11 综述——掩码语言模型

本章概述

Transformer 的问世无疑是自然语言处理领域的一次划时代突破。它巧妙地利用自注意力机制，破解了传统 RNN 和 CNN 在处理长距离依赖和并行计算中的难题。可以这样形容：Transformer 的架构就像给序列中的每个元素装上了“探照灯”，它们可以无视距离，直接捕捉全局信息。这种“全局扫描”的方式不仅效率惊人，还为后续语言模型的崛起奠定了技术基础。想象一下，曾经我们只能一页一页地翻阅书本，而现在，Transformer 能同时阅读整本书，这种飞跃令人赞叹。以 Transformer 为基础的**因果语言模型**（如 GPT 系列）在生成文本时表现出色，但它们的单向建模方式在理解语义时显得稍显局限。而**掩码语言模型**则选择了一条完全不同的路径：通过“填空游戏”式的任务设计，强迫模型在双向上下文中找到答案。与其说它是在“填空”，不如说它是在推理。掩码语言模型不仅要理解上下文，还要在多个可能性中权衡取舍，这种深度学习语义的方式赋予了它更强的理解能力。

BERT（Bidirectional Encoder Representations from Transformers）就是掩码语言模型的明星选手。它的训练目标分为两部分：**掩码语言建模（MLM）**和**下句预测（NSP）**。在 MLM 任务中，模型需要从上下文中预测被掩盖的词语，就像我们读书时遇到生词时根据上下文猜测它的意思。这种设计不仅让模型学会单词间的关系，还在一定程度上模拟了人类的阅读推理能力。至于 NSP 任务，虽然后来在 RoBERTa 中被移除，但它曾帮助 BERT 更好地理解句子间的逻辑关系。

BERT 的架构本身也非常值得玩味。它采用多层 Transformer **编码器**，每一层都包含多头自注意力机制和前馈神经网络模块。这种层次化设计赋予了它逐步捕捉深层语义信息的能力。 它的 **Contextual Embedding** 技术，即根据上下文动态生成词的语义表示。与传统的静态词嵌入（如 Word2Vec 和 GloVe）不同，BERT 能够为同一个单词在不同语境下生成不同的向量。例如，想到“bank”，是河岸还是银行？BERT 可以通过上下文轻松区分。这种动态调整的能力，几乎像是为模型装上了一副“思维眼镜”。更有意思的是，BERT 的力量并非只存在于预训练阶段，而是在**微调**中真正释放出来。微调过程就像是让一名通才变身为专家。拿**命名实体识别**（NER）任务来说，BERT 的预训练表示已经“知道”了实体的类别和边界，而微调则让它更精准地完成这项特定任务。更令人遐想的是，未来的语言模型是否能够在任务执行中实时调整自身？也许我们距离这样“自适应”的模型已经不远了。

BERT 的成功还激发了许多改进模型的出现。例如，**RoBERTa** 去掉了 NSP 任务，同时增加了训练时间，结果性能大幅提升。这一系列简单调整让我们意识到，原始 BERT 的潜力远未被完全挖掘。而 **ALBERT** 则通过参数共享等技巧实现了模型的轻量化，让性能和资源消耗找到了一种微妙的平衡点。更进一步的，**T5** 和 **BART** 等模型将 BERT 的语义理解能力扩展到生成任务。T5 的统一框架尤其值得一提：它将所有 NLP 任务都转化为“文本到文本”的形式，带来了惊人的灵活性。可以设想，如果语言模型能够同时拥有 BERT 的理解能力和 GPT 的生成能力，那或许将是一种真正的“全能型”语言智能。

从 Transformer 的问世到 BERT 的成功，再到后续模型的不断突破，语言模型的进化不仅是一场技术革新，更是一种人类语言理解和生成能力的延伸与模拟。BERT 的**双向建模**和**动态语义表示**，改变了遗忘对语言的认知。

掩码语言模型

直觉： 掩码语言模型（MLM）是另一种独特的语言建模方法。简单来说，它通过“隐藏”输入文本中的部分单词，迫使模型去猜测这些被隐藏的单词是什么。这种方式的魅力在于，模型需要依赖其前后文的提示来预测这些词汇，从而充分挖掘上下文信息的潜力。可以把MLM想象成一场“填空游戏”。比如，给定句子 “[MASK] is a powerful model,”，随机去掉了“[MASK]”这个词，模型的任务就是猜测它到底应该是什么。通过这种方式，模型不仅要了解句子的局部含义，还要理解整个句子的整体语义，像一个侦探一样拼凑出隐藏的信息。这种设置让MLM能够很好地理解词汇间的复杂关系，突破了仅仅依赖单一方向信息的限制。

