图解Transformer架构设计

韩冰 AI大模型实验室 2024-03-24 03:32 美国

近年来，Transformer 技术在自然语言处理（NLP）领域引起了巨大关注。**Transformer 是一种利用注意力机制（Attention）极大提升深度学习 NLP 翻译模型性能的新型架构。**它首次出现在《Attention is all you need》这篇论文中，迅速成为了文本数据处理的主流架构。

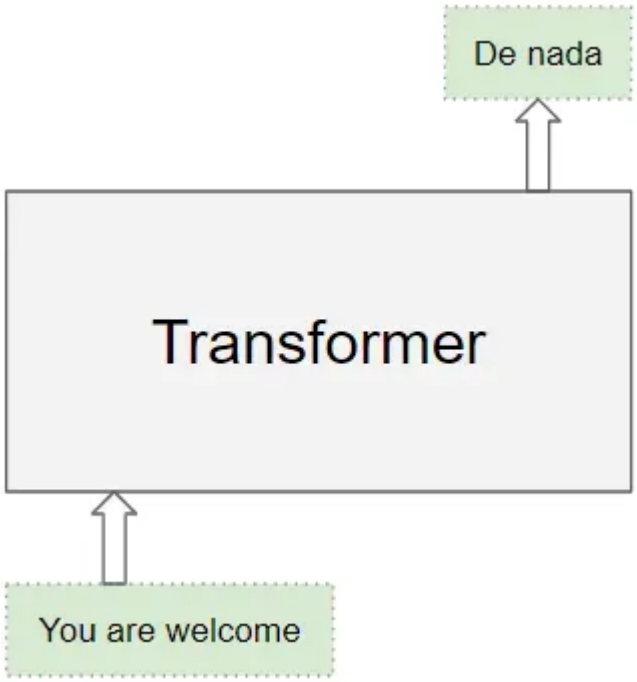
自此以后，谷歌的 BERT 和 OpenAI 的 GPT 系列等众多项目，都基于这一架构构建，表现远超现有的最先进技术。

在接下来的系列文章中，我将深入浅出地讲解 Transformer 的基本原理、架构设计及其内部运作机制。我们会逐步揭示 Transformer 的核心功能。在后续的文章里，我们还将深入剖析系统的运行细节，特别是多头注意力（multi-head attention）机制 ——Transformer 的关键所在。

