BERT:深度双向变压器的预训练语言理解

Jacob Devlin 张明伟 Kenton Lee Kristina Toutanova
谷歌人工智能语言
{jacobdevlin,mingweichang,kentonl,kristout}@google.com

抽象的

我们引入了一种新的语言表示模型,称为BERT,它代表

双向编码器表示来自
变形金刚。与最近的语言表示模型（Peters et al., 2018a;
Rad ford et al., 2018)不同， BERT 旨在预训练来自

通过联合调节未标记的文本
所有层的左右上下文。因此,预训练的 BERT 模型可以通过一个额外的输出层进行微调

为广泛的领域创建最先进的模型
任务范围,例如问答和
语言推理,无需大量特定于任务的架构修改。

BERT 在概念上简单且经验丰富
强大的。它在 11 个自然语言处理上获得了新的最先进的结果

任务,包括将 GLUE 分数推送到
80.5%（7.7%点绝对提升），
MultiNLI 准确度高达 86.7%（绝对值 4.6%
改进）,SQuAD v1.1 问答测试 F1 到 93.2（1.5 分绝对改进)和 SQuAD v2.0 测试 F1 到 83.1

（5.1分绝对提升）。

1 简介

语言模型预训练已被证明
对提高许多自然语言有效
处理任务（Dai 和 Le， 2015； Peters 等人，
2018a;拉德福德等人， 2018;霍华德和鲁德，
2018）。这些包括句子级别的任务,例如
自然语言推理（Bowman 等人， 2015 年；
Williams 等人， 2018 年)和释义（Dolan
和 Brockett， 2005）,旨在通过分析句子来预测句子之间的关系
整体上,以及令牌级别的任务,例如
命名实体识别和问答，
模型需要产生细粒度的
代币级别的输出（Tjong Kim Sang 和
德默德， 2003； Rajpurkar 等人， 2016 年）。

将预训练的语言表示应用于下游任务有两种现有策略:基于特征和微调。这

基于特征的方法,例如 ELMo (Peters
et al., 2018a),使用特定于任务的架构，
包括预训练的表示作为附加特征。微调方法,例如

生成式预训练变压器（OpenAI
GPT) (Radford et al., 2018),引入了最小
特定于任务的参数,并在
通过简单地微调所有预先训练的参数来完成下游任务。这两种方法共享
预训练期间的相同目标函数,其中
他们使用单向语言模型来学习
通用语言表示。

我们认为目前的技术限制了
预训练表示的力量,尤其是微调方法。主要限制是标准语言模型是

单向的,这限制了可以在预训练期间使用的架构的选择。为了

例如,在 OpenAI GPT 中,作者使用从左到右的架构,其中每个标记只能倾向于自我注意层中的先前标记

变压器（Vaswani 等人， 2017 年）。这种限制对于句子级任务来说是次优的，
并且在将基于微调的方法应用于令牌级任务时可能非常有害,例如
作为问答,从两个方向结合上下文至关重要。

在本文中,我们改进了基于微调的
通过提出 BERT:双向
来自Transformers 的编码器表示。
受完形填空任务(Taylor, 1953)的启发,BERT 通过使用“掩码语言模型”（MLM)预训练目标来缓解前面提到的单向性约束。这

屏蔽语言模型随机屏蔽一些
来自输入的标记,目标是
预测被掩码的原始词汇 id

单词仅基于其上下文。与从左到右的语言模型预训练不同,MLM 目标使表示能够融合左

以及正确的上下文,这使我们能够预训练一个深度双向 Transformer。除了掩码语言模型,我们还使用

联合预训练文本对表示的“下一句预测”任务。贡献

我们的论文如下:

- 我们展示了双向的重要性
  语言表示的预训练。不像Radford 等人。 (2018),使用单向语言模型进行预训练,BERT

  使用掩码语言模型来启用预先训练的深度双向表示。这

  也与彼得斯等人相反。 （2018a）,其中
  使用独立的浅连接
  训练了从左到右和从右到左的 LM。
- 我们表明预训练的表示减少了
  需要许多精心设计的任务特定架构。 BERT 是第一个基于微调的表示模型,它实现了

