0. Transformer概述

1. 概述与综述

1.1. 研究背景

- 上下文无关表示：上下文敏感

模式	含义	特点/例子
上下文无关表示	固定 $e^{token(k)}$	同词同向量，不区分语义
上下文敏感表示	固定 $e^{token(k)}$ + $\sum_{j=1}^{L} \alpha_j h_j^{(j)}$	同词同向量 $f(s, e^{token(k)})$	Tag:LM, CoVe, ELMo

- 语言模型的预训练
  - 含义：在大量未标注的文本数据上训练模型，以获得通用的、深度的上下文表示的理解
  - 方式：常见的预训练任务

方式	含义	示例
自回归语言模型	从左到右或从右到左，预测下一个词的概率	GPT
完形填空（Cloze）	通过遮蔽部分输入词（token）来以模型学习被遮蔽的内容	BERT

- 预训练模型如何用于下游任务
  - 基于特征的方法：将固定输入

  - 原理：预训练时对BERT所有层或部分层进行上下文表示一由下到上的加权组合（Weighted Combination），然后将这个下游任务定制的“上游模型结构”
  - 特点：冻结BERT主干参数，只训练下游任务的参数，计算成本低，灵活性高
  - 例子：ELMo中将不同层的隐状态进行加权组合
- 微调（Fine-tuning）
  - 含义：将预训练模型的参数作为下游任务模型的初始参数，在下游任务上继续进行训练
  - 特点：为每个下游任务创建独立的模型副本，以获得最佳性能
  - 例子：BERT在GLUE基准测试中对整个模型进行端到端微调

1.2. BERT的主要贡献

- 引入掩码语言模型
  - 原理：预训练过程中，随机计算被打乱的词（token），并在预测这些被掩盖的原词时进行
  - 作用：使模型同时融合左侧和右侧上下文一种双向表示能力，这是单向模型（如GPT）无法实现的
- 引入下一句预测（NSP）
  - 原理：通过预测一个句子是否紧跟在另一个句子后面，以学习句子级别的关系
  - 作用：为问答和自然语言推理等任务提供有益的先验
- 证明了大规模预训练的有效性
  - 原理：在海量无标注数据上进行预训练，可以显著提高模型在下游任务上的表现
  - 结果：BERT在大量任务（涵盖不同任务类型、语言环境）上取得了当时最先进的结果

2. BERT原理：预训练&微调

2.1. BERT的结构

- 模型的两个阶段

阶段	主要任务
预训练	在无标注语料上进行掩码语言模型（MLM）和下一句预测（NSP）任务的预训练
微调	根据具体下游任务（如问答、情感分析）调整预训练模型的参数，以适应特定任务

- 模型的参数规模

模型	参数量	特点
BERT-Base	约1.1亿	平衡性能与计算资源，适合大多数应用场景
BERT-Large	约3.4亿	性能更强，但计算资源需求高，适合资源充足的场景

- 输入的构建：三种嵌入的加和

- 基本概念

	说明	含义
词元(Token)	BERT处理的最小单元，为WordPiece分词器切分的子词（subword）	
句子(Sentence)	任意长度的连续文本片段，用于一段要处理的文本	
序列(Sequence)	输入BERT的不同时刻（time-step），可以是一个，多个句子	

- 序列的表示
  - 单句子：以为CLS开头
  - 多句子：以为CLS开头，以[SEP]作为句子间的分隔符（separator）
- 输入的构建：三种嵌入的加和

	维度	含义
Token Embeddings	d_{model}	词元的语义表示
Segment Embeddings	d_{model}	区分不同句子的信息
Position Embeddings	d_{model}	编码词元的位置信息

2.2. BERT的预训练：两种关键任务

- 第一种任务：Masked语言模型
  - 原理与Word2Vec的CBOW模型类似，用周围词的上下文来预测被遮盖的“目标”词本身
  - 作用：强迫模型学习双向表示，以获得一个通用的、高质量的词表示（Token）
- 掩码策略：选择15%的Token进行预测，其中
  - 80%：直接用[MASK]替换原Token
  - 10%：从词汇表中随机一个Token替换原Token
  - 10%：不做任何改变
- 这种策略为了防止预训练-微调（Pre-training - Fine-tuning）阶段的模型输入分布不一致的问题（Mismatch）
- 预测目标：不计算损失的计算被屏蔽掉的Token，仅计算被掩盖Token的交叉熵损失
- 第二种任务：预测下一个句子NSP，目的在于上游任务中理解句子间的关系
  - 引入NSP的原因：
    - 对段落级别的语义理解，如QA（问答）和NLI（自然语言推理）
    - 单句语言模型（Left-to-right or right-to-left）无法捕捉句子间的关系
  - NSP的实现
    - 数据构建

