3大主流语言模型架构

主流的语言模型架构包括纯编码器(Encoder-only)、纯解码器(Decoder-only)以及编码器-解码器(Encoder-Decoder)架构。本文将对这三种架构进行介绍。

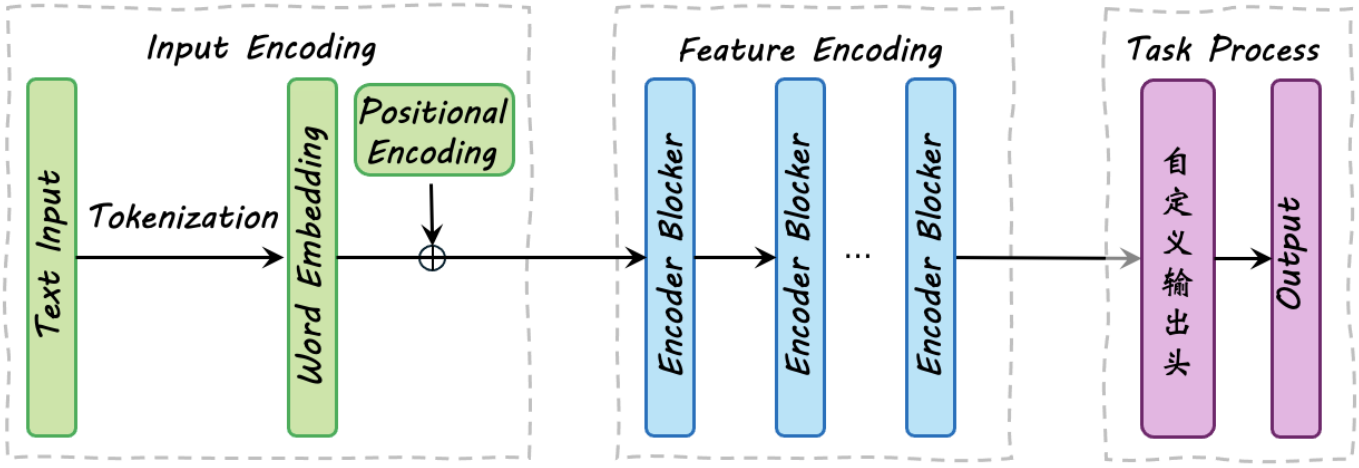
主流语言模型架构类别

Encoder-only 架构

Encoder-only架构仅使用 Transformer 的编码器(Encoder)部分来处理输入文本并生成上下文相关特征。

如下图所示，该架构包含输入编码(Input Encoding)、特征编码(Feature Encoding)和任务处理(Task Processing)三个主要部分。

- **输入编码**
  - 分词(Tokenization)：使用分词器(Tokenizer)将文本转换为token序列。
  - 向量化(Vectorization)：使用词嵌入矩阵(Embedding Matrix)将token转换为向量。
  - 位置编码(Positional Encoding)：为token添加位置信息。
- **特征编码**
  - 由多个编码模块(Encoder Block)堆叠而成,每个模块包含自注意力(Self-Attention)和前馈网络(Feed Forward Network)层。
- **任务处理**
  - 根据具体任务需求灵活设计输出层结构
  - 预训练阶段:
    - 使用全连接层作为输出头
    - 主要完成掩码语言建模(MLM)等自监督预训练任务
  - 下游任务应用:
    - 文本分类: 添加简单的分类器头部即可
    - 序列标注: 为每个token添加标注预测层
    - 文本生成: 需要通过全连接层逐token预测,存在以下问题:
      - 生成速度慢: 需要串行预测每个token
      - 连贯性差: 缺乏解码器的自回归特性
      - 生成质量有限: 无法充分利用上下文信息



