CS 330 MIP – Lecture 8

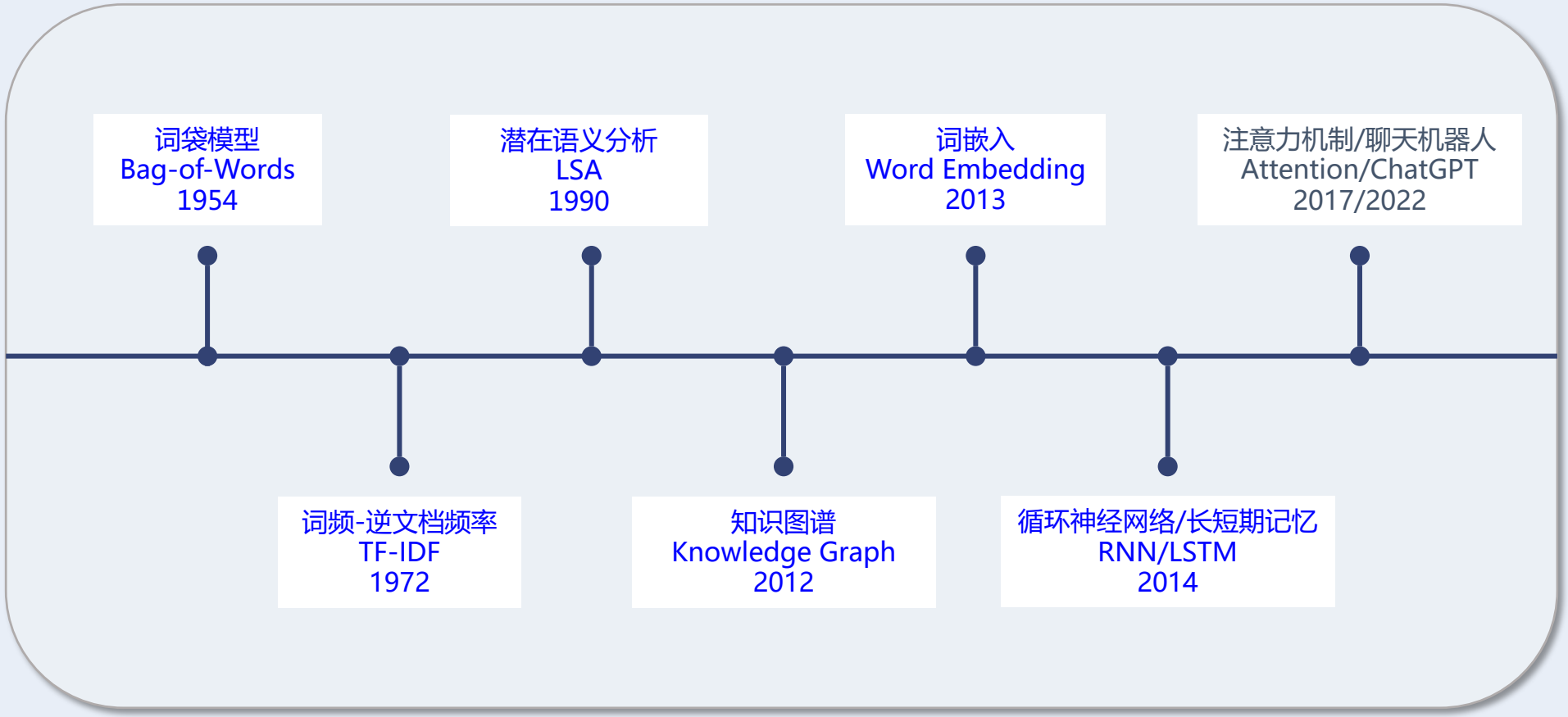
自然语言处理

Natural Language Processing

Jimmy Liu 刘江

2024-04-10

文本处理发展里程碑


词袋模型
Bag-of-Words
1954
词频-逆文档频率
TF-IDF
1972
潜在语义分析
LSA
1990
知识图谱
Knowledge Graph
2012
词嵌入
Word Embedding
2013
循环神经网络/长短期记忆
RNN/LSTM
2014
注意力机制/聊天机器人
Attention/ChatGPT
2017/2022

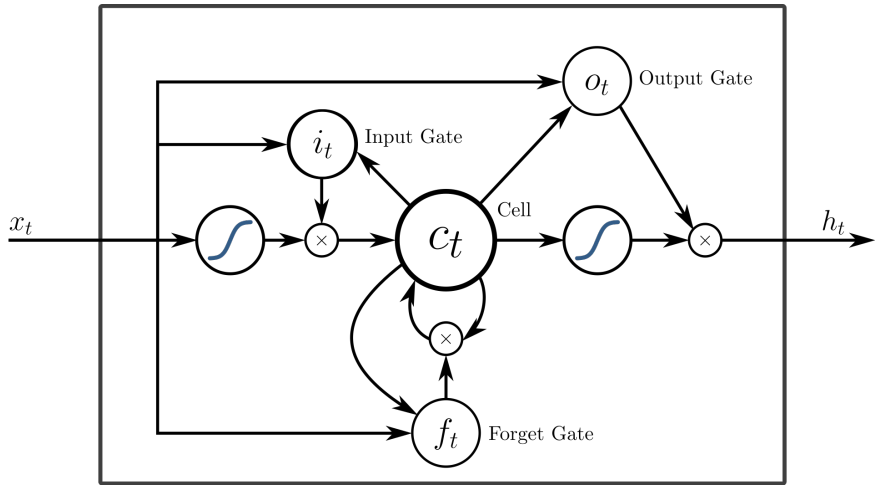
循环神经网络 RNN/LSTM

- 循环神经网络（RNN）是一种在处理序列数据时广泛应用的神经网络模型。与传统的前馈神经网络不同，RNN具有记忆能力，可以在处理序列数据时保留先前的信息。
- RNN的关键特点是它能够处理不定长度的输入序列。RNN会接收当前的输入和上一个时间步的隐藏状态，并输出当前时间步的隐藏状态和相应的输出。这种隐藏状态的传递机制使得RNN能够在处理序列数据时考虑上下文和历史信息。
- RNN的隐藏状态可以被看作是网络对过去观察的累积记忆，它能够捕捉到序列数据中的时间依赖关系。这使得RNN在自然语言处理领域中非常有用，可以用于语言建模、机器翻译、文本生成等任务。