  大型套房的一流性能
  句子级和令牌级任务,优于许多特定于任务的架构。

- BERT 将最先进技术提升了 11 个
  NLP 任务。代码和预训练模型可在https://github.com/获得

  谷歌研究/伯特。

2 相关工作

预训练通用语言表示有很长的历史,我们简要回顾一下

本节中使用最广泛的方法。

2.1 无监督的基于特征的方法

学习广泛适用的表示
词一直是一个活跃的研究领域
几十年,包括非神经（Brown et al., 1992;
安藤和张， 2005； Blitzer 等人， 2006)和
神经（Mikolov 等人， 2013； Pennington 等人，
2014)方法。预训练的词嵌入
是现代 NLP 系统不可或缺的一部分,与嵌入相比有显着改进

从头开始学习（Turian 等人， 2010 年）。为了预训练词嵌入向量,使用了从左到右的语言建模目标（Mnih

和 Hinton， 2009 年）,以及区分正确和不正确单词的目标

正确的上下文（Mikolov 等人， 2013）。

这些方法已推广到
更粗的粒度,例如句子嵌入（Kiros 等人， 2015； Logeswaran 和 Lee，
2018)或段落嵌入（Le 和 Mikolov，
2014）。为了训练句子表示,先验
工作已使用目标对候选人进行排名
句子（Jernite 等人， 2017； Logeswaran 和
Lee， 2018）,从左到右生成下一个句子单词,给出前一个句子的表示
句子(Kiros et al., 2015),或去噪自动编码器派生目标(Hill et al., 2016)。

ELMo 及其前身(Peters et al., 2017,
2018a)沿不同维度推广传统词嵌入研究。他们提取

从左到右的上下文相关特征和
从右到左的语言模型。每个标记的上下文表示是

从左到右和从右到左的表示。
集成上下文词嵌入时
使用现有的特定于任务的架构,ELMo
推进了几个主要 NLP 的最新技术
基准(Peters et al., 2018a) ,包括问题回答(Rajpurkar et al., 2016)、情绪
分析（Socher 等人， 2013 年)和命名实体
认可（Tjong Kim Sang 和 De Meulder，
2003）。梅拉姆德等人。 (2016)提出学习
通过任务从左右上下文预测单个单词的上下文表示

使用 LSTM。与 ELMo 类似,他们的模型是
基于特征而不是深度双向。费杜斯
等。 （2018)表明可以使用完形填空任务
提高文本生成模型的鲁棒性。

2.2 无监督微调方法

与基于特征的方法一样,第一个
在这个方向上工作仅来自未标记文本的预训练词嵌入参数（Collobert 和 Weston， 2008）。

最近,句子或文档编码器
产生上下文令牌表示
已经从未标记的文本中进行了预训练,并且
为有监督的下游任务（Dai
和乐， 2015;霍华德和罗德， 2018;拉德福
等人， 2018）。这些方法的优点
是需要学习的参数很少
刮。至少部分由于这个优势,
OpenAI GPT (Radford et al., 2018)在 GLUE 基准(Wang) 的
许多句子级任务上取得了先前最先进的结果