本文是这个系列文章的第一篇，主要介绍 Transformer 的使用方法，为何优于 RNN，同时介绍了其架构组成以及在训练和推理期间的行为特点。

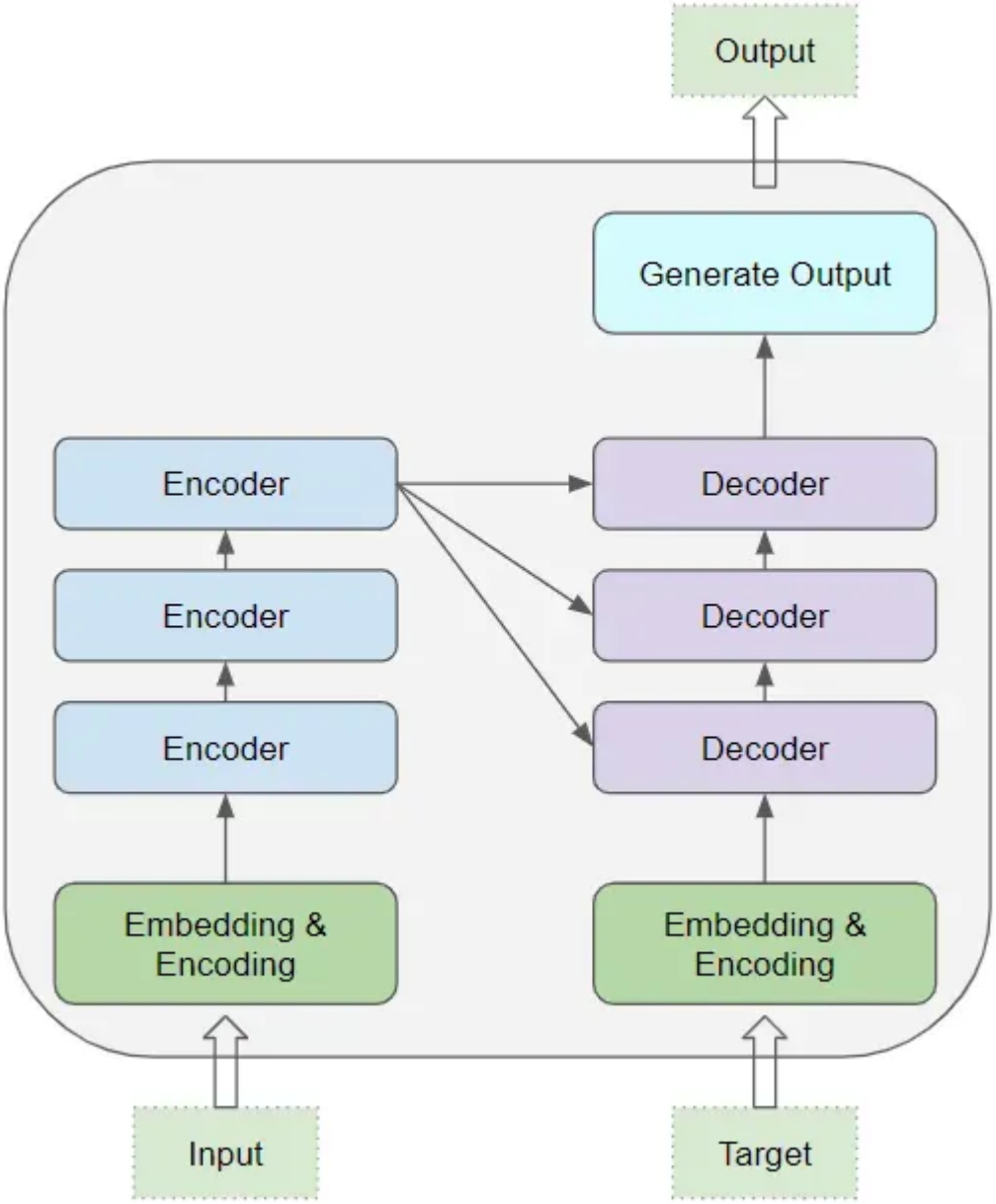
#01
什么是 Transformer？

Transformer 架构特别擅长处理顺序性强的文本数据。简单来说，它可以将一段文本作为输入，并输出另一段文本，比如将英语句子翻译成西班牙语。

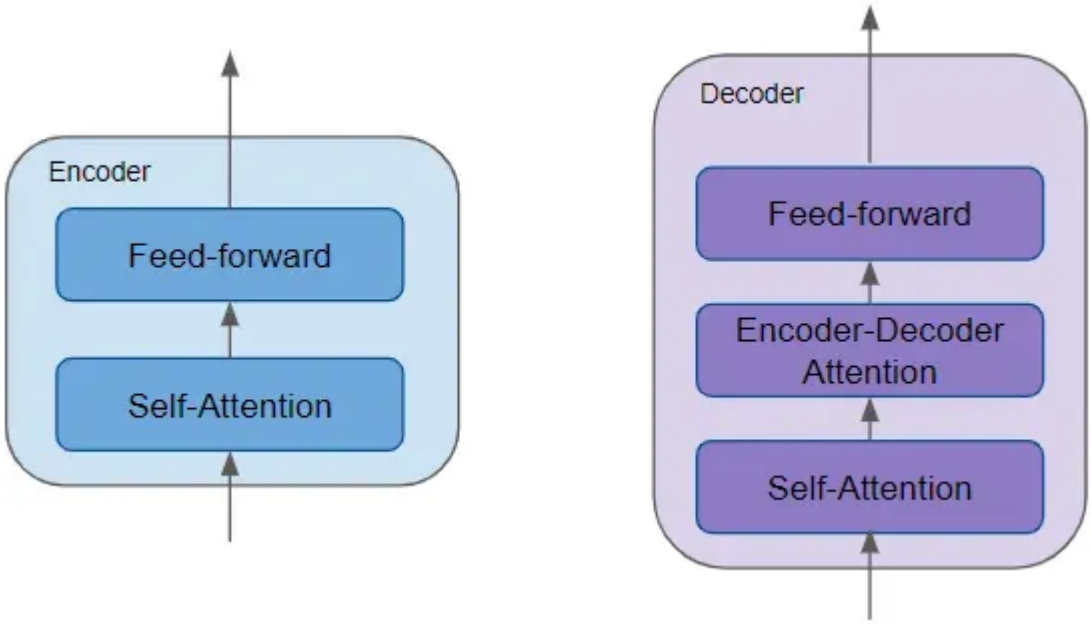


Transformer 的核心由多个编码层（Encoder）和解码层（Decoder）构成。这里，我们将单个层称为编码器或解码器，而一组这样的层则称为编码器组或解码器组。