长短时记忆 LSTM

- 输入门i：根据X_t和h_{t-1}选择接收什么输入信息（0到1值）
- 遗忘门f：根据X_t和h_{t-1}决定保留或遗忘历史信息（0到1值）
- 状态更新g：根据X_t和h_{t-1}决定对细胞状态的一个潜在更新（-1到1值）
- 输出门o：根据X_t和h_{t-1}决定下一个隐状态（0-1值）；
- 细胞单元c：根据f_t ,i_t ,g_t和C_{t-1}决定维护单元状态；
- 隐藏状态h：根据o_t ,c_t (-1到1值)

$$
\begin{aligned}
i_t &= \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi}) \\
f_t &= \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}) \\
g_t &= \tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}) \\
o_t &= \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}) \\
c_t &= f_t \odot c_{t-1} + i_t \odot g_t \\
h_t &= o_t \odot \tanh(c_t)
\end{aligned}
$$



RNN/LSTM 应用

语言模型 (Language Model, LM)

语言模型是RNN的一个重要的应用，其目的是预测下一个单词的概率。对于一个文本序列 $X = \{x_1, \dots, x_{t-1}, x_t, \dots, x_T\}$，下一个单词$x_t$的概率可以表示为：

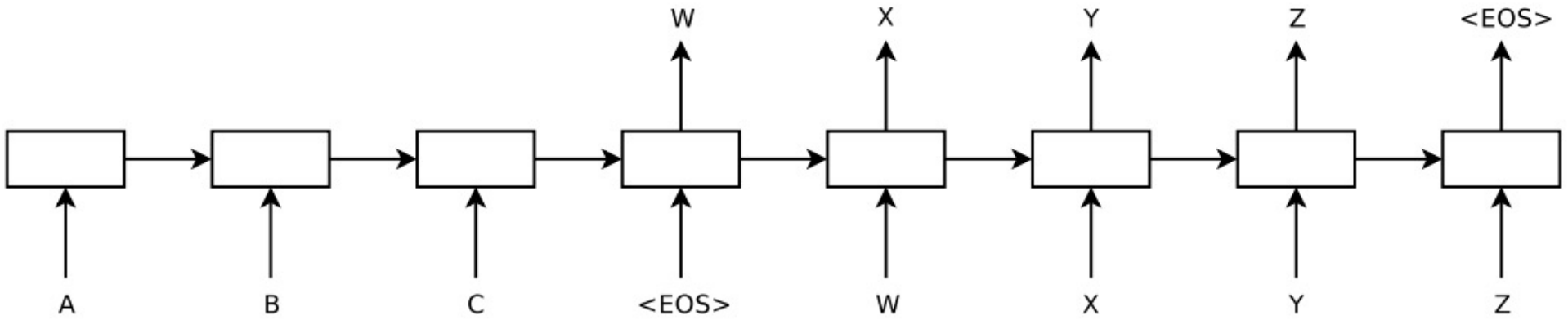
$$p(x_t | x_{t-1}, \dots, x_1)$$

而序列X的概率则表示为：

$$p(X) = p(x_1, \dots, x_T) = p(x_1) \prod_{t=1}^{T} p(x_t | x_{t-1}, \dots, x_1)$$

Sequence-to-Sequence 序列到序列学习

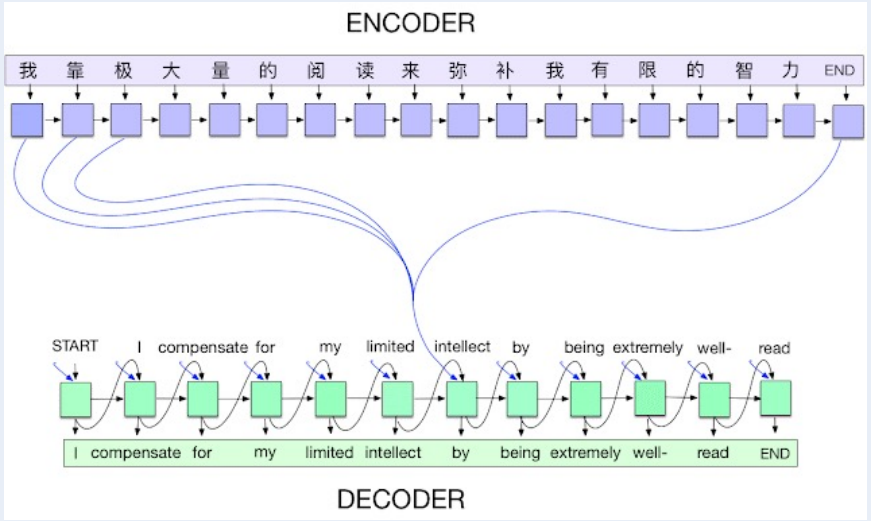
- S2S由编码器(Encoder)和解码器(Decoder)组成：
  - 编码器：将输入文本序列编码成向量；
  - 解码器：根据输入，生成对应的文本序列；