等人， 2018a）。从左到右的语言模型

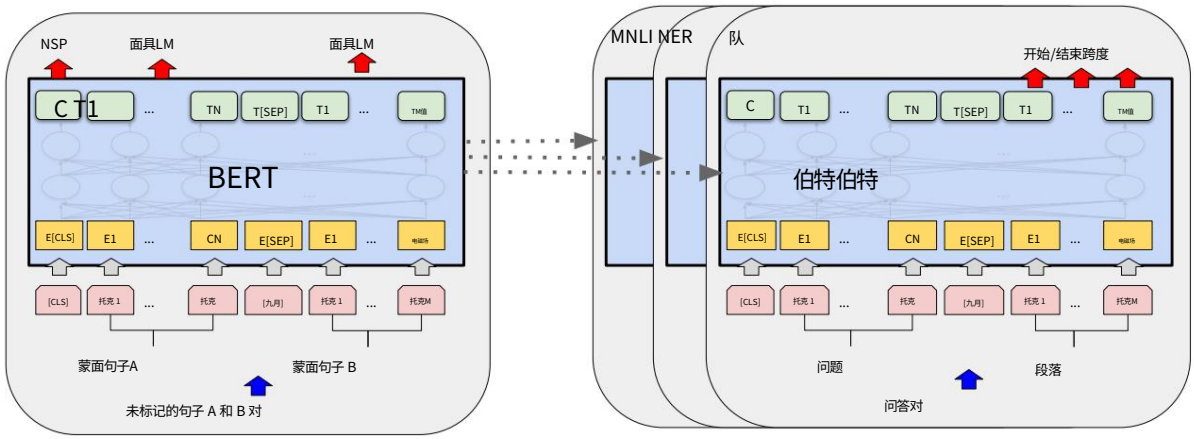

图 1:BERT 的整体预训练和微调程序。除了输出层外,预训练和微调都使用相同的架构。相同的预训练模型参数用于初始化

不同下游任务的模型。在微调期间,所有参数都被微调。 [CLS] 是一个特殊的
在每个输入示例前面添加符号,[SEP] 是一个特殊的分隔符（例如分隔问题/答案）。

荷兰国际集团和自动编码器目标已被使用
用于预训练此类模型（Howard 和 Ruder，
2018;拉德福德等人， 2018;戴和乐， 2015）。

2.3 监督数据的迁移学习

也有工作显示从具有大型数据集的监督任务中进行有效转移,例
如
作为自然语言推理（Conneau 等人，
2017)和机器翻译（McCann 等人，
2017）。计算机视觉研究也证明了迁移学习的重要性

大型预训练模型,其中有效的配方
是微调使用 Ima geNet 预训练的模型（Deng 等人，
2009； Yosinski 等人， 2014）。

3 伯特

我们将在本节介绍 BERT 及其详细实现。我们有两个步骤

框架:预训练和微调。在预训练期间,模型在未标记的

不同预训练任务的数据。为了进行微调,BERT 模型首先初始化为

预训练的参数,并且所有参数都使用来自

下游任务。每个下游任务都有单独的微调模型,即使它们是用相同
的预训练参数初始化的。这

图1中的问答示例将提供
作为本节的运行示例。
BERT 的一个显着特点是其统一的 ar
跨不同任务的架构。有迷你

预训练架构和最终下游架构之间的最大差异。

模型架构BERT 的模型架构师

ture 是基于Vaswani 等人描述的原始实现的多层双向
Transformer 编码器。 (2017)并发布于

tensor2tensor library.1因为使用
变形金刚已经变得很普遍,我们的即时通讯

补充几乎和原来的一样,
我们将省略模型架构的详尽背景描述,请读者参考

瓦斯瓦尼等人。 (2017)以及优秀的指南
例如“带注释的变压器”。2

在这项工作中,我们表示层数
（即 Transformer 块)为 L,隐藏大小为
H,self-attention head 的数量为 A。
我们主要报告两种模型尺寸的结果:
BERTBASE (L=12, H=768, A=12, Total Param
eters=110M) 和BERTLARGE (L=24, H=1024,
A=16,总参数=340M)。
BERTBASE被选中拥有相同的模型
大小作为 OpenAI GPT 进行比较。
然而,至关重要的是,BERT Transformer 使用
双向自注意力,而 GPT Trans 模型使用约束自注意力,其中每个

令牌只能关注其左侧的上下文。4

[1] https://github.com/tensorflow/tensor2tensor
[2] http://nlp.seas.harvard.edu/2018/04/03/attention.html
[3] 在所有情况下,我们将前馈/滤波器大小设置为 4H,
即,H = 768 为 3072,H = 1024 为 4096。
4我们注意到,在文献中,双向 Trans-

输入/输出表示来制作 BERT
处理各种下游任务,我们的输入
表示能够明确表示
一个句子和一对句子
（例如,h Question、Answeri)在一个标记序列中。
在整个工作中，“句子”可以是连续文本的任意跨度,而不是实际的
语言句子。 “序列”是指输入到 BERT 的令牌序列,它可能是一个句子,也可能是两个打包在一起的句子。

我们使用 WordPiece 嵌入（Wu 等人，
2016)具有 30,000 个标记词汇表。首先
每个序列的标记总是一个特殊的分类标记（[CLS]）。最终隐藏状态
对应于这个token的被用作分类的聚合序列表示