	样本来源	标签含义	比例
IsNext	从原始语料中连续两个句子中选取	下一句	50%
NotNext	从原始语料中随机选取一个句子	随机选取一个句子	50%

- 输入构建：sentence-level Transformer
- 预测输出：在CLS位置的隐状态（token）上接一个二分类器，将CLS的隐状态映射为二维向量，用于区分IsNext和NotNext
- 预训练的目标函数：最小化Masked LM与NSP两个任务中的交叉熵损失

2.3. BERT微调的概述

- 微调是为预训练的模型

方式	描述
传统方式	先冻结预训练模型的大部分参数，只训练顶部的少量参数
现代方式	通常采用全参数微调，即对模型所有参数进行更新，**以获得更优的性能**

- 微调的原理
  - 输入构建
    - 输入格式：sentence-level Transformer
    - 输出表示：第一个特殊标记（CLS）的隐状态被用作整个句子的表示
  - 输出结构：
    - 单句任务：使用单个分类器，其输出维度为 $|C| \times d^{model}$，$|C|$ 为类别数量，d^{model} 为隐状态维度
    - 句对任务：将 $u, v, |u-v|$ 三个向量拼接后输入分类器
  - 微调的目标：在Task-dline-32，微调的loss=3，学习率=1/3/2e-5情况下
    - BERT_base | BERT_Large 在平均表现上，BERT_Large在各种规模数据上的平均BERT_base
    - BERT在GLUE上的提升，并通过加深模型进一步提升效果
- 微调的优势或数据效率对比
  - 与非NLU任务
    - 原理：在预训练**不修改任务特定结构**的前提下，在自定义数据上100%可开展
    - 结果：使用句子A、句子B句子对，四元组，(w_1, w_2, w_3, w_4)，在随机或自定义句子对的70-80%的情况
  - 微调的特性
    - 输入构造：两段令牌序列，即[CLS] + 句子A令牌 + [SEP] + 句子B令牌 + [SEP]
    - 任务适配：仅为一个Task-specific的输出层，即NLU分类任务的分类器
    - 参数共享：在整个Pre-training和下游任务中，作为跨任务的通用特征提取器
  - 微调的优势：在Hyper-Search=20个Trials中，BERT_Base与BERT_Large的平均表现
  - 与NLU任务
    - 原理：在预训练**不修改任务特定结构**的前提下，在自定义数据上100%可开展
    - 结果：使用句子A，句子B句子对，四元组，(w_1, w_2, w_3, w_4)，在随机或自定义句子对的70-80%的情况
  - 微调的优势：在Hyper-Search=20个Trials中，BERT_Base与BERT_Large的平均表现

2.4. Token级任务

- 基于特征的微调方法（NER v1.1）
  - 关于NLU的v1.1：
    - 新内容：包括对NER的特别关注和数据集更新
    - 任务目标：给定句子，提取出其中具有特定语义类别的实体（如人名、地名、组织名等）
  - 微调流程：

  - 输入构造：sentence-level Transformer，[CLS]句子A[SEP]句子B[SEP]
  - 非特殊令牌：**忽略**，理论上应该用于$E_{[X]}$，但为简化实现直接复制$E_{[CLS]}$的隐状态
  - 输出：
    - 单句任务：使用单个分类器，其输出维度为 $|C| \times d^{model}$，$|C|$ 为类别数量，d^{model} 为隐状态维度
    - 句对任务：将 $u, v, |u-v|$ 三个向量拼接后输入分类器
  - 微调的优势：在Hyper-Search=20个Trials中，BERT_Base与BERT_Large的平均表现
- 关于NLU的v1.1/v2.0
  - 新内容：包括NER v1.1/SLU v1.1在内的新增任务，以及GLUE官方测试集的更新
  - 目标：为这些任务提供一个统一的、强大的基线模型
- 关于微调的优势
  - 计算资源：无需重新训练整个模型
  - 灵活性：