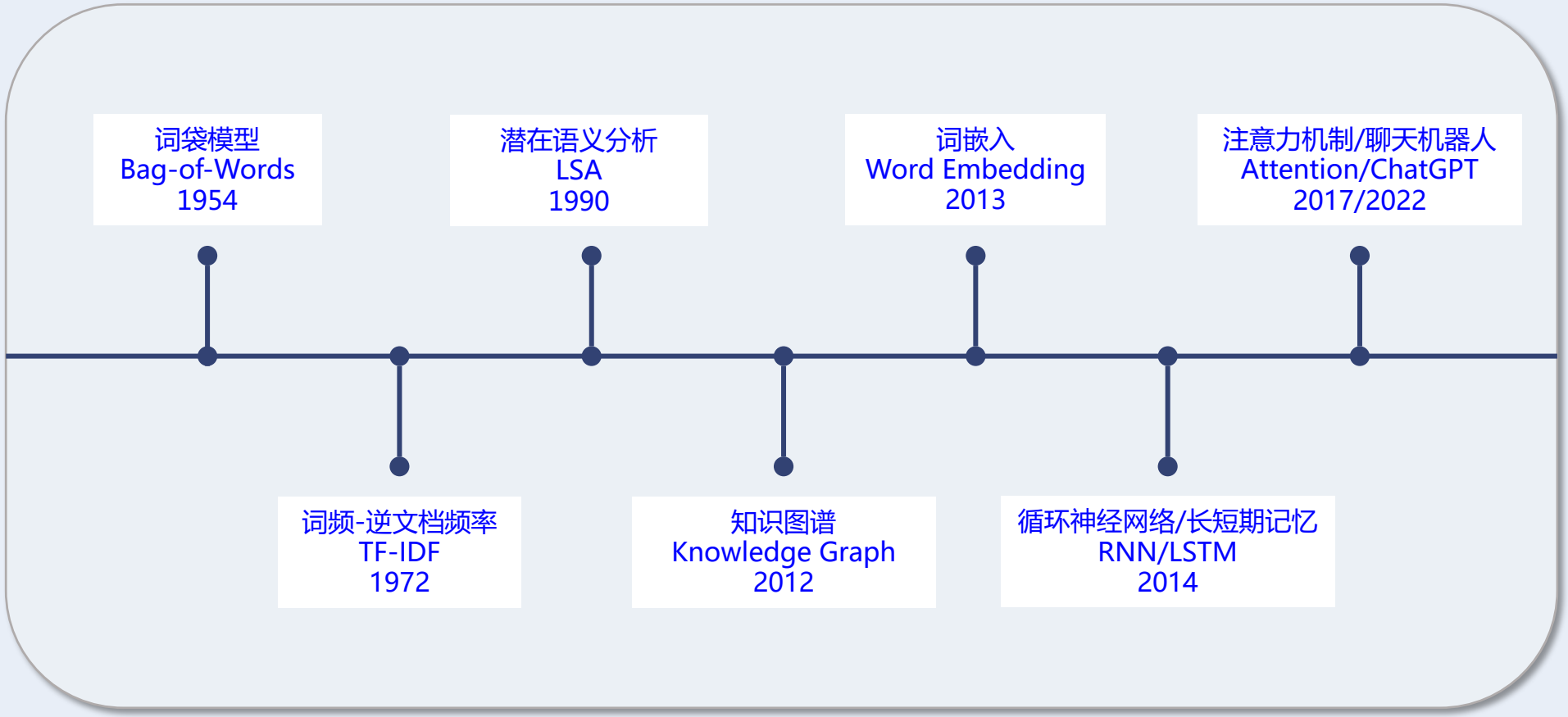
Sequence-to-Sequence 应用

机器翻译 (Machine Translation)

- S2S（Sequence-to-Sequence）模型适合语言翻译的原因主要在于其能够处理输入序列和输出序列之间的映射关系。这种模型通常包括编码器（Encoder）和解码器（Decoder）两个部分，编码器将源语言序列输入网络，并逐个生成隐藏状态，这些隐藏状态捕捉序列中的信息。随后，解码器根据这些隐藏状态生成目标语言序列。
- 与传统的基于规则引擎和统计模型的语言翻译方法相比，S2S模型具有显著优势。在面对复杂的语言表达和多样性的语言结构时，传统方法的表现力有限。而S2S模型通过自动学习语言的复杂结构和规律，能够实现更加准确和自然的翻译。



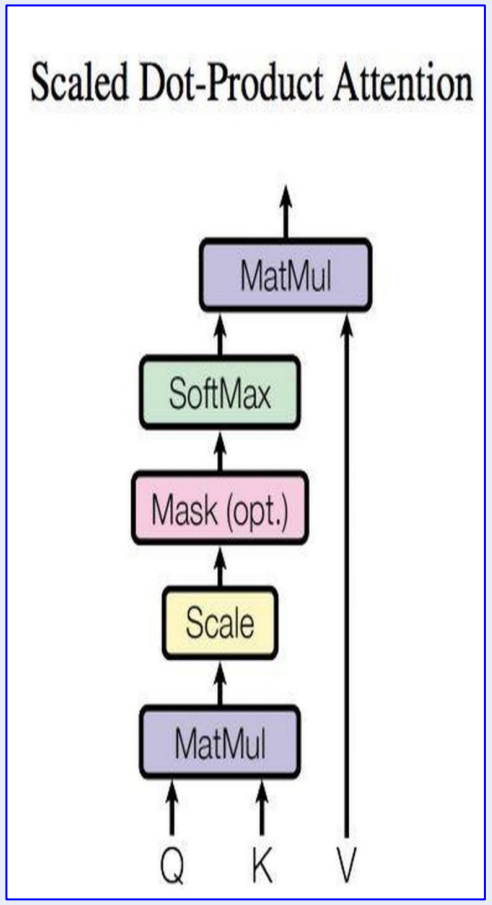
文本处理发展里程碑


词袋模型
Bag-of-Words
1954
词频-逆文档频率
TF-IDF
1972
潜在语义分析
LSA
1990
知识图谱
Knowledge Graph
2012
词嵌入
Word Embedding
2013
循环神经网络/长短期记忆
RNN/LSTM
2014
注意力机制/聊天机器人
Attention/ChatGPT
2017/2022

注意力机制 (Attention)

- 注意力(Attention)模拟了人类在处理信息时的注意力分配过程。 注意力机制的核心思想是根据输入的不同部分的重要性，动态地分配模型的注意力。
- 广义上说，注意力被表达为一个函数，将查询 (query) 和一组键值对 (Key, Value)映射到一个输出。其中，查询、键、值和最终输出都是向量。输出则被计算为值的加权和，每个值的权重由查询与相应键的兼容性函数来表示。
- 实际上，注意力机制使神经网络能够近似模拟人类使用的视觉注意机制。就像人们处理新场景时，模型会将强烈的“高分辨率”关注点放在图像的某一部分上，同时以“低分辨率”感知周围区域，然后随着网络对场景的理解而调整关注点。