特点：

a. 双向性：因果语言模型是“单向”的，也就是说，它们只能根据前面的词来预测接下来的词。而MLM不同，它能够同时考虑单词前后的信息，这让它在理解句子时，能够站在一个更全面的角度，适用于需要全局语义理解的任务。

b. 上下文敏感性：通过掩码的方式，模型需要依据上下文信息来推断出隐藏的单词。这种方法促使模型学习“动态词表示”，即根据上下文来调整词语的含义。想象一下，“bank”在不同的语境下可能是“河岸”也可能是“银行”，MLM能够根据前后文的线索，精准地抓住这些细微的语义差异。

c. 训练效率高：由于每次训练时都可以同时处理多个掩盖的单词，MLM在训练过程中能够高效地从每个输入中提取多个目标。这样一来，模型不仅能快速学习，而且能在相对较短的时间内掌握更多的语言规律。

本质： 掩码语言模型的核心本质，是通过上下文来进行语义建模，理解单词和词语之间的关系。与传统语言模型通常只注重从单向生成文本不同，MLM的重点在于让模型“理解”语义，而不是仅仅“产生”语言。换句话说，MLM在学习如何填补空缺的同时，实际上是训练模型如何从更丰富的上下文中抓取信息，进而获得更强的语义理解能力。这种双向建模方式，不仅让模型能够处理更多复杂的语言任务，也为自然语言处理的发展打开了新的可能性。

数学表达形式

给定一个句子 $S = \{w_1, w_2, \ldots, w_n\}$，其中 w_i 表示第 i 个单词，掩码语言模型会随机选择其中 m 个单词进行掩码，构造一个部分掩盖的句子 $S' = \{w_1, \ldots, [\text{MASK}], \ldots, w_n\}$。模型的目标是根据上下文 S' 预测被掩盖的单词。

假设掩盖的单词集合为 $\mathcal{M} \subset \{1, 2, \ldots, n\}$，对于每个 $i \in \mathcal{M}$，模型需要最大化以下条件概率：

$$P(w_i | S'_{-i}),$$

其中 S'_{-i} 表示从 S' 中移除 w_i 的部分掩盖句子。

整个目标函数可以表示为掩盖单词的对数似然求和：

$$\mathcal{L}_{MLM} = -\sum_{i \in \mathcal{M}} \log P(w_i | S'_{-i}).$$

具体的实现中，使用多层 Transformer 编码器对输入 S' 进行编码，得到每个位置的上下文表示 h_i。通过一个全连接层和 softmax 激活函数将 h_i 映射为词汇表上的概率分布，从而预测被掩盖的单词。

与因果语言模型的对比

掩码语言模型与因果语言模型（如 GPT）在建模目标和方法上存在根本性的不同。这种差异体现在以下几个方面：

属性	掩码语言模型（MLM）	因果语言模型（CLM）
建模方向	双向建模：同时利用单词左侧和右侧的上下文	单向建模：仅利用当前单词左侧的上下文，按顺序生成
目标任务	语义理解：预测被掩盖单词	文本生成：预测下一个单词
训练数据结构	输入句子中部分单词被掩盖	输入句子作为完整序列，预测序列的每个词
典型模型	BERT、RoBERTa、ALBERT	GPT 系列（GPT、GPT-2、GPT-3）
优点	1. 能够捕获全局上下文语义 2. 适合语义理解任务（如分类、消歧）	1. 简单高效的生成能力 2. 适合文本生成任务
缺点	1. 难以直接用于生成任务 2. 掩码方式引入了非自然的输入数据	1. 无法捕获右侧上下文 2. 长序列依赖生成效率较低

