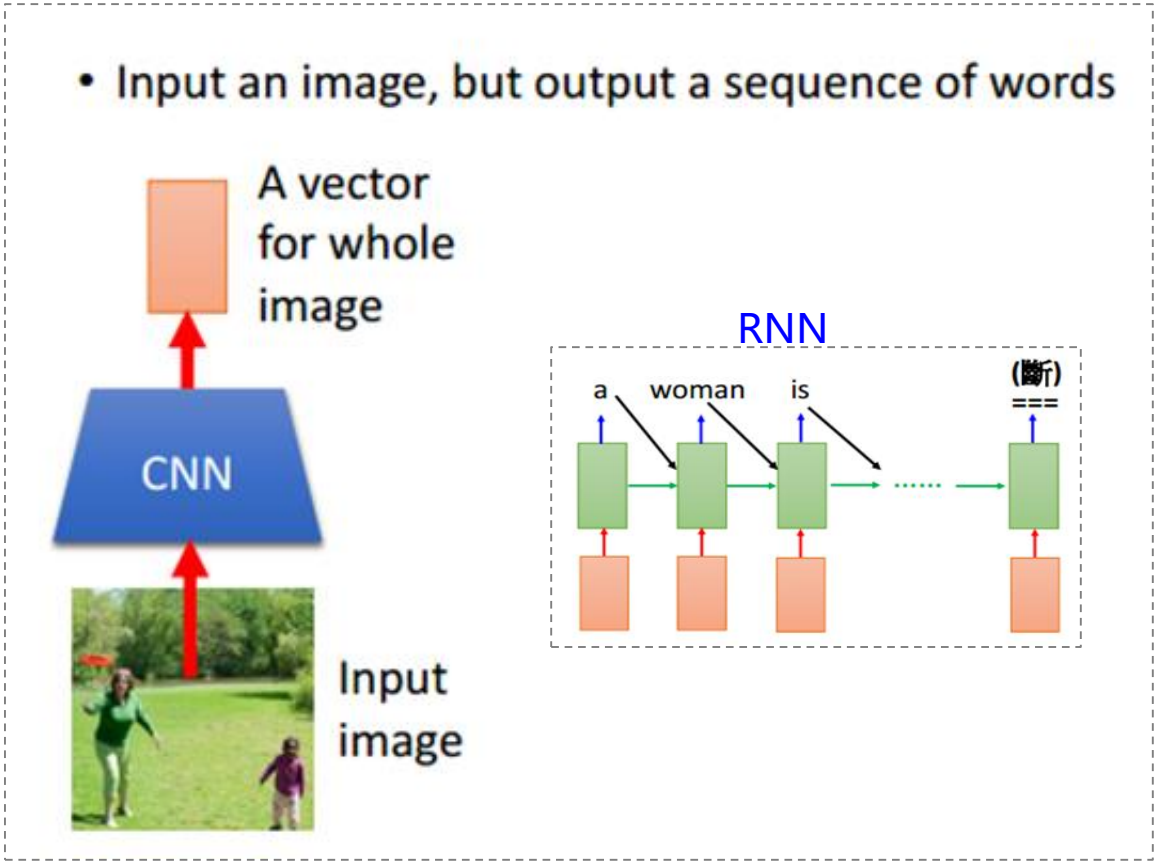
Model	All	No UNK°
RNNencdec-30	13.93	24.19
RNNsearch-30	21.50	31.44
RNNencdec-50	17.82	26.71
RNNsearch-50	26.75	34.16
RNNsearch-50*	28.45	36.15
Moses	33.30	35.63