编码器组和解码器组各有其对应的嵌入层（Embedding layers），处理各自的输入数据。最终，通过一个输出层生成最后的结果。

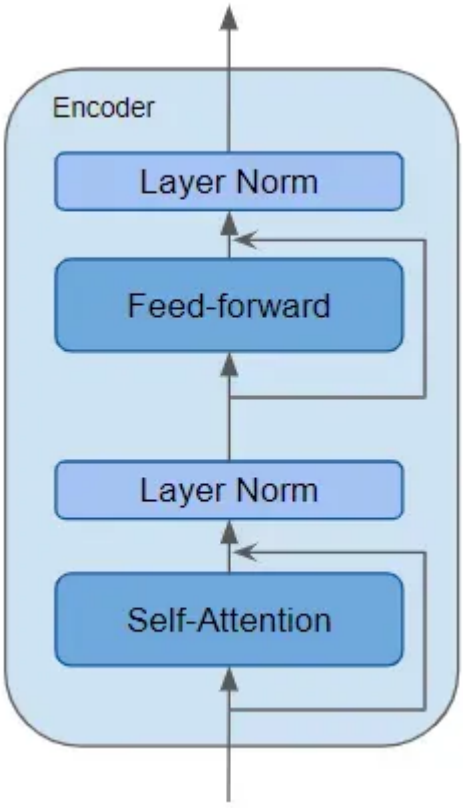


所有的编码器结构相同，解码器也是如此。



编码器包含一个关键的自注意力层，用来计算文本序列中不同单词间的联系，还有一个前馈层。而解码器则包括自注意力层、前馈层以及额外的编码器 - 解码器注意力层。每个编码器和解码器都有自己的权重集合。

编码器是可复用的模块，是定义 Transformer 架构的核心部分。除了上述两层，它还包括围绕这些层的残差跳跃连接和两个 LayerNorm 层。



Transformer 有许多不同的变体，其中一些变体甚至没有解码器，完全依赖编码器来工作。

#02

注意力机制的作用是什么？

Transformer 的强大性能源于其使用的注意力机制。

这种机制允许模型在处理某个单词时，同时关注输入中与该单词紧密相关的其他单词。

例如，单词 “Ball” 与 “blue” 和 “holding” 紧密相关，而与 “boy” 无关。

Transformer 通过自注意力机制，将输入序列中的每个单词与其他所有单词联系起来。

举个例子：

1. The cat drank the milk because it was hungry.

2. The cat drank the milk because it was sweet.

在第一句中，“it” 指代的是 “cat”，而在第二句中，“it” 指的是 “milk”。当模型处理 “it” 这个词时，自注意力机制提供了更多关于它的含义信息，从而帮助模型准确地将 “it” 与相关联的单词对应起来。

为了更细致地处理句子的意图和语义，Transformer 对每个单词赋予了多重注意力得分。

比如，在处理 “it” 时，第一个得分会突出 “cat”，第二个得分则突出 “hungry”。因此，当它把 “it” 翻译成另一种语言时，会把 “cat” 和 “hungry” 的某些元素融入到翻译中。

