中国科学技术大学
University of Science and Technology of China

《人工智能数学原理与算法》

第5章 Transformer

5.1注意力与自注意力机制

宋彦
songyan@ustc.edu.cn

本课重点

01 循环神经网络

02 注意力

03 自注意力

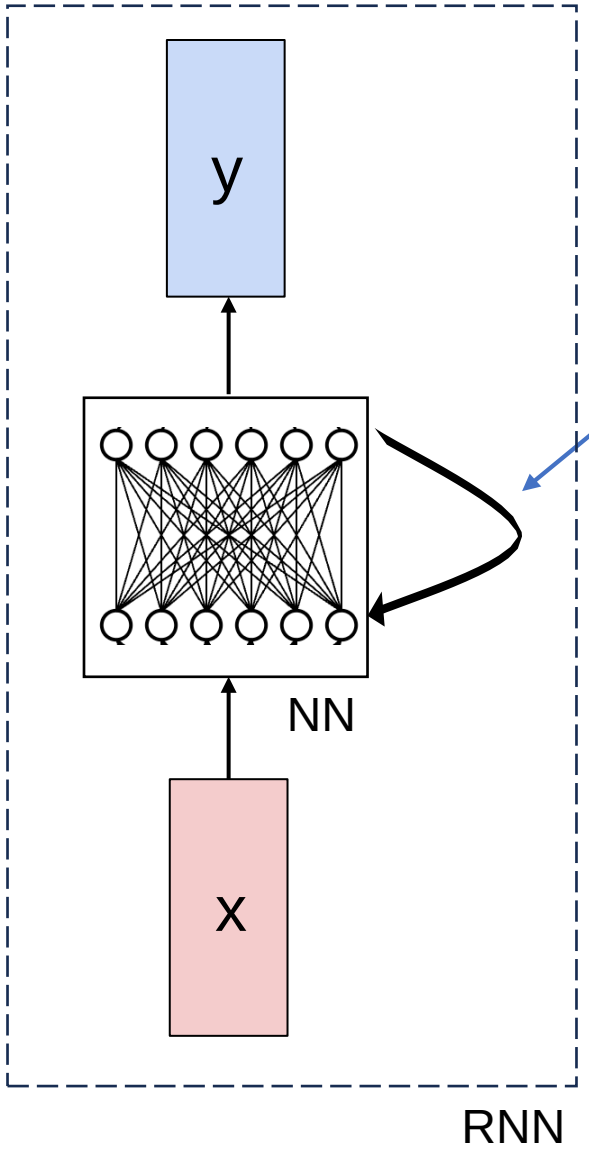
循环神经网络(RNN)

- 到目前为止，我们介绍了基本的神经网络以及图神经网络。除了这些网络之外，还有其他类型的神经网络。

- 循环神经网络（Recurrent Neural Networks，RNN）使用一种自我迭代使用的机制，使其无需指定任意固定大小的输入长度，也能处理一维线性序列数据（例如文本）。

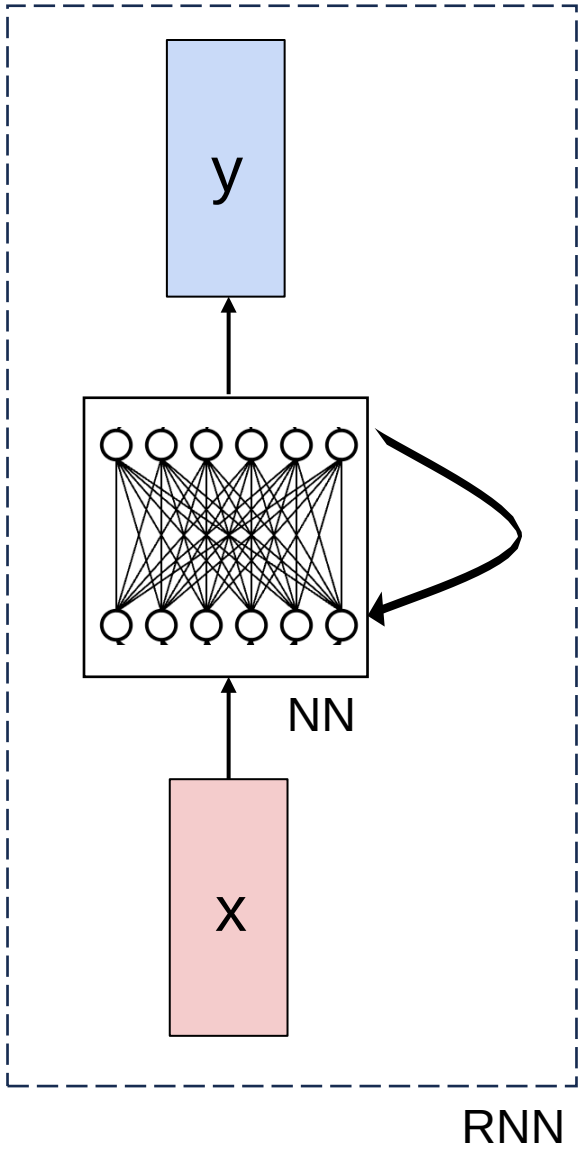
- RNN提供了一种新的方法建模线性序列数据，使得模型的输出可以迭代式依赖于历史信息。

循环神经网络(RNN)



核心思想：RNN具有一种在处理序列时不断更新的“内部状态”。

循环神经网络(RNN)



通过在每个时间步骤应用递归公式处理向量 x 序列：

t 是时间步

$$h^{(t)} = f_W(h^{(t-1)}, x^{(t)})$$

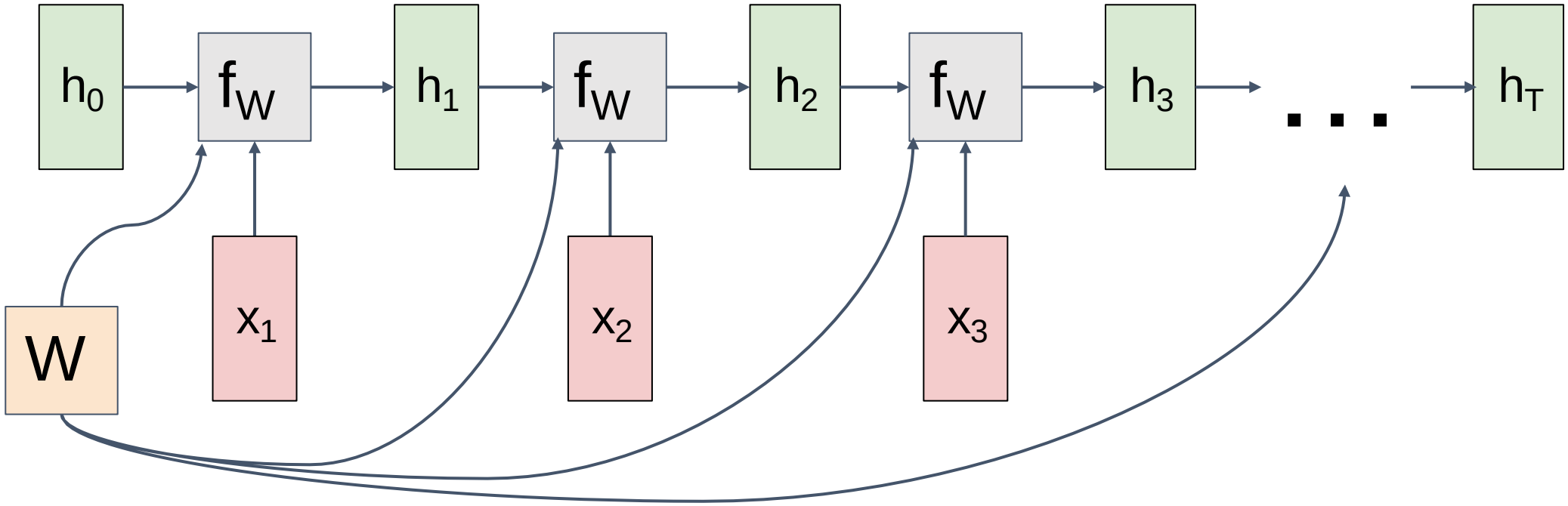
新状态 **旧状态** **t时间步的输入向量**

一个函数

注意：每个时间步骤都使用相同的函数和同一组参数。

循环神经网络(RNN)