- 在句子限长为30和50的情况下，加AM模型效果优于不加AM模型
- 句子长度增加时，加AM模型效和不加AM模型的效果均变差，但AM模型鲁棒性较好

Neural machine translation by jointly learning to align and translate. ICLR 2015

14.1 传统注意力机制

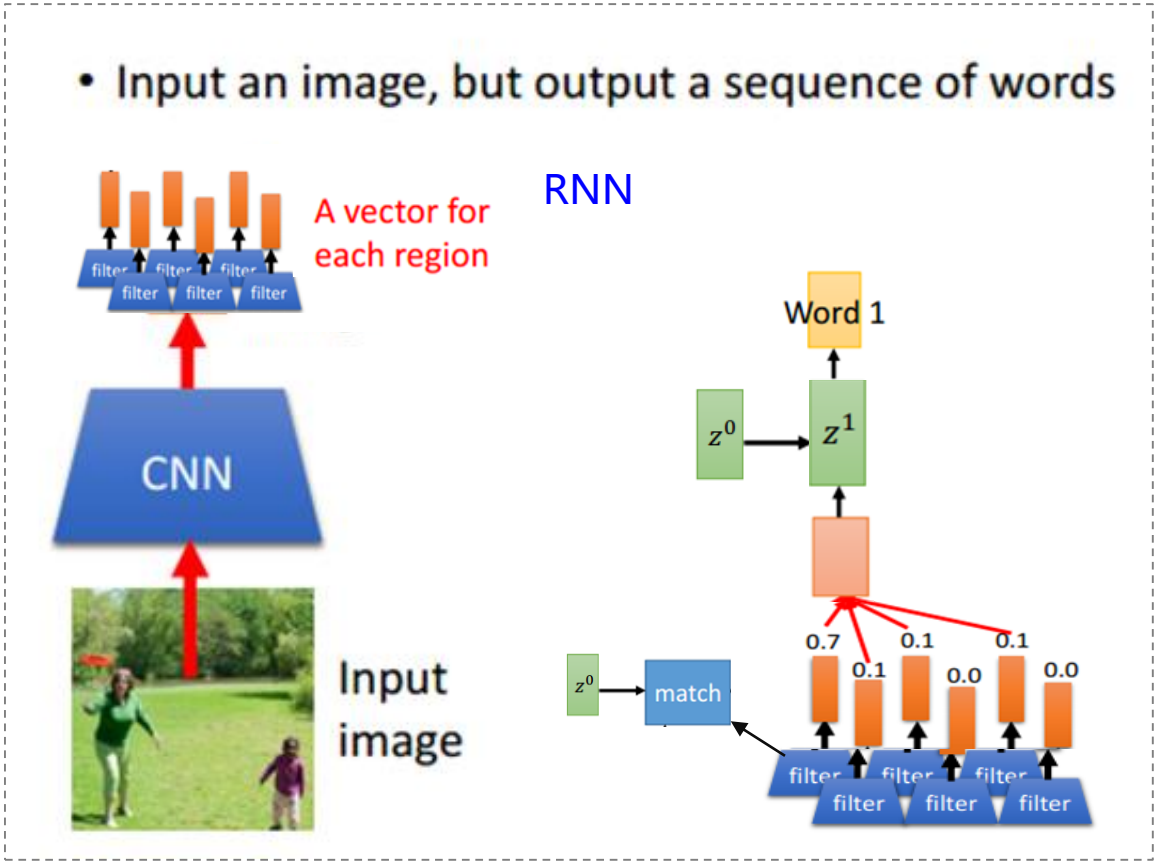
例：　　图片标题生成



A woman is throwing a Frisbee in a park

14.1 传统注意力机制

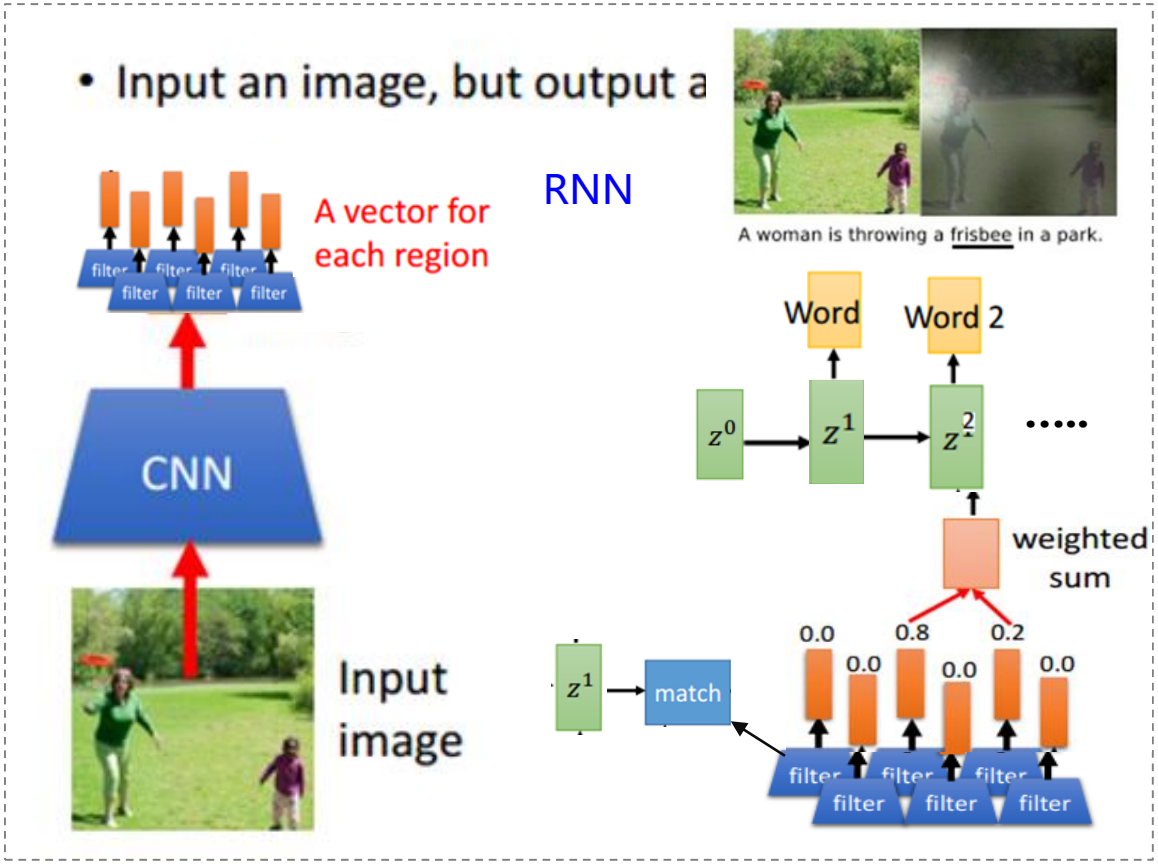
例：　　图片标题生成



A woman is throwing a Frisbee in a park

14.1 传统注意力机制

例：　　图片标题生成



A woman is throwing a Frisbee in a park

14.1 传统注意力机制

图片标题生成实验

- Good captions

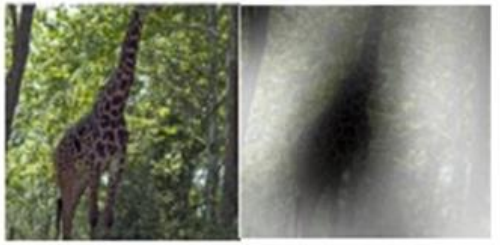
A woman is throwing a frisbee in a park.

A dog is standing on a hardwood floor.

A stop sign is on a road with a mountain in the background.

A little girl sitting on a bed with a teddy bear.

A group of people sitting on a boat in the water.

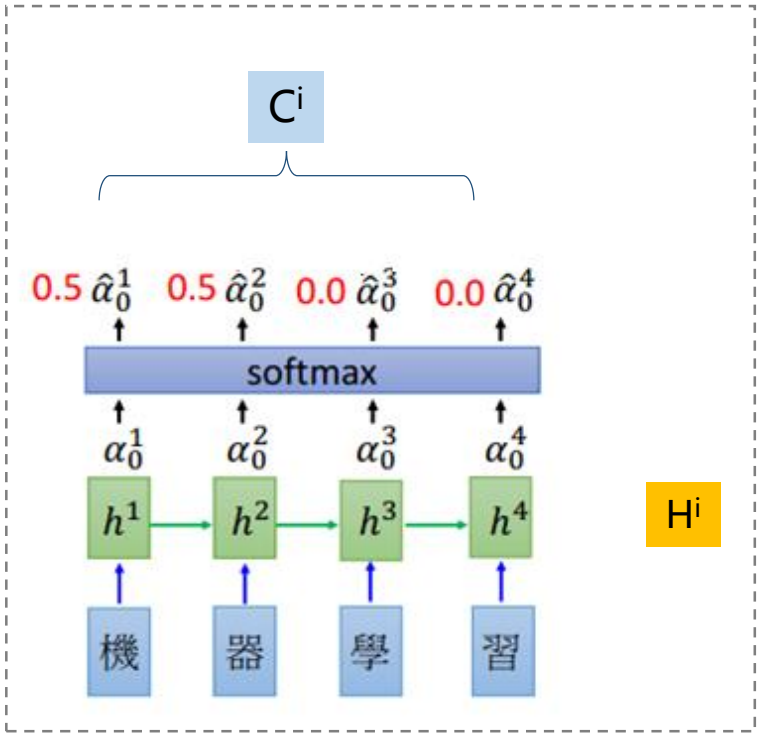


A giraffe standing in a forest with trees in the background.

14.1 传统注意力机制

□ 软注意力 Hard Attention

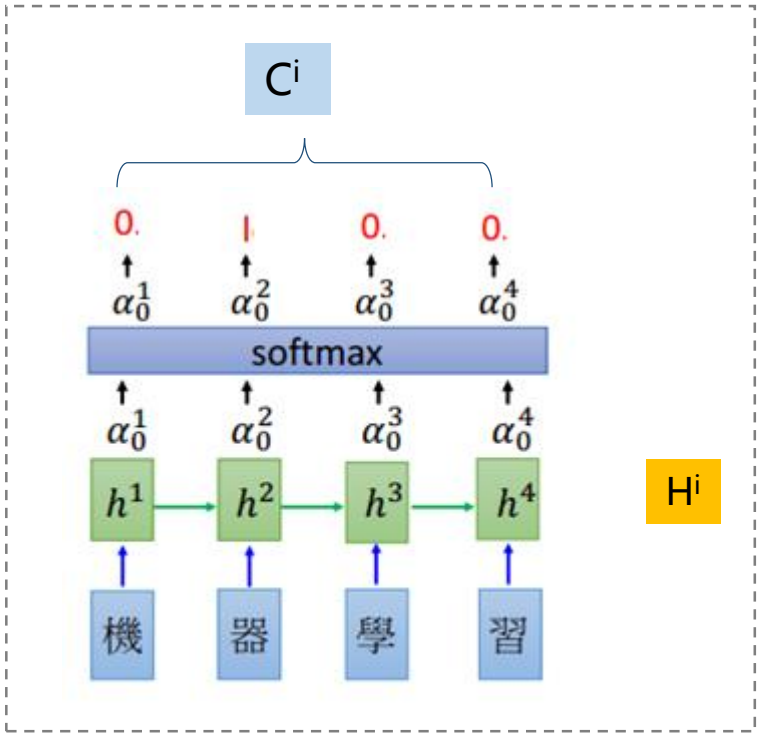
Soft AM：在求注意力分配概率分布的时候，对于输入句子X中任意一个单词都给出个概率，是个概率分布。