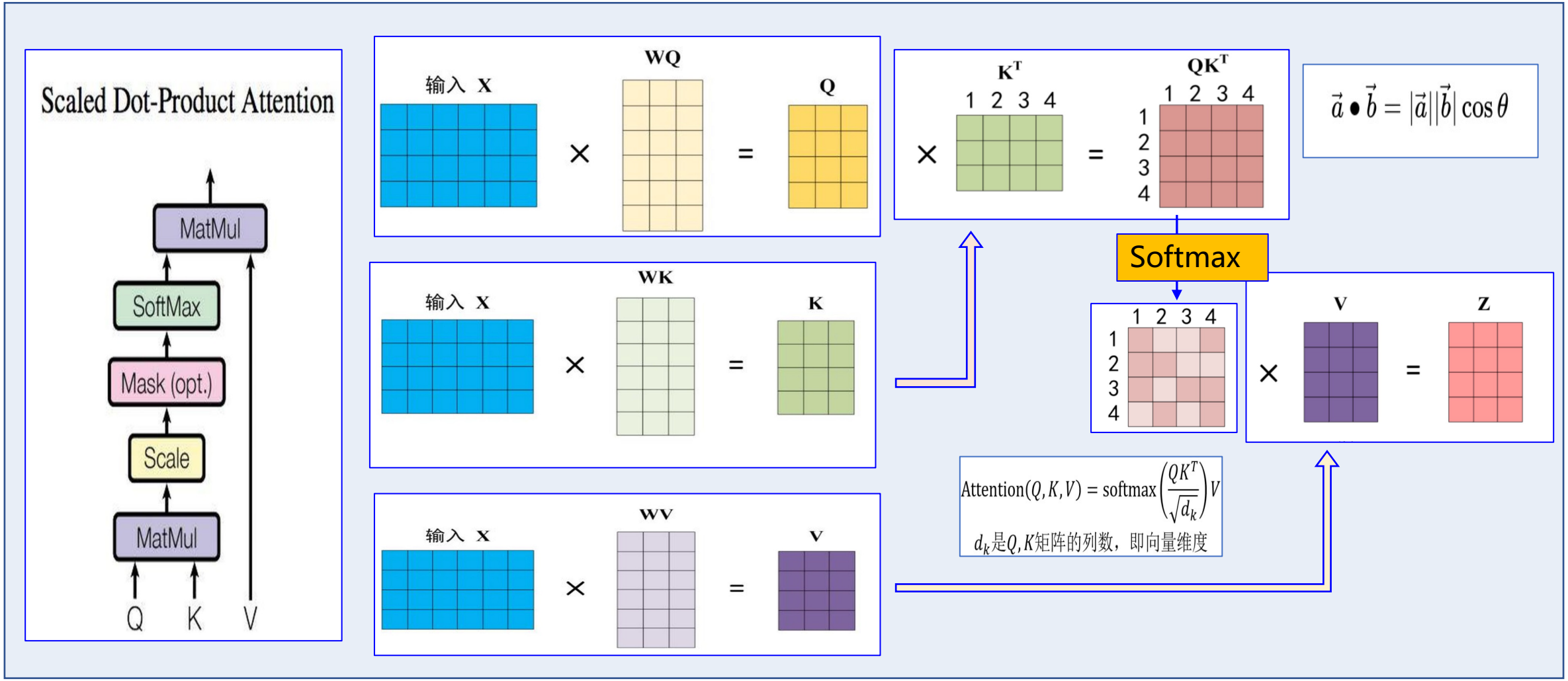
Transformer中的注意力机制



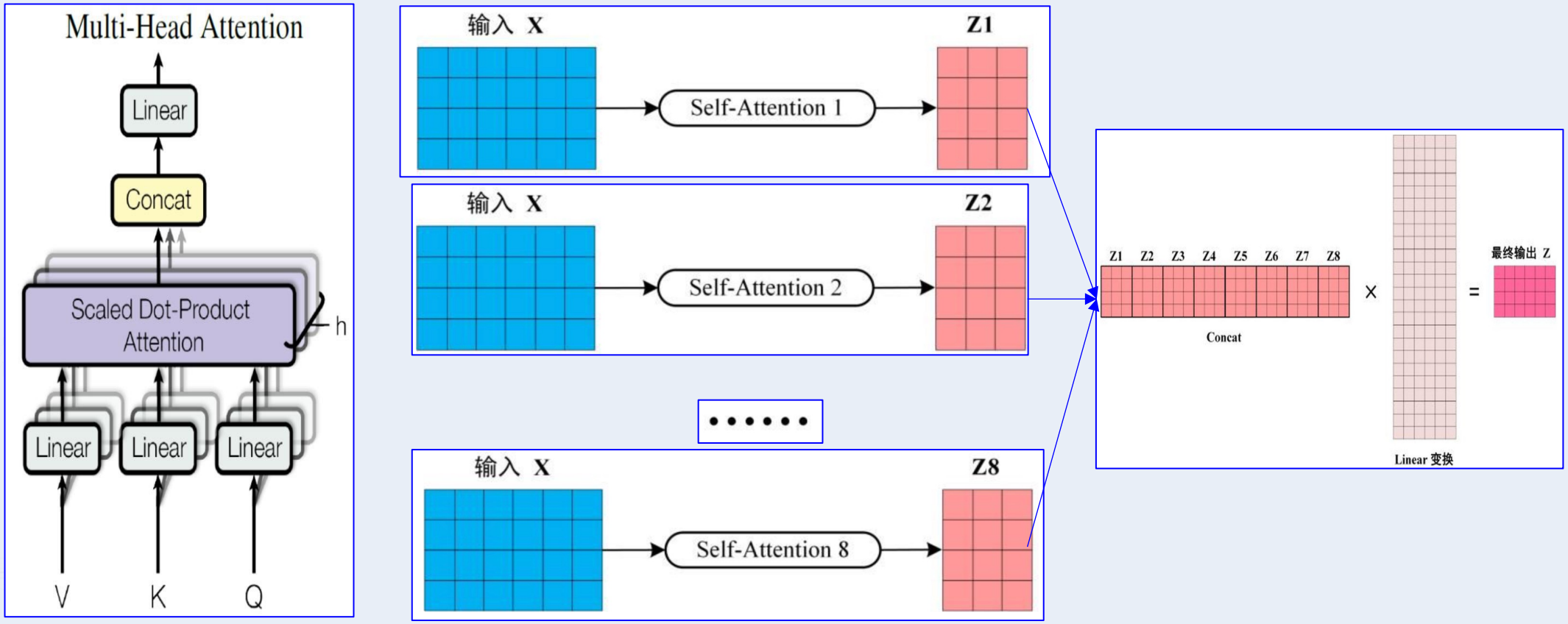
- Scaled Dot-Product Attention（缩放点积注意力）是一种常用的自注意力机制，用于在深度学习中对序列数据进行建模。其核心计算部分包括三个主要步骤：

1. **点积计算（Dot Product）**：首先，将查询（Query）矩阵Q和键（Key）矩阵K进行点积运算。这一步骤计算了查询向量和所有键向量之间的相似度得分，即衡量查询向量与每一个位置上的键向量有多匹配。
2. **缩放（Scaling）**：由于随着维度d的增加，点积的结果也会迅速增大，可能导致softmax函数梯度变得极小，影响训练效果。因此，对点积结果除以d的平方根进行缩放。这样可以保持各个位置上的注意力得分在softmax之前具有相近的尺度，确保模型收敛性能更好。
3. **加权求和（Weighted Sum）**：最后，将注意力权重矩阵与值（Value）矩阵V进行点积运算，得到最终的上下文向量（Context Vector）。每个输出位置的向量是由值向量经过注意力权重调整后综合而成的。