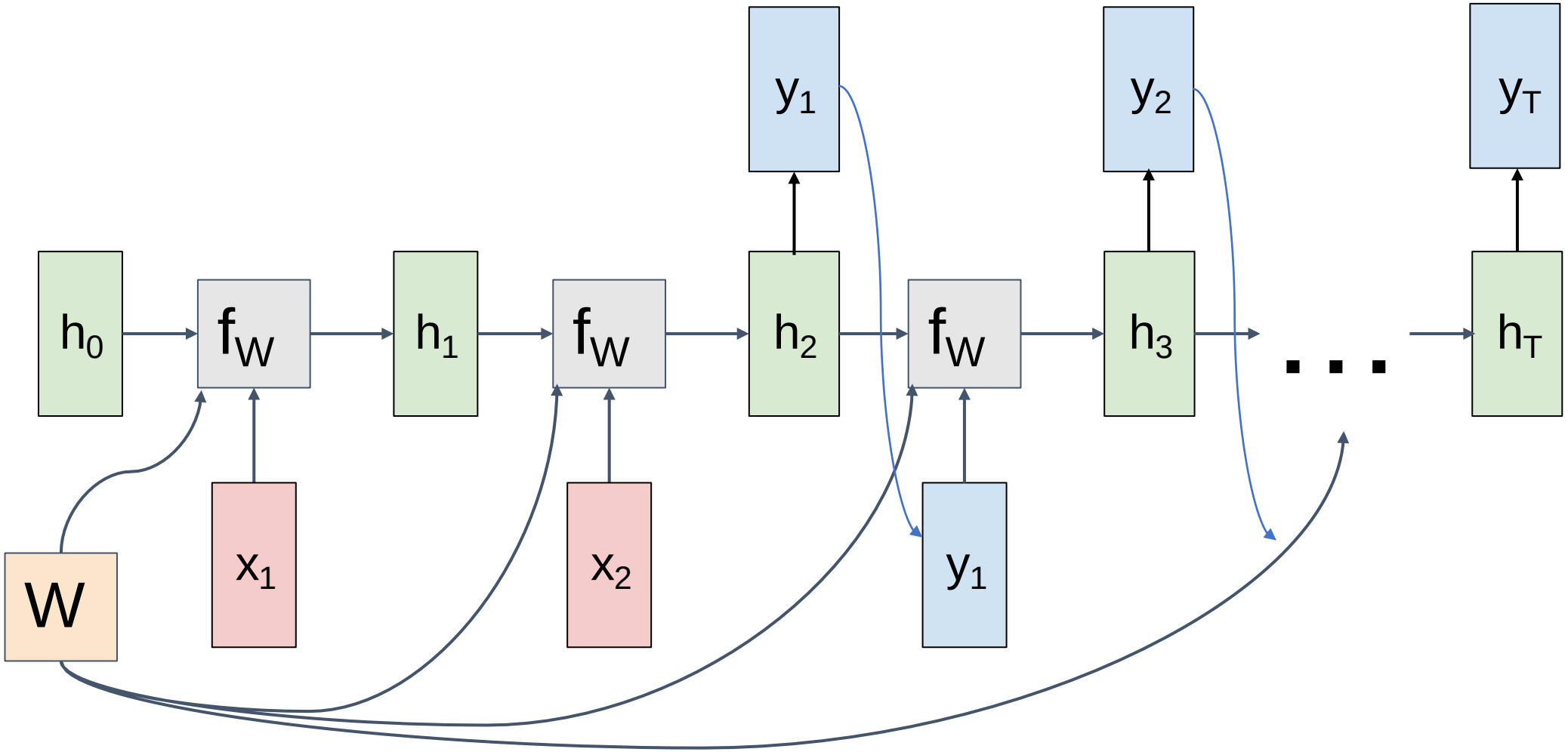
在每个时间步重复使用相同的权重矩阵



循环神经网络(RNN)

- **上述过程展现了RNN编码序列的过程。**
  - **编码阶段（Encoding）：** 在每个时间步 t，RNN 单元 f_W 根据当前输入 x_t 与上一隐藏状态 h_{t-1}更新隐藏状态 h_t，从而逐步聚合序列上下文。
  - **汇聚表示（Context Vector）：** 最终隐藏状态 h_T 汇集了整条输入序列的信息，常作为解码器的初始上下文或语义表示。
- **除了编码序列，RNN还可以进行解码生成**
  - **解码初始化（Decoder Initialization）：** 将编码得到的 h_T 传入解码器，作为初始隐藏状态，或与其他神经网络结合，指导后续生成。
  - **逐步生成（Step-by-Step Generation）：** 在解码阶段，每一步由解码器根据前一输出（或真实标签）y_{t-1}和当前隐藏状态预测下一输出 y_t，并更新隐藏状态。

循环神经网络(RNN)



循环神经网络(RNN)的局限

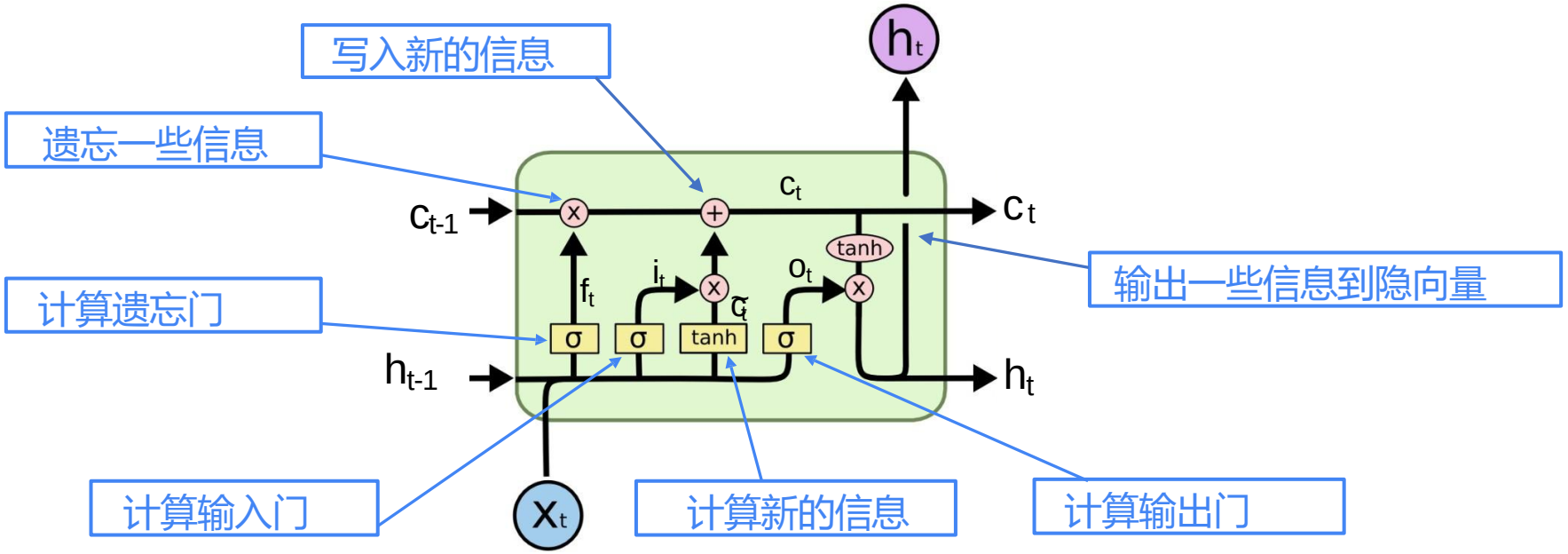
- **在训练过程中，梯度通过RNN的循环连接向后传播，以更新权重。**

- **如果权重矩阵中的数值较小**
  - 梯度在经过多次权重矩阵相乘后会变得非常小
  - 这将导致梯度消失问题（vanishing gradients problem）。

- **如果权重矩阵中的数值较大**
  - 梯度在经过多次权重矩阵相乘后会变得非常大
  - 这将导致梯度爆炸问题（exploding gradients problem）。

长短期记忆神经网络（LSTM）

- Hochreiter 和 Schmidhuber 在 1997 年提出了一种 RNN 的变体，用于解决梯度消失问题，即长短期记忆网络（LSTM）。

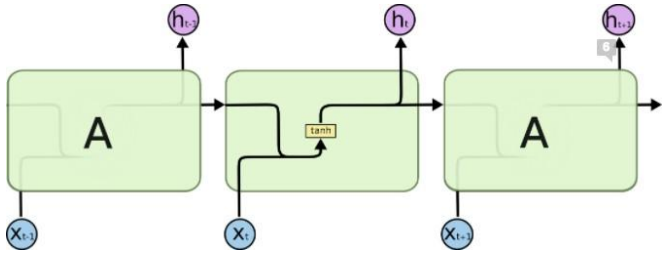


- 但 LSTM 真正变得广为人知，则是 2013 年 Hinton 将其引入谷歌之后， 而 Alex Graves 此前曾是 Hinton 的博士后研究员。

Hochreiter and Schmidhuber, 1997. Long short-term memory. https://www.bioinf.jku.at/publications/older/2604.pdf Gers, Schmidhuber, and Cummins, 2000. Learning to Forget: Continual Prediction with LSTM. https://dl.acm.org/doi/10.1162/089976600300015015 Graves, Fernandez, Gomez, and Schmidhuber, 2006. Connectionist temporal classification: Labelling unsegmented sequence data with recurrent neural nets. https://www.cs.toronto.edu/~graves/icml_2006.pdf

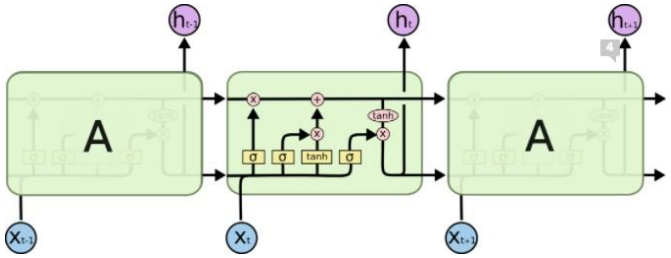
RNN与LSTM的比较

□ LSTM使用记忆机制存储历史信息



$$h_t = \tanh(W_{xh}x_t + W_{hh}h_{t-1} + b_h)$$

RNN



$$f_t = \sigma\left(W_{xf}x_t + W_{hf}h_{t-1} + b_f\right)$$

$$i_t = \sigma\left(W_{xi}x_t + W_{hi}h_{t-1} + b_i\right)$$

$$\tilde{C}_t = \sigma\left(W_{xc}x_t + W_{hc}h_{t-1} + b_c\right)$$

$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t$$

$$o_t = \sigma\left(W_{xo}x_t + W_{ho}h_{t-1} + b_o\right)$$

$$C_t = f_t \odot C_{t-1}$$

LSTM

RNN与LSTM的比较

	Model	Perplexity
	Interpolated Kneser-Ney 5-gram (Chelba et al., 2013)	67.6
标准RNN 模型	RNN-1024 + MaxEnt 9-gram (Chelba et al., 2013)	51.3
标准RNN 模型	RNN-2048 + BlackOut sampling (Ji et al., 2015)	68.3
	Sparse Non-negative Matrix factorization (Shazeer et al., 2015)	52.9
LSTM 模型	LSTM-2048 (Jozefowicz et al., 2016)	43.7
LSTM 模型	2-layer LSTM-8192 (Jozefowicz et al., 2016)	30
LSTM 模型	**Ours small** (LSTM-2048)	43.9
LSTM 模型	**Ours large** (2-layer LSTM-2048)	39.8

困惑度（Perplexity）
越低越好

https://engineering.fb.com/2016/10/25/ml-applications/building-an-efficient-neural-language-model-over-a-billion-words/