14.1 传统注意力机制

口 硬注意力 Hard Attention

Hard AM：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

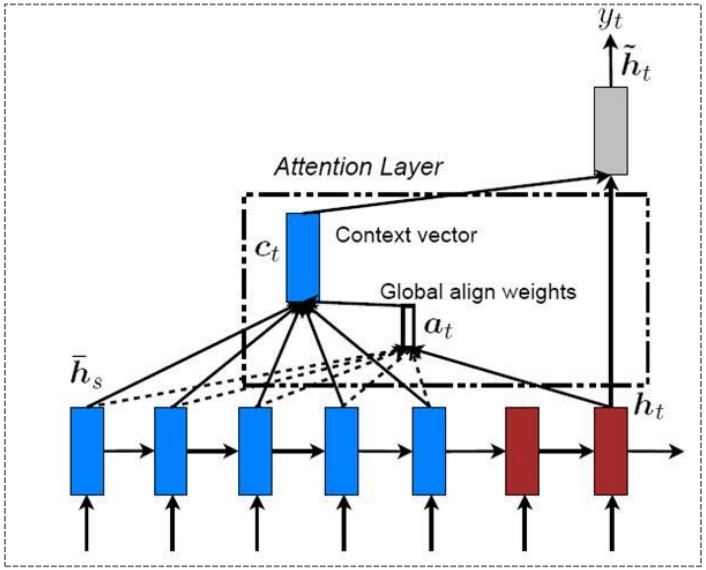


14.1 传统注意力机制

□ 全局注意力 Global Attention

Decode端Attention计算时要考虑输Ecoder端序列中所有的词

Global Attention Model 是Soft Attention Model

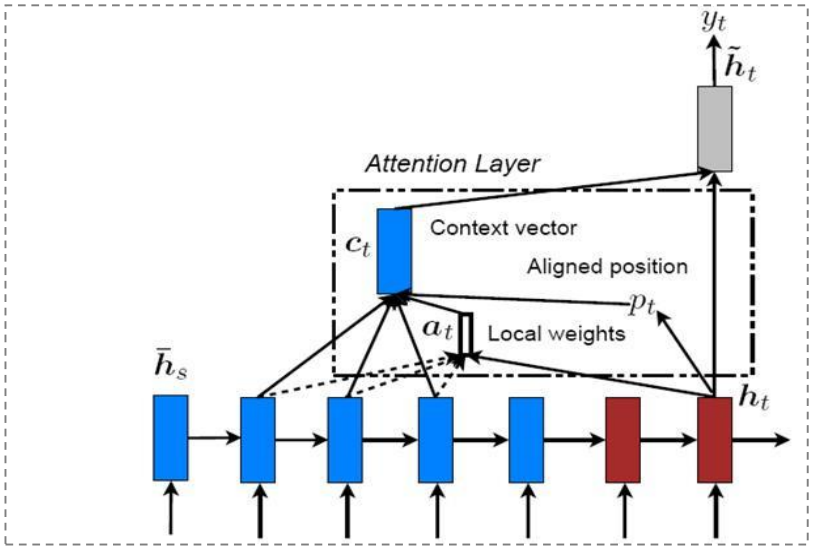


Effective Approaches to Attention-based Neural Machine Translation. ACL 2015.

14.1 传统注意力机制

□ 局部注意力 Local Attention

Local Attention Model本质上是Soft AM和 Hard AM的一个混合或折衷。一般首先预估一个对齐位置Pt，然后在Pt左右大小为D的窗口范围来取类似于Soft AM的概率分布。



Effective Approaches to Attention-based Neural Machine Translation. ACL 2015.

内 容 提 要

14.1　传统注意力机制

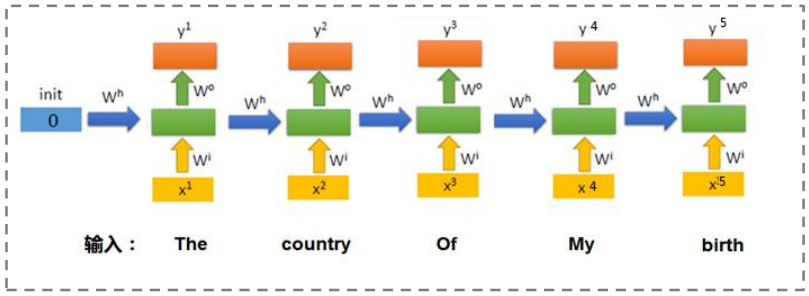
14.2　Transformer 注意力机制

14.2 Transformer 注意力机制

序列编码

深度学习的NLP 方法，基本上都是先将句子分词，然后每个词转化为对应的词向量序列这样一来，每个句子都对应的是一个矩阵 $X=(X_1,X_2,...,X_n)$，其中X_i 代表着第 i 个词的词向量（行向量），维度为 d 维，故 $X\in R^{n\times d}$。于是，各种模型就变成编码这些序列。

