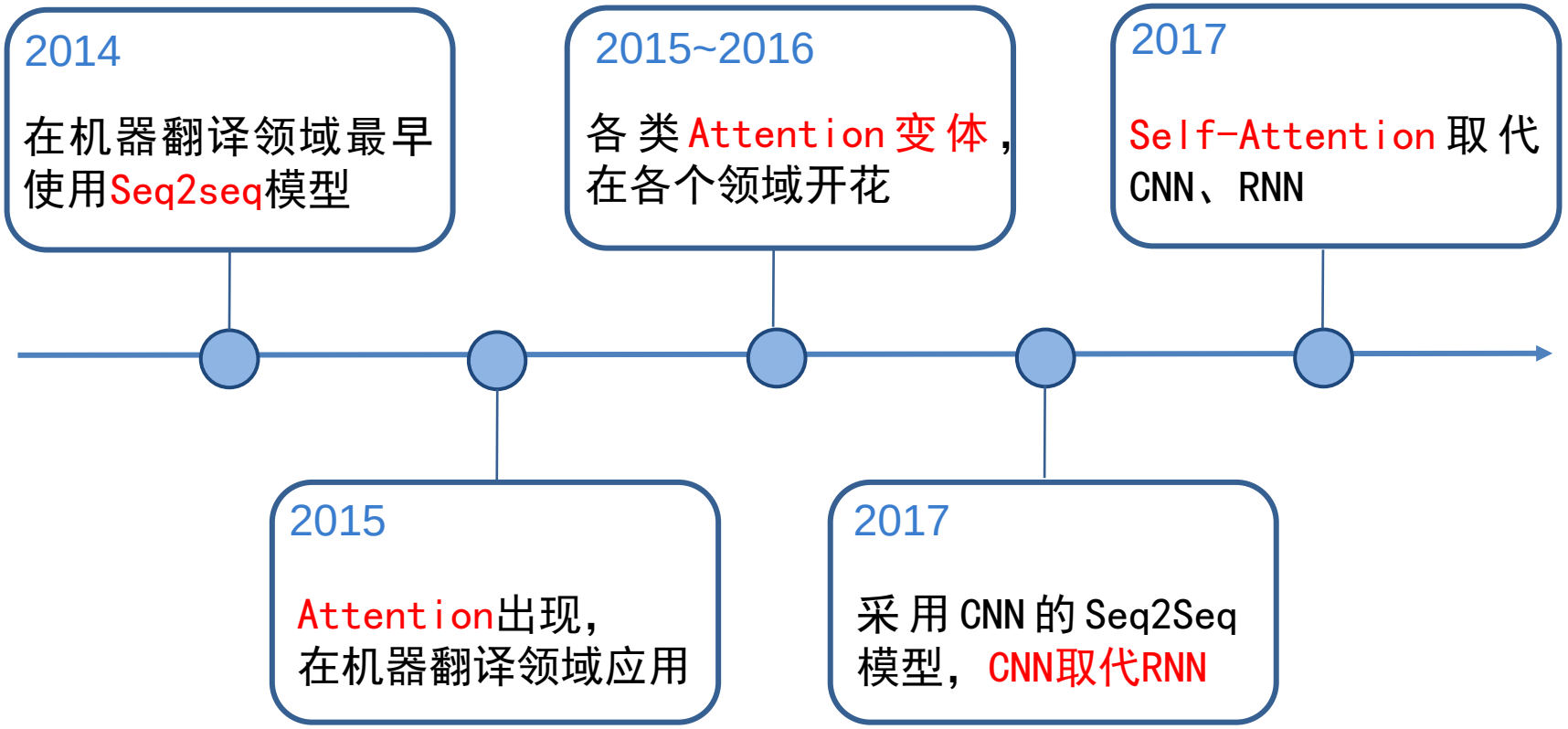
2021年度暑期强化课程

4. NLP中的注意力机制

授课人：曹亚男

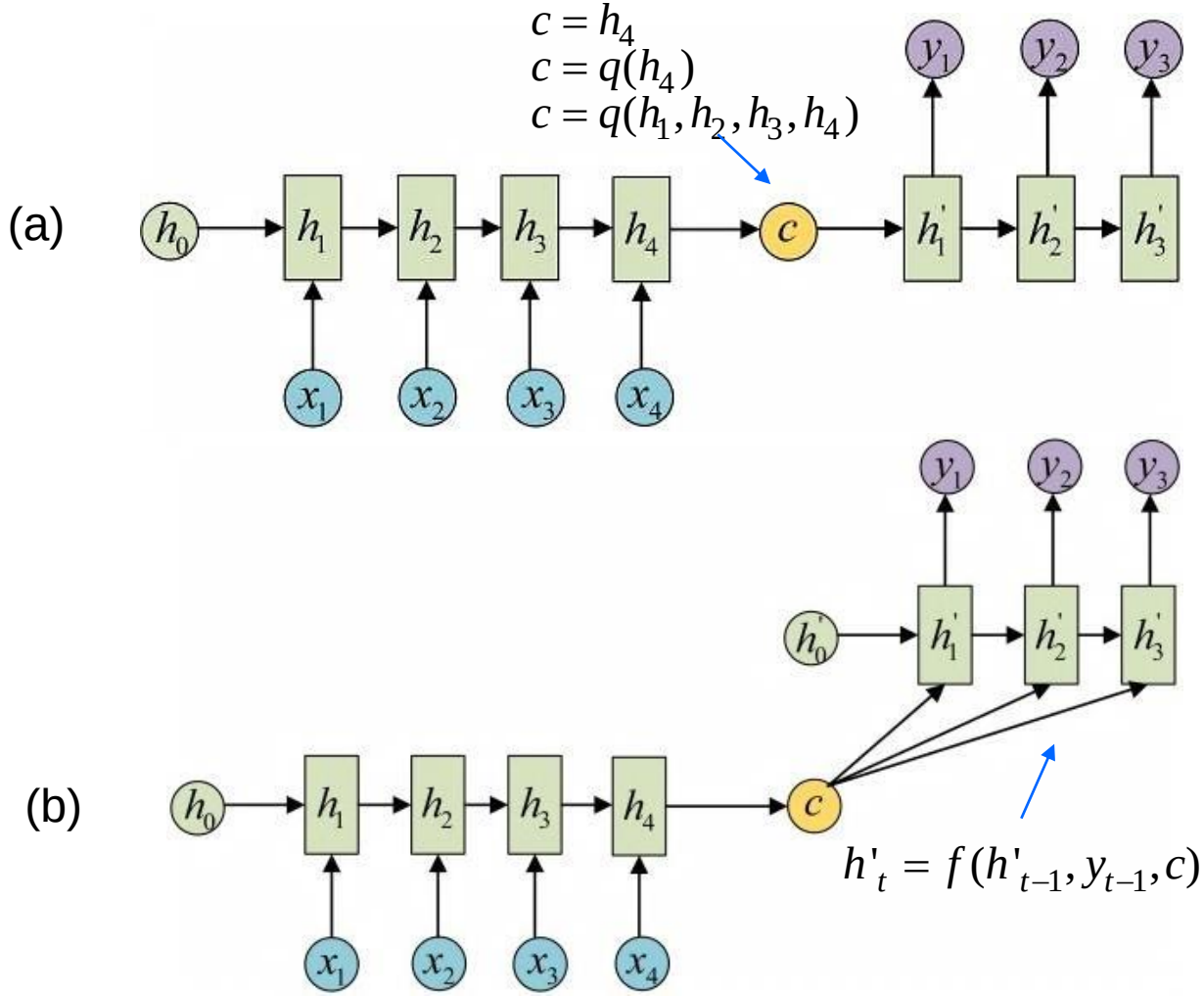
Attention发展史



4. NLP中的注意力机制

4. NLP中的注意力机制

回顾：Sequence to Sequence Model



回顾：Sequence to Sequence Model

- **两个问题**
  - **定长的中间向量*c*限制了模型性能**

Zhuge Liang--*Northern Expedition Memorial*

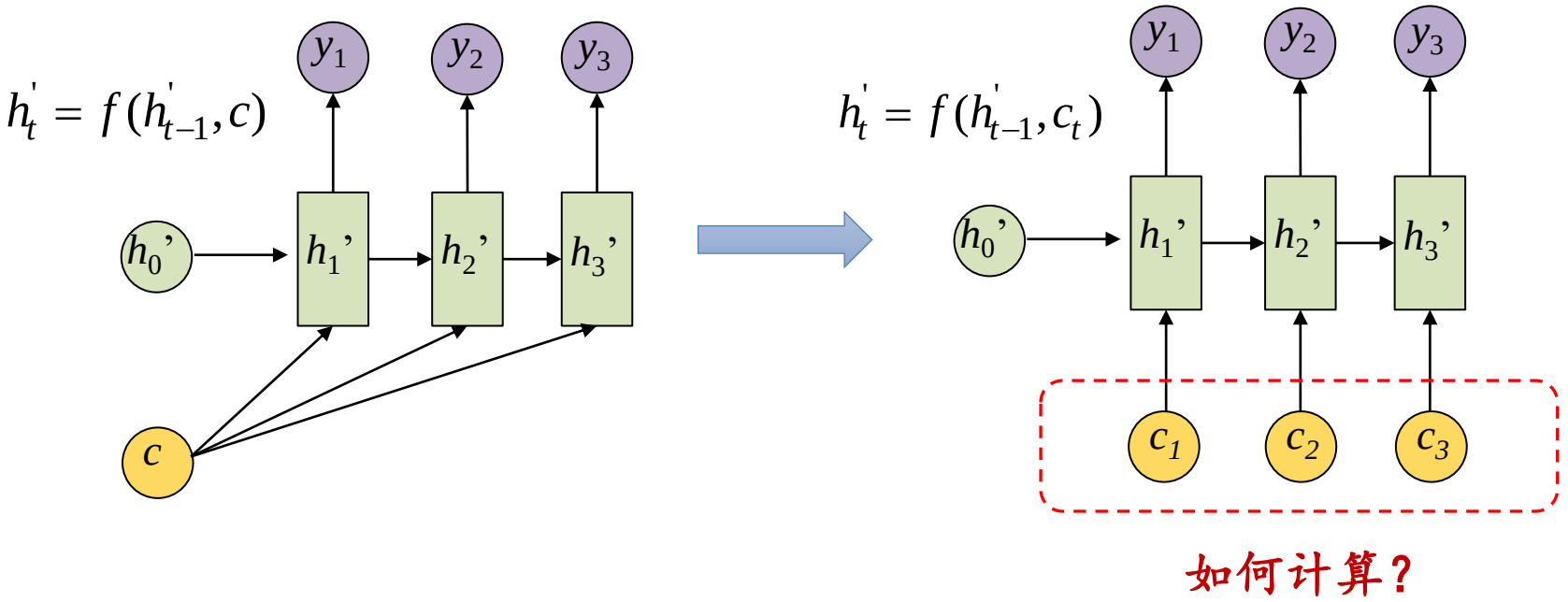
Permit me to observe: the late Emperor was taken from us before he could finish his life's work, the restoration of the Han. Today, the empire is still divided in three, and our very survival is threatened. Yet still, the officials at court and the soldiers throughout the realm remain loyal to you, your majesty. Because they remember the late emperor, all of them, and they wish to repay his kindness in service to you. This is the moment to extend your divine influence, to honor the memory of the late Emperor and strengthen the morale of your officers. It is not the time to listen to bad advice or close your ears to the suggestions of loyal men. The emperors of the Western Han chose their courtiers wisely, and their dynasty flourished.

  - **输入序列的不同部分对于输出序列的重要性不同**

先帝创业未半而中道崩殂，今天下三分，益州疲弊，此诚危急存亡之秋也。然侍卫之臣不懈于内，忠志之士忘身于外者，盖追先帝之殊遇，欲报之于陛下也。诚宜开张圣听，以光先帝遗德，恢弘志士之气，不宜妄自菲薄，引喻失义，以塞忠谏之路也。

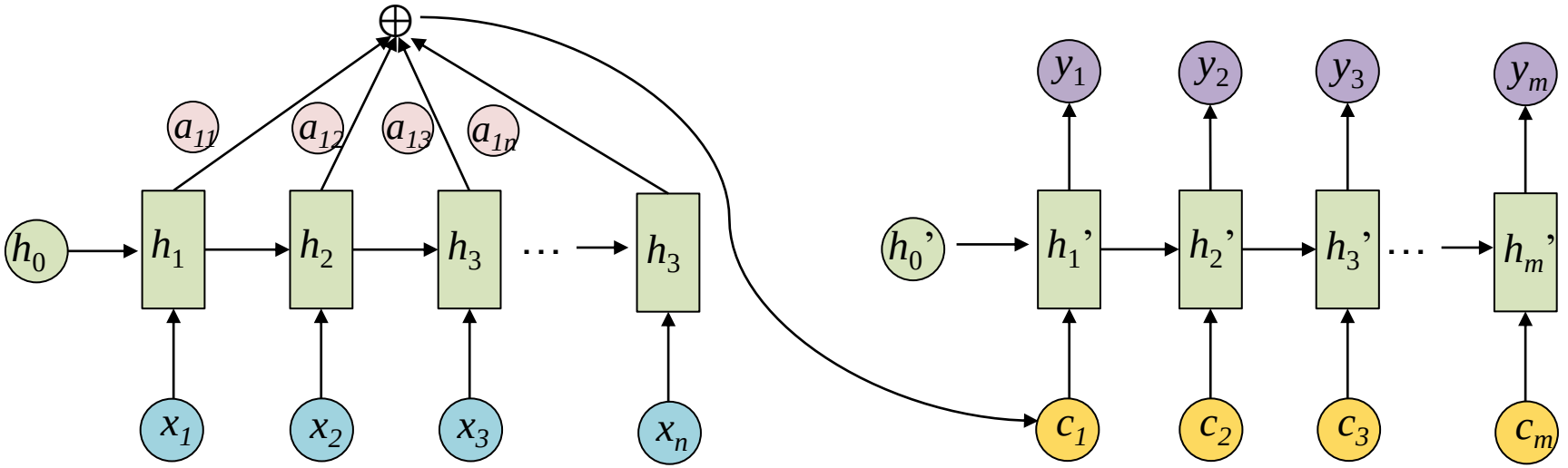
Attention Mechanism

- 解决方案
  - 解码器中的每个时刻不是输入固定的c，而是输入不同的c_i
  - 每个时刻的c自动选取与当前输出最相关的上下文



如何计算？

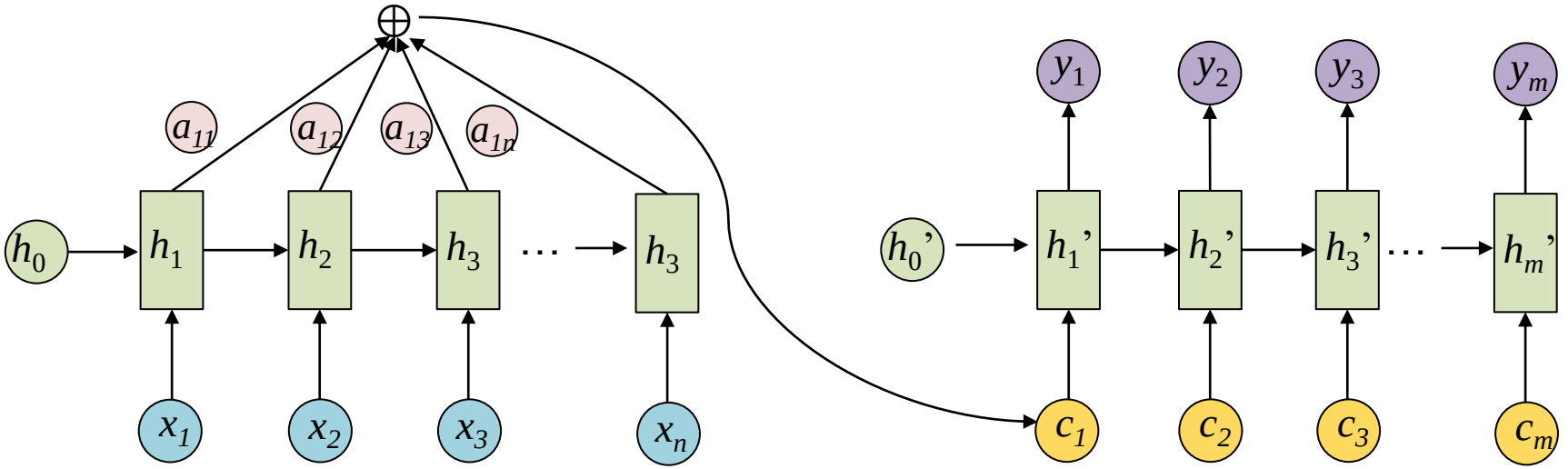
C_i 如何计算?