本课重点

01 循环神经网络

02 注意力

03 自注意力

注意力机制的基本想法

- 在为特定任务建模文本时，输入文本的某些部分更为重要，而这些重要部分可能出现在输入的任何位置（可能分布集中或分散）

- 如果重要部分远离当前状态，RNN 很难捕获重要信息

- 为了利用重要部分中的信息，我们可以为每个输入单元分配一个注意权重，然后计算它们的加权平均值。其中，注意权重越高意味着输入单元越重要

前馈注意力机制

- **前馈注意力计算过程**
  - 前馈过程：$a(\cdot)$是一个可学习的函数，h_t是输入隐向量

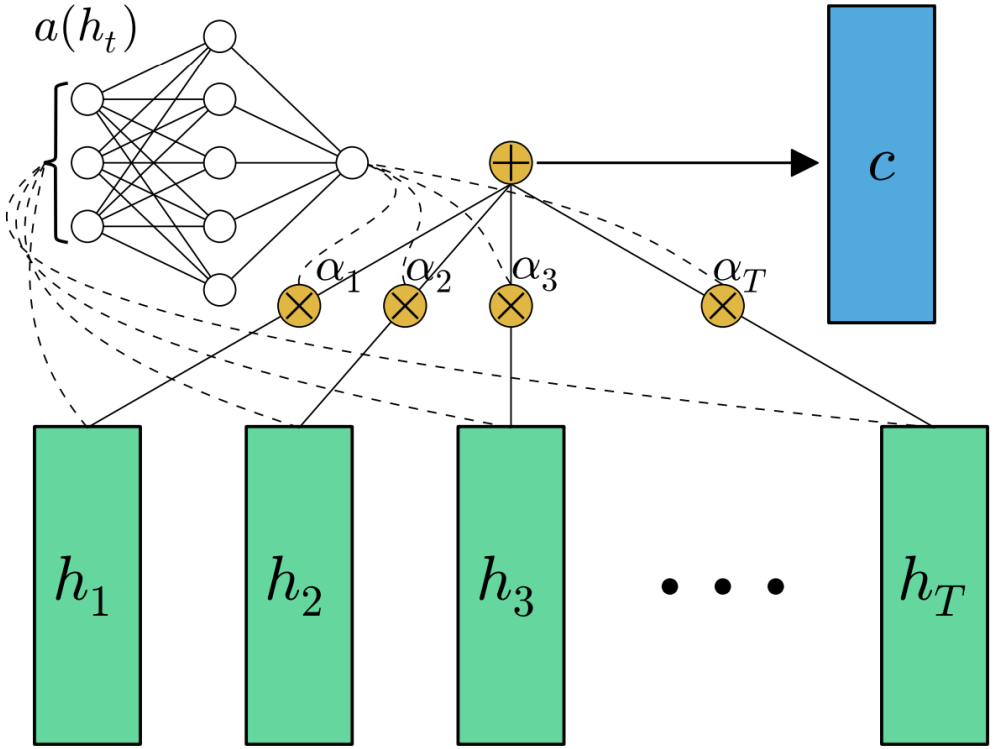
$$e_t = a(h_t)$$

  - 计算注意力权重： a_t

$$a_t = \frac{\exp(e_t)}{\sum_{t=1}^{T} \exp(e_t)}$$

  - 计算输出的存储重要信息的**上下文向量c**

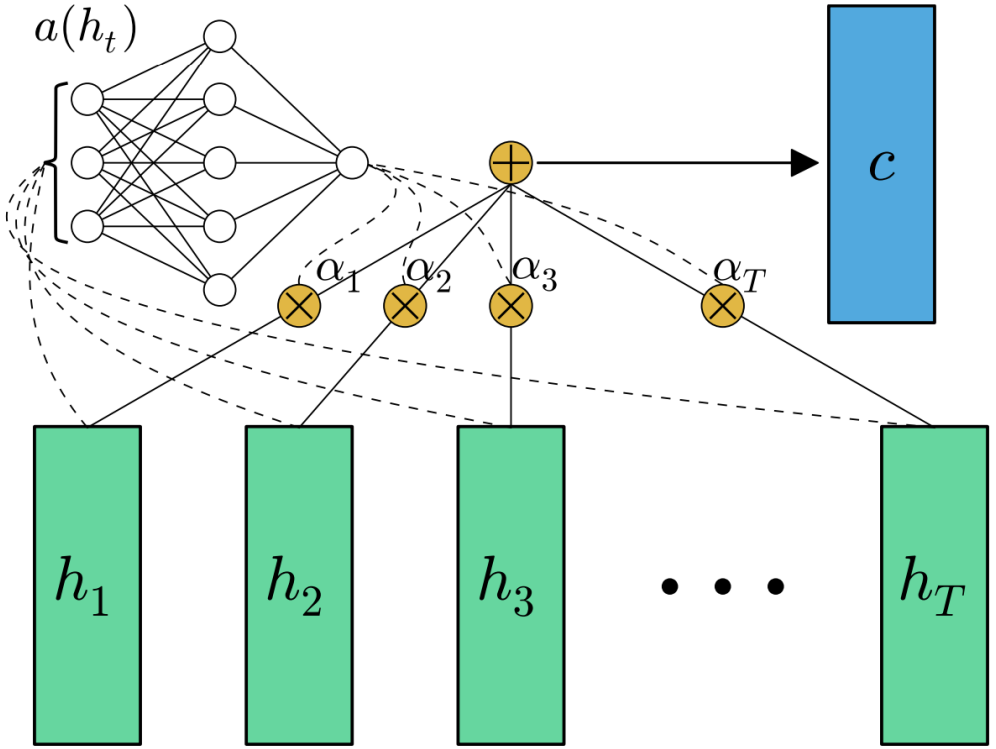
$$c = \sum_{t=1}^{T} a_t \cdot h_t$$



前馈注意力机制

- 上下文向量是所有隐藏状态的加权和，允许模型捕获长距离依赖关系
- 上下文向量不包含有关隐藏状态顺序的信息
- 如何衡量重要性?

$$e_t = a(h_t) \qquad a_t = \frac{\exp(e_t)}{\sum_{t=1}^{T} \exp(e_t)} \qquad c = \sum_{t=1}^{T} a_t \cdot h_t$$



如何衡量重要性?

- **在前馈注意力中,**

$$e_t = \tanh(w \cdot h_t)$$

w **是可学习的向量**

$$\left.\begin{aligned} e_1 &= \tanh(w \cdot h_1) \\ e_2 &= \tanh(w \cdot h_2) \\ &\cdots\cdots \\ e_T &= \tanh(w \cdot h_T) \end{aligned}\right\} \text{softmax} \left\{\begin{aligned} a_1 &= \frac{\exp(e_1)}{\sum_{t=1}^{T}\exp(e_t)} \\ &\cdots\cdots \\ a_T &= \frac{\exp(e_T)}{\sum_{t=1}^{T}\exp(e_t)} \end{aligned}\right.$$

那些与 w 相似的隐向量获得了更高的权重

如何衡量重要性?

- 在前馈注意力中,

$$e_t = \tanh(w \cdot h_t)$$

- w 是可学习的向量， 在训练中学习如何决定输入的重要性
- 那些与 w 相似的隐向量获得了更高的权重，从而通过选择重要的隐向量并且据此利用它们取得性能提升

- 学习一个好的 w 是取得好结果的重要条件

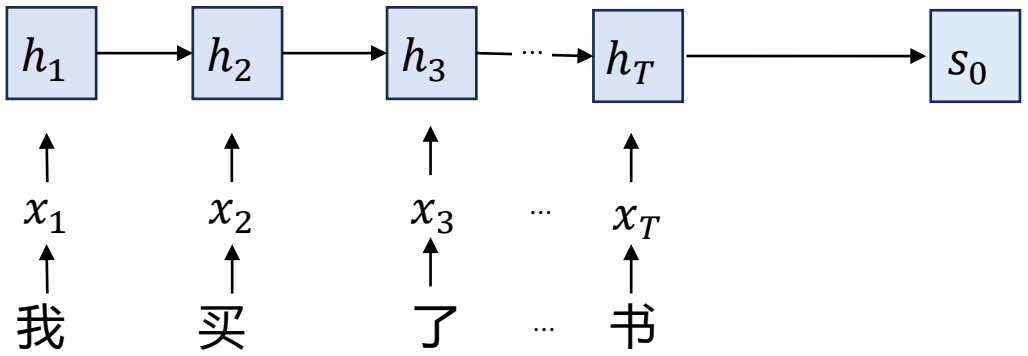
在RNN中使用注意力

以从中文到英文的机器翻译任务为例：

输入: 序列 $x_1, \ldots x_T$ （我买了他昨天读的书）
输出: 序列 $y_1, \ldots, y_{T'}$ （I bought the book that he read yesterday）

编码器: $h_t = f_W(x_t, h_{t-1})$

来自最后一个隐向量:
解码器初始隐向量s_0



Bahdanau et al, “Neural machine translation by jointly learning to align and translate”, ICLR 2015