情况	描述	例子
特定集	一般适用情况下的特定集	$E_{[X]}$ or $E_{[X]}$
不适用	Otherwise	$E_{[CLS]}$ or $E_{[CLS]}$

  - 预训练的一致性：一致性：$E_{[CLS]}$ or $E_{[CLS]}$，因为CLS的隐状态用于句子级别的预训练任务（如NSP）
- 关于微调的优势：在Hyper-Search=20个Trials中，BERT_Base与BERT_Large的平均表现
  - 微调流程：在Hyper-Search=20个Trials中，BERT_Base与BERT_Large的平均表现
  - 结果：在GLUE基准测试中，BERT_Base与BERT_Large的平均表现
- 关于NLU的v2.0
  - 新内容：包括NLU v1.1/SLU v1.1在内的新增任务，以及GLUE官方测试集的更新
  - 目标：为这些任务提供一个统一的、强大的基线模型
- 关于微调的优势
  - 计算资源：无需重新训练整个模型
  - 灵活性：

情况	描述	例子
特定集	一般适用情况下的特定集	$E_{[X]}$ or $E_{[X]}$
不适用	Otherwise	$E_{[CLS]}$ or $E_{[CLS]}$

  - 预训练的一致性：一致性：$E_{[CLS]}$ or $E_{[CLS]}$，因为CLS的隐状态用于句子级别的预训练任务（如NSP）
- 关于微调的优势：在Hyper-Search=20个Trials中，BERT_Base与BERT_Large的平均表现
  - 微调流程：在Hyper-Search=20个Trials中，BERT_Base与BERT_Large的平均表现
  - 结果：在GLUE基准测试中，BERT_Base与BERT_Large的平均表现
- 关于微调的优势：在Hyper-Search=20个Trials中，BERT_Base与BERT_Large的平均表现
  - 微调流程：在Hyper-Search=20个Trials中，BERT_Base与BERT_Large的平均表现
  - 结果：在GLUE基准测试中，BERT_Base与BERT_Large的平均表现

4. BERT实验：微调&消融

4.1. BERT在不同任务上的微调

4.1.1 序列级任务

- 基础流程图解析

- 关于GLUE（包含多项NLU任务的综合，部分数据集除外）

数据集	描述	任务类型	指标
MNLI	判断句子对语义蕴涵/矛盾/中性	三分类	GLUE中最大规模最常使用
QQP	判断两条问题是否语义等价	二分类	N/A
SST-2	判断句子情感倾向（正面/负面）	二分类	N/A
CoLA	判断句子是否语法正确/可接受	二分类	N/A
STS-B	预测句子的语义相似度（1-5分）	回归	N/A
.....	.....	.....	.....

- 微调的实验

模型	动机与目标	是否使用了一句预测
No-NSP	Masked LM(MLM)	✗
LTR & No-NSP	从左到右	✗
BERT	Masked LM(MLM)	✓

- 实验结果

机制	影响	结论
NSP	在推理/问答任务上显著提升	NSP可有效帮助模型上下文之间的理解
Masked LM	在所有任务上性能提升	掩码预测任务可以有效学习

- 预训练的消融研究

预训练	实验结果	结论
任务微调	BERT_base在所有BERT_base及其它小模型Transformer	模型参数量是性能提升的关键
小模型	蒸馏模型（知识蒸馏）在数据量较小的情况下表现优秀	可用于部署小任务

- 基于特征的方法与BERT
  - 两种模型

模型	对预训练BERT的影响	适用场景/优势
微调(BERT)	最终加一个分类层	适合微调所有参数
特征(BERT)	提取第12个文本Token特征	输入特征提取到BiLSTM的NER，在输入层提

- 实验结果：NER

  - 数据集：CoNLL-2003 NER
  - 任务：为每个Token分类一个命名实体标签（不使用CRF）
- 实验结果
  - 微调方法：性能最佳
  - 特征方法：接近微调方法的效果，也说明了BERT表征的有效性与实用价值