Encoder-Decoder 架构

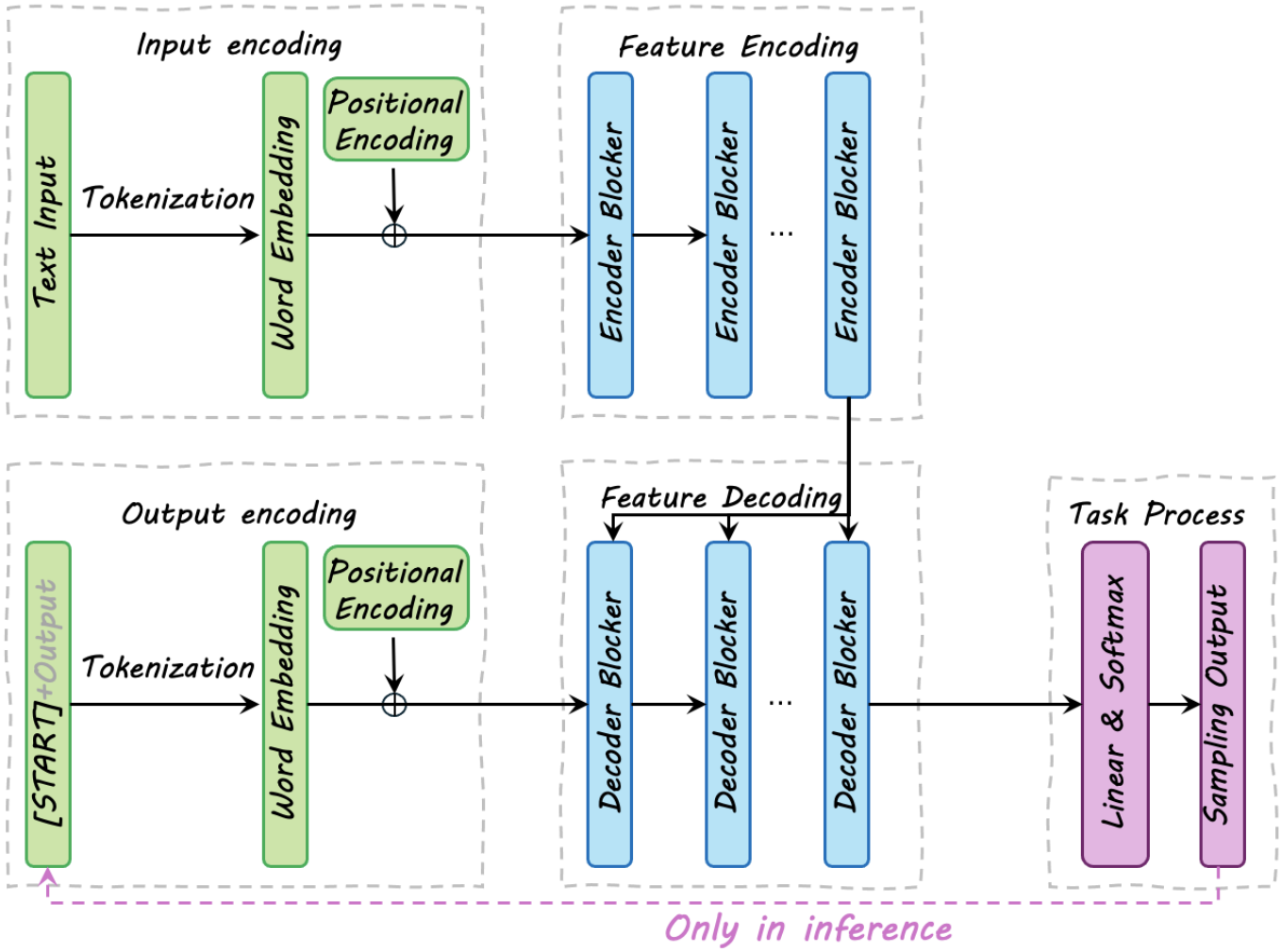
Encoder-Decoder架构通过引入解码器(Decoder)和交叉注意力(Cross-Attention)机制,有效解决了Encoder-only架构在生成任务上的局限。

解码器包含输出编码、特征解码和输出生成三个部分。

- **输出编码**
  - 与编码器的输入编码结构相同。
- **特征解码**
  - 包含掩码自注意力(Masked Self-Attention)、交叉注意力和前馈网络模块,其中掩码自注意力(Masked Self-Attention)确保模型只关注已生成内容,交叉注意力(Cross-Attention)负责编解码器间的信息传递。
- **输出生成**
  - 包含线性层及 Softmax 层
  - 负责将特征解码后的向量通过线性变换映射到词表维度,然后使用Softmax函数得到词表上的概率分布。