- **RNN层：**



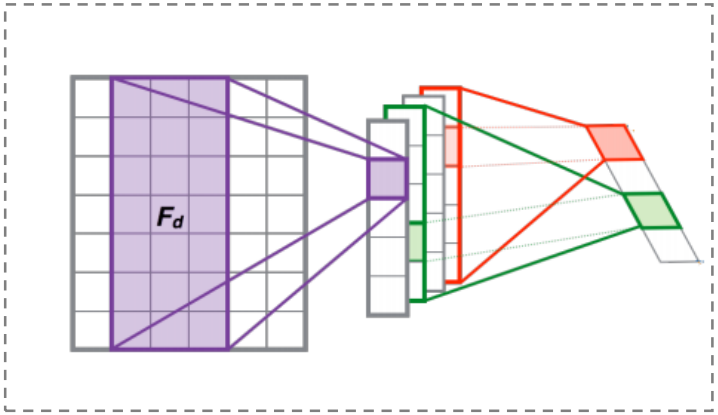
RNN 的方案很简单，递归式进行： $y_t=f(y_{t-1},x_t)$

优点：结构比较简单，适合序列建模；

不足：无法并行，速度较慢，无法很好地学习到全局的结构信息。要用双向 RNN逐步递归才能获得全局信息。

14.2 Transformer 注意力机制

- **CNN 层：** CNN 的方案也是很自然的，窗口式遍历



如，采用卷积核（窗口）为 3 的卷积，CNN为：

$$y_t=f(x_{t-1},x_t,x_{t+1})$$

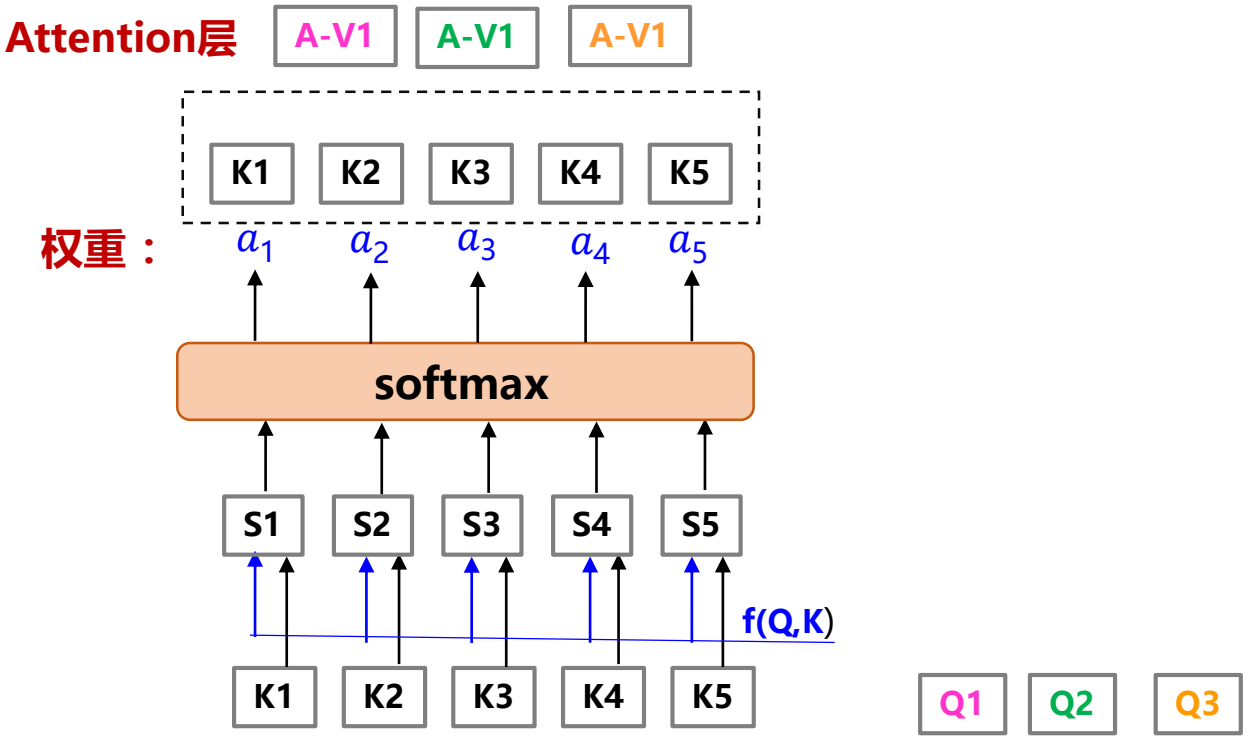
CNN 事实上只能获取局部信息，是通过层叠来增大感受野

14.2 Transformer 注意力机制

- **Attention层**

Google 的 Attention 思路也是一个编码序列的方案，可以认为它RNN、CNN 一样，都是一个序列编码的层。

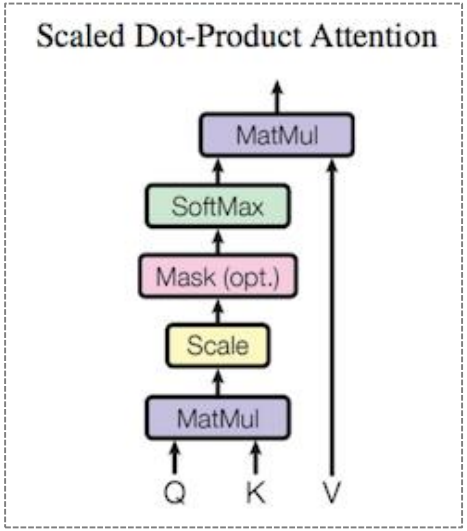
$$\text{Att-V} = a1 \times K1 + a2 \times K2 + a3 \times K3 + a4 \times K4 + a5 \times K5$$



14.2 Transformer 注意力机制

- **Attention层**

Google 的一般化 Attention编码的层。



$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

其中， $Q \in R^{n\times dk}, K \in R^{m\times dk}, V \in R^{m\times dv}$

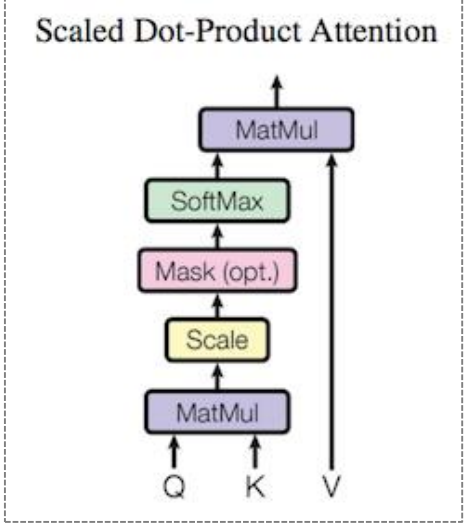
scale factor $\sqrt{d_k}$ 起 到调节作用，使得内积不至于太大（否则softmax会进入饱和区）

如果忽略激活函数 softmax 的话，那么事实上它就是三个 $n\times d_k, d_k\times m, m\times d_v$ 的矩阵相乘，最后的结果就是一个 $n\times d_v$ 的矩阵。这是一个 Attention 层，将 $n\times d_k$ 的序列 Q 编码成了一个新的 $n\times d_v$ 的序列。**问题：Attention层是词袋模型**

14.2 Transformer 注意力机制

自注意力 Self-Attention

其实就是 Attention($\mathbf{X},\mathbf{X},\mathbf{X}$)，$\mathbf{X}$ 为输入序，其含义为在序列内部做 Attention，寻找序列内部的联系



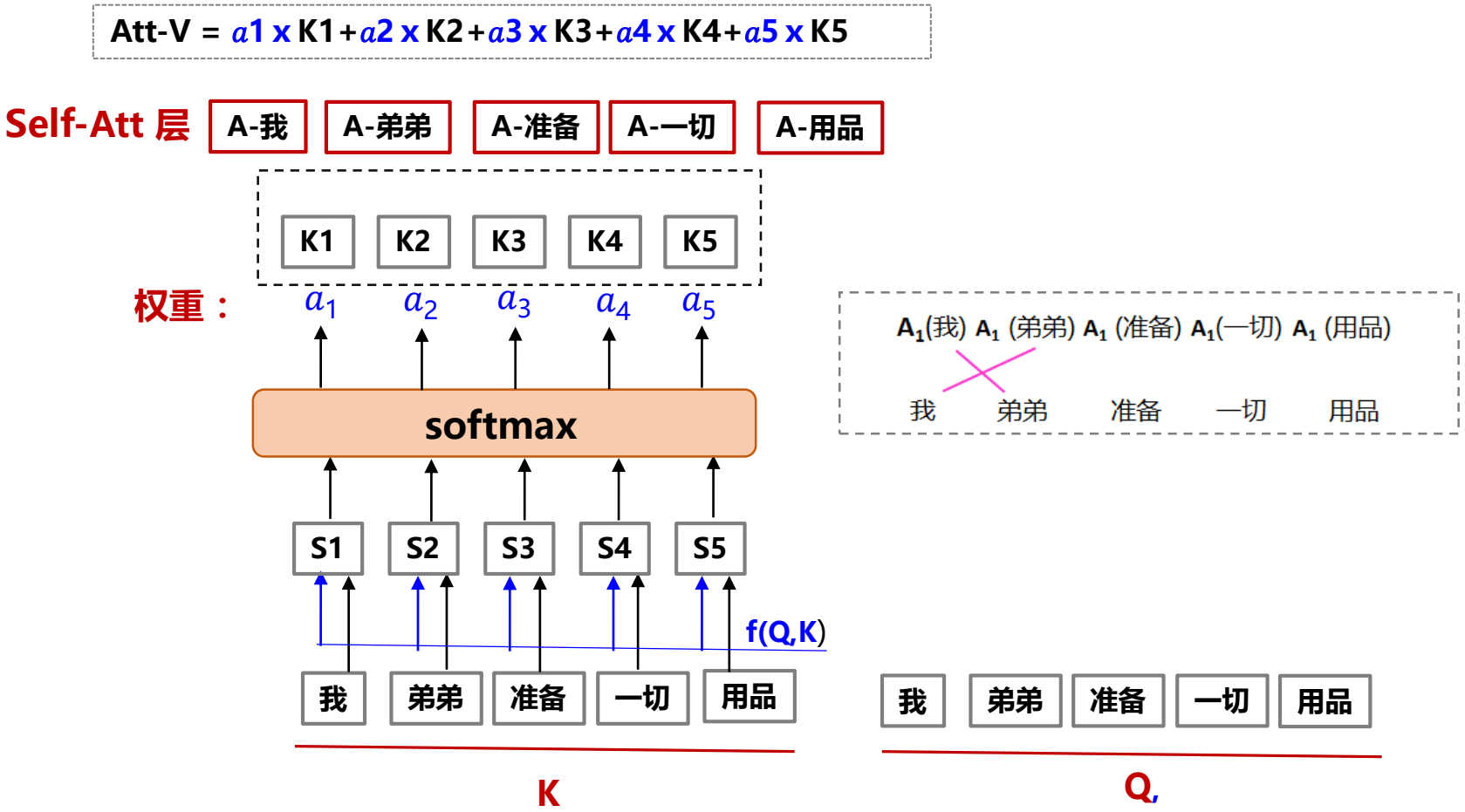
$$\text{Attention}(Q,K,V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