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

- c_i是编码器中隐状态的加权和
- a_{ij}是目标词y_i与源词x_j对齐的概率

C_i 如何计算?



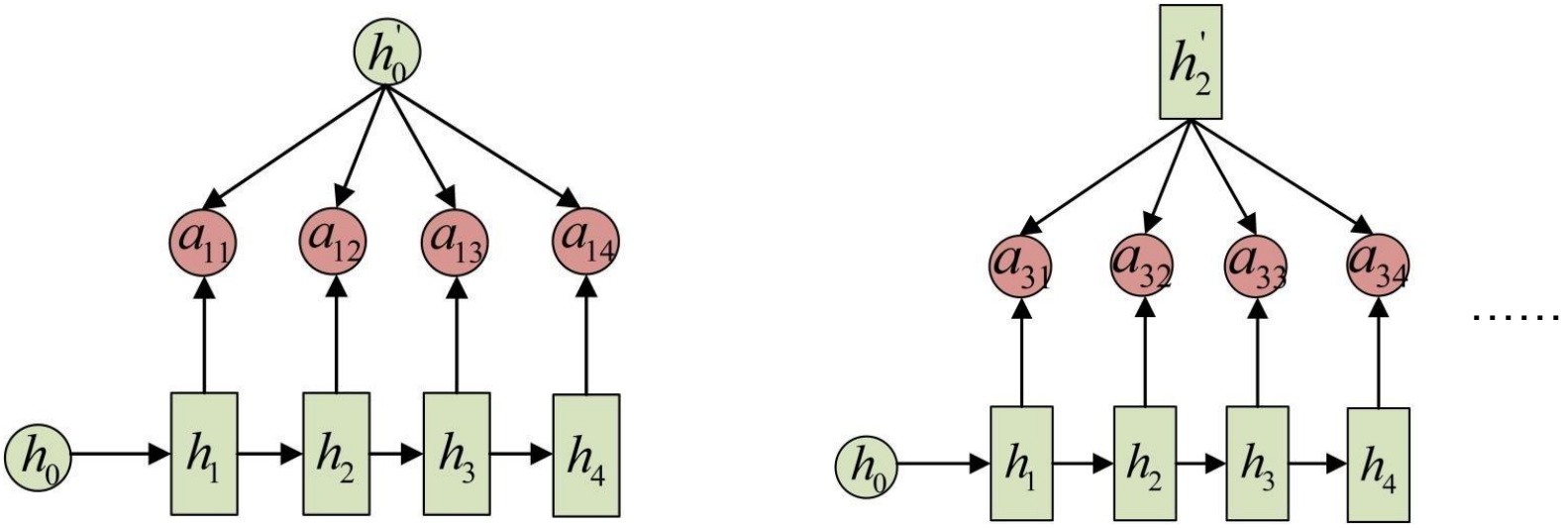
Today empire is divided in three

$h_1 * a_{11} + h_2 * a_{12} + h_3 * a_{13} + h_4 * a_{14} + h_5 * a_{15} + h_6 * a_{16} = c_1 \rightarrow$ 今

$h_1 * a_{21} + h_2 * a_{22} + h_3 * a_{23} + h_4 * a_{24} + h_5 * a_{25} + h_6 * a_{26} = c_2 \rightarrow$ 天下

$h_1 * a_{31} + h_2 * a_{32} + h_3 * a_{33} + h_4 * a_{34} + h_5 * a_{15} + h_6 * a_{36} = c_1 \rightarrow$ 三分

a_{ij} 如何计算?



$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \quad e_{ij} = \varphi(h'_{i-1}, h_j) = V^T \tanh(Wh'_{i-1} + Uh_j)$$

- a_{ij} 是目标词y_i与源词x_j对齐的概率
- e_{ij} 是a_{ij}对应的能量函数
- φ 是一个对齐模型，用于衡量j位置输入与i位置输出的匹配程度

AM in Machine Translation: Model

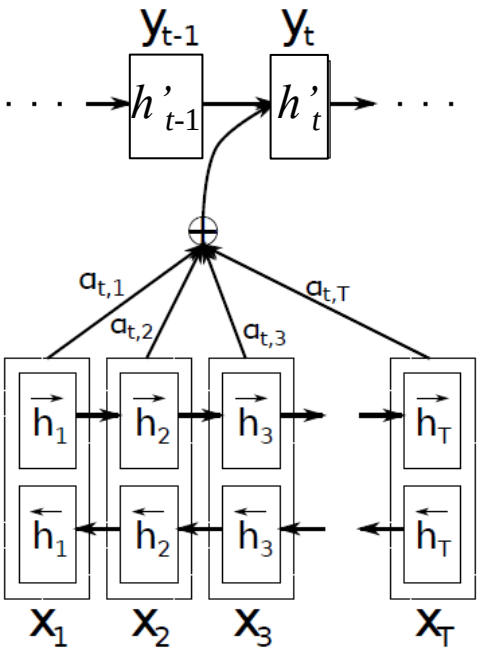


Figure 1: The graphical illustration of the proposed model trying to generate the t-th target word y_t given a source sentence $(x_1, x_2, \ldots, x_T)$.

Decoder

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

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

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

模型结构

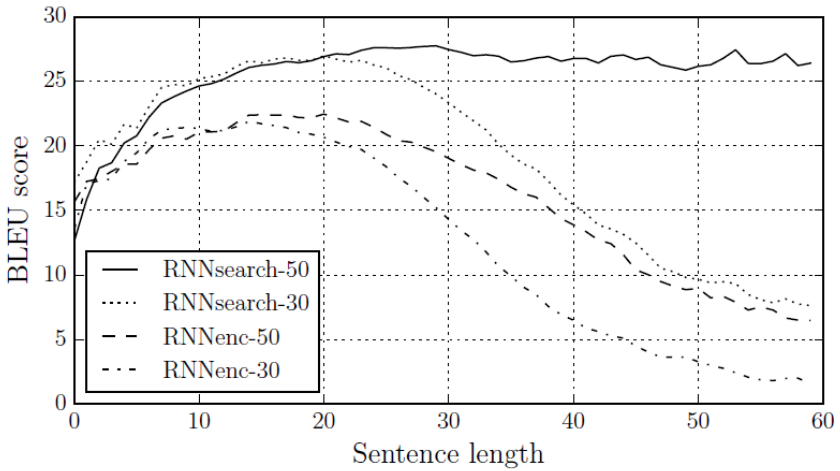
- 编码器是双向RNN，解码器是单向RNN
- 编码器和解码器均是多层网络，基本单元是maxout激活函数
- 编码器和解码器的隐层均有1000个隐单元

训练方式

- 采用mini-batch SGD算法，minibatch大小为80
- 训练样本长度最大值分别是30、50

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

AM in Machine Translation: Results



评价指标

BLEU值

- 对待评价译文和参考译文的n-gram进行比较，计算出匹配n-gram的个数，匹配数目越多模型效果越好

$$BLEU = \mathrm{BP} \cdot \exp(\sum_{n=1}^{N} w_n \log p_n)$$

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

实验结果

- 在句子限长为30和50的情况下，都是AM模型效果较好
- 句子长度增加时，seq2seq模型和使用AM模型效果均变差，但AM模型鲁棒性较好

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

AM in Machine Translation: Results

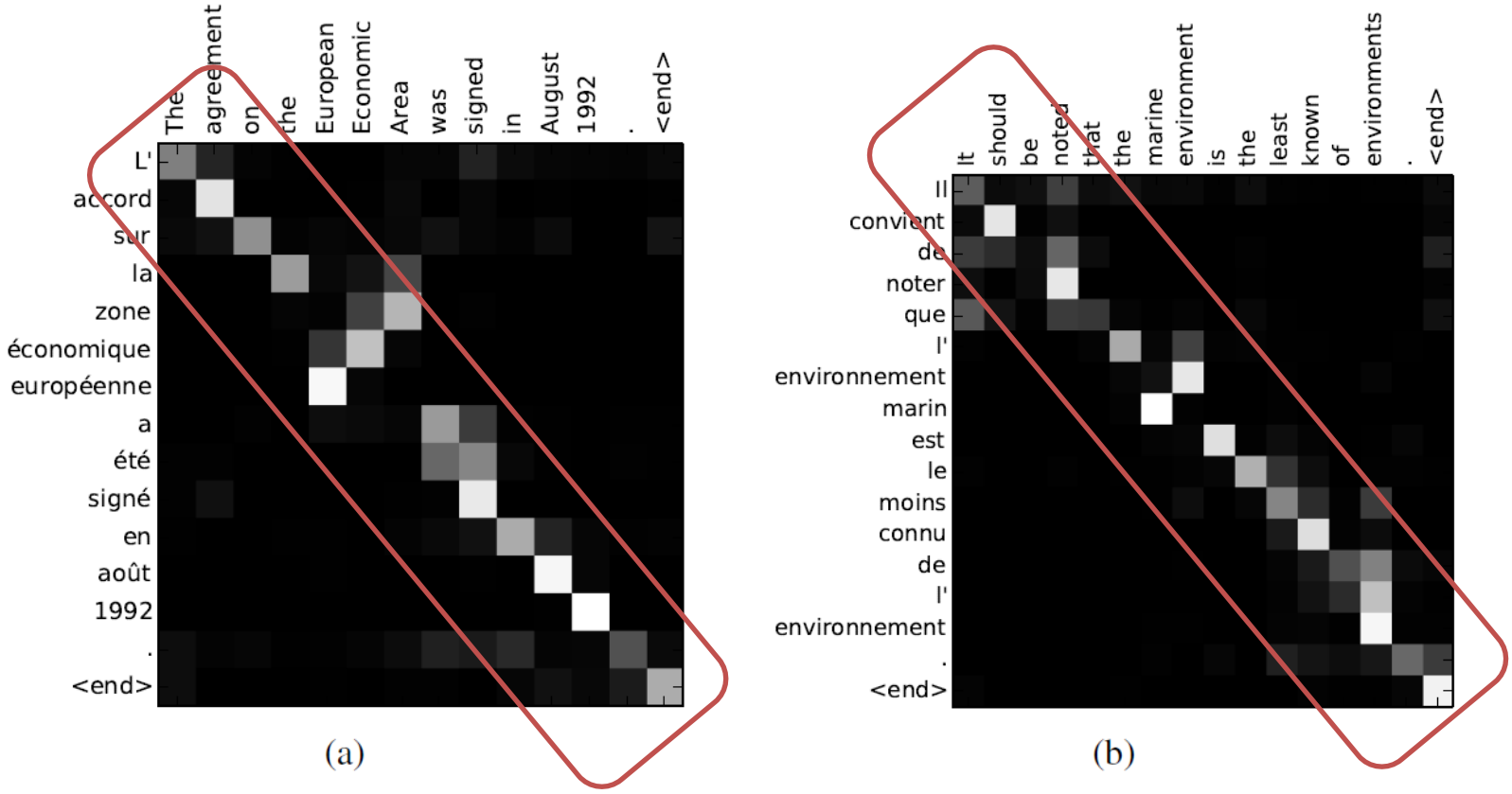
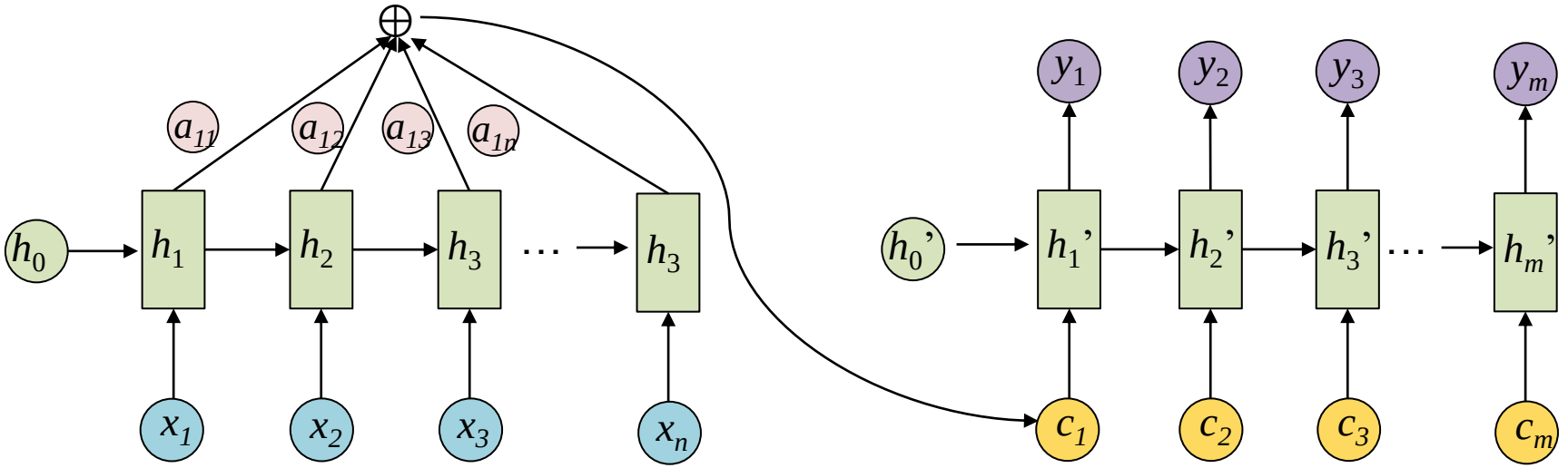


图. 展示注意力机制的双语对齐（英语→法语）效果，每个像素的灰度表示第j个源词对于生成第i个目标词的权重a_{ij}（0:黑，1:白）

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

4. NLP中的注意力机制

注意力机制



- c_i是编码器中隐状态的加权和

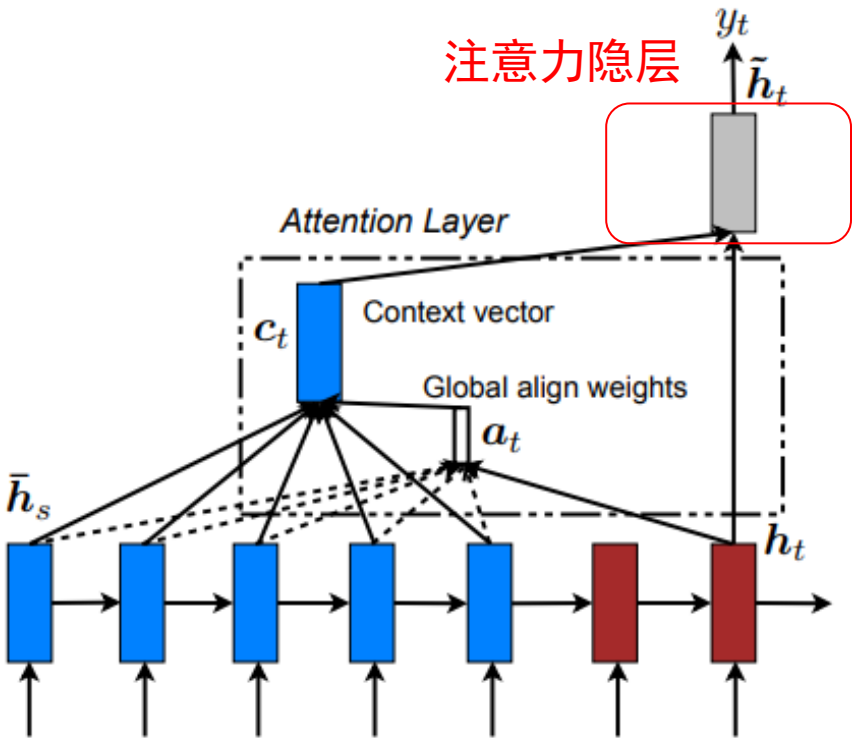
$$c_i = \sum_{j=1}^{T_x} a_{ij} h_j$$

- a_{ij}是目标词y_i与源词x_j对齐的概率

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \quad e_{ij} = \varphi(h'_{i-1}, h_j) = V^T \tanh(W h'_{i-1} + U h_j)$$