任务。句子对被打包成一个
单序列。我们区分句子
两种方式。首先,我们用一个特殊的
令牌（[SEP]）。其次,我们为每个令牌添加一个学习嵌入,指示它是否属于
到句子 A 或句子 B。如图1 所示，
我们将输入嵌入表示为 E,最终隐藏
特殊 [CLS] 标记的向量为 $C \in R H$，
和 i 的最终隐藏向量 th 输入令牌
为Ti $\in$ R H。

对于给定的令牌,其输入表示为
通过对相应的令牌求和构造，
段和位置嵌入。这种结构的可视化可以在图2 中看到。

3.1 预训练BERT

不像彼得斯等人。 (2018a)和Radford 等人。
(2018),我们不使用传统的从左到右或
从右到左的语言模型来预训练 BERT。
相反,我们使用本节中描述的两个无监督任务来预训练 BERT。这一步
如图1 左侧所示。

任务 #1:Masked LM直觉上,我们有理由相信深度双向模型是

严格比从左到右更强大
模型或左到的浅连接
右和从右到左的模型。很遗憾，
标准条件语言模型只能是
从左到右或从右到左进行训练,因为双向条件反射允许每个单词
直接“看到自己”,并且模型可以简单地

在多层上下文中预测目标词。

为了训练深度双向表示,我们简单地屏蔽了输入的一些百分比

随机标记,然后预测那些被屏蔽的
令牌。我们将此过程称为“屏蔽
LM”（传销）,虽然它通常被称为
文献中的完形填空任务(Taylor, 1953)。在这个
情况下,最终的隐藏向量对应于
掩码令牌被输入到输出 softmax
词汇,就像在标准 LM 中一样。在我们所有的
实验中,我们将所有 WordPiece 的 15% 随机屏蔽
为每个序列中的 kens。相比之下
去噪自动编码器（Vincent et al., 2008）,我们
只预测被屏蔽的词而不是重建整个输入。

尽管这使我们能够获得双向预训练模型,但缺点是我们

正在造成预训练和训练之间的不匹配
微调,因为 [MASK] 令牌在微调期间不会出现。为了减轻这种情况,我们做
并不总是用实际的 [MASK] 标记替换“被屏蔽”的单词。训练数据生成器
随机选择 15% 的令牌位置
预言。如果选择了第 i 个令牌,我们替换
具有 (1) [MASK] 令牌 80% 的第 i 个令牌
时间 (2) 随机令牌 10% 的时间 (3)
10% 的时间保持不变的第 i 个令牌。然后，
Ti将用于预测原始令牌
交叉熵损失。我们比较这个的变化
附录C.2 中的程序。

任务 #2:下一句预测 (NSP)
许多重要的下游任务,例如问答 (QA) 和自然语言推理 (NLI),都是基于理解两个句子之间的关系,而语言建模无法直接捕捉到这一点。为了

为了训练一个理解句子关系的模型,我们预先训练一个二值化的
下一个句子预测任务,该任务可以从任何单语语料库中轻松生成。具体来说，

在为每个预训练示例选择句子 A 和 B 时,50% 的时间 B 是实际的
A 之后的下一个句子（标记为IsNext），
并且 50% 的时间是随机句子
语料库（标记为NotNext）。正如我们展示的
在图1 中,C 用于下一个句子预测 (NSP)。5尽管它很简单,但我们
在第5.1节中演示了针对此的预训练