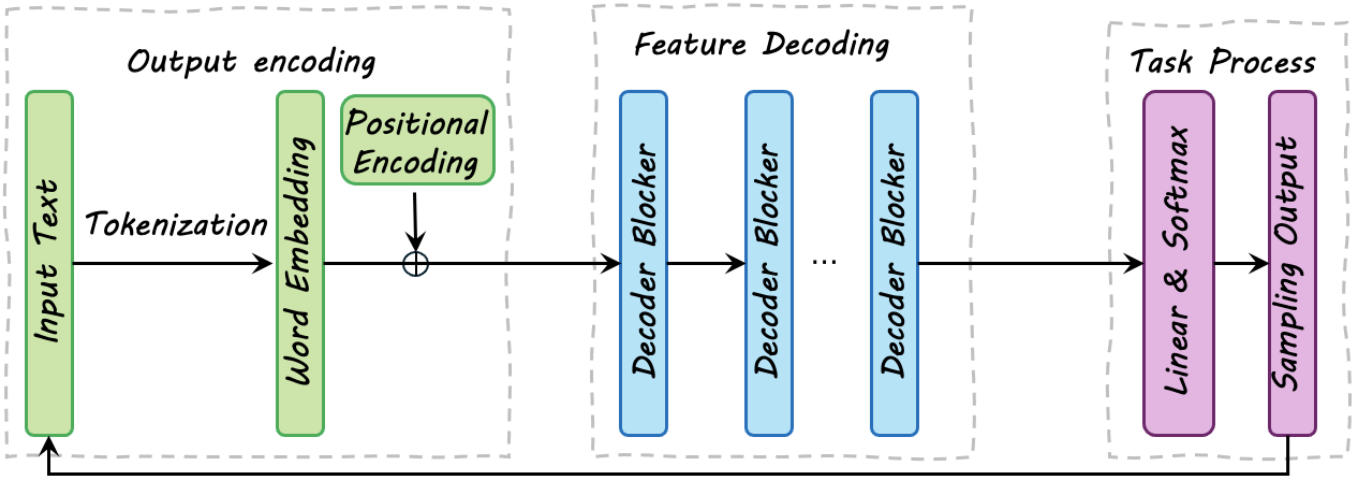
在训练阶段,模型同时处理输入文本和标签文本(Ground Truth)。输入文本经过编码器处理生成上下文表示,而标签文本在添加开始标记`[START]`后进入解码器。模型使用前面文章提到的Teacher Forcing训练方式,利用标签文本的已知部分预测下一个token。

在推理阶段,当出现`[START]`标记时,模型开始通过自回归(Auto-regressive)方式逐步生成文本。每生成一个token都会被用于下一轮预测,直到生成结束标记或达到长度限制。这种串行生成方式虽然确保了文本连贯性,但也限制了生成速度。