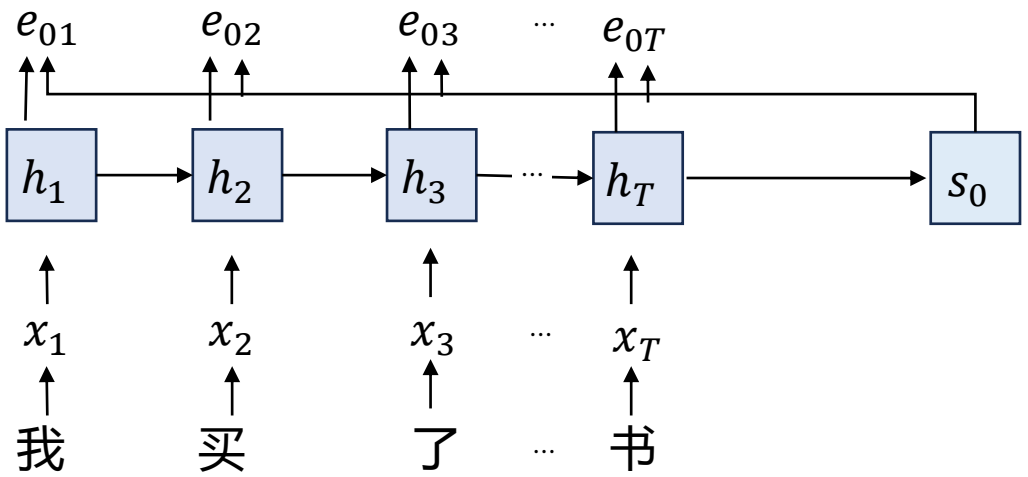
在RNN中使用注意力

计算对齐分数

$e_{i,j} = a(s_{i-1}, h_j)$

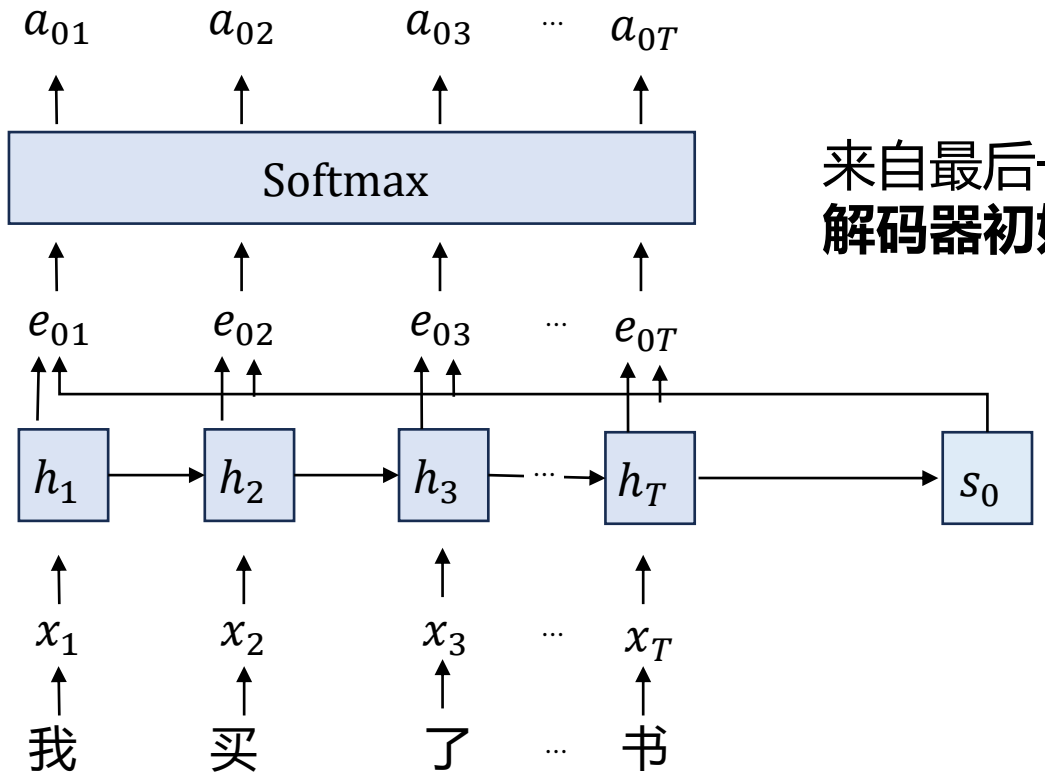
a 是前馈注意力机制

来自最后一个隐向量:
解码器初始隐向量s_0



Bahdanau et al, “Neural machine translation by jointly learning to align and translate”, ICLR 2015

在RNN中使用注意力



来自最后一个隐向量:
解码器初始隐向量s_0

计算对齐分数

$$e_{i,j} = a(s_{i-1}, h_j)$$

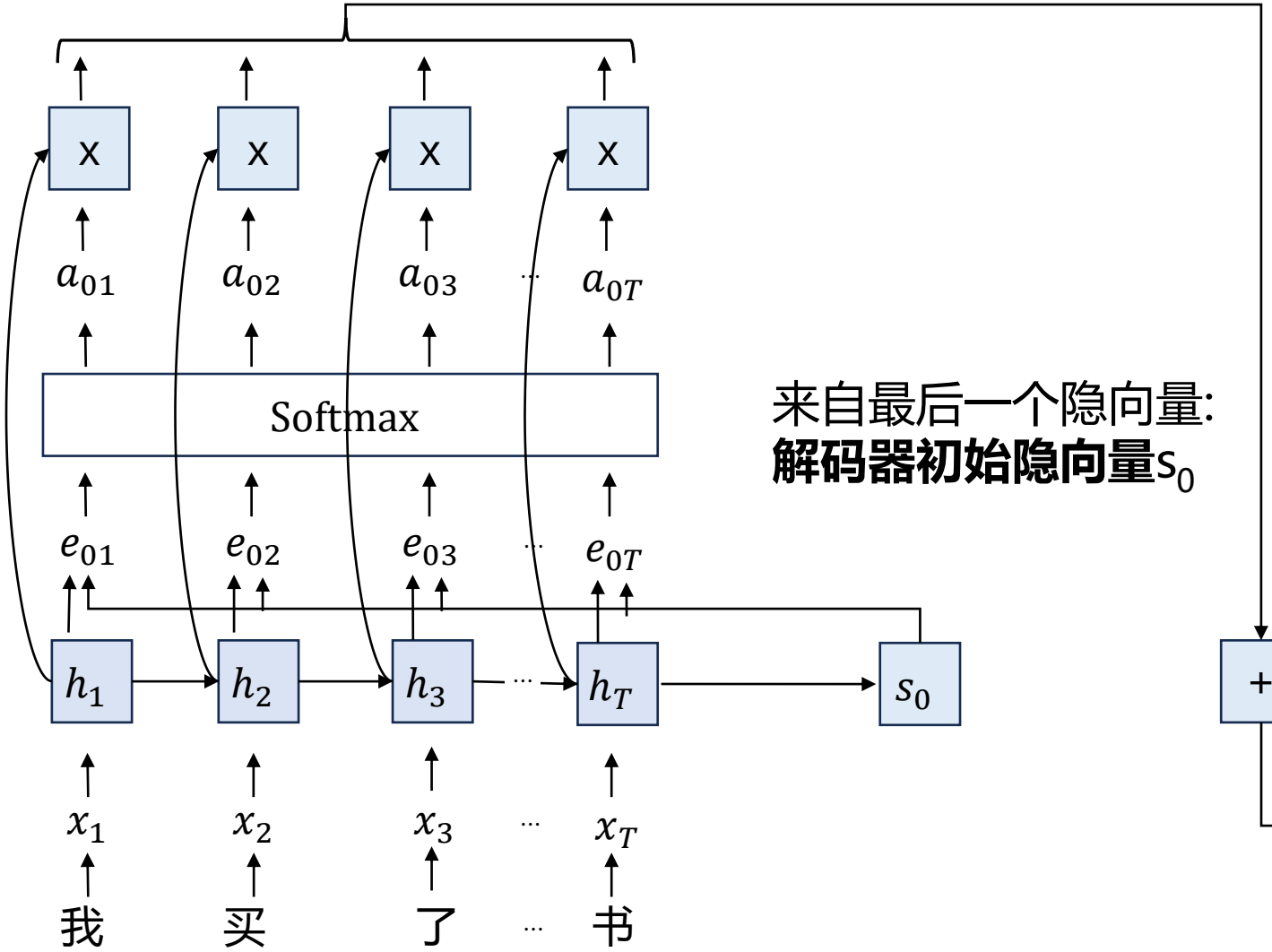
a 是前馈注意力机制

得到注意力权重

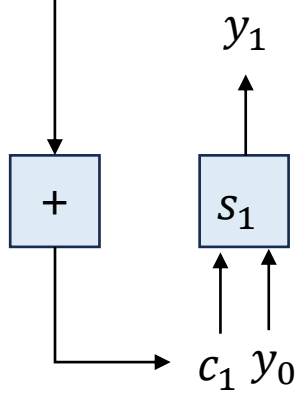
$$a_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k=1}^{T} \exp(e_{i,k})}$$

Bahdanau et al, “Neural machine translation by jointly learning to align and translate”, ICLR 2015