BERT

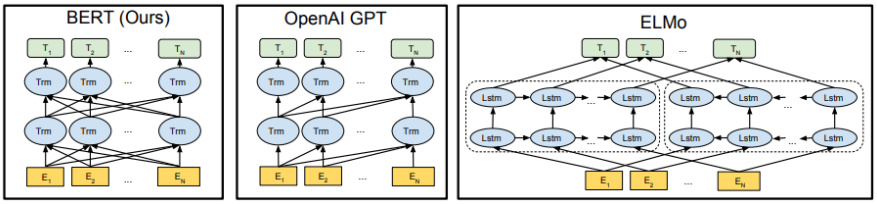
BERT（Bidirectional Encoder Representations from Transformers）是 Google AI 语言团队于 2018 年推出的一款突破性的语言模型。它的核心创新在于采用了双向编码器——这意味着在训练时，模型不仅可以看到一个词前面的上下文，还能看到它后面的部分。这一设计赋予了 BERT 前所未有的语义理解能力，使其在自然语言处理（NLP）的各项任务中大放异彩。与传统的自回归模型（例如 GPT）不同，BERT 并不依赖于逐词生成，而是采用了掩码语言建模（Masked Language Modeling, MLM）的预训练方式。这种方法让模型在训练过程中能够“猜测”被隐藏的词汇，从而根据上下文动态地调整每个词的表示。这种灵活性使得 BERT 能够根据实际语境生成词的向量，而不是仅仅依赖固定的词嵌入。

除了双向编码器的创新，BERT 的另一个亮点就是其卓越的迁移学习能力。通过微调（Fine-tuning），BERT 可以迅速适应各种具体任务，而无需从头开始训练一个全新的模型。这一点，使得 BERT 成为在实际应用中非常高效的工具。

自发布以来，BERT 的影响力巨大，甚至催生了多个变体模型，如 RoBERTa、ALBERT 和 DistilBERT 等，这些模型在 BERT 的基础上进行了一些优化，进一步提升了在不同任务中的表现。

模型架构

BERT 的模型架构是基于 Transformer 编码器的，采用了多个 Transformer 层的堆叠，形成一个深层次的结构。这种结构的核心在于通过 **多头自注意力机制** 和 **前馈神经网络** 来捕捉输入文本中的复杂语义。



首先，BERT 的输入并不仅仅是简单的单词，它会将每个单词映射为一个向量，并且为每个单词加上位置信息和句子类型信息。BERT 通过以下几种方式来处理输入：

a. 词嵌入（Token Embeddings）：这部分将输入的每个词（或者子词）转换成一个向量，实际上是将词汇表中的每个词与一个数字向量一一对应。

b. 位置编码（Position Embeddings）：由于 Transformer 并不能像人类一样直觉地理解词汇的顺序，BERT 就为每个词加入了它在句子中的位置编码，帮助模型理解“顺序”。

c. 句子类型编码（Segment Embeddings）：BERT 支持处理句子对（例如在问答任务中），因此它为每个词加上了一个标记，表示它是第一个句子的一部分，还是第二个句子的一部分。

这三部分信息结合在一起，最终形成了 **输入向量**。简单来说，就是：

$$\text{Input Embedding} = \text{Token Embedding} + \text{Position Embedding} + \text{Segment Embedding}$$

在每一层 Transformer 中，自注意力机制能够让模型在处理当前单词时，参考句子中所有其他单词的上下文信息。这意味着，每个单词的意义都不是孤立的，而是与整个句子的其他部分相互关联。这就像是在听一个故事时，故事的每个部分都互相呼应，帮助我们更好地理解整体情节。每层 Transformer 中还包含一个 **前馈神经网络**，它通常由两个线性变换和一个激活函数组成。它的作用是将每个单词的表示进一步映射到更高维度，为后续的计算提供更加丰富的信息。BERT 并不是由一层 Transformer 就构成的，而是由多个 Transformer 编码器堆叠而成。BERT-base 模型有 12 层编码器，而 BERT-large 模型则有 24 层编码器。每一层都可以看作是一个更高阶的理解器，它会为每个输入的单词生成一个新的上下文表示，从而让模型能够捕捉到更深层次的语言特征。