其中， Q=K=V

self-attention的特点在于无视词之间的距离直接计算依赖关系，能够学习一个句子的内部结构，实现也较为简单并行可以并行计算（self-attention可以当成一个层和RNN，CNN，FNN等配合使用，应用于其他NLP任务）

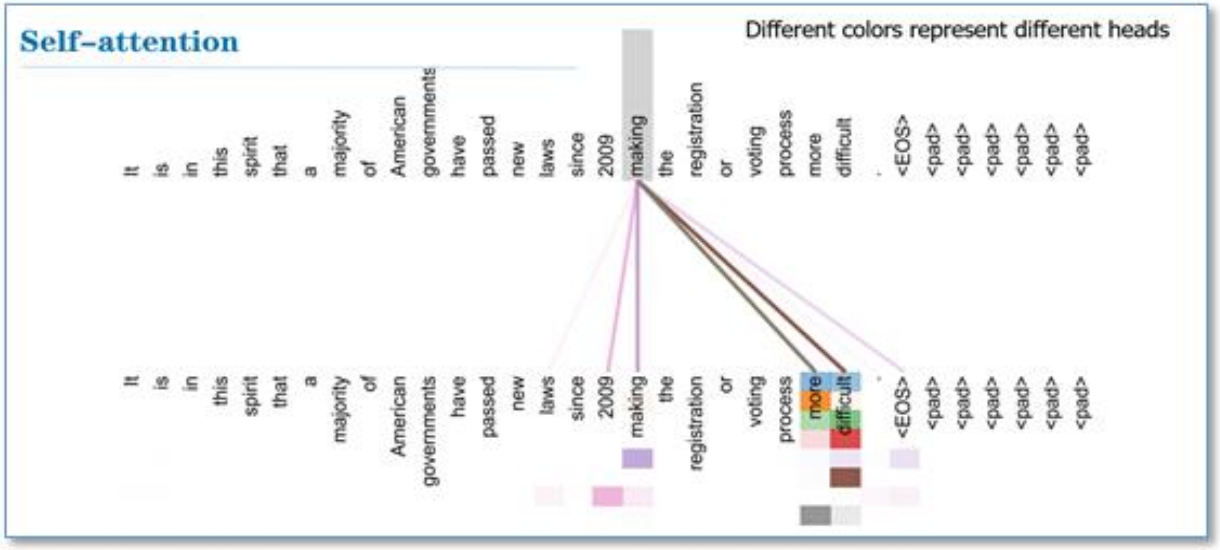
14.2 Transformer 注意力机制

自注意力 Self-Attention



14.2 Transformer 注意力机制

Self-Attention可视化的效果



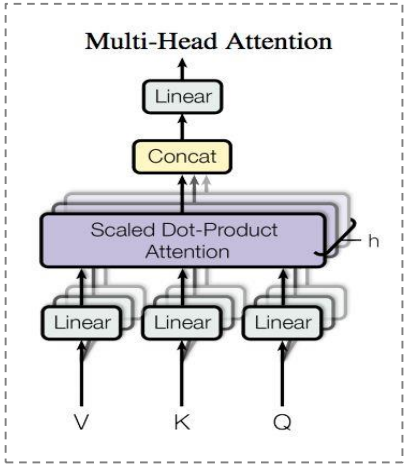
可以看到self-attention在这里可以学习到句子内部长距离依赖"making.....more difficult"这个短语

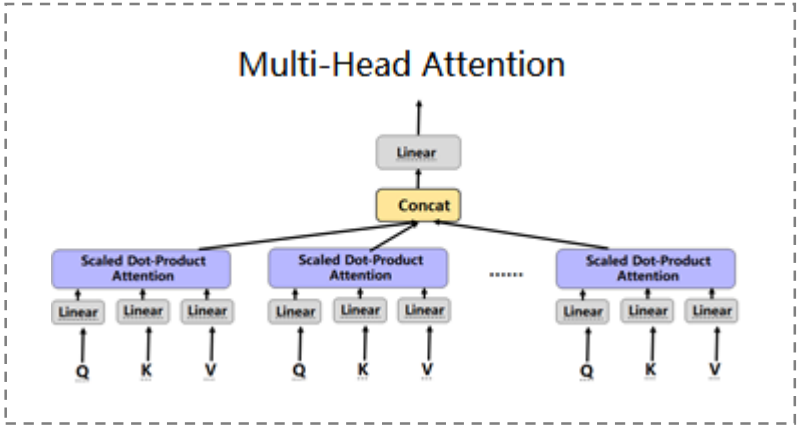
14.2 Transformer 注意力机制

多头注意力Multi-Head Attention

多头（Multi-Head）就是做多次同样的事情（参数不共享），然后把结果拼接

多头attention通过计算多次来捕获不同子空间上的相关信息。





$$\mathrm{Head}_i = \mathrm{Attention}(QW^Q_i, KW^K_i, VW^V_i)$$

其中，$W^Q_i \in R^{dk \times \sim dk}, W^K_i \in R^{dk \times \sim dk}, W^V_i \in R^{dv \times \sim dv}$

$\mathrm{Multi\text{-}Head}(Q,K,V) = \mathrm{Concat}(\mathrm{head}_1, \ldots, \mathrm{head}_h)$ **最后得到一个** $n \times (h \sim d_v)$ **的序列**

14.2 Transformer 注意力机制

多头自注意力 Multi-Head Self Attention（Transformer）

$$Y=MultiHead(X,X,X)$$

◆ Head1 Self Attention

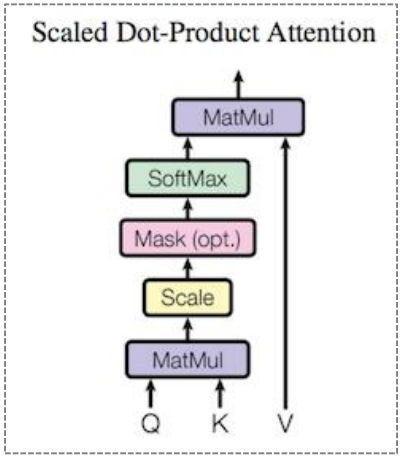
输出：A_1(我) A_1 (弟弟) A_1 (准备) A_1(一切) A_1 (用品)

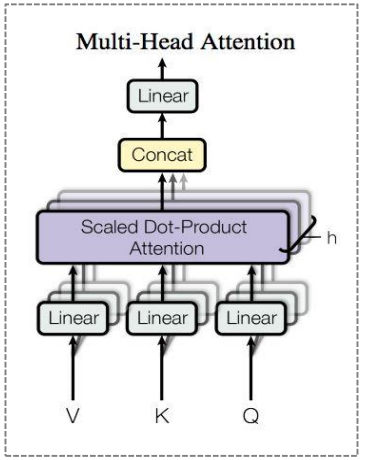
输入：我　弟弟　准备　一切　用品

◆ Head2 Self Attention

输出：A_2(我) A_2 (弟弟) A_2 (准备) A_2(一切) A_2 (用品)

输入：我　弟弟　准备　一切　用品





◆ Multi-Head Self Attention

输出：$C_{1\text{-}2}$(我) $C_{1\text{-}2}$ (弟弟) $C_{1\text{-}2}$ (准备) $C_{1\text{-}2}$(一切) $C_{1\text{-}2}$ (用品)

Concat

输入：

A_1(我) A_1 (弟弟) A_1 (准备) A_1(一切) A_1 (用品)

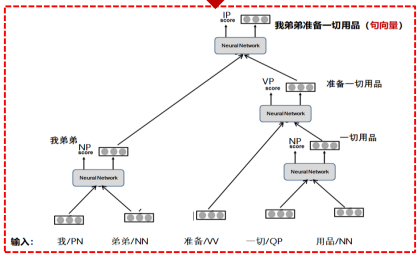
我　弟弟　准备　一切　用品

A_2(我) A_2 (弟弟) A_2 (准备) A_2(一切) A_2 (用品)