Global Attention

- 预测当前解码时要考虑输入序列中的所有词



- 不同点1

$$\tilde{h}_t = \tanh(W_c[c_t; h_t])$$

- 不同点2

$$a_t(s) = \text{align}(h_t, \bar{h}_s) = \frac{\exp\left(\text{score}(h_t, \bar{h}_s)\right)}{\sum_{s'} \exp\left(\text{score}(h_t, \bar{h}_{s'})\right)}$$

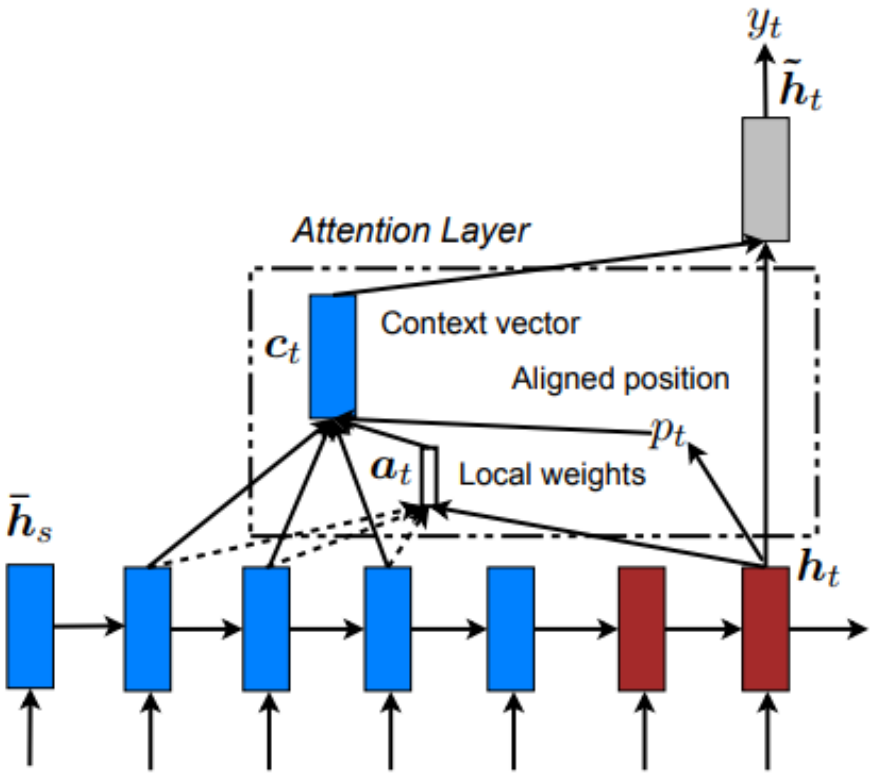
$$\text{score}(h_t, \bar{h}_s) = \begin{cases} h_t^\top \bar{h}_s & dot \\ h_t^\top W_a \bar{h}_s & general \\ W_a[h_t; \bar{h}_s] & concat \end{cases}$$

$$a_t = \text{softmax}(W_a h_t) \qquad location$$

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

Local Attention

- 预测当前解码时要对齐的源语言端的位置P
- 以p_t位置单词为轴心，建立大小为D的上下文窗口，仅考虑窗口内的词



- Monotonic alignment(local-m)

$$p_t = t$$

$$\boldsymbol{a}_t(s) = \text{align}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) = \frac{\exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s)\right)}{\sum_{s'} \exp\left(\text{score}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_{s'})\right)}$$

- Predictive alignment(local-p)

$$p_t = S \cdot \text{sigmoid}(\boldsymbol{v}_p^\top \tanh(\boldsymbol{W}_p \boldsymbol{h}_t))$$

$$\boldsymbol{a}_t(s) = \text{align}(\boldsymbol{h}_t, \bar{\boldsymbol{h}}_s) \exp\left(-\frac{(s - p_t)^2}{2\sigma^2}\right)$$

Global vs. Local Attention

System	Ppl	BLEU	
		Before	After unk
global (location)	6.4	18.1	19.3 (+1.2)
global (dot)	6.1	18.6	20.5 (+1.9)
global (general)	6.1	17.3	19.1 (+1.8)
local-m (dot)	>7.0	x	x
local-m (general)	6.2	18.6	20.4 (+1.8)
local-p (dot)	6.6	18.0	19.6 (+1.9)
local-p (general)	**5.9**	**19**	**20.9 (+1.9)**

Table 4: **Attentional Architectures** – performances of different attentional models. We trained two local-m (dot) models; both have ppl > 7.0.

Method	AER
global (location)	0.39
local-m (general)	0.34
local-p (general)	0.36
ensemble	0.34
Berkeley Aligner	0.32

一对多对齐

Table 6: **AER scores** – results of various models on the RWTH English-German alignment data.

结 论

- Global和Local Attention相比，Local Attention效果较好
- Attention的各类计算方法相比，general效果较好

4. NLP中的注意力机制

AM for Answer Selection

- 任务：给定一个问题，从候选句子集合中选出答案

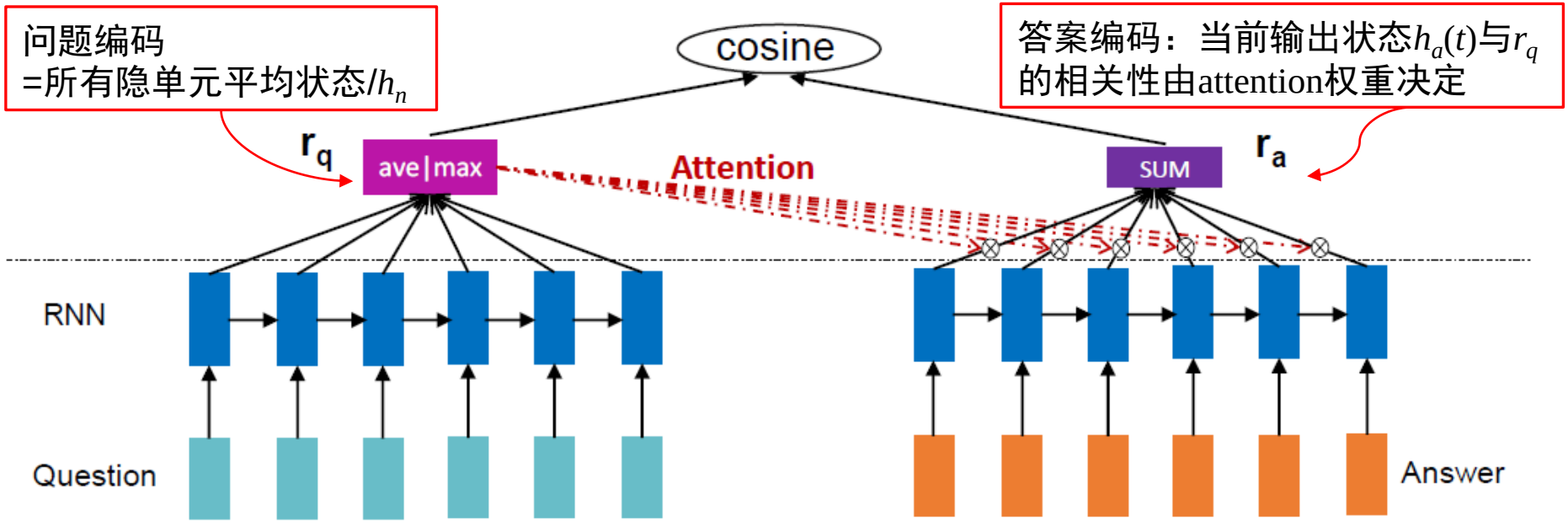
例　子

Question1：When did Michael Jordan retired form NBA?

Question2：Which sports does Michael Jordan participates after his retirement from NBA?

Answer：Michael Jordan abruptly retired from Chicago Bulls before the beginning of the 1993-94 NBA season to pursue a career in baseball.

Traditional Attention Model



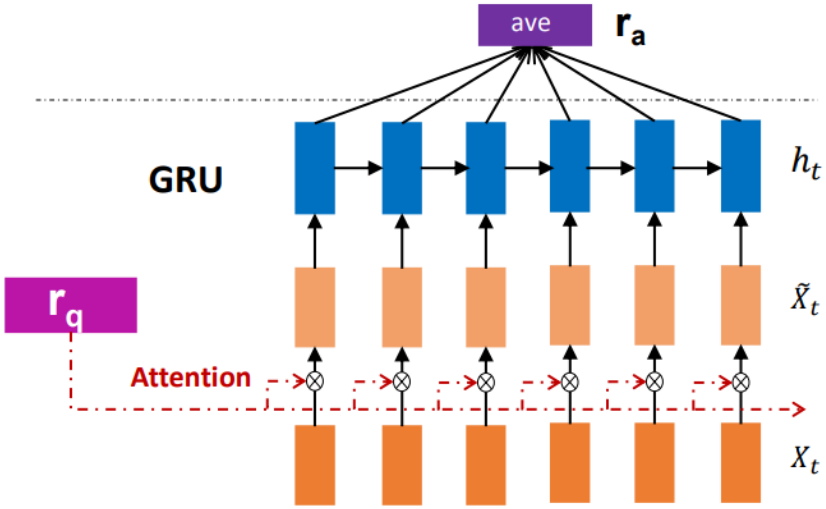
- 问题：由于越往后的隐状态包含的信息越多，容易发生attention bias，即后面的隐状态容易得到较高的权重

  eg. Michael Jordan abruptly retired from Chicago Bulls before the beginning of the 1993-94 NBA season to pursue a career in baseball.

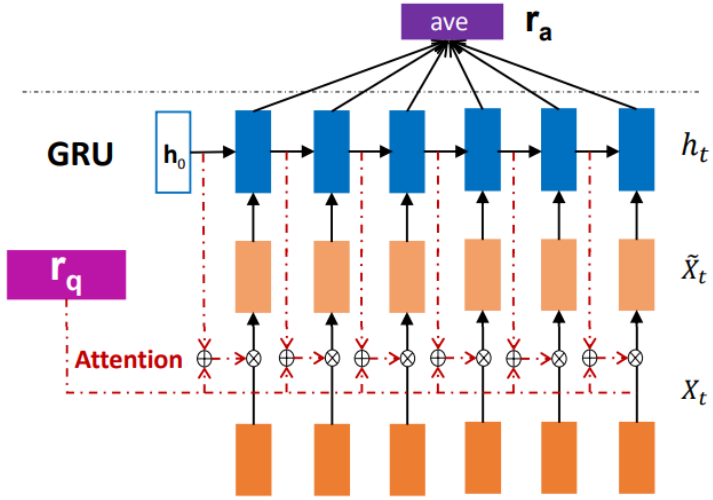
Inner Attention based Recurrent Nerual Networks for Answer Selection. ACL 2016.

Inner-Attention

- 解决方法
  - 把Attention层从RNN后移到RNN前，直接衡量答案句子中的词对r_q的贡献



IARNN-WORD



IARNN-Context

$$\alpha_t = \sigma(\mathbf{r}_q^T \mathbf{M}_{qi} \mathbf{x}_t)$$

$$\tilde{\mathbf{x}}_t = \alpha_t * \mathbf{x}_t$$

考虑上文信息

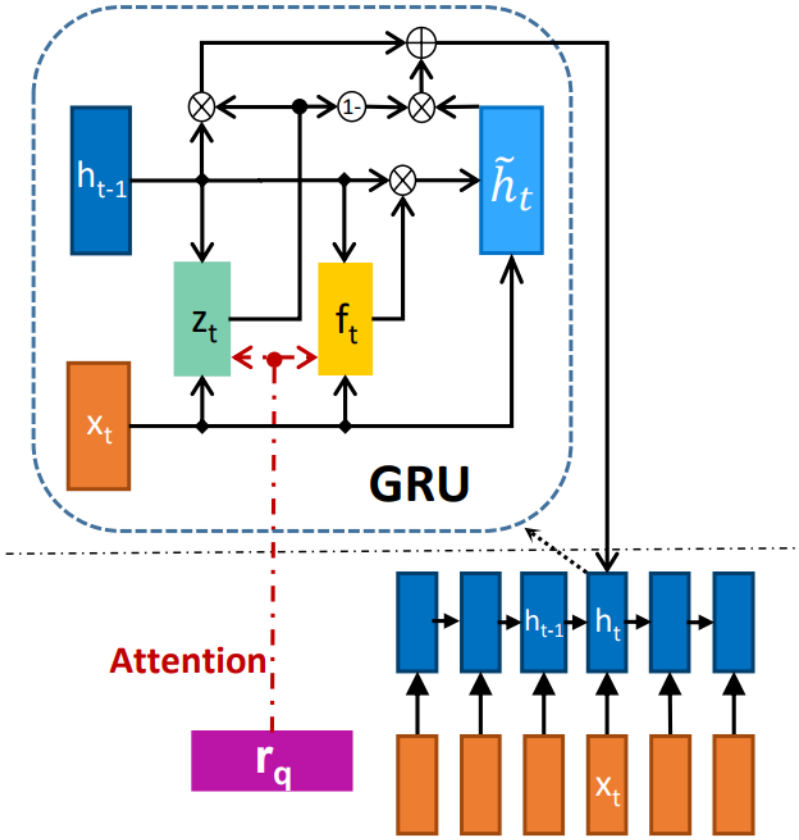
$$\mathbf{w}_C(t) = \mathbf{M}_{hc}\mathbf{h}_{t-1} + \mathbf{M}_{qc}\mathbf{r}_q$$

$$\alpha_C^t = \sigma(\mathbf{w}_C^T(t)\mathbf{x}_t)$$

$$\tilde{\mathbf{x}}_t = \alpha_C^t * \mathbf{x}_t$$

Inner Attention based Recurrent Nerual Networks for Answer Selection. ACL 2016.

Inner-Attention



IARNN-GATE

- 由于GRU中的门状态控制着网络中的信息流，并且能够让信息保持长距离传递，在重置门和更新门后加Attention层

$$\mathbf{z}_t = \sigma(\mathbf{W}_{xz}\mathbf{x}_t + \mathbf{W}_{hz}\mathbf{h}_{t-1} + \mathbf{M}_{qz}\mathbf{r}_q)$$
$$\mathbf{f}_t = \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{M}_{qf}\mathbf{r}_q)$$
$$\tilde{\mathbf{h}}_t = tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}(\mathbf{f}_t \odot \mathbf{h}_{t-1}))$$
$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t$$

Inner Attention based Recurrent Nerual Networks for Answer Selection. ACL 2016.