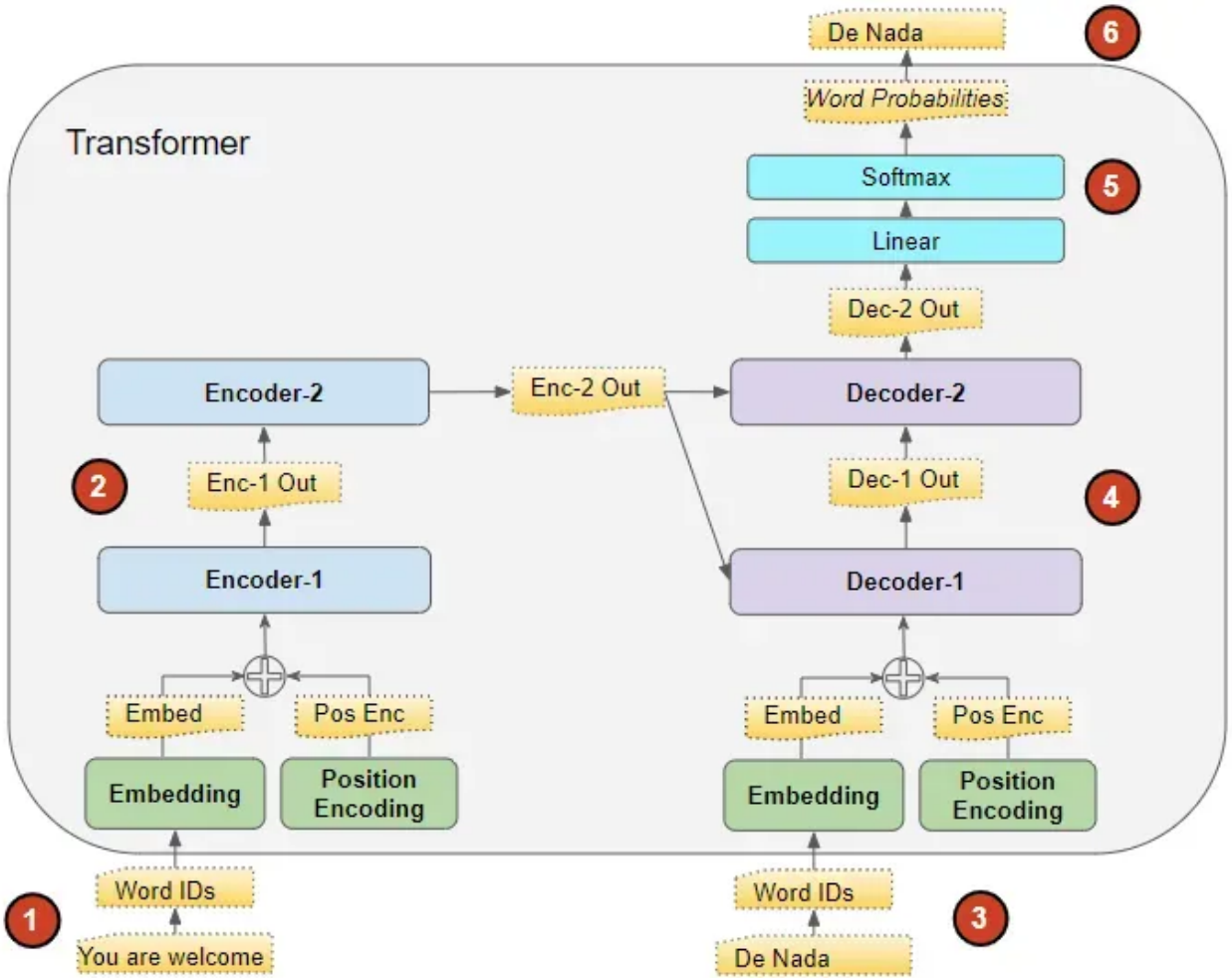
#03
训练 Transformer

在训练和推理阶段，Transformer 的运作略有不同。

首先看训练阶段的数据流程。训练数据分为两部分：

- 源序列或输入序列（如翻译问题中的英文“You are welcome”）
- 目标序列或目标序列（如西班牙语的“De nada”）

Transformer 的目标是学习如何根据输入序列和目标序列来生成目标序列。



Transformer 的处理过程如下：

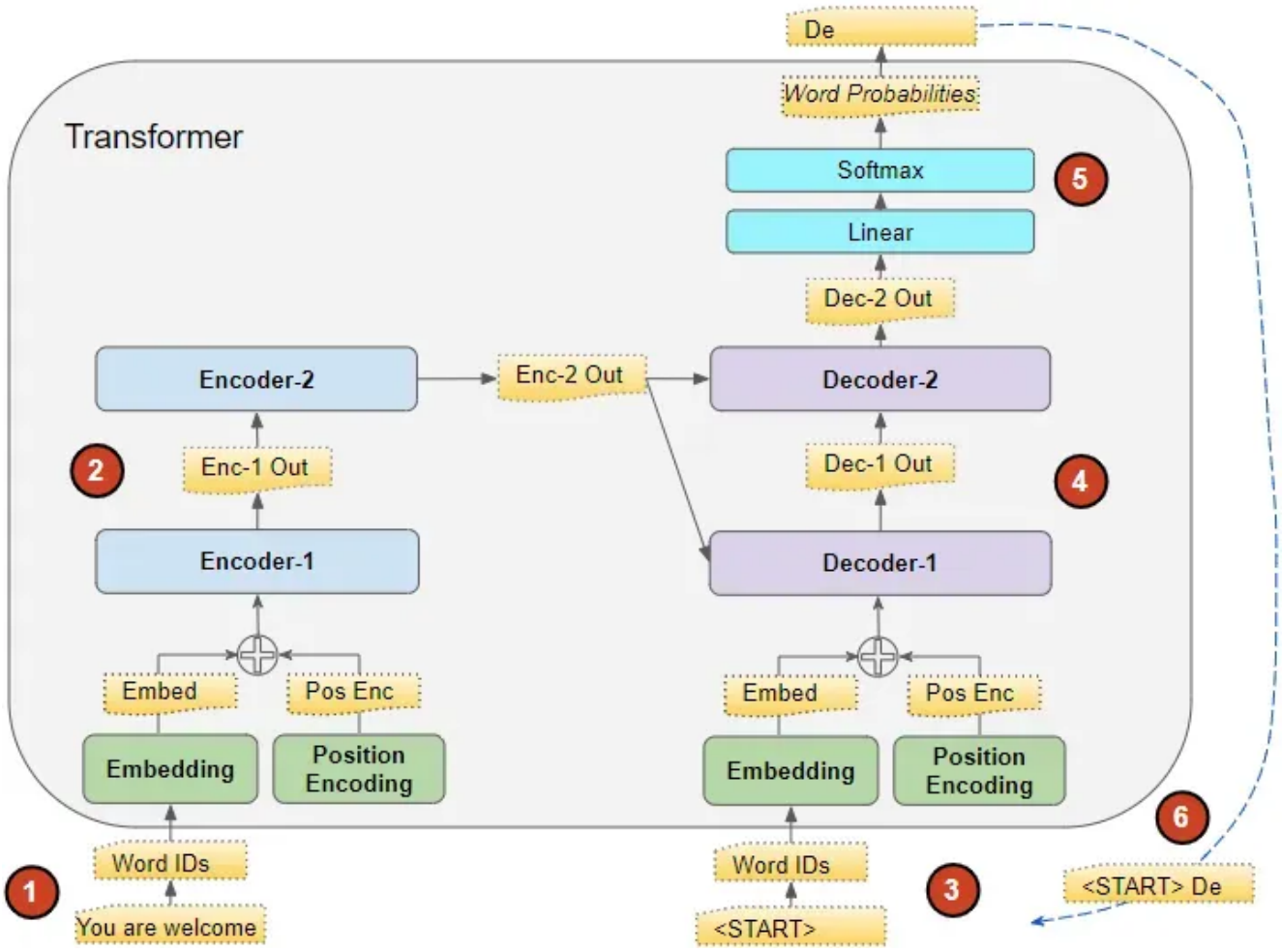
1. 将输入序列转换成嵌入式表示（含位置编码）并送入编码器。
2. 编码器组处理这些数据，生成输入序列的编码表示。
3. 将目标序列加上句首标记，转换成嵌入式表示（含位置编码）并送入解码器。
4. 解码器组在处理这些数据的同时，结合编码器组生成的编码表示来生成目标序列的编码表示。
5. 输出层将其转换成单词概率和最终的输出序列。
6. Transformer 的损失函数将这个输出序列与训练数据中的目标序列进行比较，这个损失用于在反向传播过程中训练 Transformer。

#04
推理