最终，BERT 的输出是每个输入单词的上下文表示。对于大多数下游任务，通常使用 [CLS] 标记的输出作为最终的表示（例如，在分类任务中），而其他位置的输出则表示句子中相应词汇的上下文：

$$X_L = \text{Transformer}(X_{L-1}), \quad L \in \{1, 2, \ldots, L\}$$

最终的输出是每个单词的上下文相关表示 X_L，用于进行后续任务。

训练任务

BERT 的预训练任务包括两项主要任务：掩码语言建模（Masked Language Modeling, MLM） 和下一句预测（Next Sentence Prediction, NSP）。

a. 掩码语言建模（MLM）： 在预训练阶段，BERT 随机选择输入文本中的 15% 的词汇进行掩盖，并要求模型预测被掩盖的词。掩盖的词被替换为特殊的 [MASK] 标记。掩码语言建模任务是 BERT 中的核心任务之一，它让模型能够从上下文中推测被掩盖的词，并捕获更多的上下文信息。对于句子 $S = [w_1, w_2, \ldots, w_n]$ ，模型需要最大化被掩盖词的预测概率：

$$\mathcal{L}_{MLM} = -\sum_{i \in \mathcal{M}} \log P(w_i | S'_{-i}),$$

其中， $\mathcal{M}$ 是被掩盖的位置集合， S'_{-i} 是掩盖词后的部分输入。

b. 下一句预测（NSP）： NSP 任务的目标是让模型判断两个句子是否是连续的。模型通过判断输入的句子对（A, B）是否相邻，来增强其对句间关系的理解。NSP 的目标是最大化以下概率：

$$P(\text{IsNext}(A, B)) = \frac{\exp(\text{score}(A, B))}{\exp(\text{score}(A, B)) + \exp(\text{score}(A, C))}$$

其中， A 和 B 是句子对，而 C 是与 A 不相邻的句子。模型需要学习区分句子对是否相邻。

BERT微调

BERT凭借着在大规模语料上的训练，它能捕捉到丰富的上下文信息和语言特征。在BERT的预训练阶段，它通过“掩码语言建模”（MLM）和“下一句预测”（NSP）任务来理解句子的深层语义。然而，BERT作为一个通用模型，在面对特定任务时，仍然需要进一步的优化和调整——这就是微调（Fine-tuning）的重要性所在。

什么是微调?

微调是指在BERT的预训练基础上，通过在特定任务的数据集上进行进一步的训练，从而调整模型的参数，使其在特定任务中表现得更好。与从零开始训练一个全新的模型相比，微调利用了BERT在大量语料上所学到的语言知识，从而加速了学习过程，并显著提高了模型在具体任务上的表现。

微调的步骤

a. **任务头**：首先，根据具体任务（例如文本分类、命名实体识别等），我们需要在BERT模型的顶部添加一个任务头。在分类任务中，通常会加一个简单的全连接层，而在序列标注任务中（如NER），则需要添加标注层。

b. **训练目标**：训练的目标是通过任务的数据，使用相应的损失函数来优化模型的预测。比如，命名实体识别（NER）任务通常使用交叉熵损失来减少预测标签与真实标签之间的误差。

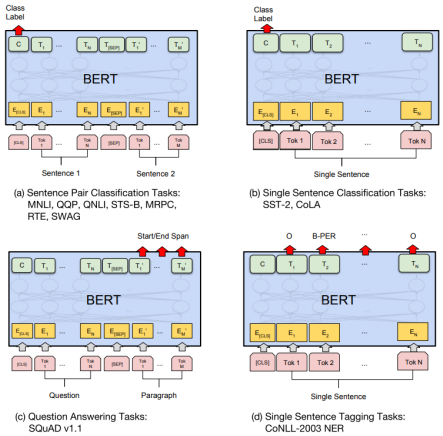
c. **端到端训练**：在微调的过程中，BERT的所有层都会参与训练。也就是说，不仅仅是任务头部分，整个BERT模型都会根据下游任务的需求进行调整，从而最大化任务的性能。

微调的优势

a. **迁移学习的威力**：BERT在预训练阶段学到的语言知识，能够迅速迁移到不同的NLP任务中。这样，我们就能避免从零开始训练的巨大开销。

b. **少量数据的良好表现**：即便任务的标注数据并不多，借助微调，我们仍然能使BERT表现得相当优秀，提升模型的效果。

c. **高度灵活**：BERT能够适应各种不同的NLP任务，包括文本分类、情感分析、问答、命名实体识别等，具有极强的通用性和灵活性。



本章问题

在大语言模型时代到来之前，NLP领域还是BERT家族的天下，他们的表现往往比GPT系列的表现更好。如今情况却似乎反转。试问在LLM时代，Encoder-only模型是否还有地方发挥出更优于LLM的表现？