在RNN中使用注意力



来自最后一个隐向量:
解码器初始隐向量s_0



计算对齐分数

$$e_{i,j} = a(s_{i-1}, h_j)$$

a 是前馈注意力机制

得到注意力权重

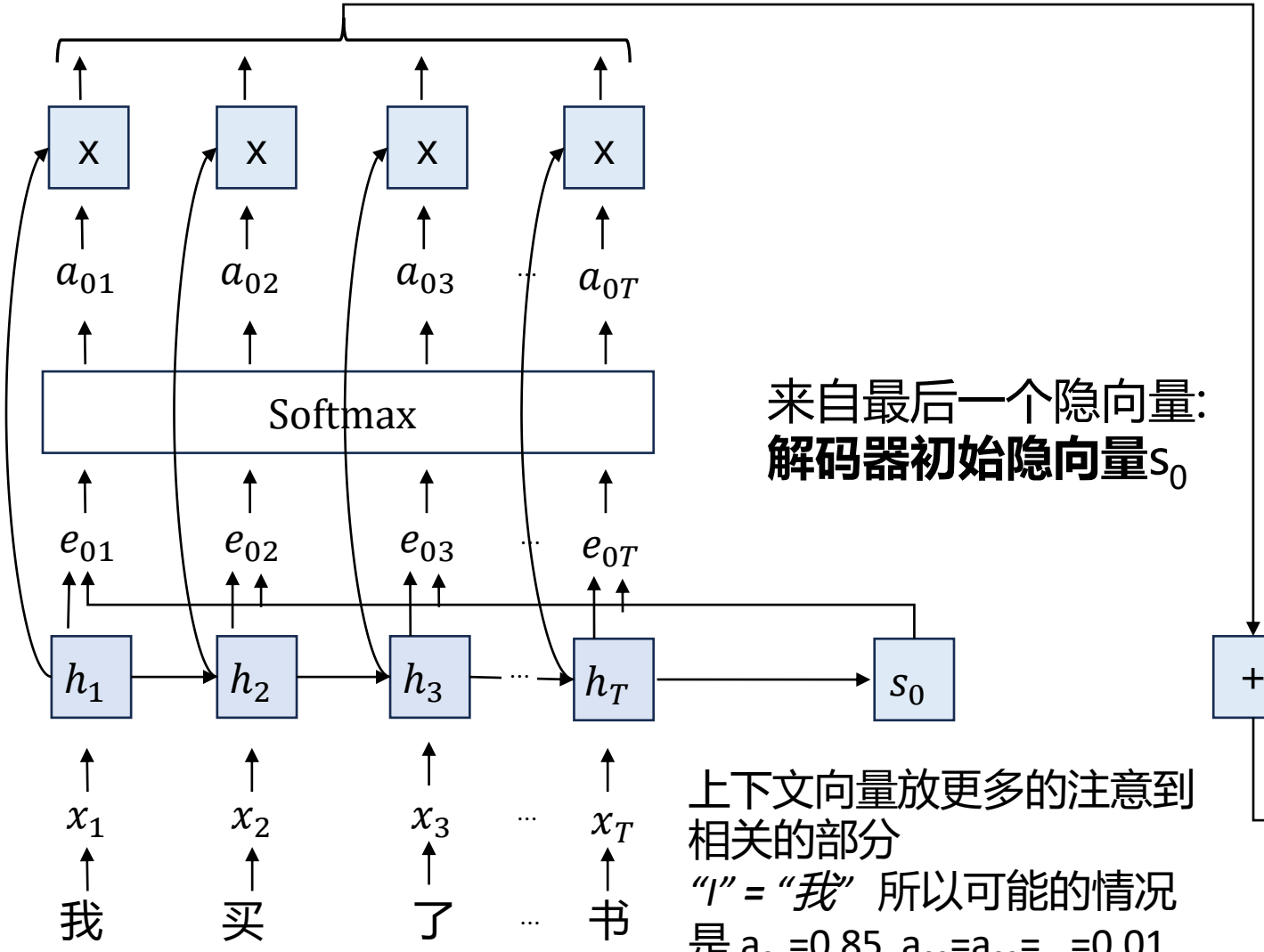
$$a_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k=1}^{T} \exp(e_{i,k})}$$

通过加权平均计算上下文向量

$$c_i = \sum_{j=1}^{T} a_{i,j} h_j$$

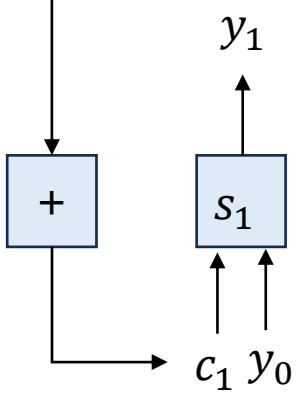
Bahdanau et al, “Neural machine translation by jointly learning to align and translate”, ICLR 2015

在RNN中使用注意力



来自最后一个隐向量:
解码器初始隐向量s_0

上下文向量放更多的注意到相关的部分
"I" = "我" 所以可能的情况是 a_{01}=0.85, a_{02}=a_{03}=…=0.01



计算对齐分数

$$e_{i,j} = a(s_{i-1}, h_j)$$

a 是前馈注意力机制

I

得到注意力权重

$$a_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k=1}^{T} \exp(e_{i,k})}$$

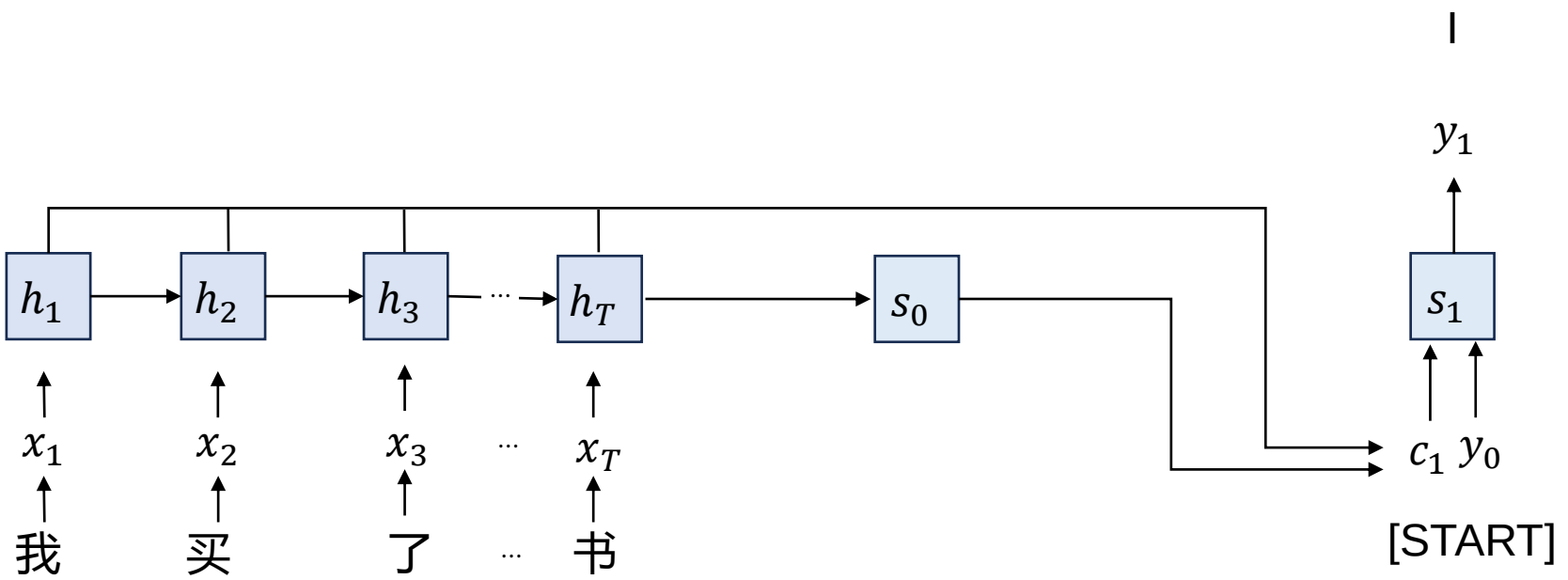
通过加权平均计算上下文向量

$$c_i = \sum_{j=1}^{T} a_{i,j} h_j$$

Bahdanau et al, "Neural machine translation by jointly learning to align and translate", ICLR 2015

在RNN中使用注意力

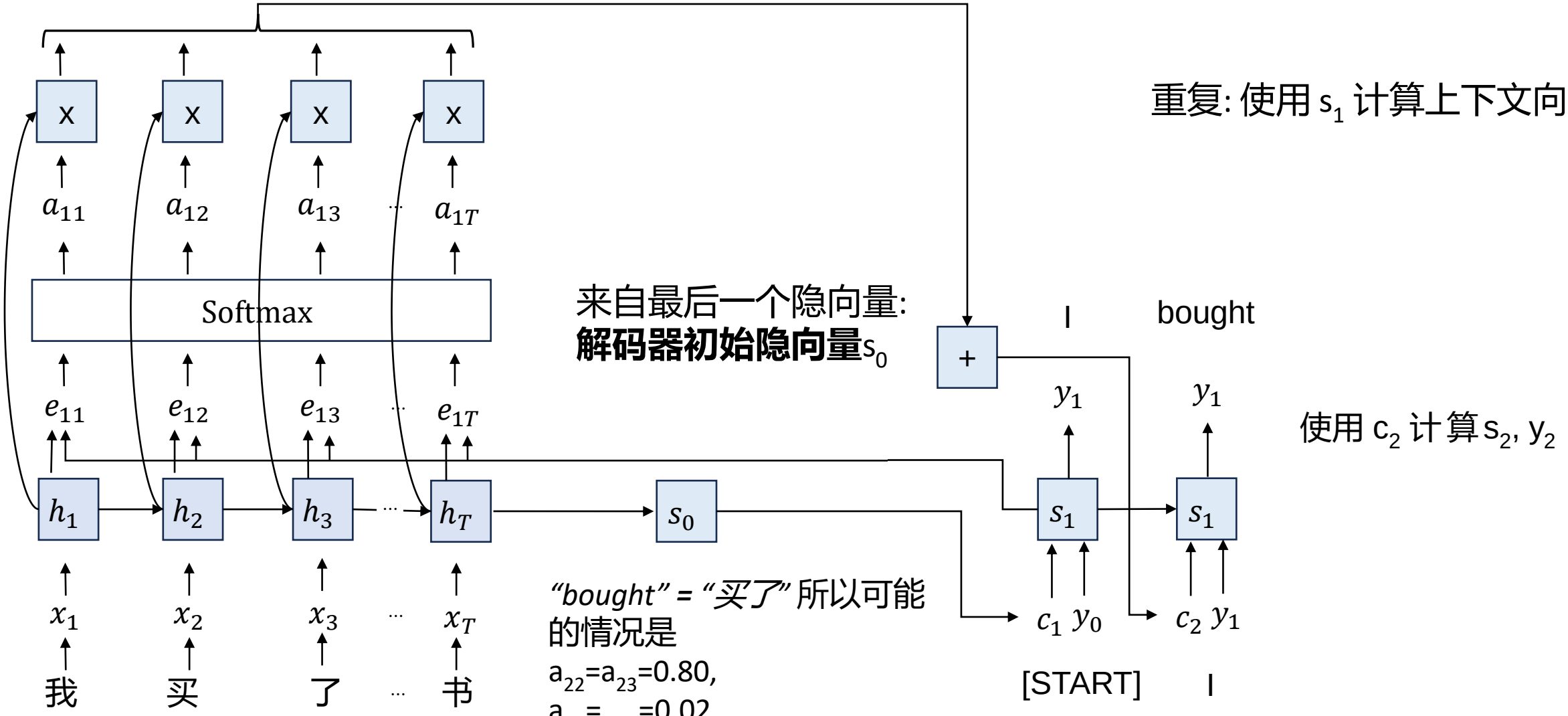
重复: 使用 s_1 计算上下文向量c_2



使用 c_2 计算s_2, y_2

Bahdanau et al, “Neural machine translation by jointly learning to align and translate”, ICLR 2015