在推理阶段，我们只有输入序列，而没有目标序列作为解码器的输入。Transformer 的目标是仅从输入序列生成目标序列。

因此，就像在序列到序列（Seq2Seq）模型中一样，我们在一个循环中生成输出，并把每个时间步的输出序列整体输入到下一个时间步的解码器中，直至出现句子结束标记。

与 Seq2Seq 模型的不同之处在于，我们每次不是只输入上一步的单词，而是重新输入到目前为止生成的整个序列。



在推理过程中，数据流动如下：

1、将输入序列转换成嵌入式表示（含位置编码）并送入编码器。

2、编码器组处理这些数据，生成输入序列的编码表示。

3、此时不使用目标序列，而是用一个只含有句首标记的空序列。将其转换成嵌入式表示（含位置编码）并送入解码器。

4、解码器组结合编码器组的编码表示处理这些数据，生成目标序列的编码表示。

5、输出层将其转换成单词概率，并生成输出序列。

6、我们选取输出序列的最后一个单词作为预测词。然后，将这个单词填入解码器输入序列的第二个位置，这个序列现包含一个句首标记和第一个单词。

7、返回步骤 3，重复以上操作，每次将新生成的单词添加到解码器序列中，直至预测出句子结束标记。值得注意的是，由于每次迭代编码器序列不变，我们无需重复前两步（感谢 Michal Kučírka 的指出）。