IARNN-WORD vs. IARNN-Context vs. IARNN-GATE

System	MAP	MRR
(Yang et al., 2015)	0.652	0.6652
(Yin et al., 2015)	0.6921	0.7108
(Santos et al., 2016)	0.6886	0.6957
GRU	0.6581	0.6691
OARNN	0.6881	0.7013
IARNN-word	0.7098	0.7234
IARNN-Occam(word)	0.7121	0.7318
IARNN-context	0.7182	0.7339
IARNN-Occam(context)	**0.7341**	**0.7418**
IARNN-Gate	0.7258	0.7394

Table 2: Performances on WikiQA

System	Dev	Test1	Test2
(Feng et al., 2015)	65.4	65.3	61.0
(Santos et al., 2016)	66.8	67.8	60.3
GRU	59.4	53.2	58.1
OARNN	65.4	66.1	60.2
IARNN-word	67.2125	67.0651	61.5896
IARNN-Occam(word)	69.9130	69.5923	63.7317
IARNN-context	67.1025	66.7211	63.0656
IARNN-Occam(context)	69.1125	68.8651	**65.1396**
IARNN-Gate	**69.9812**	**70.1128**	62.7965

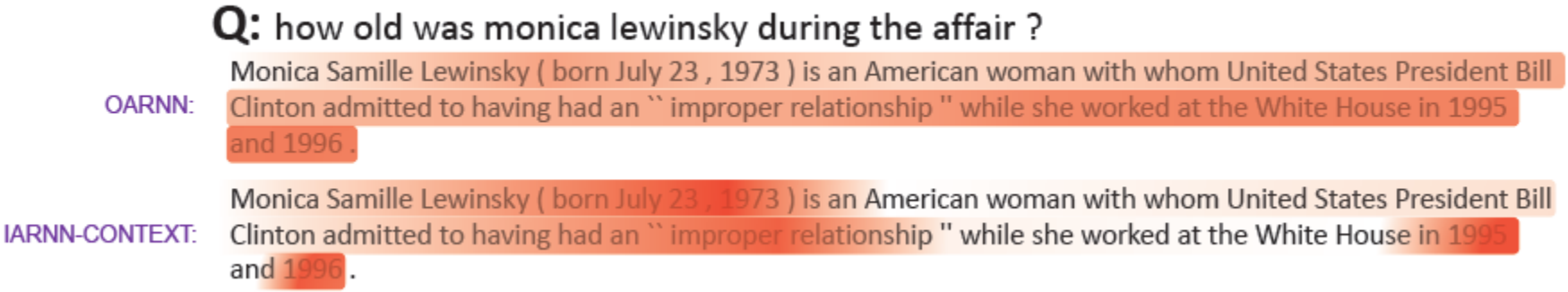
Table 3: Experiment result in InsuranceQA, (Feng et al., 2015) is a CNN architecture without attention mechanism.

实验结果

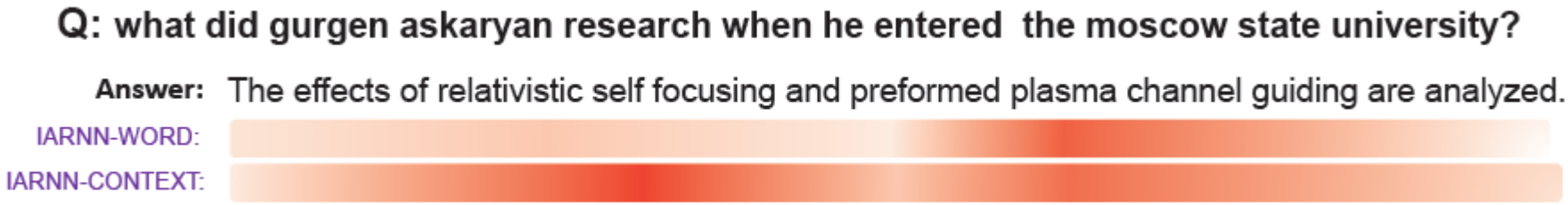
- 有attention比没有好
- Inner-attention比outer-attention好
- IARNN-GATE好于IARNN-Context好于IARNN-WORD

IARNN-WORD vs. IARNN-Context vs. IARNN-GATE

- IARNN-CONTEXT有效解决了OARNN中的attention bias问题

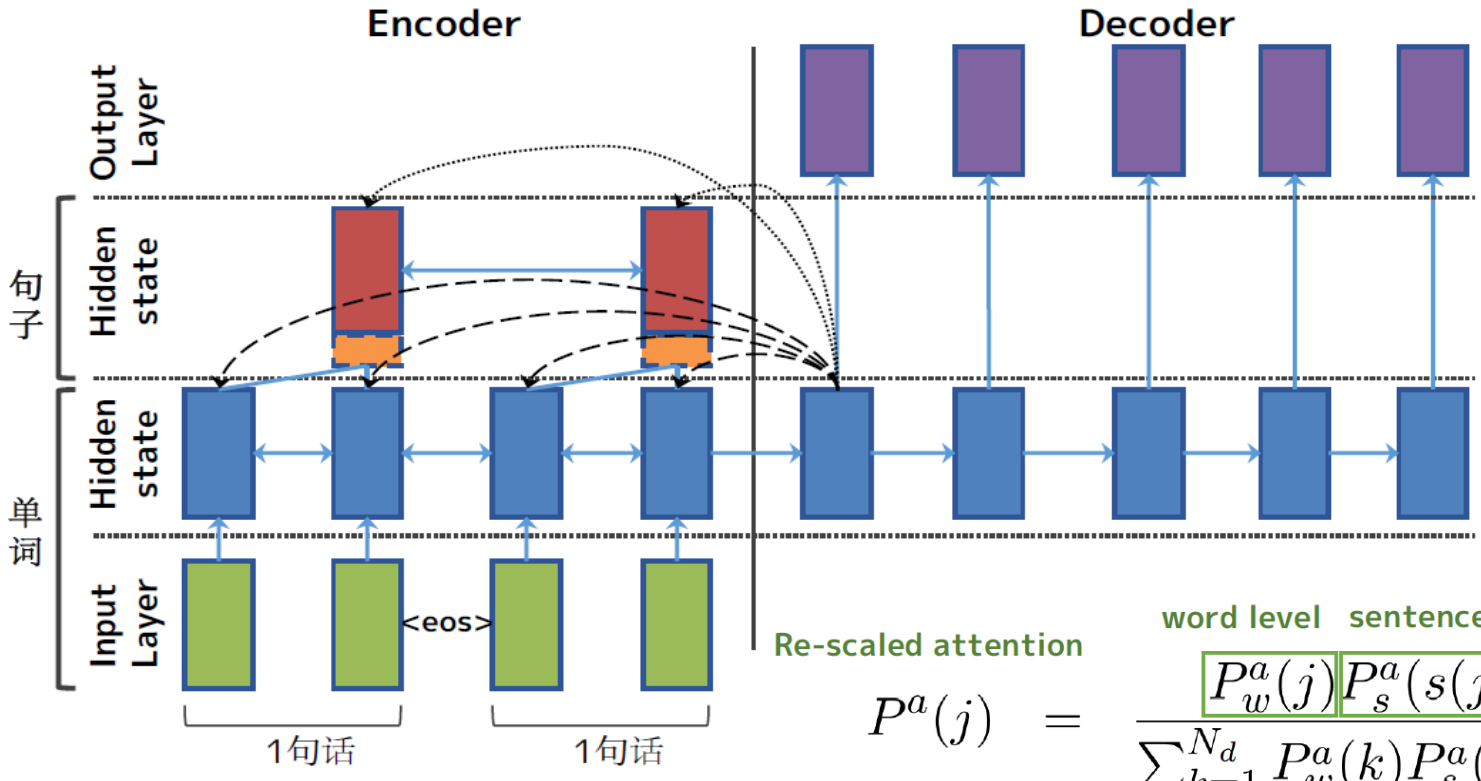


- IARNN-CONTEXT有效地识别了与问题相关的多词表达



Hierarchical Attention

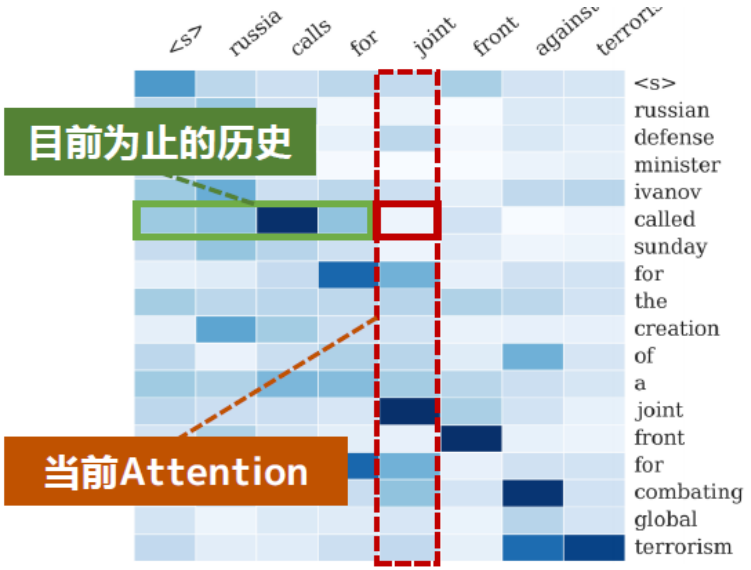
- 某些任务中不同句子的重要程度不同，在源端使用两个双向RNN分别捕获单词级和句子级两个级别的特征，在这两个层面上使用Attention



Nallapati, Ramesh, et al. "Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond." CoNLL 2016 (2016): 280.

Temporal Attention

- 问题：
  - 解码端容易出现重复的高频词：e.g. Germany beat Germany beat beat …
- 解决思想：
  - 解码时输出哪些词与Attention有关
  - 利用到目前为止的Attention历史信息，如果之前对某一个单词已经赋予过很高的Attention值，则适度降低这个值



<s> Russia calls for [ ]

$$\alpha_t \propto \frac{\alpha'_t}{\beta_t}$$

α'_t：词t当前的Attention

β_t：词t过去的Attention总和

$$\beta_t = \sum_{k=1}^{t-1} \alpha'_k$$

Attention Coverage Model [See+ 2017]
Intra-Attention Model [Paulus+ 2017]

Sankaran, Baskaran, et al. "Temporal attention model for neural machine translation." arXiv preprint arXiv:1608.02927 (2016).

4. NLP中的注意力机制

Transformer

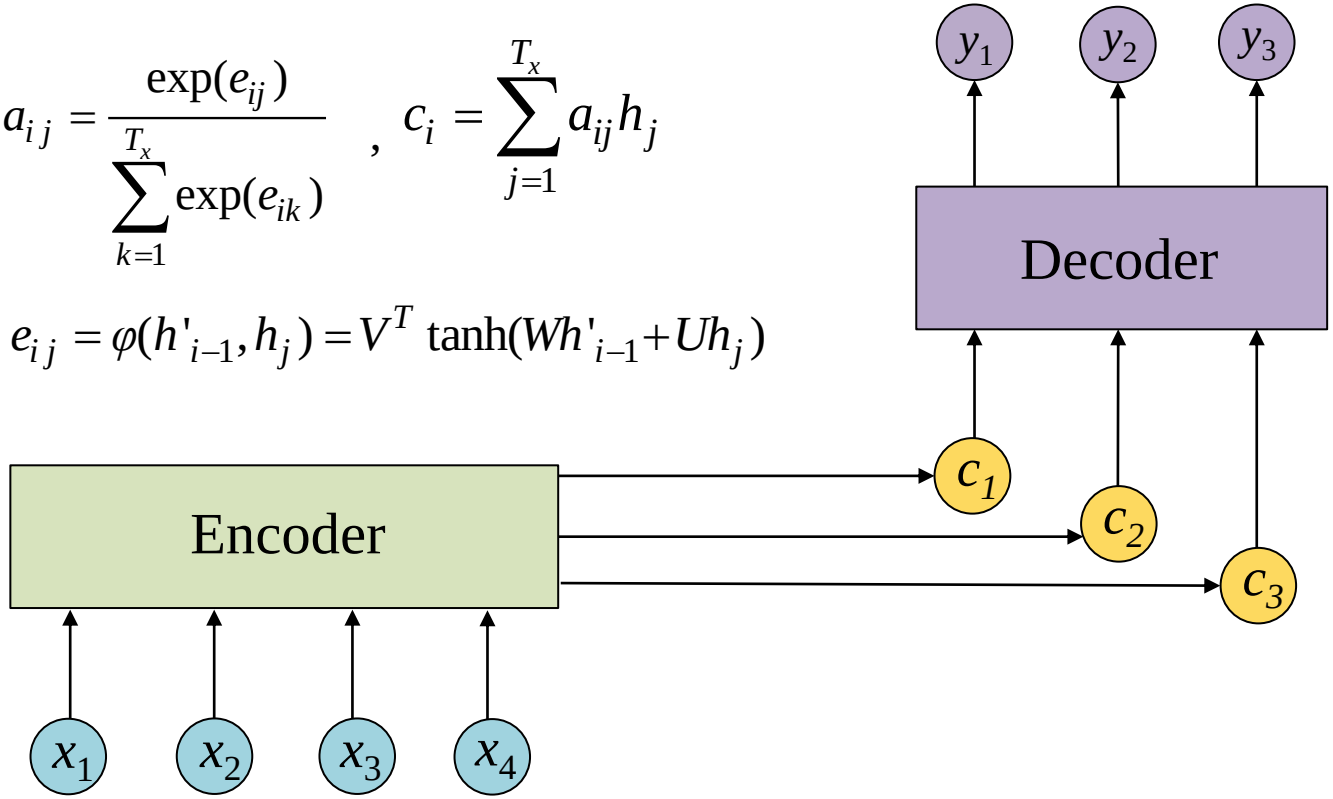
Are you kidding me?

你逗我呢?