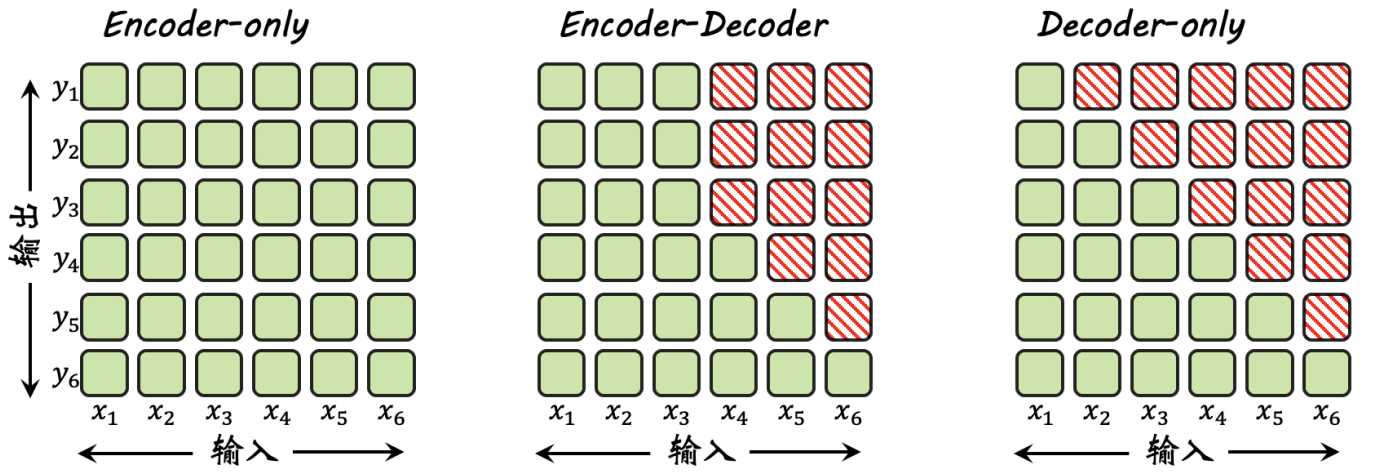
Decoder-only 架构

Decoder-only架构是为了降低计算复杂度,去除了Encoder和交叉注意力模块。该架构仅保留解码器,通过自回归机制实现连贯文本生成。架构包含输入编码、特征解码和输出生成三部分,其特点是省略了交叉注意力子模块。



模型架构的功能对比

这三种架构虽源自 Transformer,但在注意力机制和适用任务上各有特色。注意力矩阵(Attention Matrix)作为核心组件,决定了token间的依赖关系。下图为三种架构的注意力矩阵示意图。



- Encoder-only架构：采用完全注意力,允许每个token关注整个序列，也称为双向注意力机制，这种机制让模型能够深入利用上下文信息，理解复杂的语义。
- Encoder-Decoder架构：结合了完全注意力和下三角注意力,编码器使用双向注意力机制，解码器使用单向注意力机制，确保生成当前token时，只关注已生成的token和当前token，避免泄露未来信息。
- Decoder-only架构：仅使用下三角注意力,确保生成过程的单向性。

任务适用性

Encoder-only架构的双向注意力机制使其能够全面理解序列中的上下文信息,在自然语言理解(NLU)任务上表现优异,如情感分析、文本分类等。但由于缺少专门的解码组件,该架构在生成任务上存在明显局限。

Encoder-Decoder架构通过编码器和解码器的协同工作,既保留了强大的理解能力,又增强了生成能力。解码器可以基于编码器提供的上下文表示,逐步生成高质量的输出序列。这种架构特别适合需要深度理解输入并生成相应输出的任务,如机器翻译、文本摘要和问答系统等。但引入双重组件也带来了更大的计算开销。

Decoder-only架构通过精简设计,仅保留解码器部分,显著降低了计算复杂度。该架构采用掩码注意力和自回归生成方式,能够产生连贯的文本序列。在海量预训练数据的支持下,这类模型在开放式生成任务中表现出色。虽然早期模型(如GPT-1/2)在处理复杂输入时可能不如Encoder-Decoder架构,但随着规模扩大,这一劣势已被显著改善。

如今,以GPT-3/4为代表的大型Decoder-only模型已展现出超越人类的能力。它们不仅在基础的文本生成上表现优异,在记忆、推理和复杂任务执行等高阶认知能力上也取得了突破性进展,逐渐成为大语言模型的主流架构。

总结

大语言模型架构的发展历程清晰展现了技术演进的轨迹。2018年,BERT(Encoder-only)和GPT-1(Decoder-only)几乎同时问世。当时由于模型规模有限,BERT凭借强大的理解能力获得了更多关注。随着机器翻译等生成任务需求增加,Encoder-only架构的局限性逐渐显现。

2019年末,一批Encoder-Decoder架构的模型兴起,凭借出色的序列转换能力成为主流。但随着算力的快速发展,研究者开始追求更大规模的参数量以提升生成能力。Decoder-only架构因其简洁的结构和良好的可扩展性,成为参数扩充的最佳选择。自2021年以来,在GPT-3等模型的推动下,Decoder-only架构主导了大语言模型的发展方向。

尽管如此,Encoder-Decoder架构仍活跃于开源社区,不断获得改进。Encoder-only架构虽然热度有所下降,但在特定的判别任务(如文本分类、情感分析、命名实体识别等)中仍发挥重要作用。

这种架构更迭主要源于生成能力和计算效率的差异。Encoder-only架构虽然理解能力强,但生成效率低下。Encoder-Decoder架构虽然功能全面,但计算开销大。相比之下,Decoder-only架构通过简化设计,在保持强大功能的同时显著提升了效率,特别适合当前对AI生成内容(AIGC)快速增长的需求。

这三种架构的发展极大推动了自然语言处理技术进步,从搜索引擎到智能对话系统,都留下了深远影响。未来,语言模型有望在更优秀的架构设计下继续进化,为更多领域带来革新。