在RNN中使用注意力



Bahdanau et al, "Neural machine translation by jointly learning to align and translate", ICLR 2015

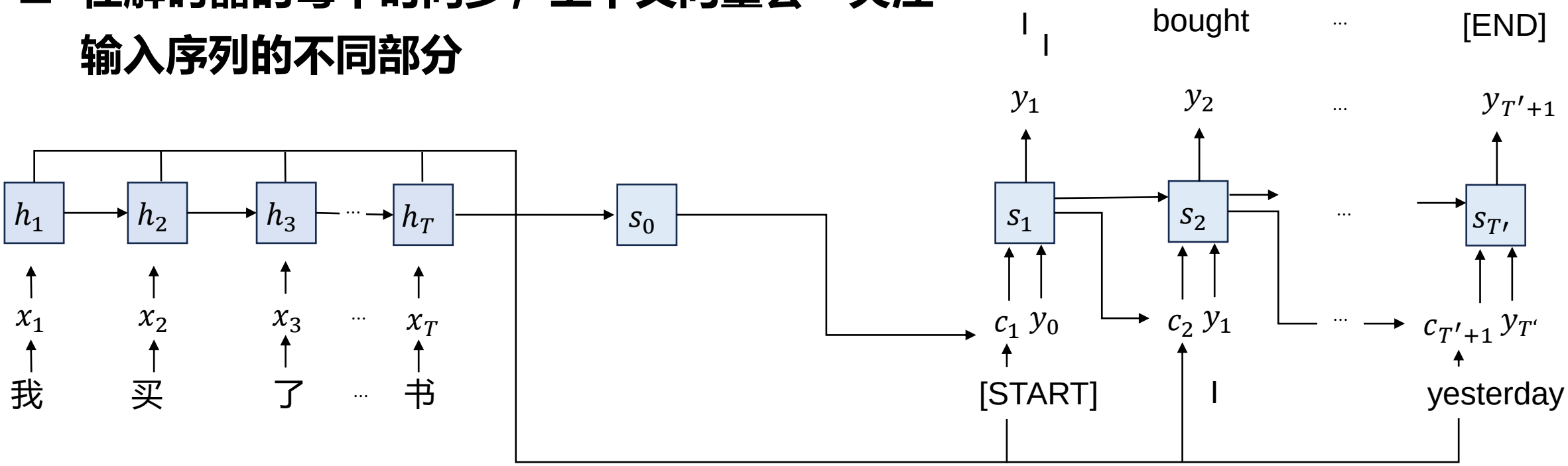
在RNN中使用注意力

- 在解码器的每个时间步使用不同的上下文向量
- 输入序列不会被单一向量所限制（不会形成瓶颈）
- 在解码器的每个时间步，上下文向量会“关注”输入序列的不同部分

$$e_{i,j} = a(s_{i-1}, h_j)$$

$$a_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k=1}^{T} \exp(e_{i,k})}$$

$$c_i = \sum_{j=1}^{T} a_{i,j} h_j$$



Bahdanau et al, “Neural machine translation by jointly learning to align and translate”, ICLR 2015

注意力的计算方法

- 在RNN中，使用前馈注意力机制

$$e_{i,j} = \tanh\left(w \cdot (s_{i-1} \oplus h_j)\right)$$

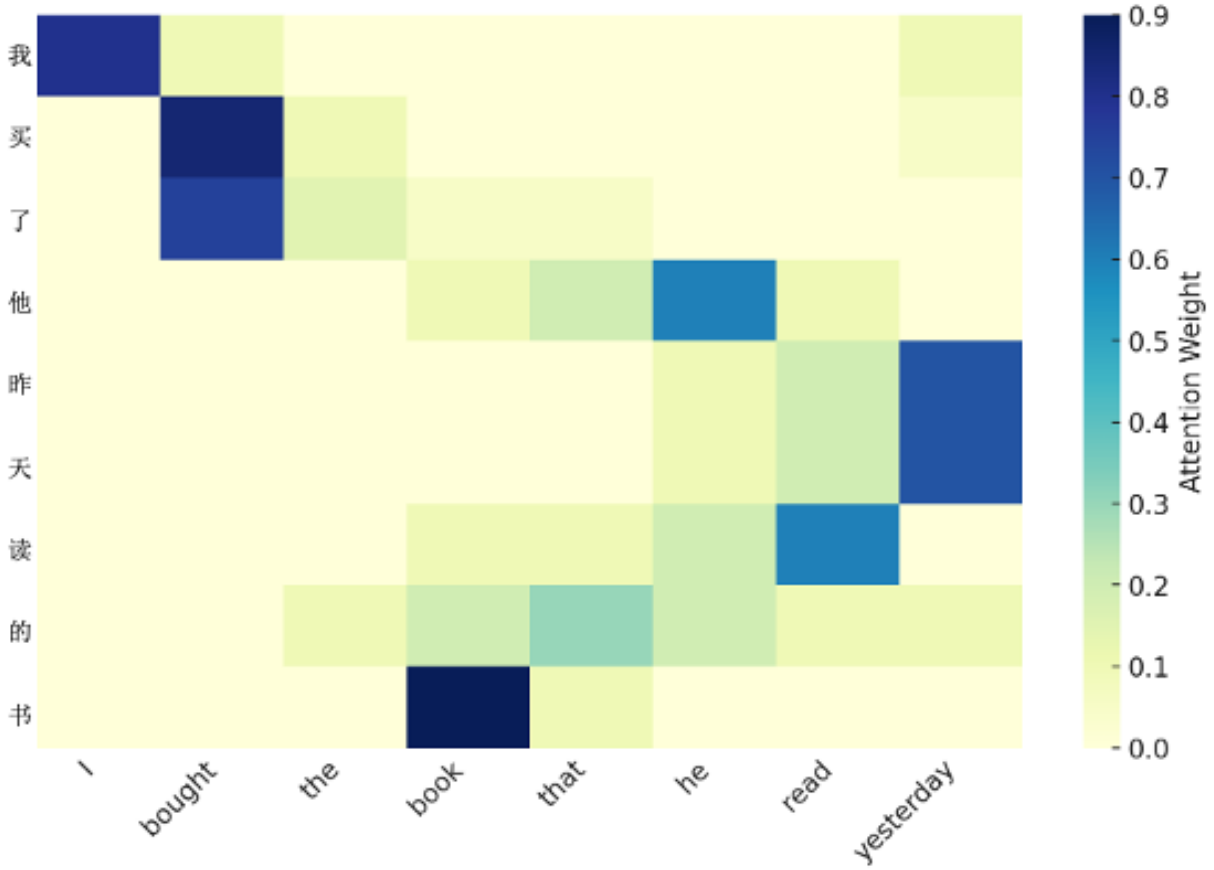
- 向量 w 在训练中被更新
- 对于中文到英文的翻译，w 学习重要的关于任务的信息，从而指导注意力机制选择重要的上下文信息

注意力的可视化

- 例子: 中文到英文的翻译
- 输入: “我买了他昨天读的书”
- 输出: “I bought the book that he read yesterday”

- 表格中各个位置的注意力表示了词汇之间的对应关系

注意力权重 $a_{t,l}$ 的可视化



Bahdanau et al, “Neural machine translation by jointly learning to align and translate”, ICLR 2015

前馈注意力机制的总结

- **前馈注意力的优势**
  - 高度并行
  - 非线性映射能力

- **前馈注意力的局限**
  - 依赖外部网络计算注意力，可能存在向量空间不匹配的问题
  - 单一视角映射，仅有一套固定参数映射，缺少多重子空间表示
  - 无法建模长距离依赖，每个位置只见自身输入，无法获取全局上下文

01 循环神经网络

02 注意力

03 自注意力

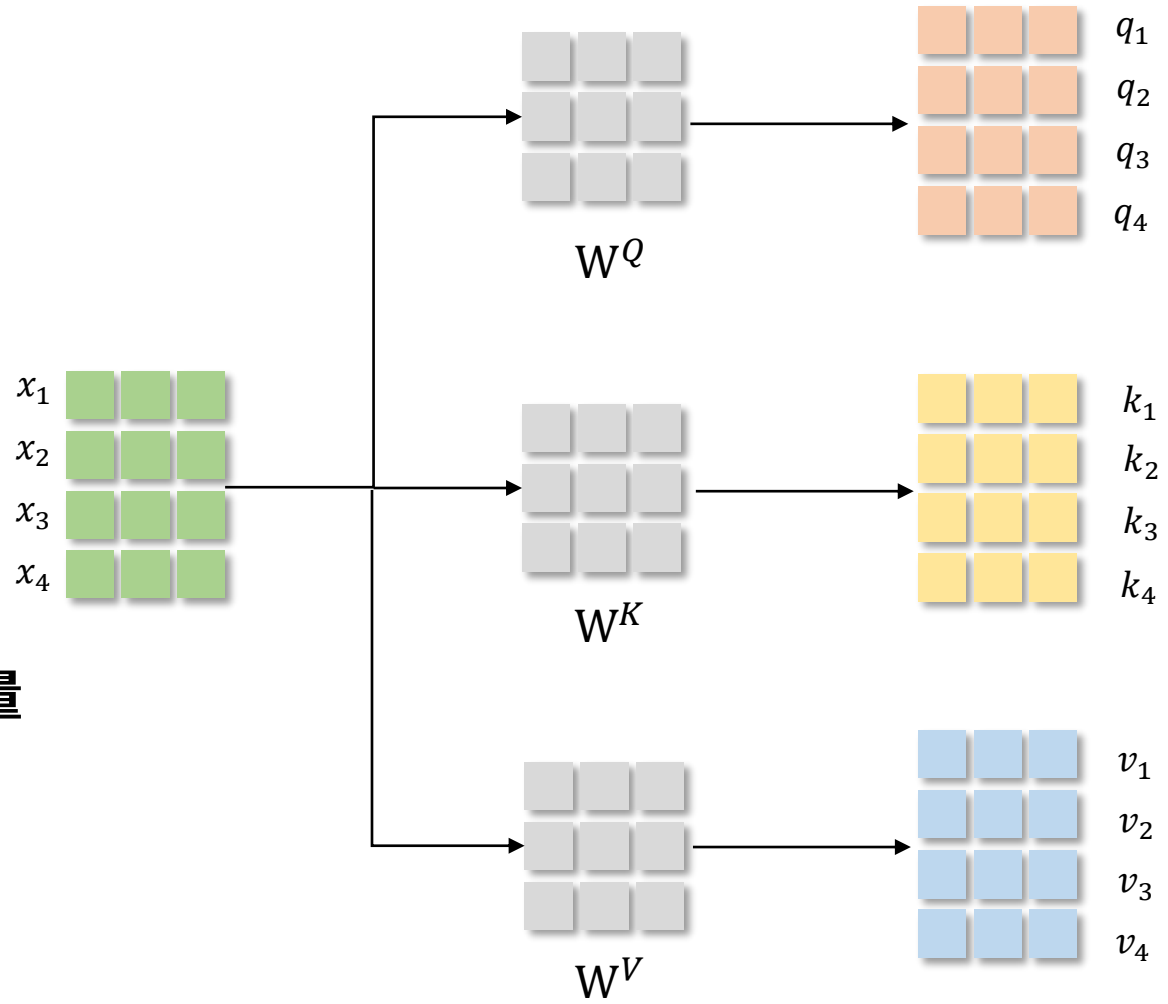
本课重点

自注意力

- 首先每个输入被分解为三个部分

$$X_{Embedding} \cdot W^Q = Q$$
$$X_{Embedding} \cdot W^k = K$$
$$X_{Embedding} \cdot W^v = V$$