Traditional Attention Model

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \quad c_i = \sum_{j=1}^{T_x} a_{ij} h_j$$

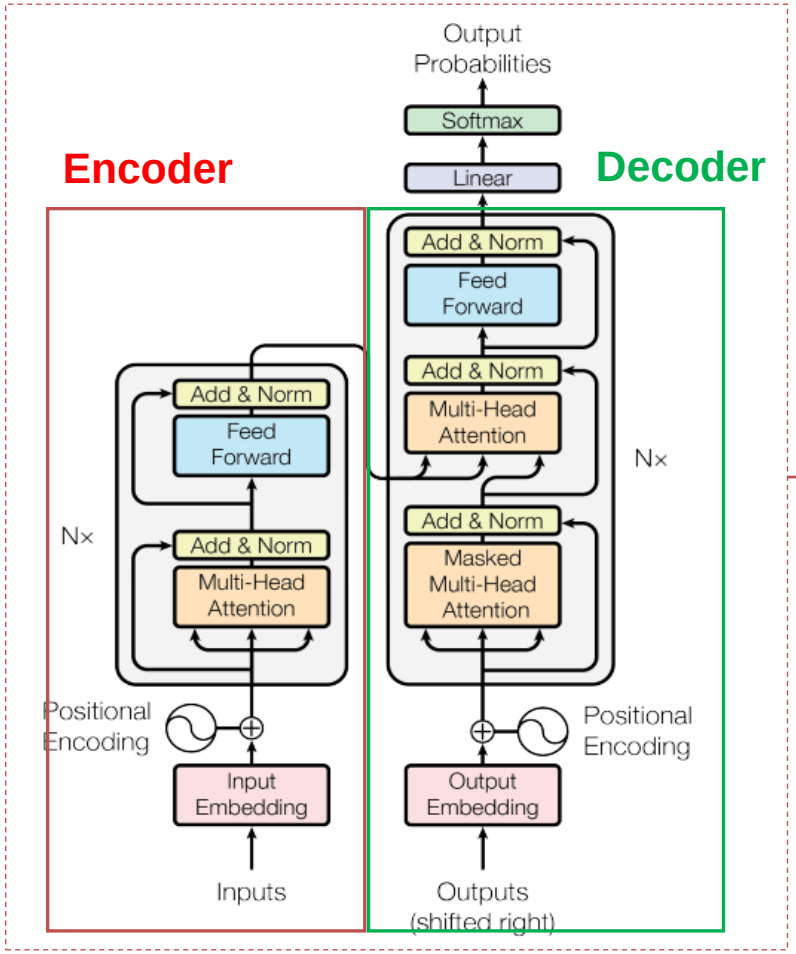
$$e_{ij} = \varphi(h'_{i-1}, h_j) = V^T \tanh(W h'_{i-1} + U h_j)$$

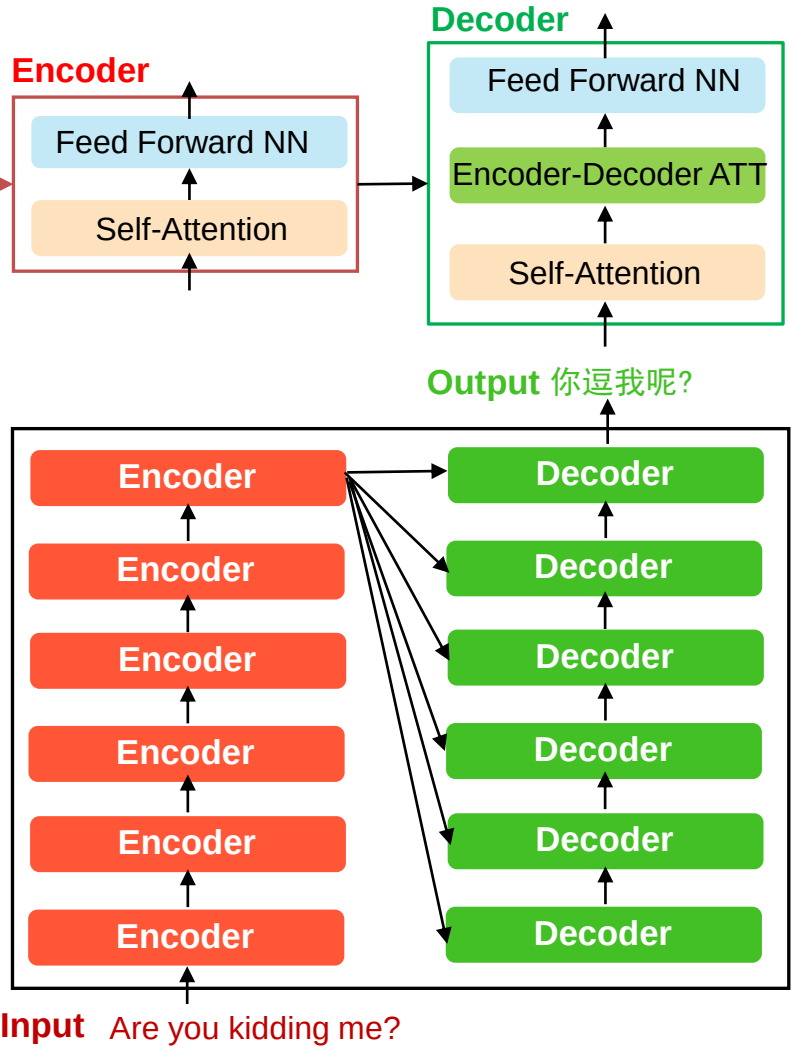


- RNN不能并行化，且忽略了句子内部词与词之间的关系，能否抛弃RNN？

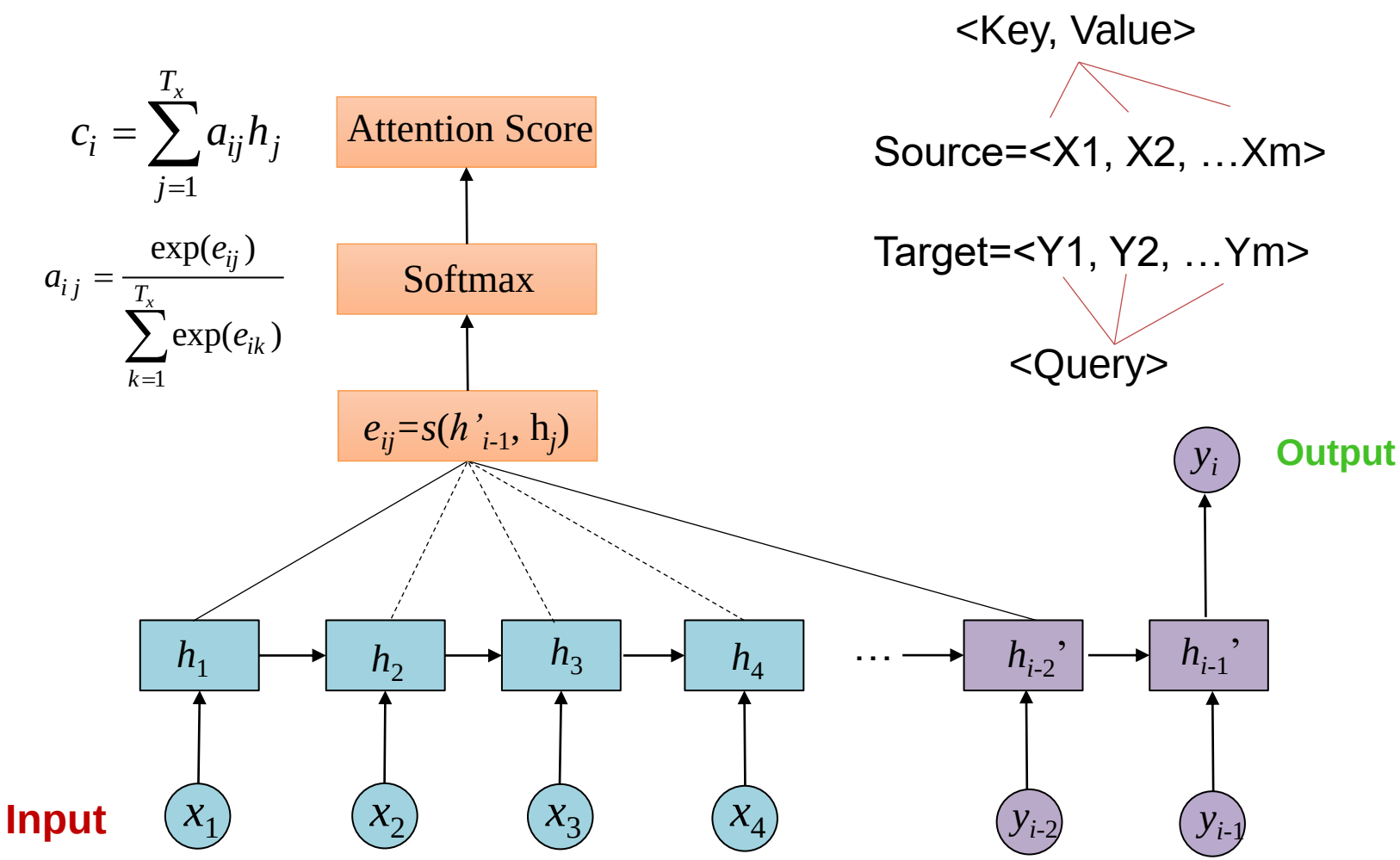
Google Brain: Attention is all you need (NIPS 2017)

- Transformer： 在encoder-decoder模型中舍弃 RNN/CNN，只用注意力模型来进行序列建模



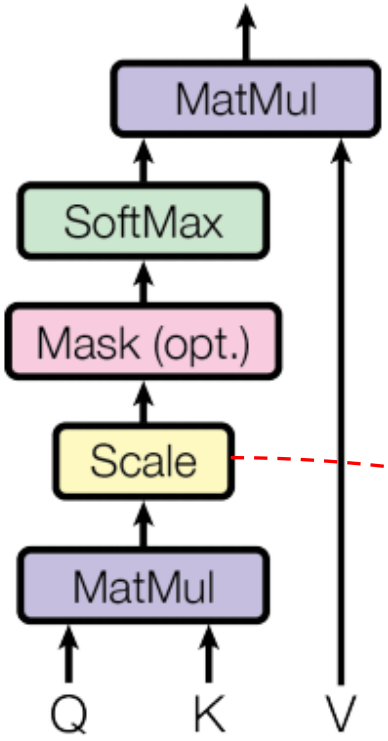


Key, Value, Query ?

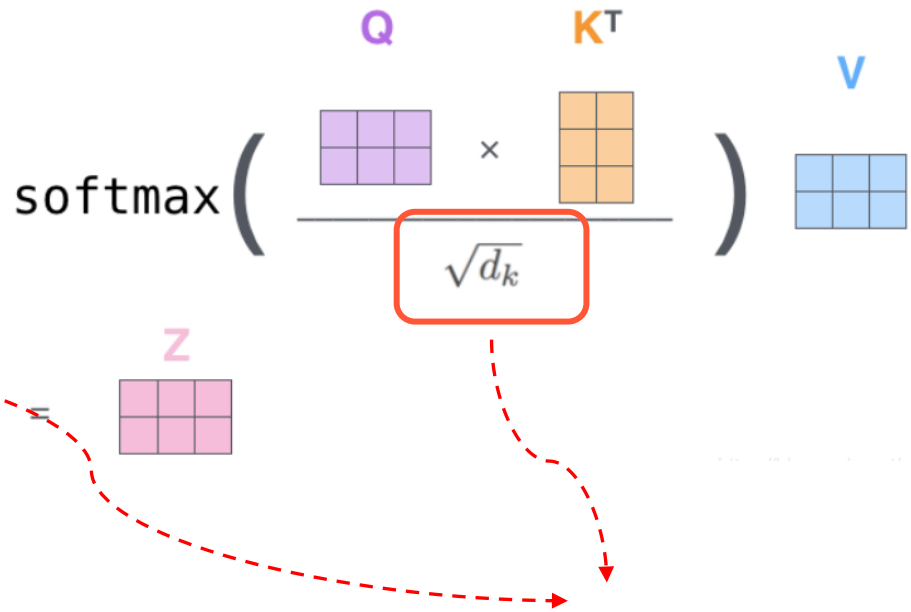


Scaled Dot-Product Attention

Scaled Dot-Product Attention

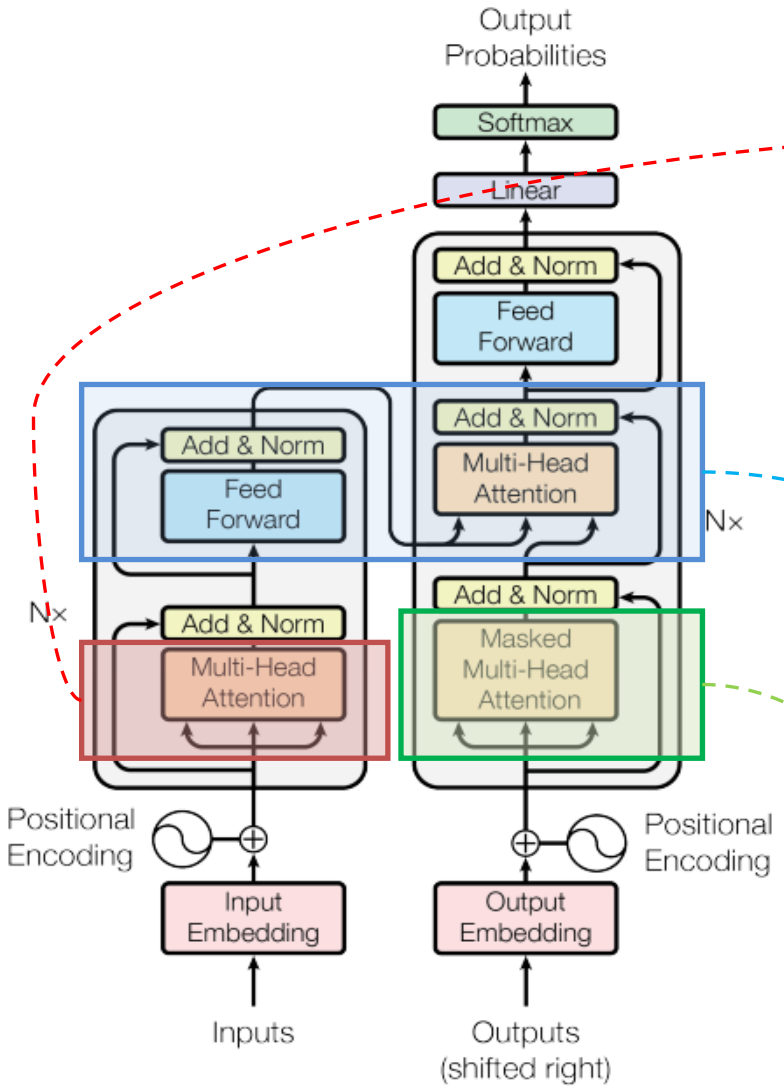


$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$



避免内积过大时，softmax产生的结果非0即1

Three kinds of Attention



Encoder self attention

1.Multi-head Attention

2.Query=Key=Value

Encoder-Decoder attention

1.Multi-head Attention
2.Encoder Self attention=Key=Value
2.Decoder Self attention=Query