我　弟弟　准备　一切　用品

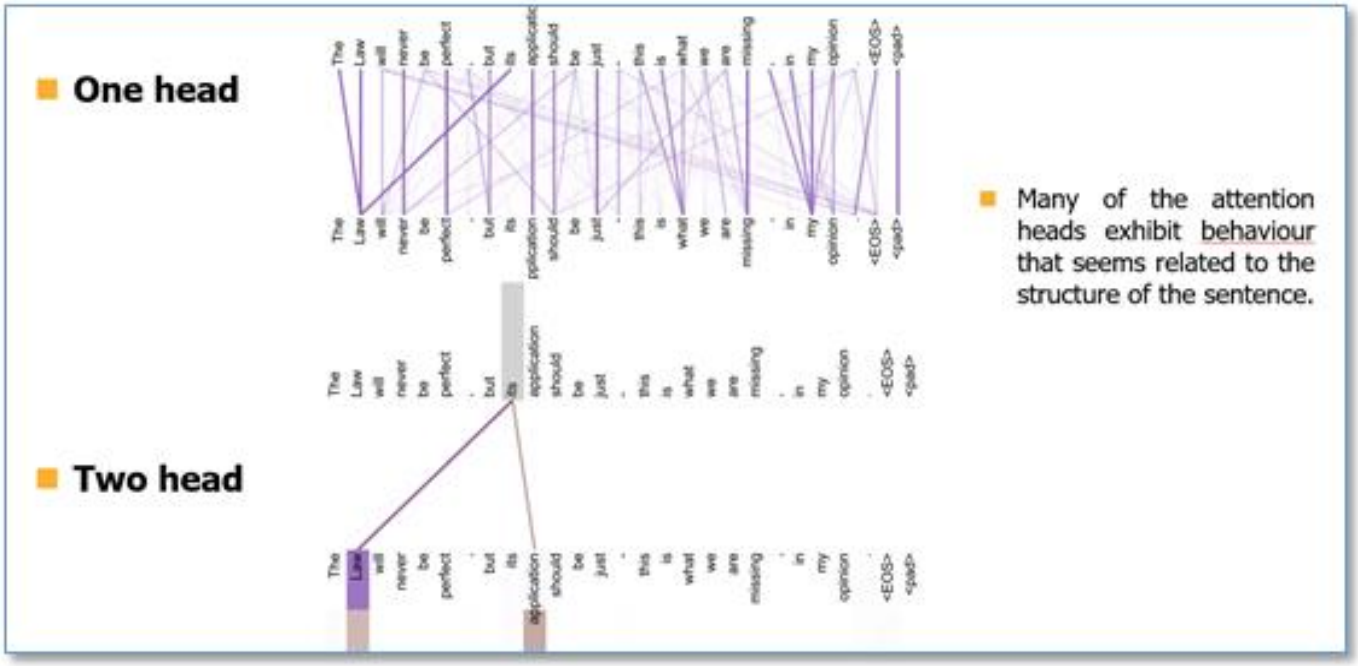
$C_{1\text{-}2}$(我) $C_{1\text{-}2}$ (弟弟) $C_{1\text{-}2}$ (准备) $C_{1\text{-}2}$(一切) $C_{1\text{-}2}$ (用品)

我　弟弟　准备　一切　用品



14.2 Transformer 注意力机制

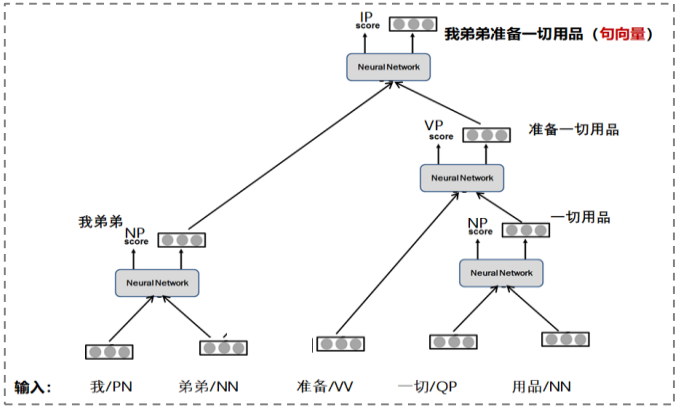
多头自注意力的可视化的效果



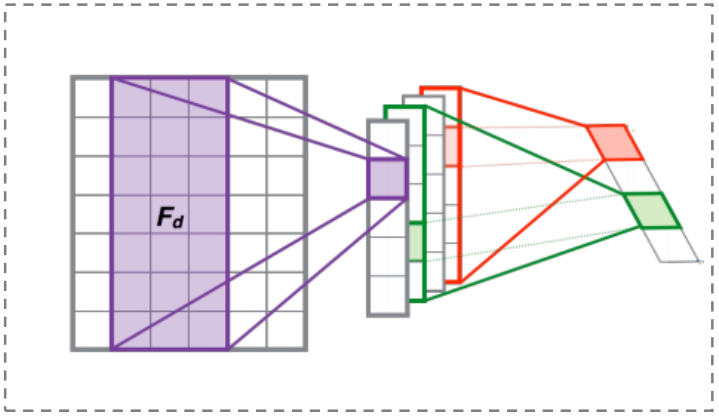
在两个头和单头的比较中，可以看到单头"its"这个词只能学习到"law"的依赖关系，而两个头"its"不仅学习到了"law"还学习到了"application"依赖关系。多头能够从不同的表示子空间里学习相关信息

14.2 Transformer 注意力机制

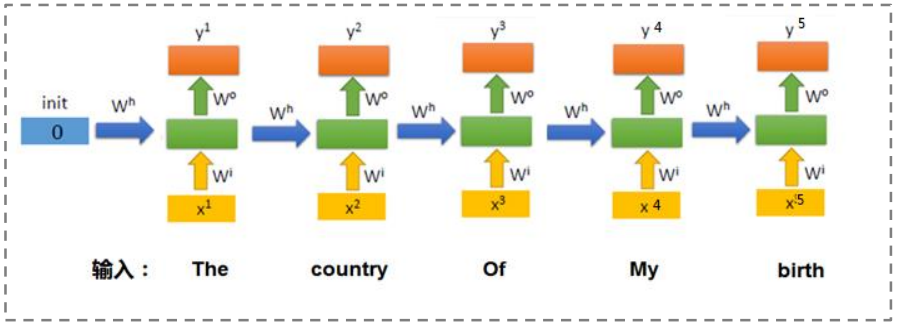
句向量



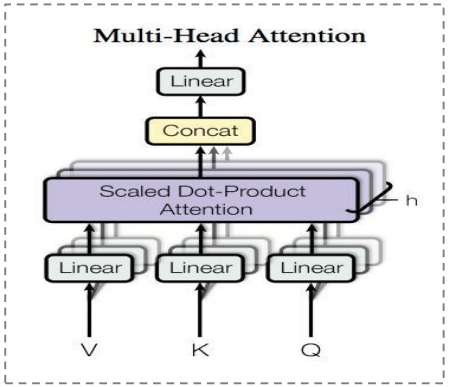
RvNN 句向量



CNN 句向量



RNN 句向量



Multi-Head Self Attention 句向量

14.2 Transformer 注意力机制

最新自注意力机制(Self-attention)论文：

1. A Structured Self-attentive Sentence Embedding（一个结构化的自注意力的句子嵌入）

2. Self-Attention with Relative Position Representations（基于相对位置表示的子注意力模型）

3. Reinforced Self-Attention Network: a Hybrid of Hard and Soft Attention for Sequence Modeling
（增强的自注意力网络:一种对序列建模的硬和软注意力的混合）