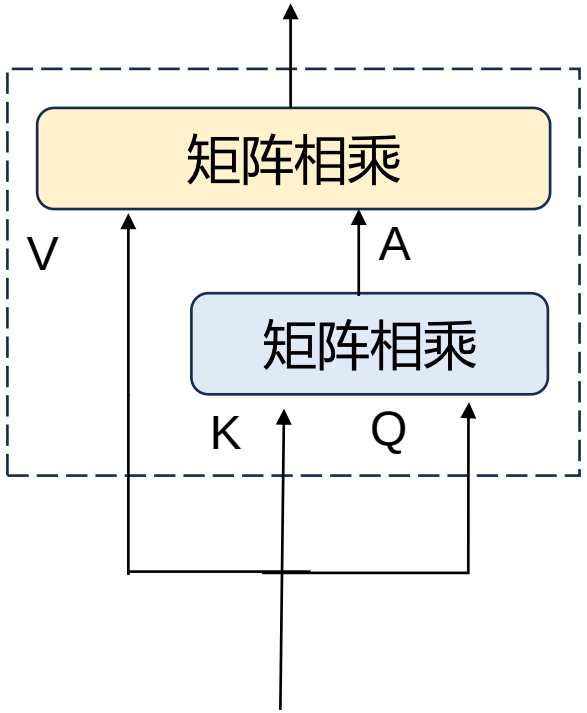
- 其次为自注意力计算准备结果
  - 查询向量、键向量和值向量都是行向量



自注意力

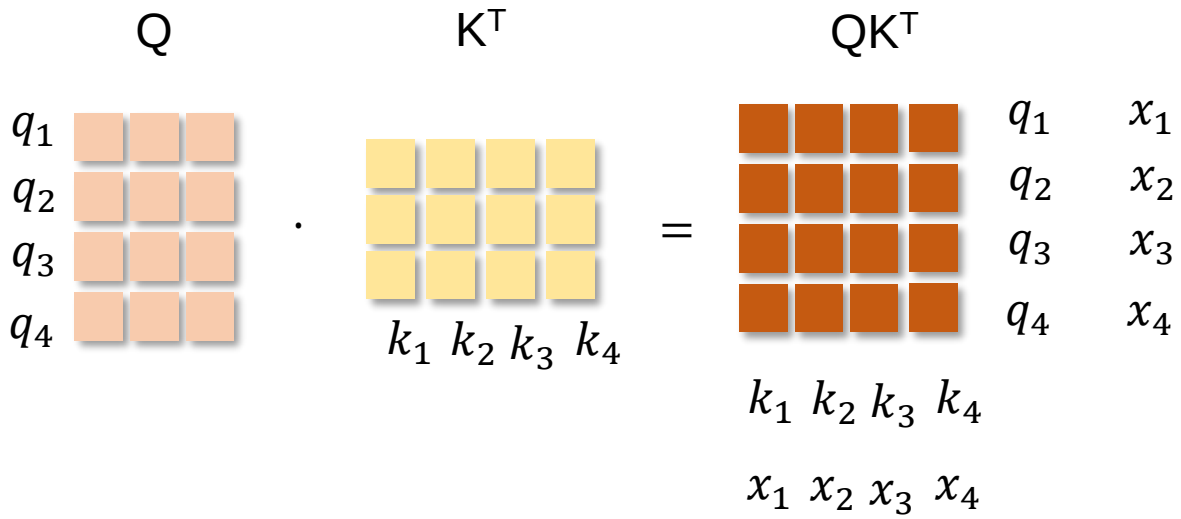
- 计算步骤:
  - 计算与缩放 $Q \cdot KT$，若在解码期间则对得到的注意力矩阵进行掩码覆盖（Mask）
  - 对每行应用Softmax实现权重的归一化
  - 把得到的归一化权重矩阵乘以值矩阵 V

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

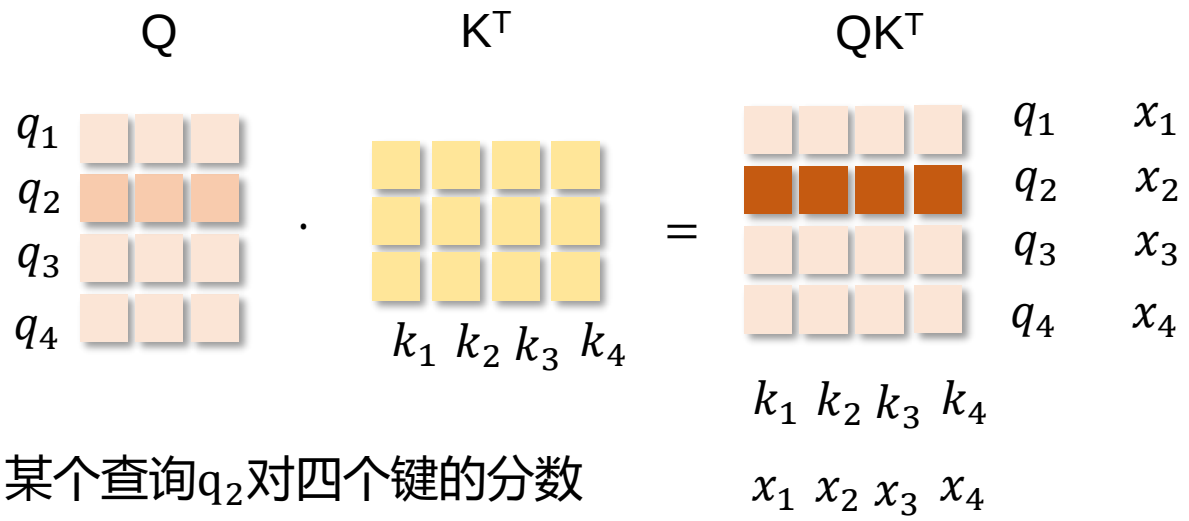


自注意力

- QK^T 代表什么?
  - 句子中的单词关联
  - 查询是行向量，键是列向量
  - QK^T中位置 (i,j) 是 (q_i, k_j) 的分数



- 具体理解QK^T的意义



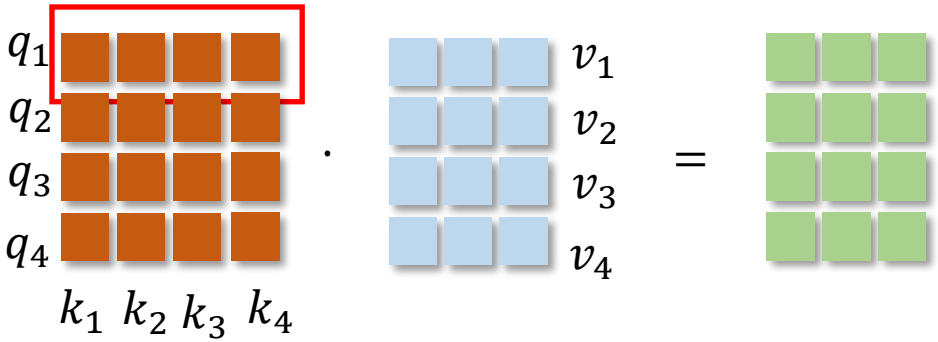
某个查询q_2对四个键的分数

自注意力

加权求和的过程

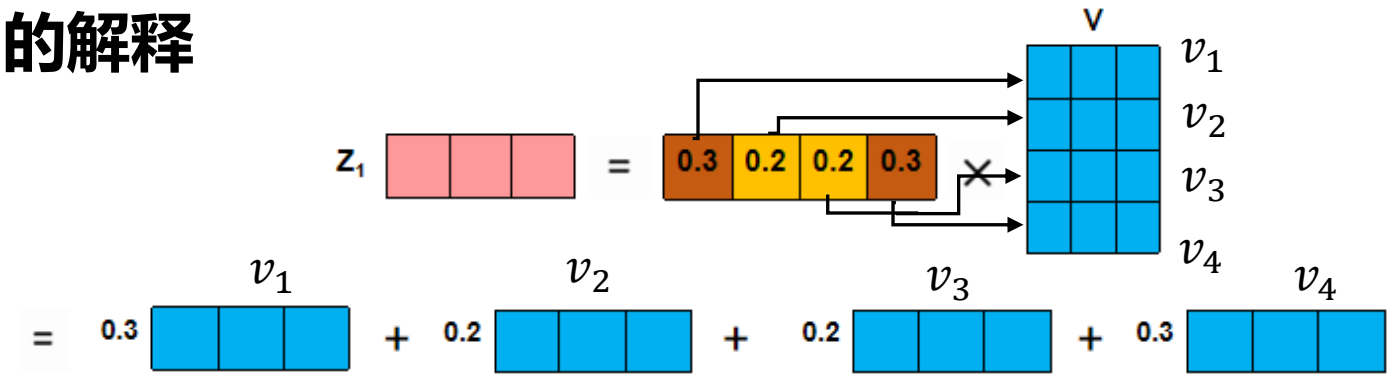
$$\text{Attention}(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