#05
教师强制法

在训练过程中，将目标序列输入解码器的方法称为教师强制法。那么，为什么要采用这种方法，这个术语是什么意思呢？

在训练期间，虽然我们可以采用与推理时相同的方法 —— 即循环运行 Transformer，逐步生成并处理输出序列的每一个单词，但这样做会让训练过程变得更长，且难度增加。因为模型需要根据之前可能预测错误的单词来预测接下来的单词。

相反，通过将目标序列直接输入到解码器，我们实际上是在给模型提供一个线索，就像教师指导学生那样。即便模型在开始时预测错误，它也可以依据正确的目标单词来调整接下来的预测，从而避免错误的累积。

此外，Transformer 可以同时并行地输出所有单词，而无需通过循环逐个处理，这大大加快了训练速度。

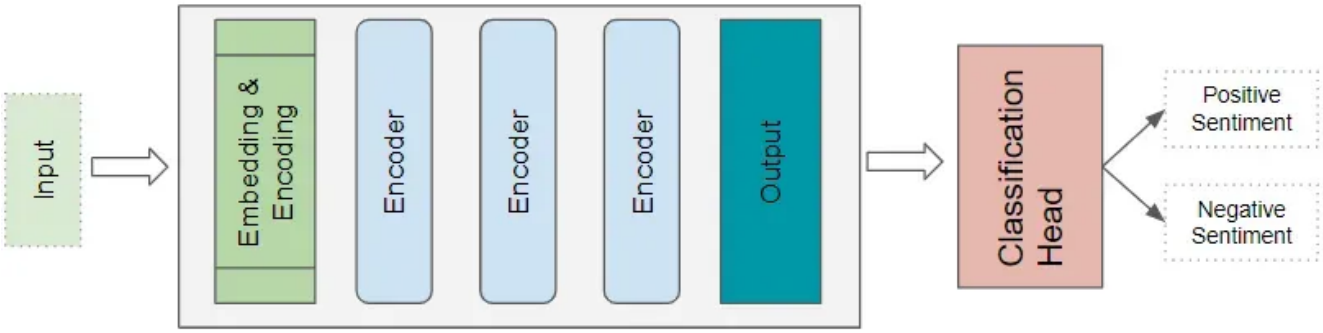
#06
Transformer 有哪些应用？

Transformer 在自然语言处理（NLP）领域中应用广泛，涵盖了语言模型、文本分类等多种任务。它们常用于机器翻译、文本摘要、问答、命名实体识别和语音识别等序列到序列模型的应用场景。

针对不同问题，Transformer 有多种变体。基本的编码器层作为这些架构的共同基石，根据不同应用需求，配备有特定的 “头部” 模块。

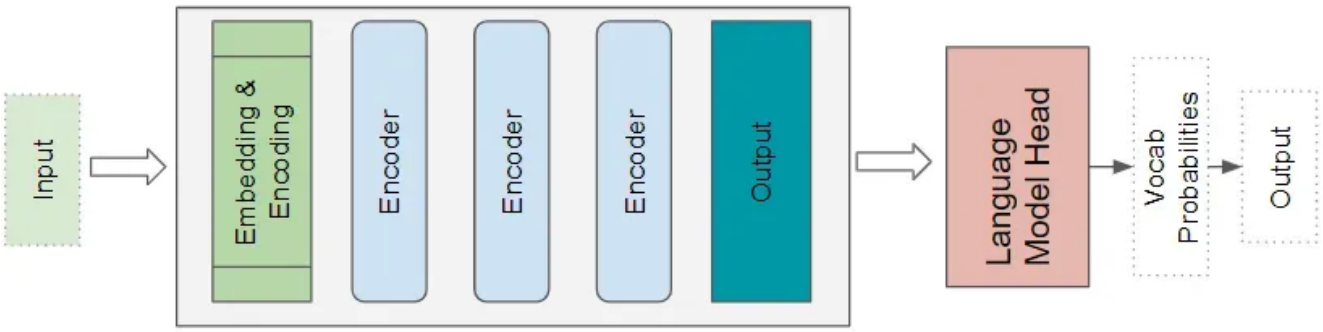
Transformer 分类架构

例如，在情感分析的应用中，Transformer 接收文本文件作为输入。其分类“头部”模块处理 Transformer 的输出，进而预测类别标签，如判断文本表达的是积极还是消极情感。