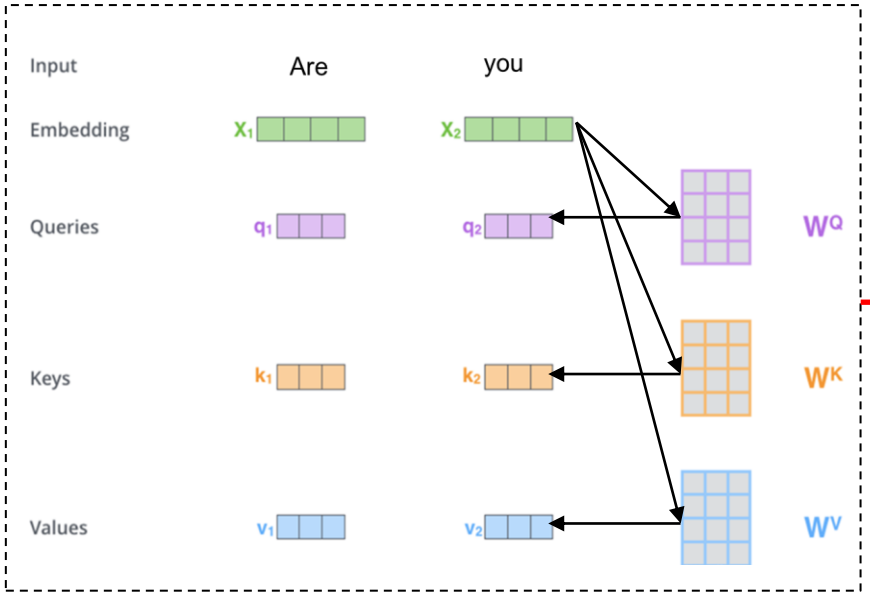
Decoder self attention

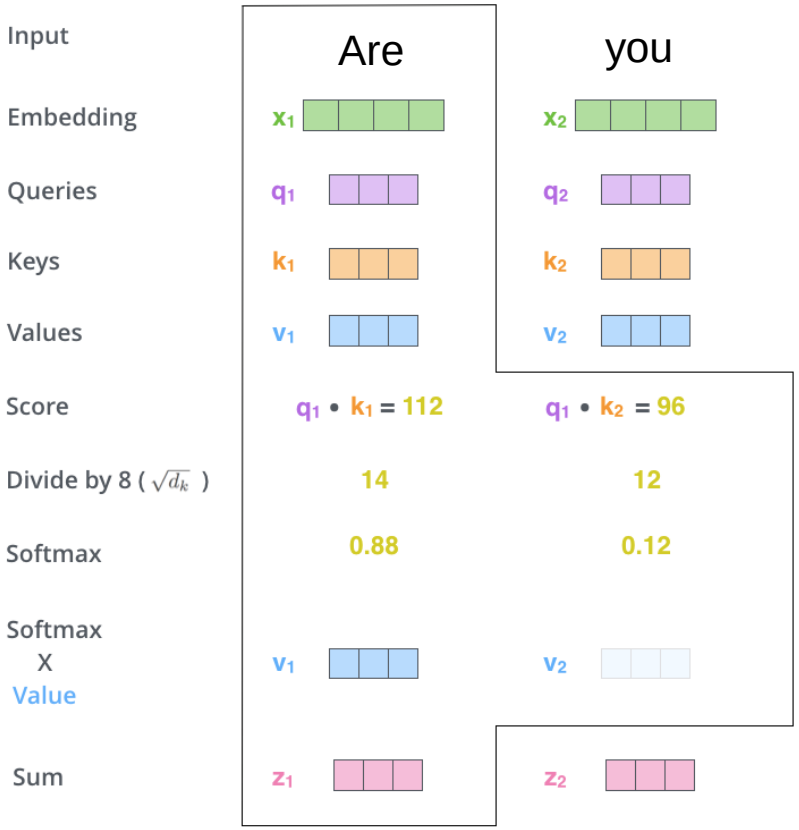
1.Masked Multi-head Attention

2.Query=Key=Value

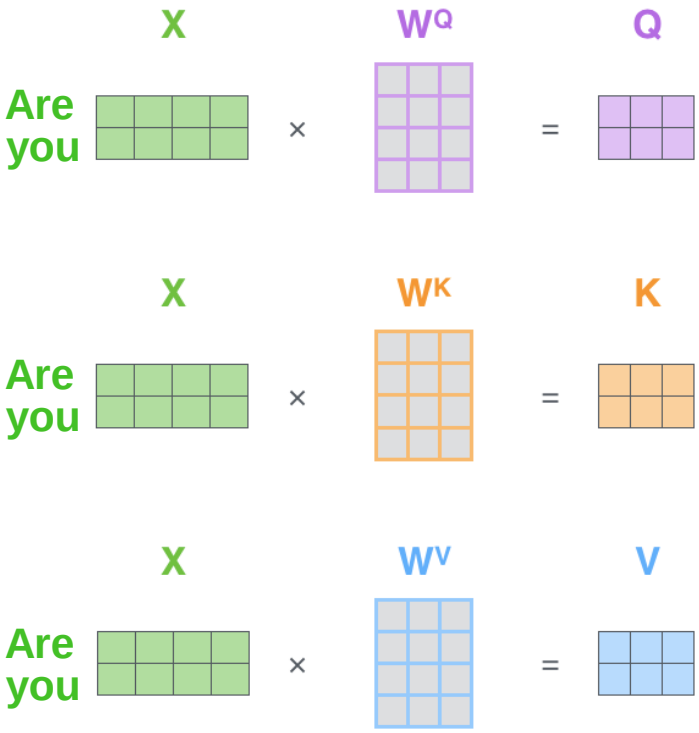
Encoder Self Attention

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

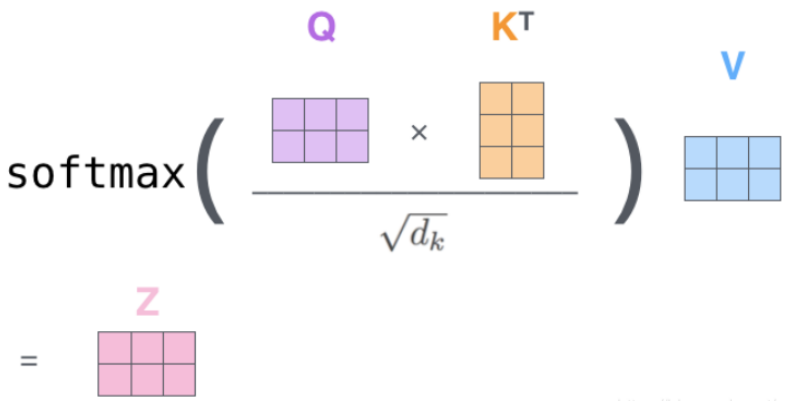




Matrix Calculation of Self-Attention

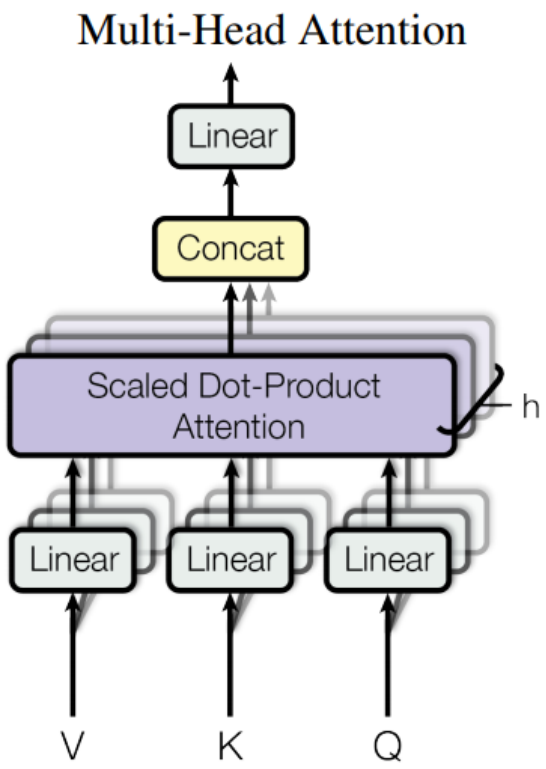


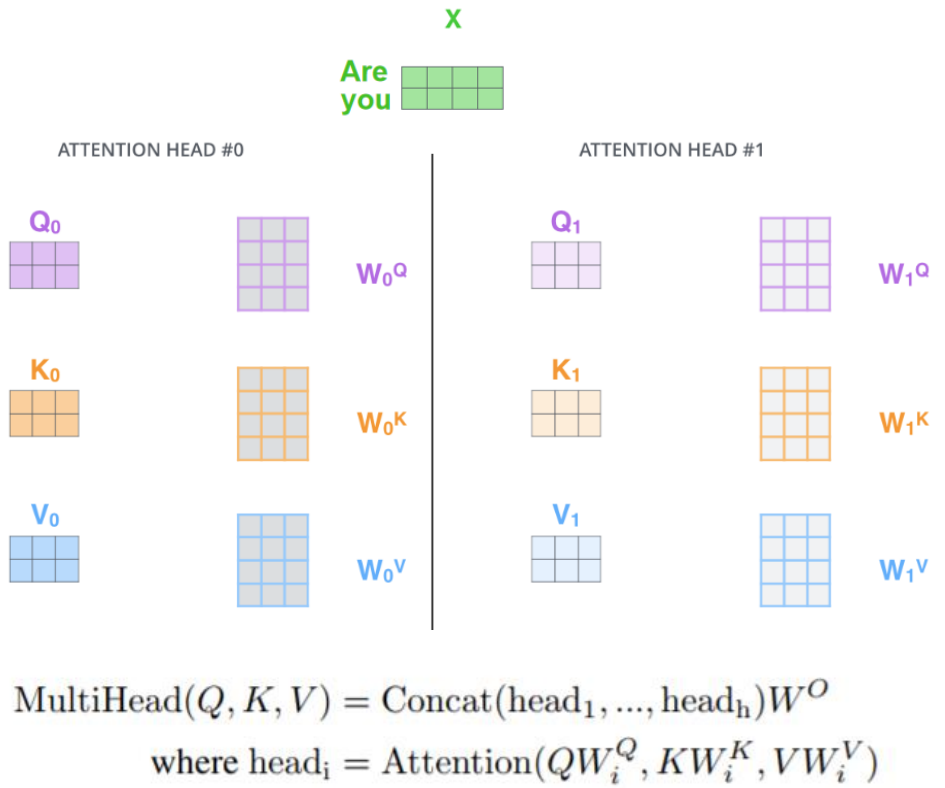
Attention(Q, K, V)=Attention(X, X, X)



Multi-head Attention

- 采用多个head，每个head有独立的Q/K/V的权值矩阵，不同的head为Attention层提供了不同的表示子空间，不同的表示关注不同位置





The animal didn't cross the street because it was too tired.

Eight Attention Head

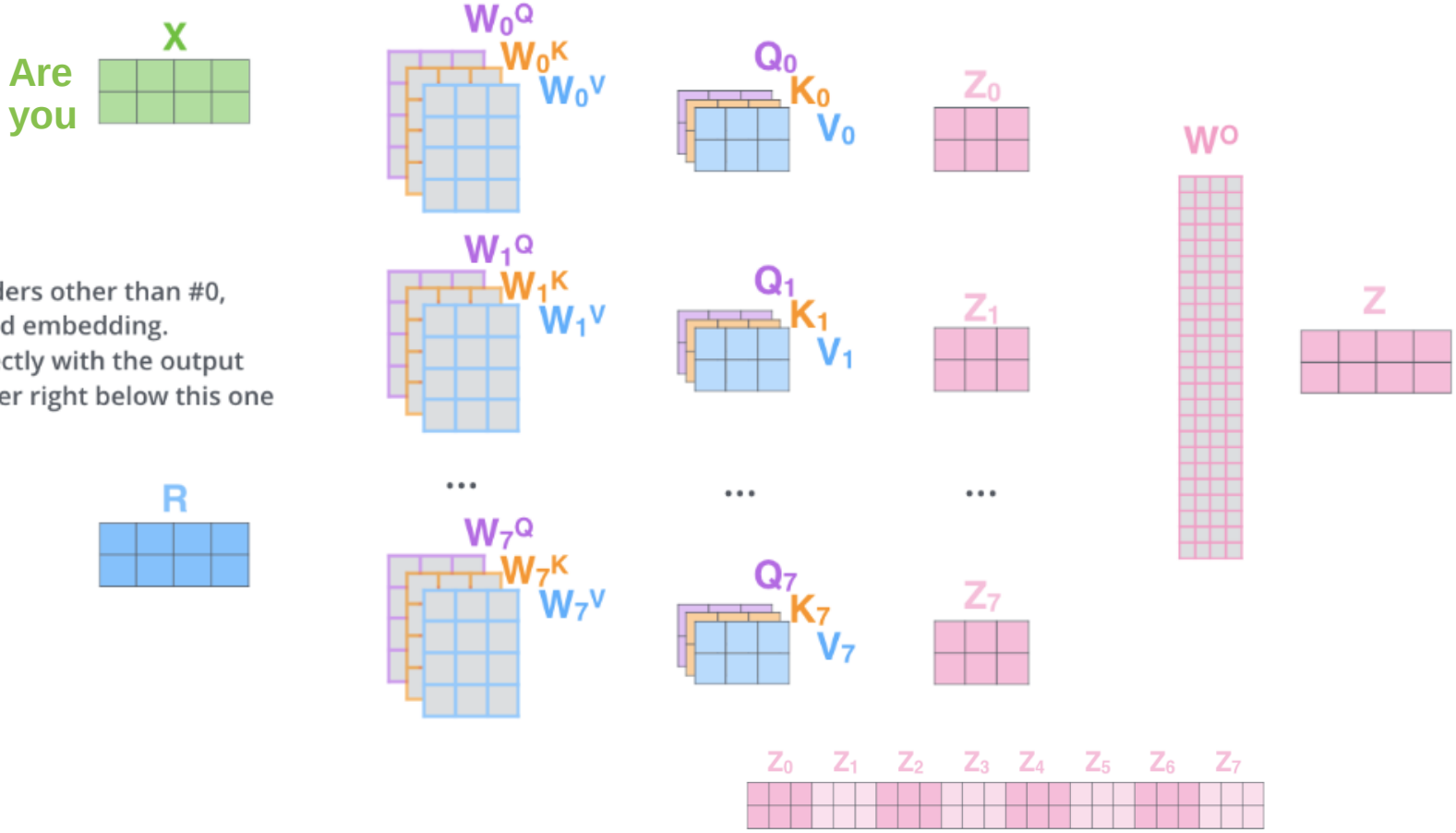
1) This is our input sentence*

2) We embed each word*

3) Split into 8 heads. We multiply X or R with weight matrices

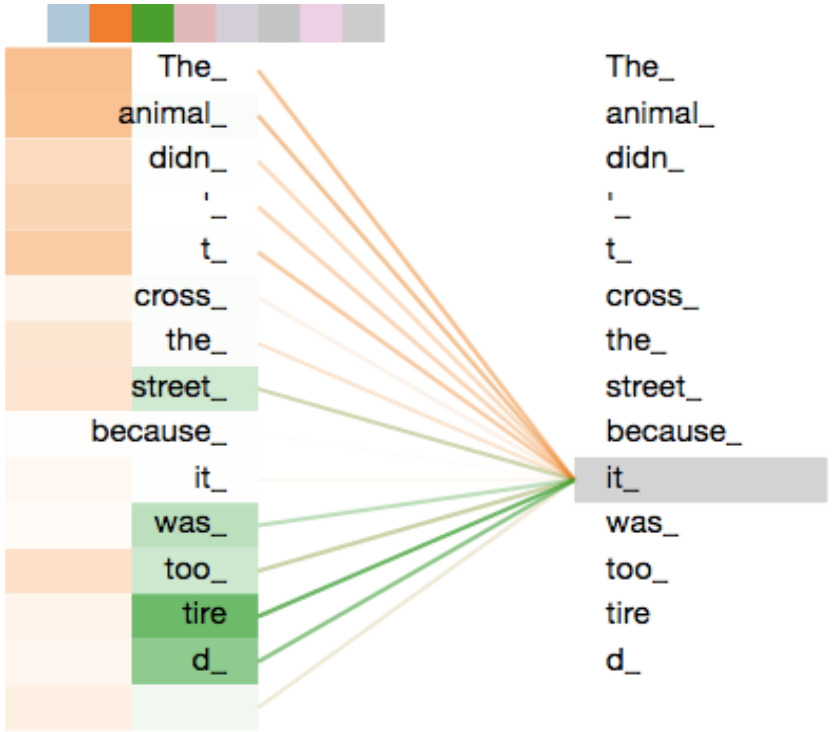
4) Calculate attention using the resulting Q/K/V matrices

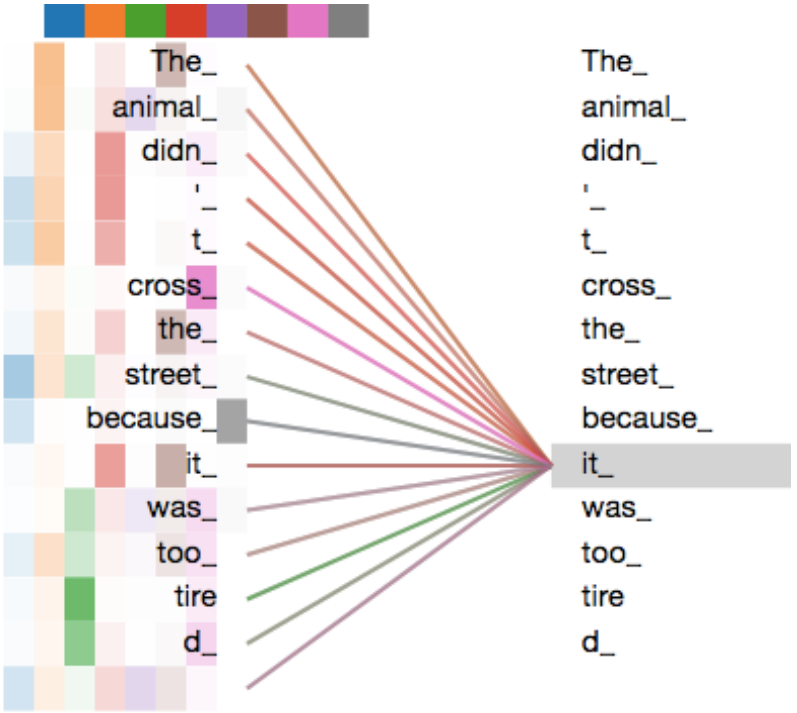
5) Concatenate the resulting Z matrices, then multiply with weight matrix W^O to produce the output of the layer



