本页PDF

Bert

架构

基于EMLO与GPT改进，融合了这两个模型的特征：

- EMLO使用的是**双向的RNN**
- GPT使用的是**transformer**

而Bert使用的是**双向transformer**

两步工作

1. 无监督学习的预训练
    - 使用大量无标记的文本进行预训练，也就是文字填空和NSP，使得模型学习到词元的特征和句子之间的关系
2. 有监督学习的微调
    - 使用有标记的文本进行微调，目的是为了让模型适应下游任务，**原始论文采用全参微调**
    - 微调会保留Bert的预训练权重，但在任务数据上继续训练，并加上**适应该任务的头部层**

无标号的大数据的训练比**有标号的一定大小的数据**的训练有用

解决的任务

1. 做文字填空（MLM）
    - 会随机mask掉一些词，然后根据上下文进行词的填空
    - 有**15%的概率选中一些词，又有80%的概率把这些词mask掉**，还有10%的概率换成词表原有的词，剩下10%的概率什么都不做
    - 这种做法会导致bert在进行微调的时候**有性能损失，因为微调的时候并不会去进行掩码**
2. 做下一个句子的预测（**NSP**）
    - 随机抽取两个句子，判断其中一个句子是不是另一个句子的下一句
    - 有**50%的正样本，50%的负样本**

输入向量

输入 = **Token Embedding** + **Segment Embedding** + **Position Embedding**

- **Token Embedding：**WordPiece分词（处理未登录词如"unhappiness"→"un", "##happiness"）。
- **Segment Embedding：**标记句子归属（第一句为0，第二句为1）。
- **Position Embedding：**绝对位置编码（最长支持512个Token）。

文本标记

- `[CLS]`：这是一个向量，它会与句中所有其他词交互，聚合全局语义信息，它的**最终隐藏层状态会作为分类器的输入特征**，可以用于分类任务
- `[SEP]`：这个标记用于明确句子边界

WordPiece

基于子词的一种分词算法

核心目标：把单词拆分为**更小的、有语义的子词单元**

- **减小词表大小：**避免为所有单词单独分配向量
- **提升泛化能力：**通过子词组合处理未见过的词

合并策略是基于**语言模型的似然概率，优化目标是最大化语料库的似然概率**

前缀标记：非词首子词用`##`标记（如`##ing`），区分词内和词首位置

训练阶段

1. 将语料库中所有单词拆分为**单字符，统计字符概率**
2. 计算候选子词对的得分，使用**似然增益公式**，将**得分最高的子词对加入词汇表**，直到词汇表达到预设大小

$$ score(s) = \log P(s)- (\log P(s_1) + \log P(s_2)) $$

其中：

- $P(s)$：合并后的子词s在语言模型中的概率
- $P(s_1)$和$P(s_2)$：未合并时**两个子词独立出现的概率**