和为1



- 尺寸越大意味着点积和中的项越多。
- 因此，logits 的方差较高。
- 大向量将产生更高的logits。
- 最终，这些大幅度的向量将导致 softmax 达到峰值，并为所有其他向量分配很少的权重
- 除以$\sqrt{d_k}$ 减少大幅度矢量的影响

更详细的解释



理解自注意力

核心思想

- 每个位置的表示不仅依赖自身，还“询问”序列中所有位置的上下文信息。

三矩阵投影

- 查询（Query）由 W^Q 决定“提问角度”
- 键（Key）由 W^K 决定“回答的重要性”
- 值（Value）由 W^V 决定“实际提供的上下文信息”

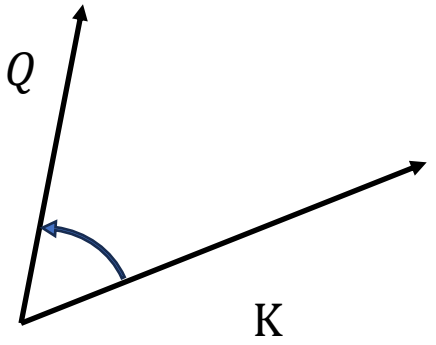
注意力计算流程

- 计算相似度：$\frac{QK^T}{\sqrt{d_k}}$
- 归一化：Softmax 得到权重分布
- 加权求和：权重乘以V，汇聚上下文

理解自注意力

- **统一语义投影：**将输入映射到查询（Q）和键（K）空间，以确保相似度计算落在同一语义空间内。
- **自适应权重分配：**根据 Q 与各位置 K 的匹配程度动态计算注意力权重，实现对不同上下文的实时关注。
- **全局信息汇聚：**加权求和值（V）将分散于序列各处的关键信息融合到目标位置的表示中，捕捉跨序列依赖。
- **语义再分配：**最终输出基于上下文重新编码原始信息，支持模型在不同任务场景中灵活提取所需特征。

理解自注意力



几何视角

- Q, K 空间中的“角度”与“距离”决定重要的输入
- 通过识别重要内容，提升模型可解释性

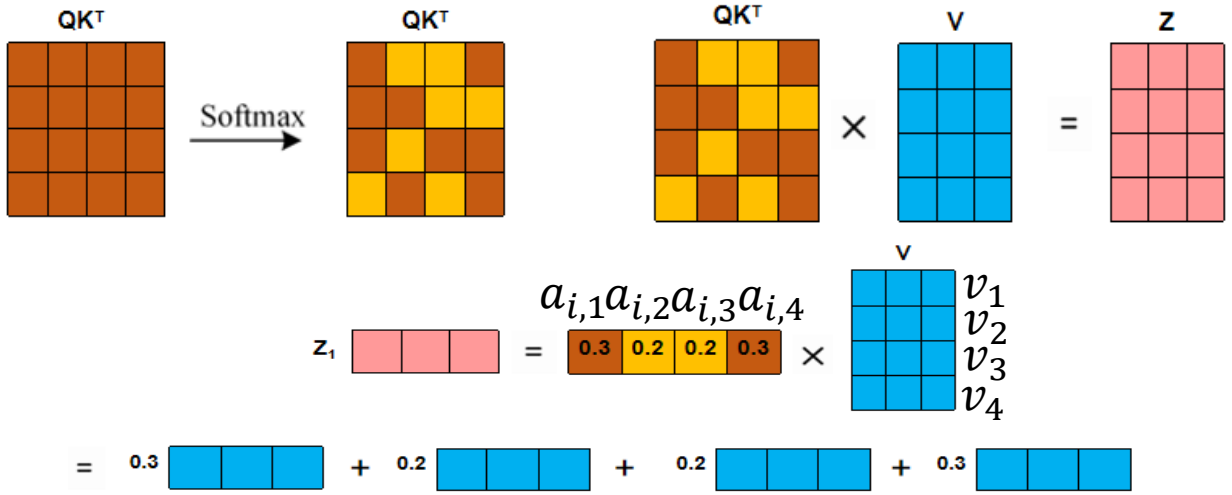
动态上下文塑形

- 不同输入间权重动态变化，体现了输入中各单元间或紧或松的关系
- 在长文本中，可针对不同语义单元自适应地集中 / 分散注意力

结构化偏置

- 可在自注意力中加入相对位置编码、图结构先验，引导模型学到更合适的依赖模式

自注意力和前馈注意力的比较



QK^T

$$e_{i,t} = \sum_{j=1}^{T} q_{i,j} \cdot k_{j,t} = q_i \cdot k_t$$

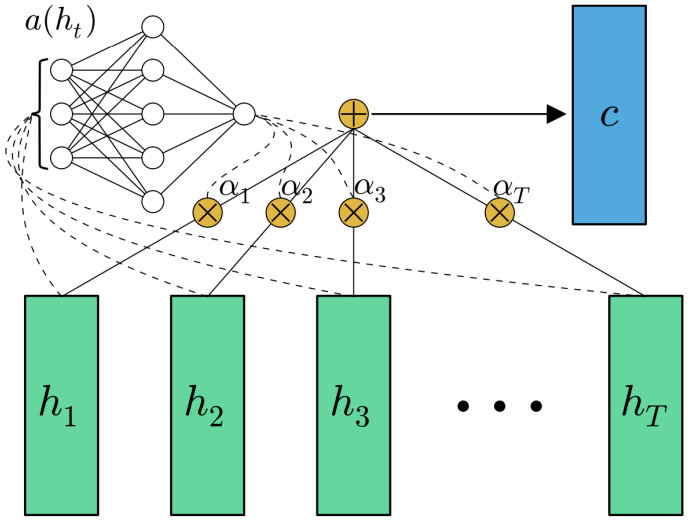
$softmax(QK^T)$

简单起见，忽视$1/\sqrt{d_k}$

$$a_{i,t} = \frac{\exp(e_{i,t})}{\sum_{t=1}^{T} \exp(e_{i,t})}$$

$softmax(QK^T)V$

$$z_i = \sum_{t=1}^{T} a_{i,t} \cdot v_t$$



$$e_t = \tanh(w \cdot h_t)$$

$q_i \rightarrow \mathrm{w}$

$k_t \rightarrow h_t$

$$a_t = \frac{\exp(e_t)}{\sum_{t=1}^{T} \exp(e_t)}$$

$$c = \sum_{t=1}^{T} a_t \cdot h_t$$

$v_t \rightarrow h_t$

自注意力和前馈注意力的比较

- **自注意力（Self-Attention）**
  - 利用查询（Q）、键（K）、值（V）计算相似度，动态调整信息权重
  - 可直接捕捉任意位置之间的长程依赖
  - 多头机制允许在不同子空间中并行学习多种模式
  - 信息交互充分，实现全局与局部特征融合
- **前馈注意力（Feedforward Network）**
  - 采用固定的非线性变换，每个位置独立处理
  - 难以显式捕捉跨位置的信息交互
  - 主要用于局部特征转换，表达能力受限

总结

- **RNN / 前馈注意力的局限**
  - 难以捕捉长程依赖与跨位置信息
  - 独立位置、静态映射、向量空间不匹配，缺乏语义对齐
- **自注意力计算流程**
  - 查询 $Q = W^Q X$;键$\mathrm{K} = W^K X$，值 $\mathrm{V} = W^V X$
  - 相似度计算：$\frac{QK^T}{\sqrt{d_k}}$
  - 归一化：Softmax 得到权重
  - 聚合：权重乘以 V，实现全局信息汇聚
- **自注意力实现动态语义对齐与再分配**

总结

自注意力的优点

- **高并行性**：无需序列顺序，支持全序列同时计算
- **长程依赖捕捉**：直接建模任意位置间关联
- **动态权重分配**：根据输入内容自适应聚焦最相关上下文

自注意力的局限

- **单一视角**：一次注意力只能聚焦一种模式，需要多头并行分担
- **深层表示不足**：纯自注意难学层次结构，需前馈层与多层堆叠
- **训练不稳定**：深层堆叠易导致梯度问题，需残差连接与归一化

参考阅读

- Dan Jurafsky and James H. Martin. Speech and Language Processing (3rd ed. draft); Chapter 8, 9. 2025
  - 系统介绍了RNN和LSTM的原理、优势以及局限
  - 系统介绍了self-attention的计算方法
- Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola & Eduard Hovy. “Hierarchical Attention Networks for Document Classification.”
  - 在双层 Bi-LSTM 结构上分别加入句子级和词级注意力，允许模型聚焦于最具判别力的词和句子，用于长文档分类。
  - 该方法在 Yelp 评论和 Yahoo Answers 等多项文档分类任务上超越传统 LSTM 和 CNN 基线。

参考阅读

- Alexander M. Rush, Sumit Chopra & Jason Weston. “A Neural Attention Model for Abstractive Sentence Summarization.”
  - 在 LSTM 编码-解码框架中加入局部注意力，用于短句摘要生成，模型在基准数据集上获得显著性能提升；
  - 该工作奠定了注意力机制在文本生成任务中替代传统提取式方法的基础。
- Abigail See, Peter J. Liu & Christopher D. Manning. “Get To The Point: Summarization with Pointer-Generator Networks.
  - 在标准的 LSTM seq2seq + attention 框架中，引入 hybrid pointer-generator 机制：模型既能像传统生成器那样生成新词，又能通过指针复制原文中的单词，显著提升了摘要的事实准确性和对 OOV 词的覆盖能力

课后思考

- 在注意力机制中，“注意力”具体指的是什么？举一个日常生活中的例子，说明注意力机制是如何帮助人类更有效地处理信息的。

- 注意力得分（Attention Score）为什么需要进行归一化（如使用Softmax函数）？如果不进行归一化，可能会出现哪些问题？