Transformer中的注意力机制



多头注意力的具体实现方式

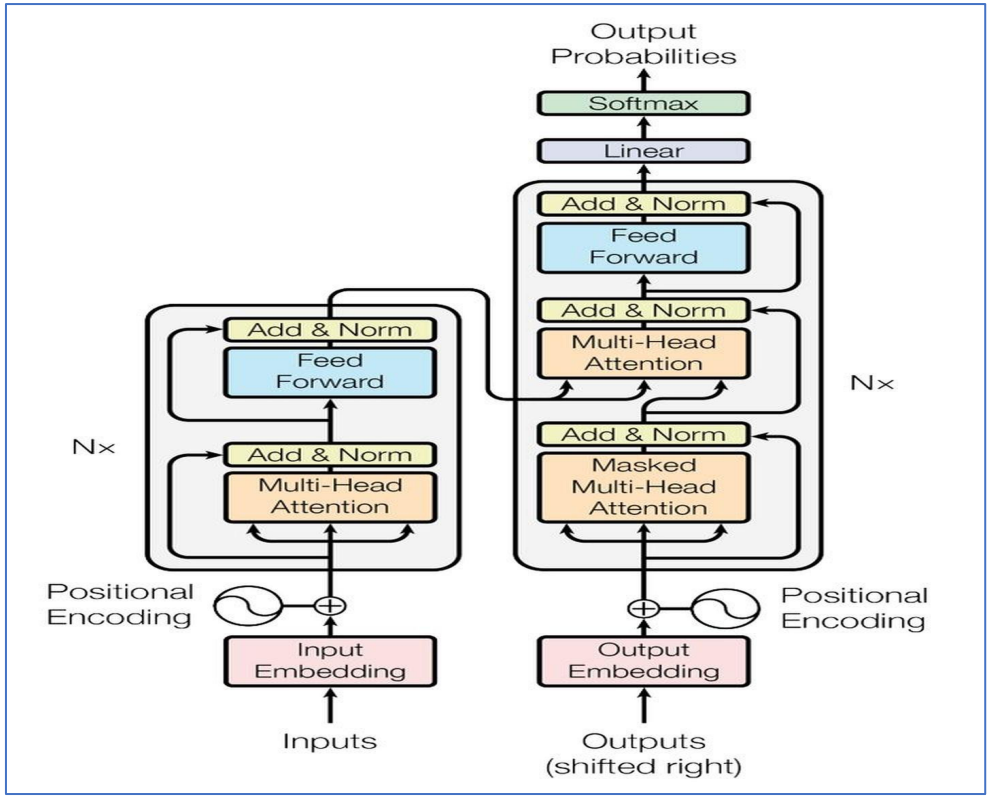


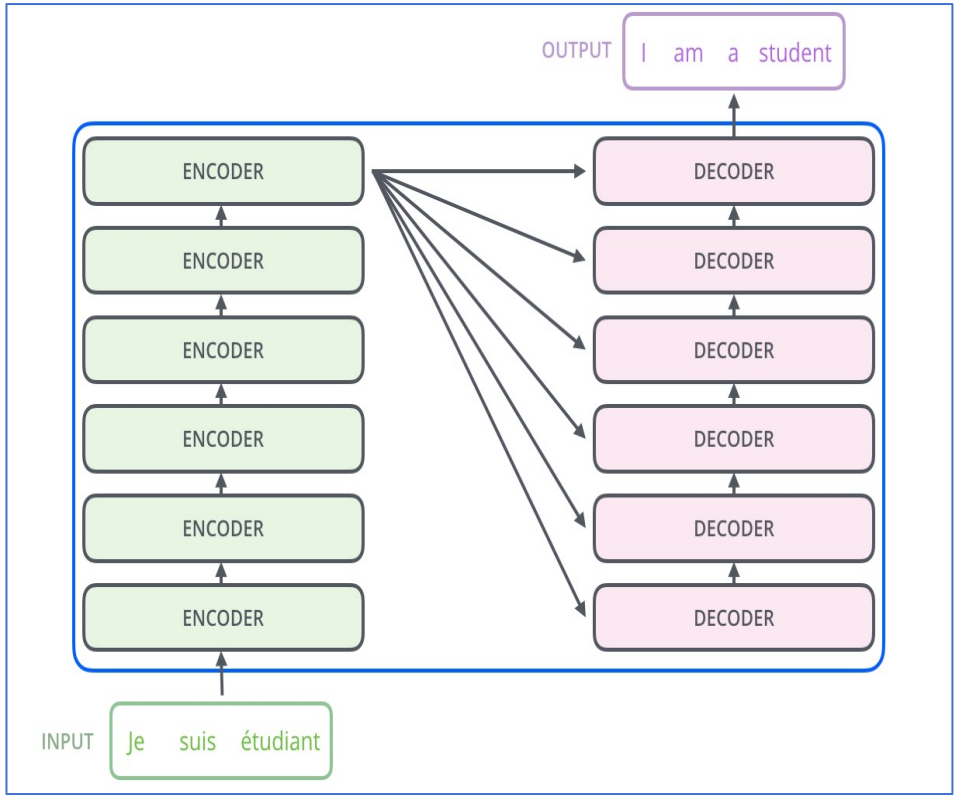
在Transformer中，注意力模块并行地多次重复其计算。其中每一个都被称为一个注意力头（Attention Head）。注意力模块将其查询（Query）、键（Key）和值（Value）参数分为N个部分，并分别将每个分割部分通过单独的注意力头独立处理。

多头注意力与NLP

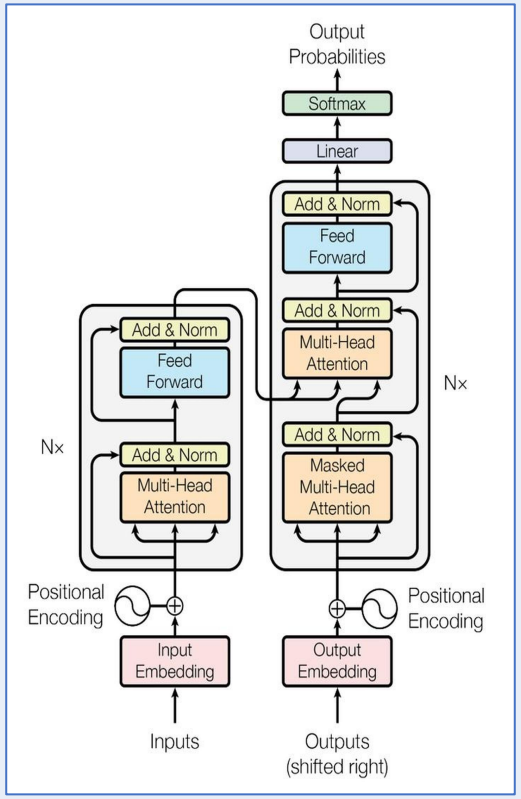
- 多头注意力（Multi-Head Attention）是一种在Transformer模型中被广泛采用的注意力机制扩展形式，通过并行地运行多个独立的注意力机制来获取输入序列的不同子空间的注意力分布，从而更全面地捕获序列中潜在的多种语义关联。
- 具体来说，在多头注意力中，输入序列首先通过三个不同的线性变换层分别得到Query、Key和Value。然后，这些变换后的向量被划分为若干个“头”，每个头都有自己独立的Query、Key和Value矩阵。通过这种方式，多头注意力能够并行地从不同的角度对输入序列进行注意力处理，提高了模型理解和捕捉复杂依赖关系的能力。
- 多头注意力机制的计算过程包括：将输入张量拆分成多个子张量，每个子张量都以不同的方式学习到的注意力信息；对每个子张量执行一次自注意力计算，得到一个输出张量；最后，将多个输出张量拼接在一起，得到最终的输出张量。
- 多头注意力机制可以处理多个关注点的问题，能够较好地处理复杂语义关系，特别是在自然语言处理领域，它能够模拟人类在理解语言时的注意力机制，将特定的注意力放在不同的词或短语上，从而提取出更有效的特征表示。