4. Distance-based Self-Attention Network for Natural Language Inference
（基于距离的自注意力网络的自然语言推理）

5. Improving Visually Grounded Sentence Representations with Self-Attention
（以自注意力的方式提高基于视觉的句子表达）

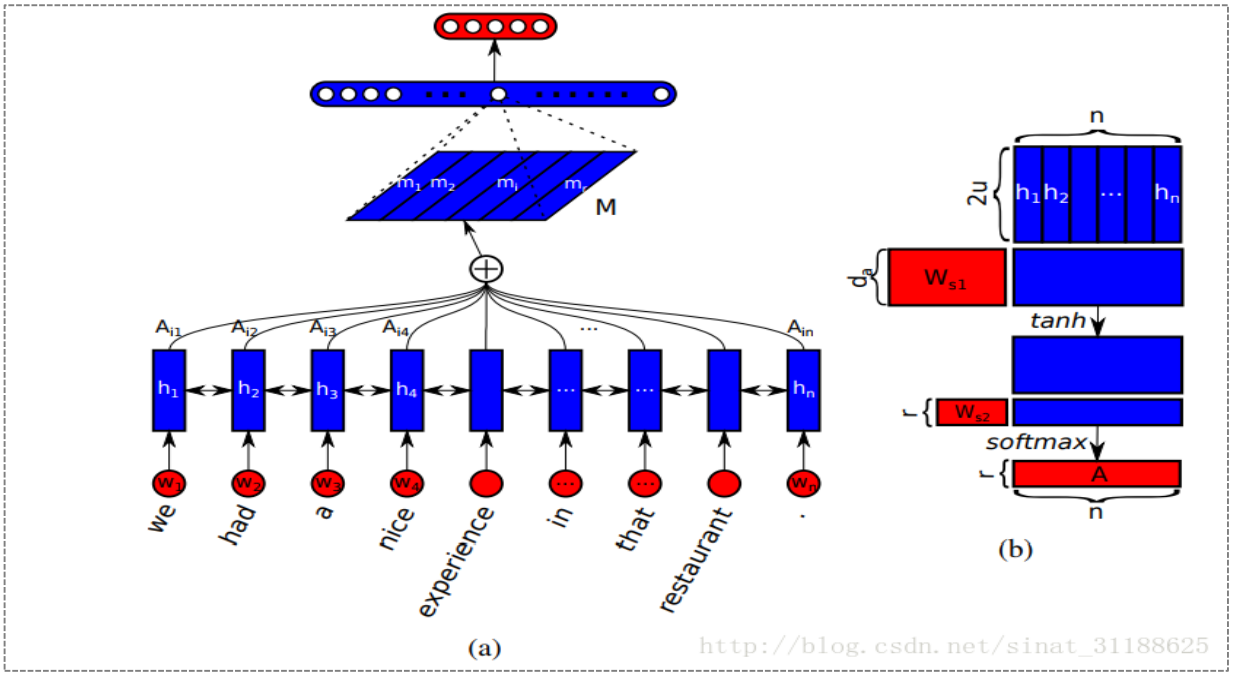
6. DiSAN: Directional Self-Attention Network for RNN/CNN-Free Language Understanding
（DiSAN:RNN/CNN-free语言理解的方向的双向自注意网络）