任务对 QA 和 NLI 都非常有益。

前者通常被称为“变压器编码器”,而
仅左上下文版本被称为“变压器
解码器”,因为它可以用于文本生成。

5最终模型在 NSP 上达到 97%-98% 的准确率。
6向量C不是有意义的句子表示
没有微调,因为它是用 NSP 训练的。

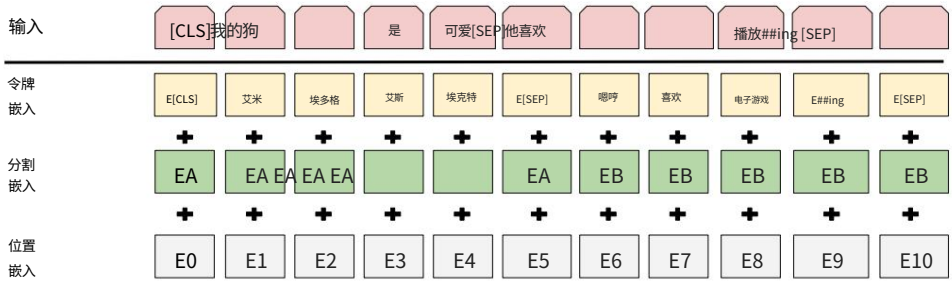

图 2:BERT 输入表示。输入嵌入是令牌嵌入、分割嵌入和位置嵌入的总和。

NSP 任务与Jernite 等人使用的表示学习目标密切相关。
（2017)和
Logeswaran 和李(2018)。然而,在之前
工作,只有句子嵌入被转移到
下游任务,其中 BERT 传输所有参数以初始化最终任务模型参数。

预训练数据预训练过程
很大程度上遵循现有的语言文献
模型预训练。对于预训练语料库,我们
使用 BooksCorpus（800M 字）（Zhu 等人，
2015)和英语维基百科（2,500M 字）。
对于维基百科,我们只提取文本段落
并忽略列表、表格和标题。使用文档级语料库而不是

打乱的句子级语料库,例如 Billion
Word Benchmark (Celba et al., 2013)
提取长的连续序列。

3.2 微调BERT

微调很简单,因为 Transformer 中的自注意力机制允许 BERT
对许多下游任务进行建模

它们是否涉及单个文本或文本对 通过
交换适当的输入和输出。
对于涉及文本对的应用程序,一个常见的
模式是在应用双向交叉注意之前独立编码文本对,例如

作为Parikh 等人。（2016）;徐等人。（2017）。 BERT
而是使用self-attention机制来统一
这两个阶段,作为对连接文本的编码
与自我注意配对有效地包括两个句子之间的双向交叉注意。

对于每个任务,我们只需将任务特定的输入和输出插入 BERT
并端到端微调所有参数。在输入处,来自预训练的句子 A 和句子 B

类似于（1)释义中的句子对，(2)蕴涵中的假设-前提对，(3)

问答中的问答对,以及

(4) 文本分类中的退化文本-∅对
或序列标记。在输出端,token 表示被输入到用于 token 级任务的输出层,例如序列标记或问题

回答,并馈送 [CLS] 表示
到输出层进行分类,例如 entailment 或情感分析。

与预训练相比,微调相对便宜。本文中的所有结果最多可在单个 Cloud TPU 上 1 小时内复制,或在 GPU 上数小时内复制,开始

来自完全相同的预训练模型。7我们在第 4 节的相应小节中描述了特定于任务的细节。更多细节可以

见附录A.5。

4个实验

在本节中,我们展示了 11 个 NLP 任务的 BERT 微调结果。

4.1 胶水

通用语言理解评估
(GLUE) 基准(Wang et al., 2018a)是各种自然语言理解的集合

任务。 GLUE 数据集的详细描述是
包含在附录B.1 中。
为了在 GLUE 上进行微调,我们表示输入
序列（用于单个句子或句子对）
如第3 节所述,并使用最终的隐藏向量 $C \in R^H$ 对应第一个
输入标记 ([CLS]) 作为聚合表示。期间引入的唯一新参数

微调是分类层权重 $W \in R^{K \times H}$,其中 K 是标签的数量。我们用 C 和 W 计算标准分类损失,
即,log(softmax(CWT))。

[7] 例如,BERT SQuAD 模型可以在
大约 30 分钟在单个 Cloud TPU 上实现开发
F1分数为91.0%。

[8] 请参阅https://gluebenchmark.com/faq中的 (10) 。

系统	MNLI-(m/mm)	QQP	QNLI	SST-2	CoLA	STS-B	MRPC	RTE	平均值
	392k	363k	108k	67k	8.5k	3.5k	5.7k	2.5k	-
Pre-OpenAI SOTA	80.6/80.1	66.1	82.3	93.2	35.0	81.0	86.0	61.7	74.0
BiLSTM+ELMo+Attn	76.4/76.1	64.8	79.8	90.4	36.0	73.3	84.9	56.8	71.0
OpenAI GPT	82.1/81.4	70.3	87.4	91.3	45.4	80.0	82.3	56.0	75.1
伯特库	84.6/83.4	71.2	90.5	93.5	52.1	85.8	88.9	66.4	79.6
伯特大	86.7/85.9	72.1	92.7	94.9	60.5	86.5	89.3	70.1	82.1