Transformer -编码层/解码层,



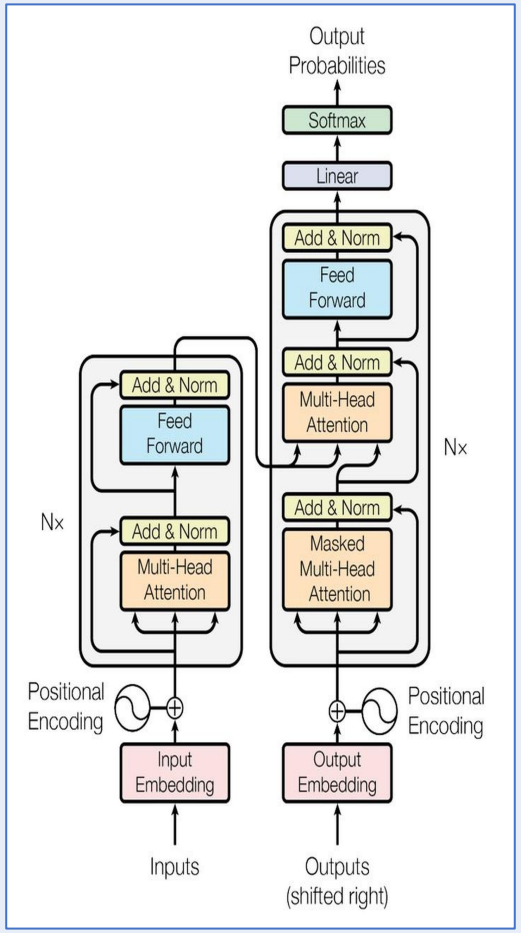


Transformer 位置偏移



- 在Transformer模型中，“output (shifted right)”通常指的是在解码器生成输出序列时，对预测的目标位置进行偏移的处理。这种偏移是由于Transformer模型在自回归生成过程中，需要防止模型将有关当前位置的信息泄露到未来位置的预测中。
- 具体来说，对于每个位置i，预测的输出实际上是i+1位置的嵌入（embedding），而不是i位置的嵌入。这是因为在解码过程中，模型是根据当前位置之前的所有信息来预测下一个位置的词或符号。通过将输出偏移一位，可以确保模型不会直接访问到它应该预测的那个位置的信息。
- 这种“shifted right”的处理通常是在实现Transformer模型时的一个技术细节。在模型训练或推理时，输入到解码器的目标序列会进行这种偏移，以便模型能够正确地生成下一个位置的输出。这种机制有助于模型更好地捕捉序列中的上下文信息，并生成连贯且准确的输出序列。

Transformer 位置编码



- 在Transformer模型中，位置编码（Positional Encoding）起着至关重要的作用。由于Transformer模型主要依赖于自注意力机制来处理输入序列，这种机制本身并不考虑序列中元素的位置信息。然而，对于许多序列处理任务（尤其是自然语言处理任务）来说，元素的位置是非常重要的。因此，需要一种方式来向模型提供位置信息，这就是位置编码的作用。
- 位置编码的主要目的是给输入序列中的每个位置分配一个独特的向量，这样模型就能够区分不同位置的信息。这些向量通常与输入序列的嵌入表示（Word Embedding）相结合，作为Transformer模型的输入。
- 有多种方法可以实现位置编码，其中最常见的是正弦和余弦函数的位置编码（也称为Sinusoidal位置编码）。这种编码方式通过不同频率的正弦和余弦函数来生成位置向量，每个位置都有一个独特的编码。这种编码方式的一个优点是它可以处理任意长度的序列，因为对于每个新的位置，都可以简单地计算其对应的位置向量

BERT

BERT（Bidirectional Encoder Representations from Transformers）是一种基于Transformer的机器学习技术，用于自然语言处理的预训练。它由Google开发，并于2018年由Jacob Devlin和他的Google团队发布。自2019年以来，Google一直在利用BERT来更好地理解用户搜索。

BERT模型架构：
BERT有两个版本：Base和Large。Base版本包含12层编码层，参数量为1.1Million，而Large版本包含24层编码层，参数量为3.4Million。根据BERT论文的实验，Large版本的效果比Base版本的好。

GPT-3 预训练数据- Common Crawl

Common Crawl

Nonprofit

commoncrawl.org

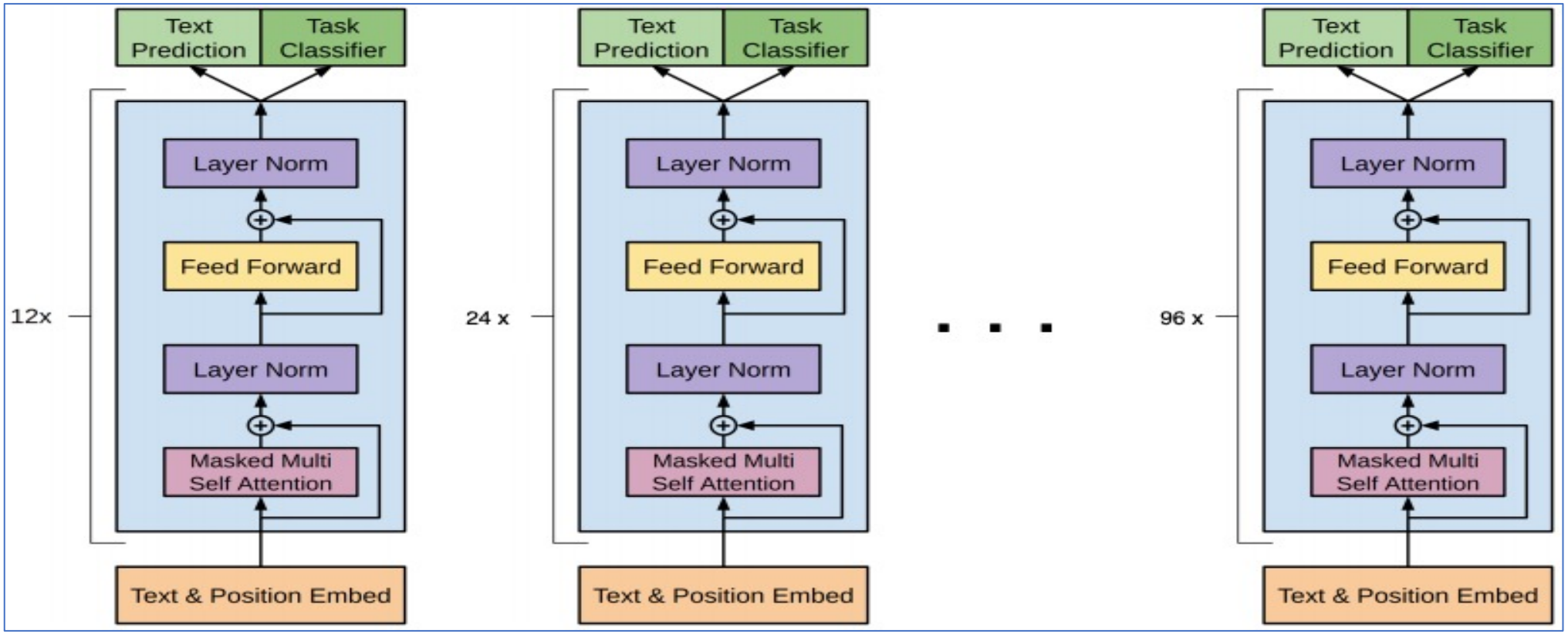
Common Crawl is a nonprofit 501 organization that crawls the web and freely provides its archives and datasets to the public. Common Crawl's web archive consists of petabytes of data collected since 2011. It completes crawls generally every month. Common Crawl was founded by Gil Elbaz. Wikipedia