7. Deep Semantic Role Labeling with Self-Attention（基于子注意力机制的深度语义角色标注）

14.2 Transformer 注意力机制

结构化的自注意力句嵌入

- **模型结构**

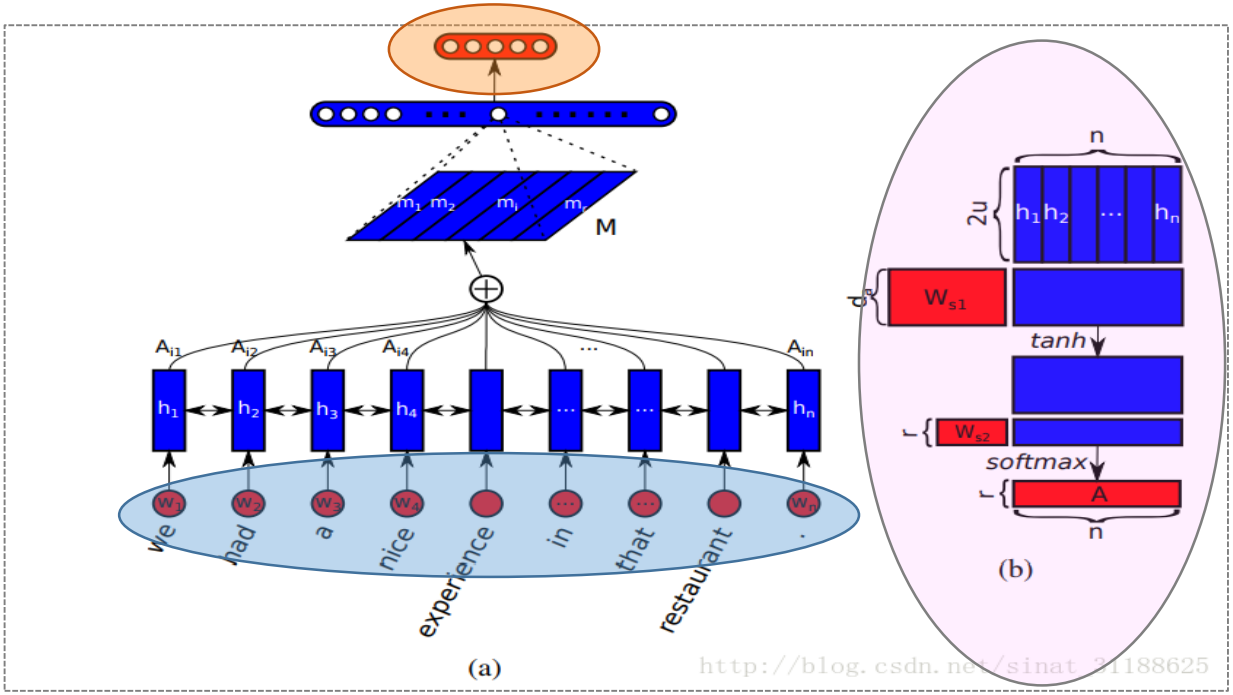


14.2 Transformer 注意力机制

结构化的自注意力句嵌入

■ **模型结构**

输出： 句向量



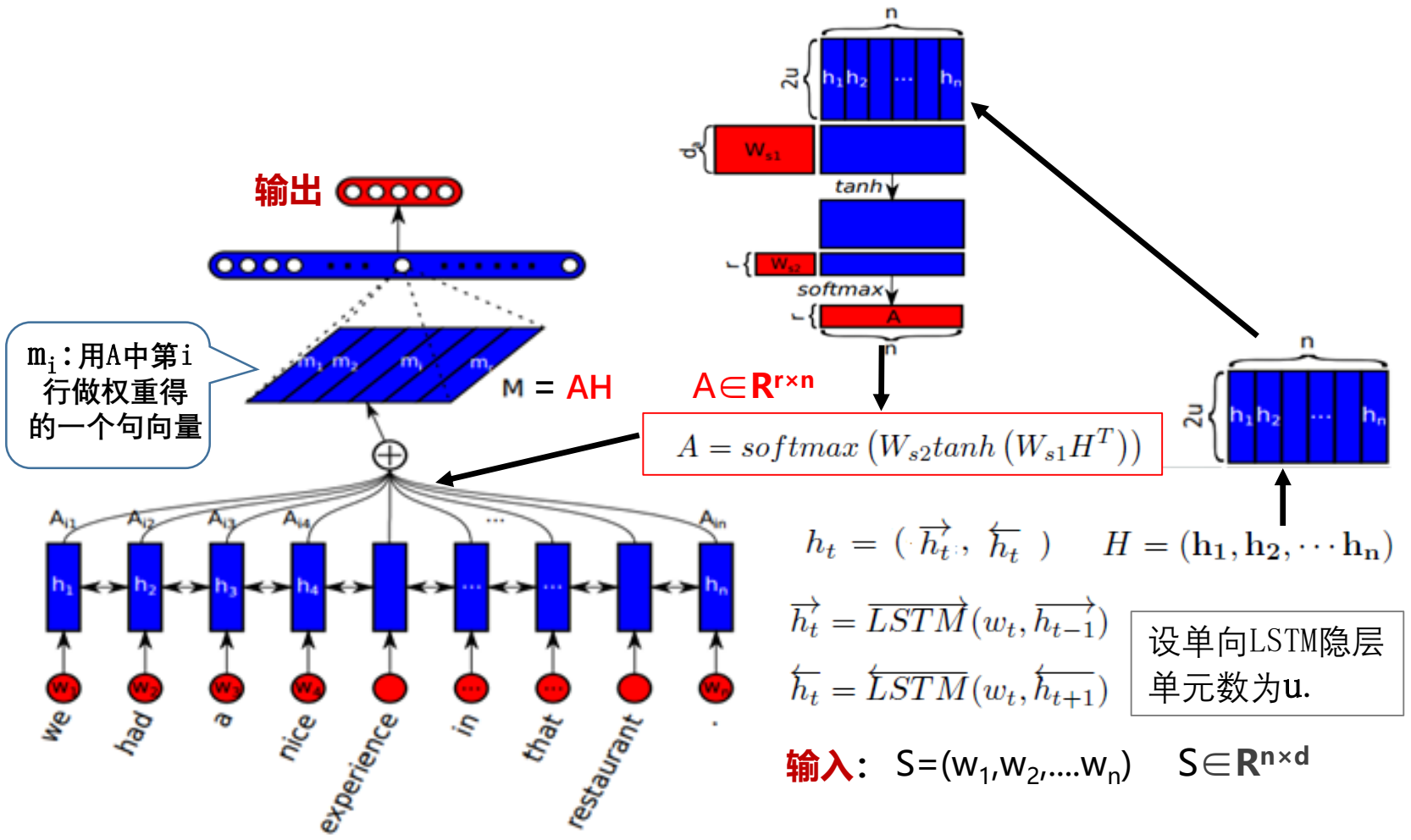
self-attention 机制

（提供对LSTM隐藏层状态进行加和的权重）

输入： 有n个单词句子，$S=(w_1,w_2,....w_n)$ $w_i \in R^{1\times d}$ d为词向量维数

$S \in R^{n\times d}$

14.2 Transformer 注意力机制



■ 正则问题

A矩阵中的r行权重表示从r个角度来生成r个系数向量，所以这r行权重要尽量不同，为此需增加一个正则项来对模型进行约束，从而保证生成的r个分布具有最大的差异性

正则项： $P = \left\| (AA^T - I) \right\|_F^2$

$$0 < a_{ij} = \sum_{k=1}^{n} a_k^i a_k^j < 1$$

最终的损失函数：

cost=交叉熵+αP，

其中，α 是调节P的比重的系数

14.2 Transformer 注意力机制

- **A的行数 r 对实验的影响**

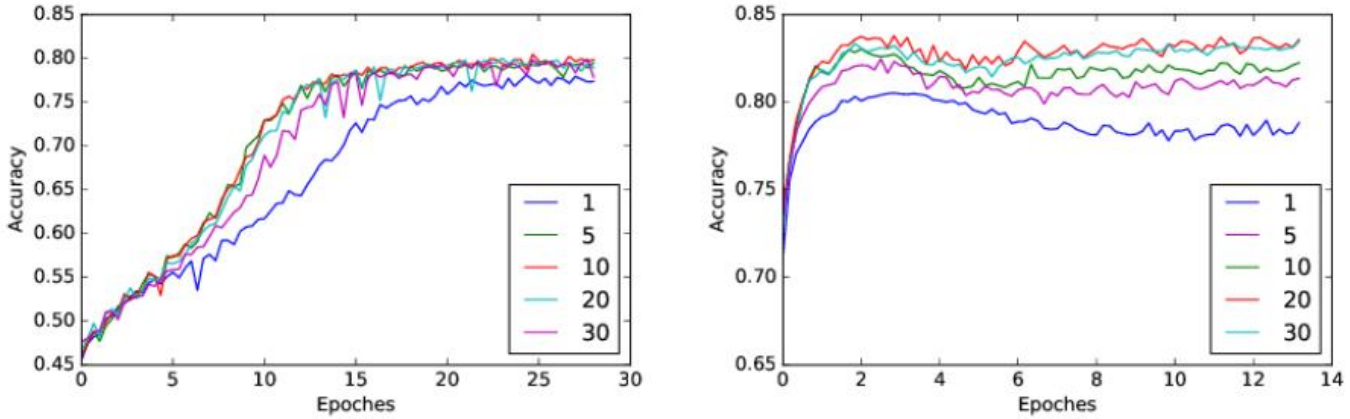
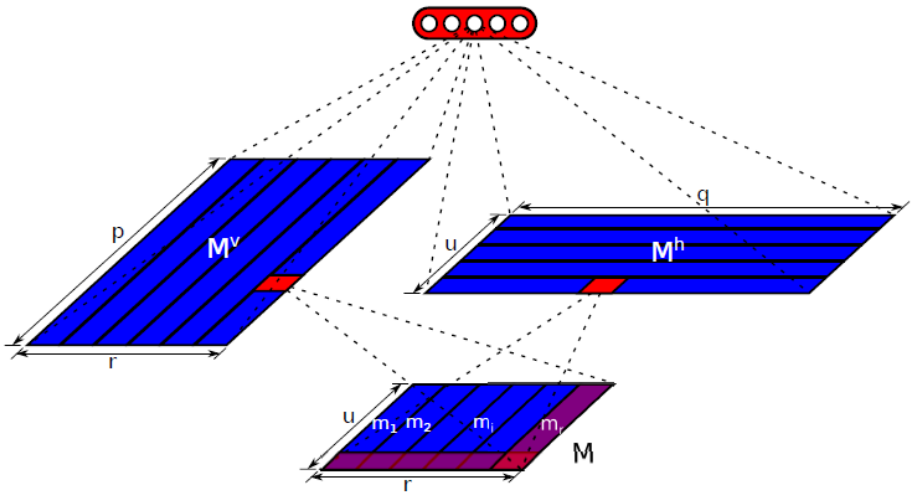


Figure : Effect of the number of rows (r) in matrix sentence embedding. The vertical axes indicates test set accuracy and the horizontal axes indicates training epoches. Numbers in the legends stand for the corresponding values of r. (a) is conducted in Age dataset and (b) is conducted in SNLI dataset.

14.2 Transformer 注意力机制

附录：剪裁网络



	Hidden layer	Softmax	Other Parts	Total	Accuracy
Yelp, Original, b=3000	54M	15K	1.3M	55.3M	64.21%
Yelp, Pruned, p=150, q=10	2.7M	52.5K	1.3M	4.1M	63.86%
Age, Original, b=4000	72M	20K	1.3M	73.2M	80.45%
Age, Pruned, p=25, q=20	822K	63.75K	1.3M	2.1M	77.32%
SNLI, Original, b=4000	72M	12K	22.9M	95.0M	84.43%
SNLI, Pruned, p=300, q=10	5.6M	45K	22.9M	28.6M	83.16%

Model Size Comparison Before and After Pruning

参考文献：

张俊林，深度学习中的注意力机制(2017版)，
https://blog.csdn.net/malefactor/article/details/78767781

苏剑林，《Attention is All You Need》浅读（简介+代码），
https://kexue.fm/archives/4765

https://blog.csdn.net/Mbx8X9u/article/details/79908973

https://www.sohu.com/a/242214491_164987

http://xiaosheng.me/2018/01/13/article121/#ii-attention层

https://cloud.tencent.com/developer/article/1086575

https://blog.csdn.net/guoyuhaoaaa/article/details/78701768

https://blog.csdn.net/sinat_31188625/article/details/78344404

李宏毅课程http://speech.ee.ntu.edu.tw/~tlkagk/courses_ML16.html

在 此 表 示 感 谢!

信息内容安全技术国家工程实验室
National Engineering Laboratory for Information Security Technologies

谢谢各位！