Transformer 语言模型架构

语言模型架构采用输入序列的起始部分，例如一段文本，作为输入，通过预测接下来的句子来生成新文本。语言模型的头部模块接收 Transformer 的输出，并为词汇表中的每个单词计算一个概率。概率最高的单词被选为下一单词的预测输出。



#07
为什么优于 RNN？

在 Transformer 问世并取代它们之前，RNN 及其衍生版本 LSTM 和 GRU 一直是自然语言处理应用的标准架构。

基于 RNN 的序列到序列模型表现不错，注意力机制最初被引入时，就是为了增强它们的性能。

但是，它们存在两个主要限制：

- 难以处理长句中相隔较远的单词之间的长距离依赖性。

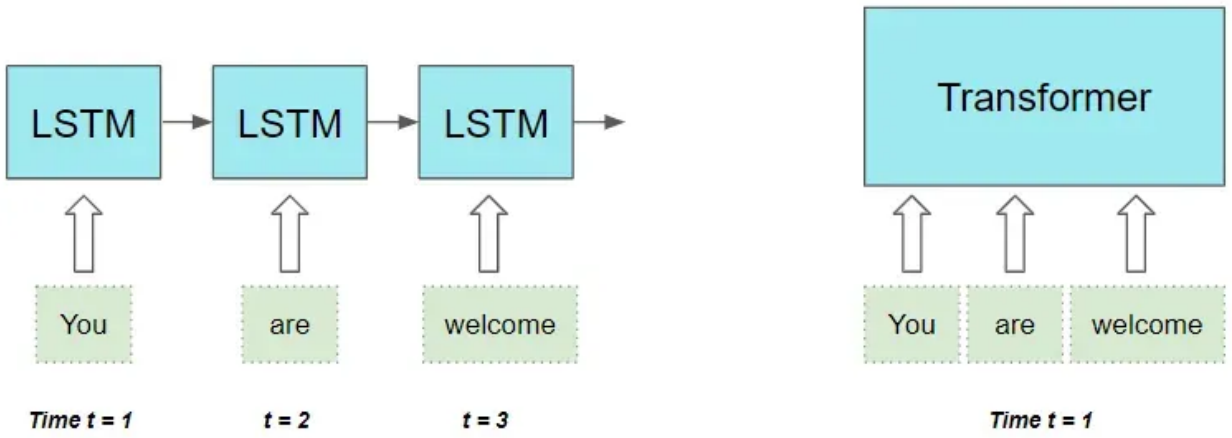
- 它们逐字处理输入序列，无法在完成前一个时间步的计算之前进行下一个时间步的计算。这使得训练和推理速度变慢。

另一方面，尽管 CNN 的所有输出都可以并行计算，从而使得卷积运算速度更快，但它们在处理长距离依赖性方面也存在限制：

- 在卷积层中，只有在内核大小范围内的图像部分（或应用于文本数据时的单词）可以相互作用。对于更远的元素，需要通过多层深度网络才能相互影响。

Transformer 架构解决了这些问题。它完全放弃了 RNN，完全依赖于注意力机制的优势。

- 它们能够并行处理序列中的所有单词，从而极大地提高了计算速度。



- 输入序列中单词之间的距离不再重要。无论是相邻单词还是相隔较远的单词，Transformer 都能很好地计算它们之间的依赖性。

现在我们已经对 Transformer 有了一个总体的了解，下一篇文章将深入探讨它的内部功能，了解它的工作细节。

原文链接：https://towardsdatascience.com/transformers-explained-visually-part-1-overview-of-functionality-95a6dd460452

现在大模型都是基于 Transformer 构建的，要了解大模型必须要了解 Transformer，欢迎进入 AI 大模型实验室微信群一起学习 Transformer。