Founder: Gil Elbaz

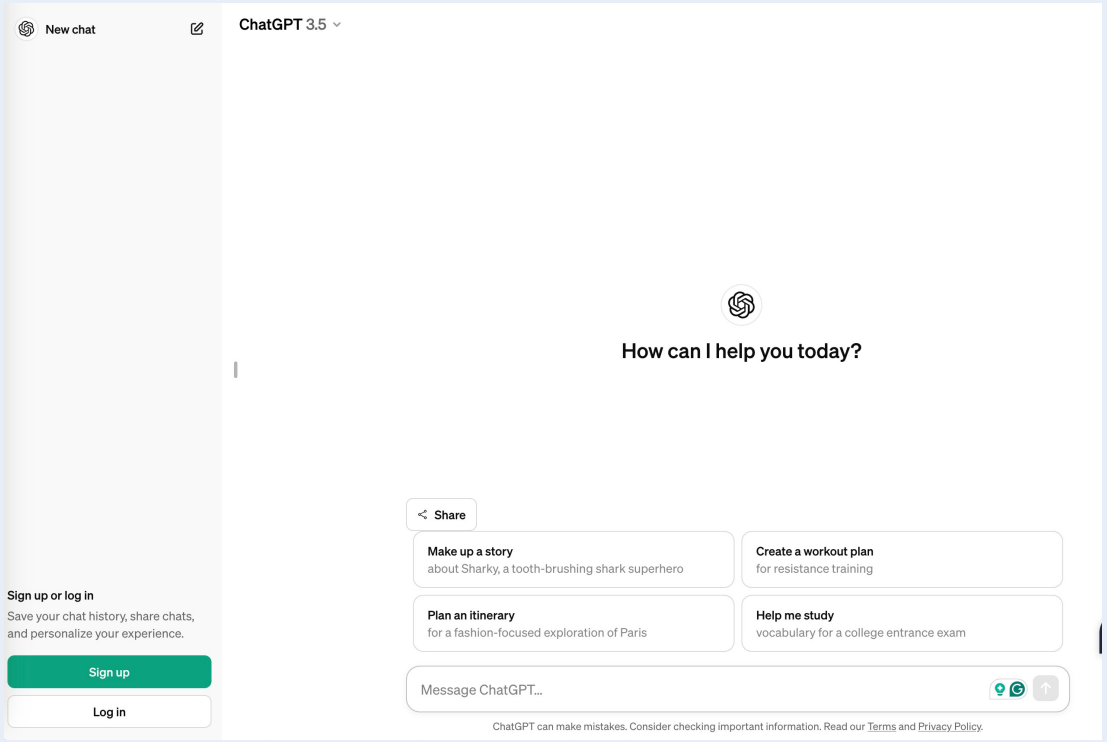
Founded: 2007

中文单位	中文简称	英文单位	英文简称	进率（Byte=1）
位	比特	bit	b	0.125
字节	字节	Byte	B	1
千字节	千字节	KiloByte	KB	2^10
兆字节	兆	MegaByte	MB	2^20
吉字节	吉	GigaByte	GB	2^30
太字节	太	TeraByte	TB	2^40
拍字节	拍	PetaByte	PB	2^50
艾字节	艾	ExaByte	EB	2^60
泽字节	泽	ZettaByte	ZB	2^70
尧字节	尧	YottaByte	YB	2^80
千亿亿亿字节	千亿亿亿字节	BrontByte	BB	2^90

GPT-3 模型架构


Text Prediction
Task Classifier
Layer Norm
Feed Forward
Layer Norm
Masked Multi Self Attention
Text & Position Embed
12x
24 x
96 x