* In all encoders other than #0, we don't need embedding. We start directly with the output of the encoder right below this one

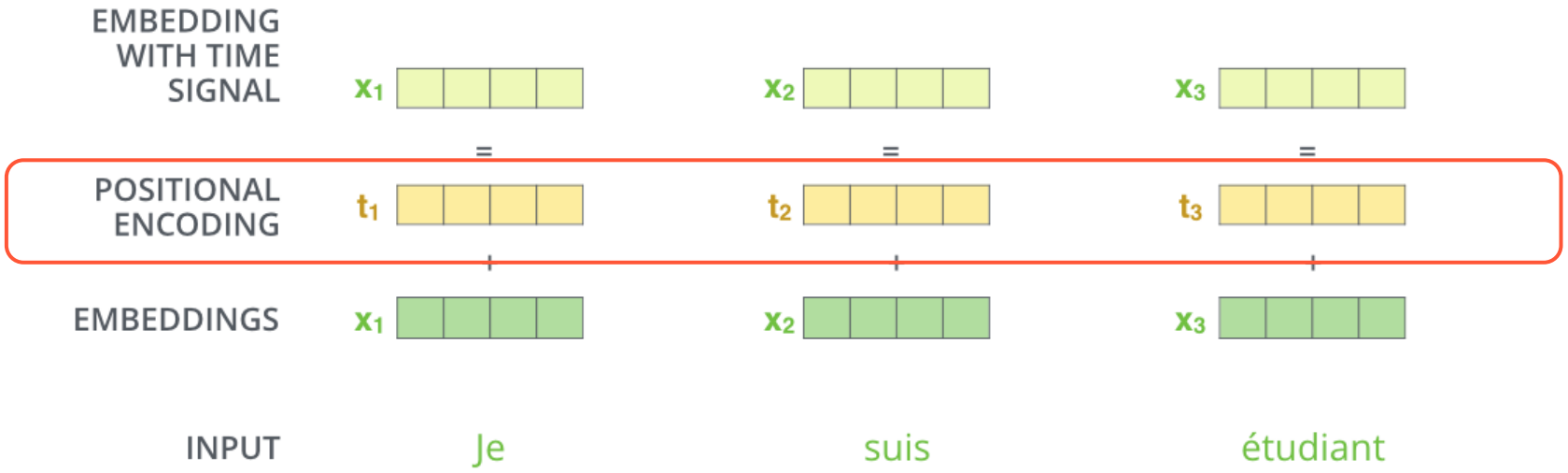
Multi-head Attention





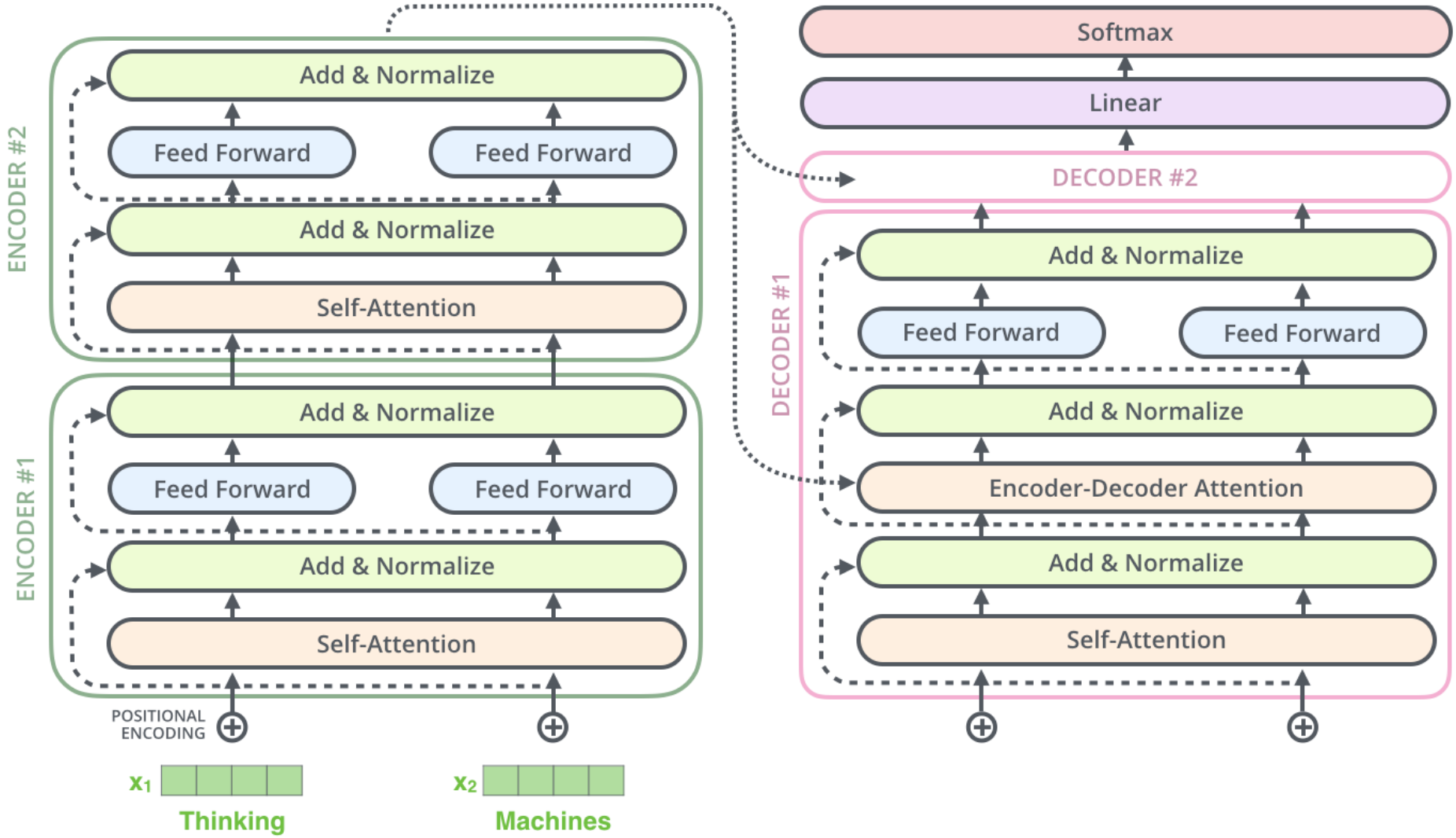
Positional Encoding

- 为了让模型捕捉到单词的顺序信息，在输入层添加位置编码向量信息。位置编码向量不需要训练，基于特定的规则产生

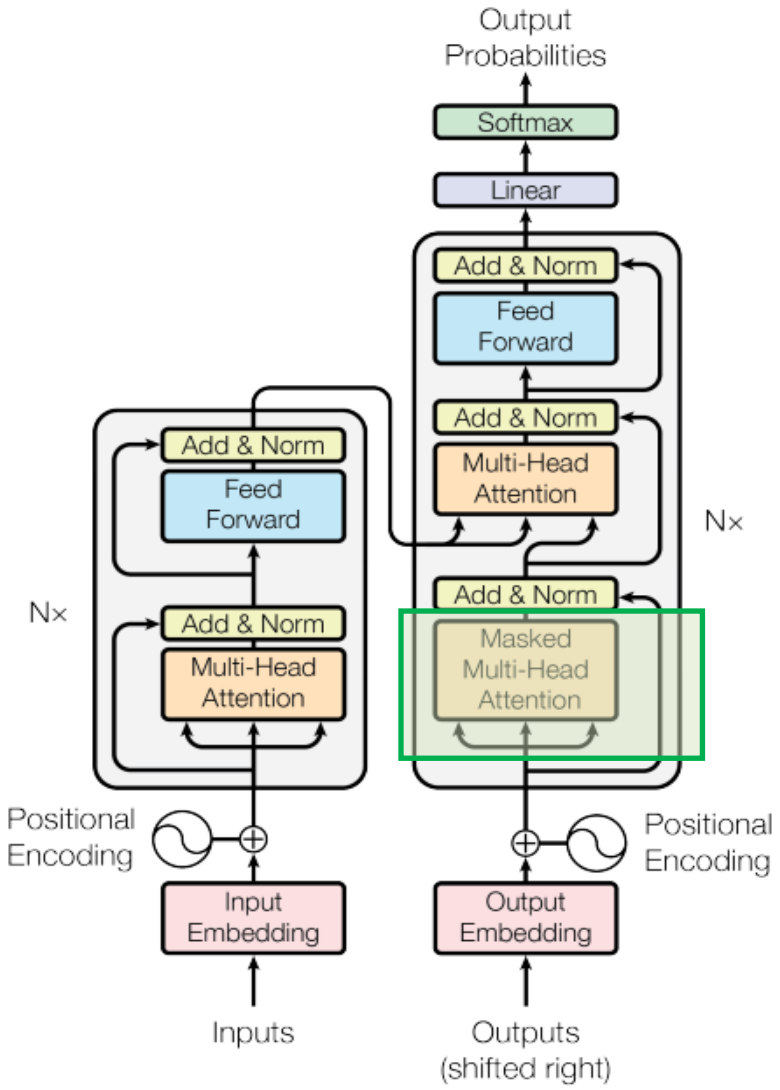


$$PE_{(pos,2i)} = sin(pos/10000^{2i/d_{\text{model}}})$$
$$PE_{(pos,2i+1)} = cos(pos/10000^{2i/d_{\text{model}}})$$

Residual network

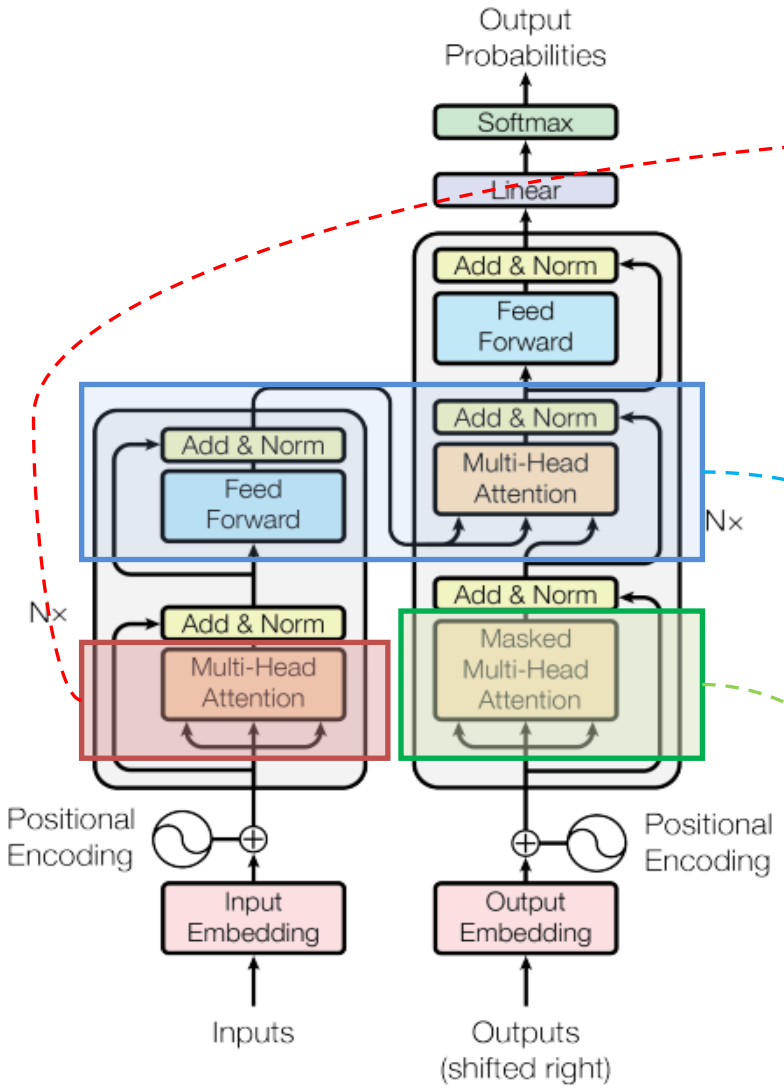


Decoder



- 每个时刻输出一个元素
- 使用Masked Multi-Head Attention：保证训练阶段和推理阶段的一致性
- 线性层+Softmax层

Three kinds of Attention



Encoder self attention

1.Multi-head Attention

2.Query=Key=Value

Encoder-Decoder attention

1.Multi-head Attention
2.Encoder Self attention=Key=Value
2.Decoder Self attention=Query

Decoder self attention

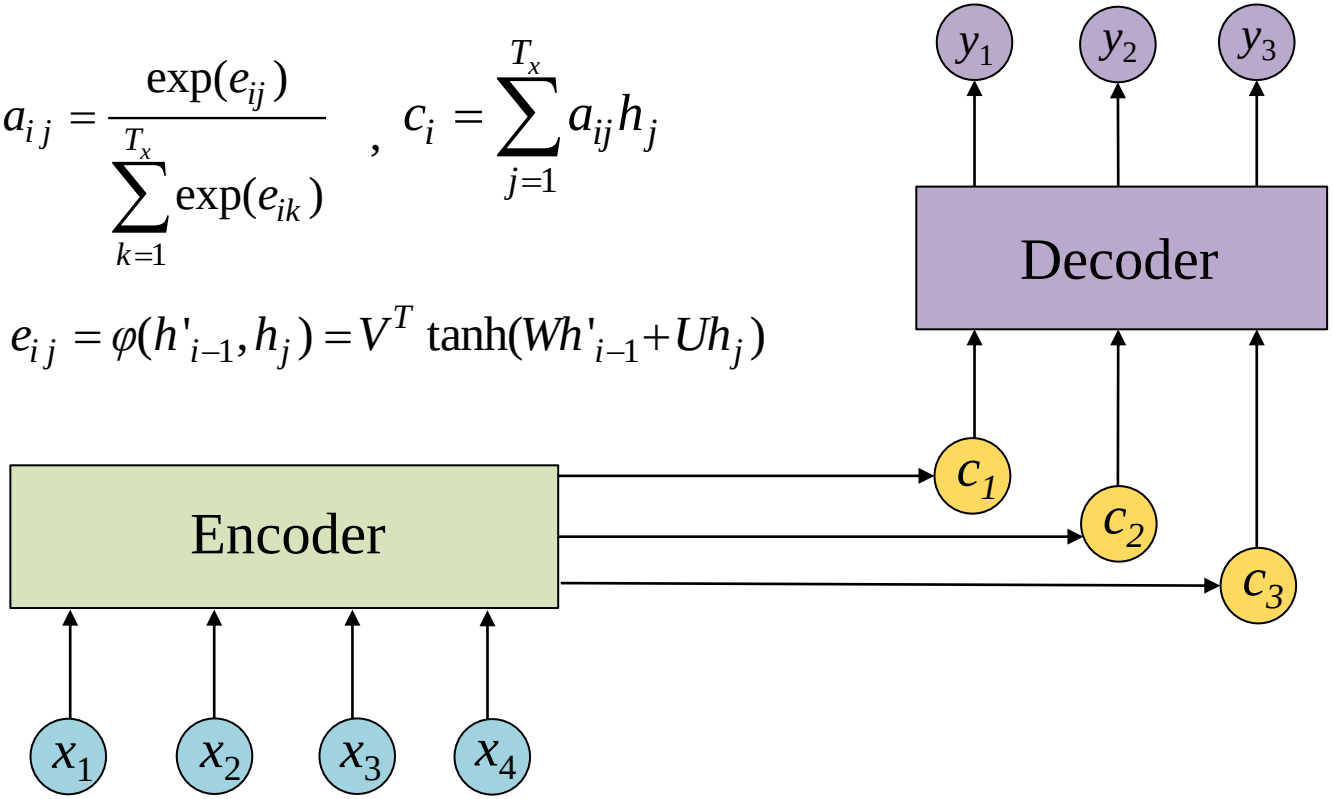
1.Masked Multi-head Attention

2.Query=Key=Value

VS. Traditional Attention Model

$$a_{ij} = \frac{\exp(e_{ij})}{\sum_{k=1}^{T_x} \exp(e_{ik})}, \quad c_i = \sum_{j=1}^{T_x} a_{ij} h_j$$