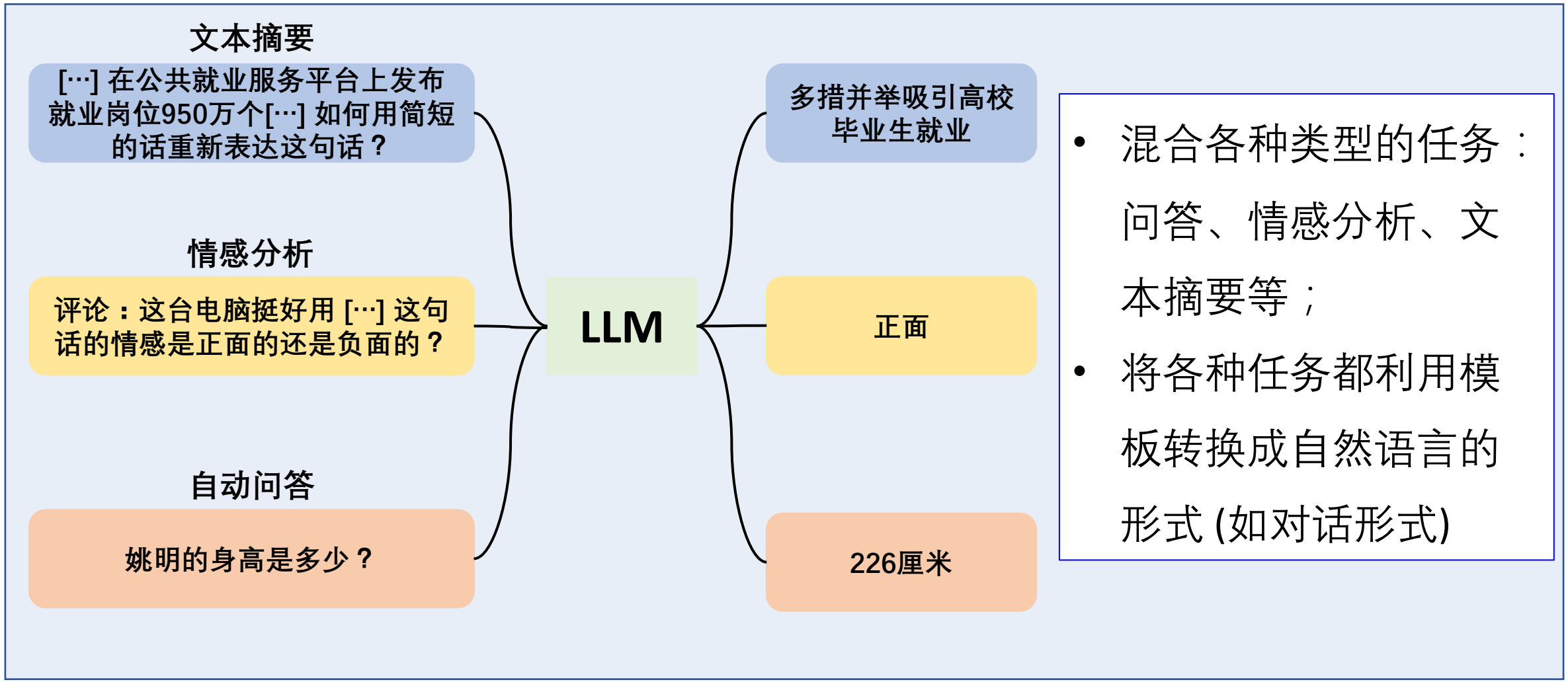
ChatGPT



对比GPT-3，ChatGPT做了以下调整：

- 指令微调(Instruction Tuning)
  - 通过混合各种任务的数据，让GPT学会服从人类的指令
- 基于强化学习的人类反馈学习(Reinforcement Learning from Human Feedback, RLHF)
  - 通过RLHF，使得模型的输出与人类的话语更接近

指令微调(Instruction Tuning)



- 混合各种类型的任务：问答、情感分析、文本摘要等；
- 将各种任务都利用模板转换成自然语言的形式 (如对话形式)

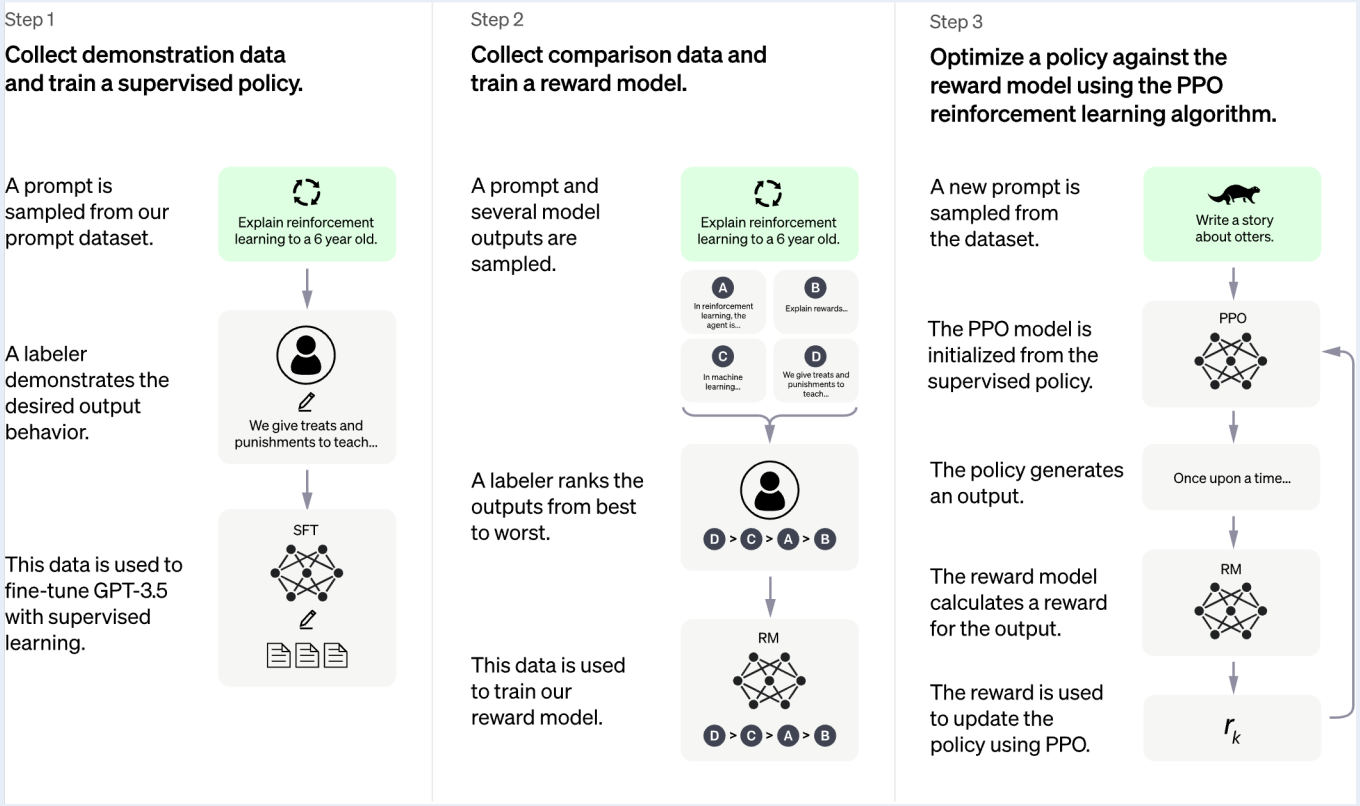
基于机器学习的人类偏好学习(RLHF)

大模型的输出对比人类的文字还有很大差距，RLHF通过以下三个步骤对齐人类偏好：

（1）收集数据，对大模型微调；

（2）人类对模型的输出打分并排序，用这些数据训练奖励模型(Reward Model);

（3）以奖励模型的得分为信号，用强化学习进一步优化大模型。



Homework 08

No Homework This Week

CS 330 MIP – Lecture 8

自然语言处理

Natural Language Processing

Jimmy Liu 刘江

2024-04-10