$$e_{ij} = \varphi(h'_{i-1}, h_j) = V^T \tanh(Wh'_{i-1} + Uh_j)$$



- RNN不能并行化，且忽略了句子内部词与词之间的关系，能否抛弃RNN？

4. NLP中的注意力机制

Attention

- 采用一种自动加权方式，将两个不同模块进行对齐

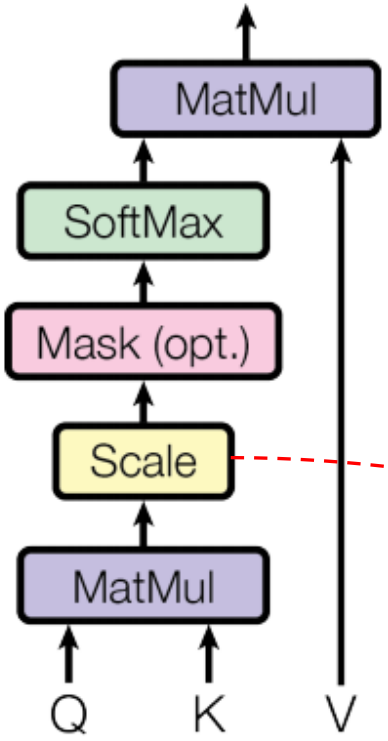
$$a_t(s) = align(m_t, m_s) = \frac{\exp(score(m_t, m_s))}{score(m_t, m_s)}$$

- 主流计算方法

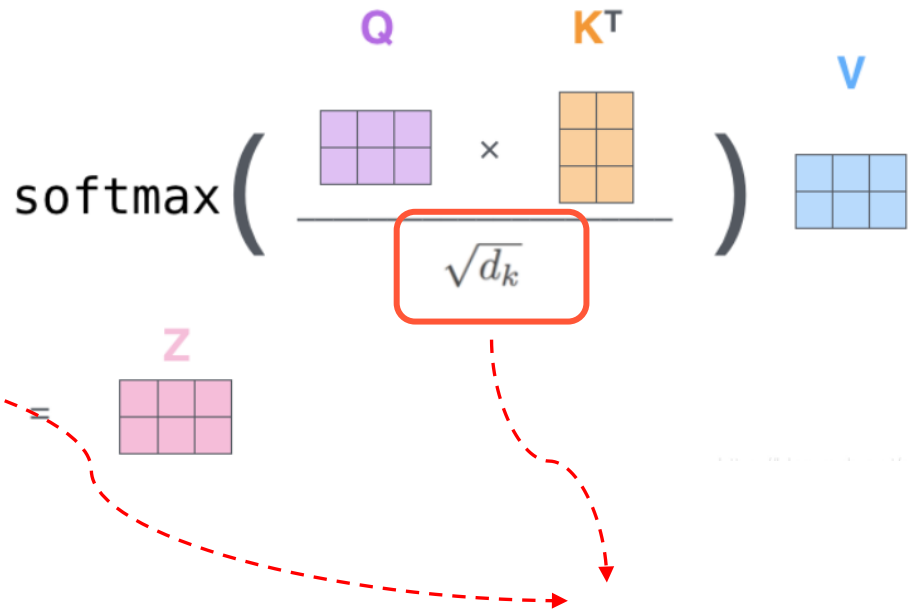
$$score(m_t, m_s) = \begin{cases} m_t^T m_s & \text{dot} \\ m_t^T W_a m_s & \text{general} \\ W_a[m_t; m_s] & \text{concat} \\ v_a^T \tanh(W_a m_t + U_a m_s) & \text{perceptron} \end{cases}$$

Scaled Dot-Product Attention

Scaled Dot-Product Attention



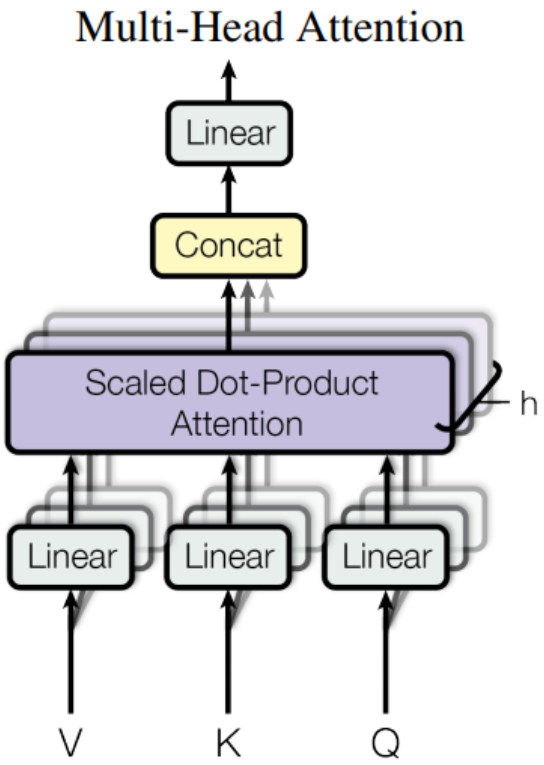
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

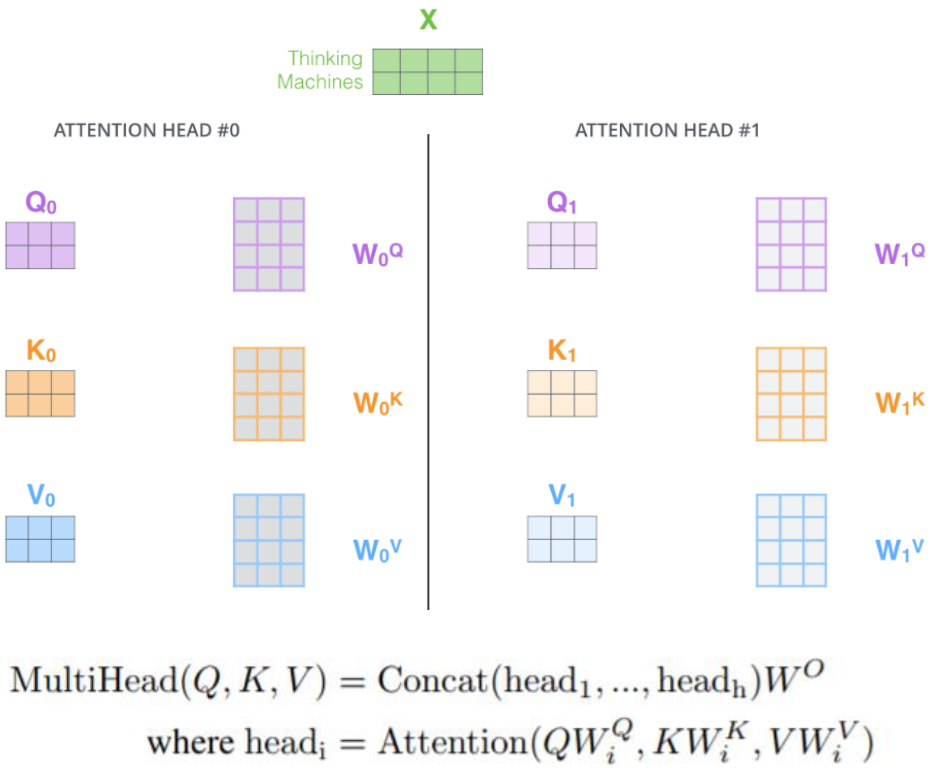


避免内积过大时，softmax产生的结果非0即1

Multi-head Attention

- 采用多个head，每个head有独立的Q/K/V的权值矩阵，不同的head为Attention层提供了不同的表示子空间，不同的表示关注不同位置

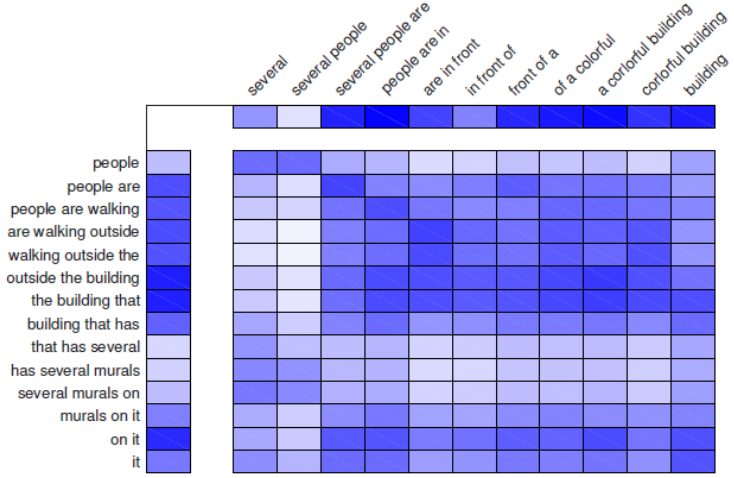




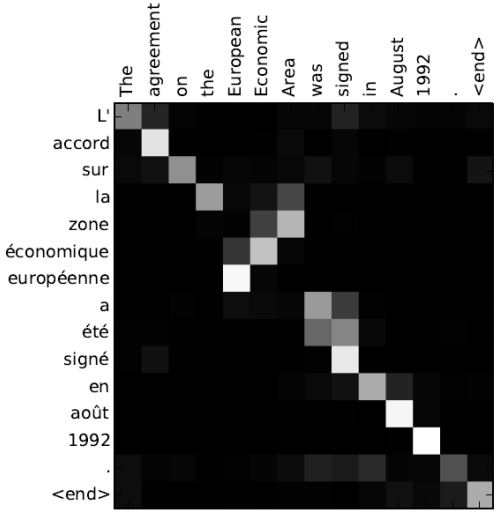
"Multi-head attention allows the model to jointly attend to information from different representation subspaces at different positions.

应 用

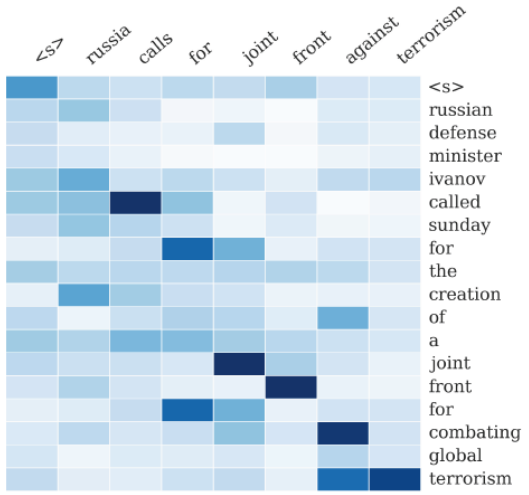
- Machine Translation
- Text Summarization
- Text comprehend (Q&A)
- Syntactic constituency parsing
- Relation classification
- Text classification



句对建模



机器翻译



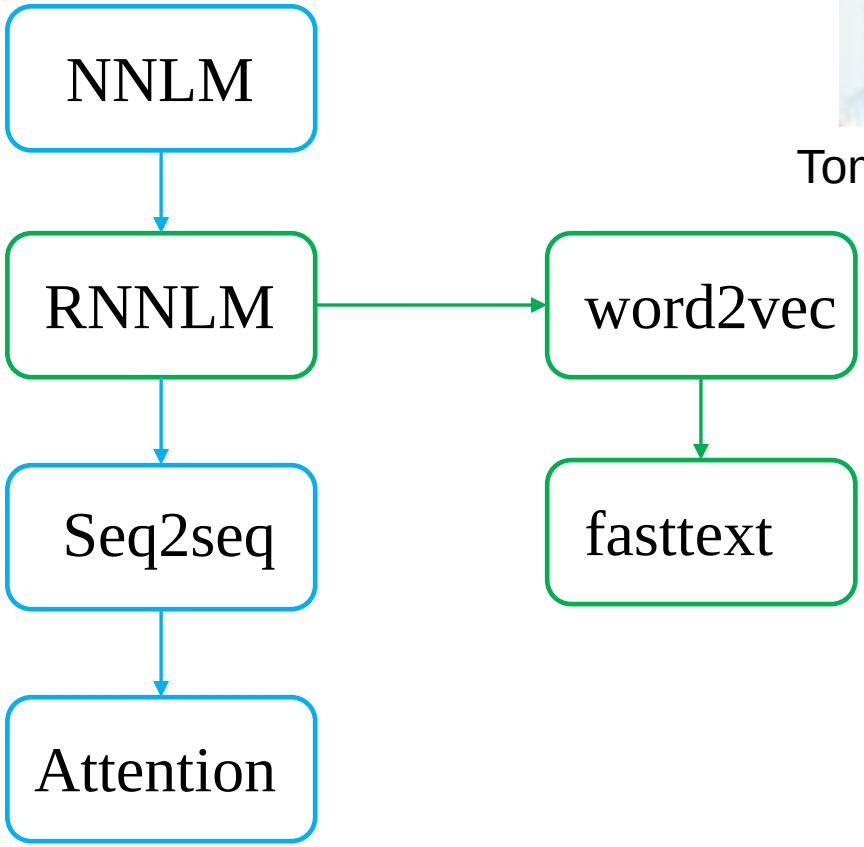
自动文摘

膜拜一下大神

Yoshua Bengio

蒙特利尔大学计算机教授
神经网络语言模型发起者
Word2vec的创始人
Theano开发团队leader
微软的人工智能研究顾问
2018年图灵奖获得者

Tomas Mikolov



参考文献

- Dzmitry Bahdanau, Kyungh yun Cho, and Yoshua Bengio.2015. Neural machine translation by jointly learning to align and translate. In ICLR 2015.
- LifengShang, Zhengdong Lu, Hang Li. Neural Responding Machine for Short-Text Conversation. ACL 2015.
- Alexander M. Rush. etc. A Neural Attention Model for Sentence Summarization. EMNLP 2015.
- Minh-Thang Luong Hieu PhamChristopher D. Manning. Effective Approaches to Attention-based Neural Machine Translation. ACL 2015.
- Volodymyr Mnihh, Nicolas Heess. et. al., Recurrent Models of Visual Attention.
- Karl Moritz Hermann.etc. Teaching Machines to Read and Comprehend.arXiv.2015. google.
- Jan Chorowski.etc. End-to-end Continuous Speech Recognition using Attention-based Recurrent NN: First Results.arXiv.2015
- Abstractive Sentence Summarization with Attentive Recurrent Neural Networks. NAACL-HLT 2016.
- Inner Attention based Recurrent Nerual Networks for Answer Selection. ACL 2016.
- Attention is all you need. NIPS 2017

欢迎加入DL4NLP！