表 1:GLUE 测试结果,由评估服务器（https://gluebenchmark.com/leaderboard)评分。
每个任务下方的数字表示训练示例的数量。 “平均”列略有不同
比官方的 GLUE 分数高,因为我们排除了有问题的 WNLI 集。8 BERT 和 OpenAI GPT 是单一模型、单一任务。 QQP
和 MRPC 报告 F1 分数,STS-B 报告 Spearman 相关性,以及
报告其他任务的准确度分数。我们排除使用 BERT 作为其组件之一的条目。

我们使用 32 的批量大小并微调 3
对所有 GLUE 任务的数据进行 epochs。对于每个
任务,我们选择了最好的微调学习率
（在开发集上的 5e-5、4e-5、3e-5 和 2e-5 中）。
此外,对于BERTLARGE ,我们发现微调有时在小型数据集上不稳定,
所以我们运行了几次随机重启并选择了
开发集上的最佳模型。随着随机重启,
我们使用相同的预训练检查点,但执行不同的微调数据混洗和分类器层初始化。 9

结果如表1 所示。
BERTBASE和BERTLARGE在所有任务上的表现都大大优于所有系统,获得
与现有技术相比,平均准确度分别提高了 4.5% 和 7.0%。注意
BERTBASE和 OpenAI GPT 几乎相同
除了注意力屏蔽之外,在模型架构方面。对于最大和最广泛的
报告 GLUE 任务,MNLI,BERT 获得 4.6%
绝对精度提高。在官方
GLUE排行榜10， BERTLARGE获得分数
80.5,与 OpenAI GPT 相比,获得
72.8 截至撰写之日。
我们发现BERTLARGE在所有任务中都明显优于BERTBASE ,尤其是那些
训练数据很少。模型效果
第5.2节对 size 进行了更深入的探讨。

4.2 小队 v1.1

斯坦福问答数据集
（SQuAD v1.1)是 10 万个众包问答对的集合（Rajpurkar 等人,
2016）。给出一个问题和一段来自
包含答案的维基百科,任务是
预测文章中的答案文本跨度。

如图1 所示,在问答任务中,我们将输入问题和段落表示为单个
打包序列,其中问题使用 A 嵌入,段落使用
B 嵌入。我们只在
tor $S \in R^H$ 和一个结束向量 $E \in R^H$
微调。单词 i 的概率
答案跨度的开始计算为Ti和 S 之间的点积,然后是一个 softmax
段落中的所有单词： $P_i = \frac{e^{S \cdot T_i}}{\sum_j e^{S \cdot T_j}}$.
类似的公式用于结束
答案跨度。候选人的分数从
位置 i 到位置 j 定义为 $S \cdot T_i + E \cdot T_j$,
$j \geq i$ 的最大得分跨度为
用作预测。训练目标是
正确开始的对数似然和
结束位置。我们用 a 微调 3 个 epoch
学习率为 5e-5,批量大小为 32。

表2还显示了顶级排行榜条目
作为顶级发布系统的结果（Seo 等人,
2017;克拉克和加德纳， 2018;彼得斯等人,
2018a;胡等人， 2018）。最好的结果来自
SQuAD 排行榜没有最新的公开
系统描述可用,11并且允许
在训练他们的系统时使用任何公共数据。
因此,我们在
我们的系统首先在 TriviaQA (Joshi
et al., 2017)在对 SQuAD 进行微调之前。
我们性能最佳的系统优于顶级系统
排行榜系统 +1.5 F1 在合奏和
+1.3 F1 作为一个单一的系统。其实我们的单
BERT 模型在 F1 分数方面优于顶级集成系统。没有 TriviaQA 罚款

9GLUE数据集分布不包括Test
标签,我们只制作了一个 GLUE 评估服务器
为每个BERTBASE和BERTLARGE 提交。

10https://gluebenchmark.com/leaderboard

Yu 等人描述了 11QANet 。 (2018),但系统
出版后有了很大的改善。

系统	开发 EM	F1	测试 EM	F1
顶级排行榜系统（2018 年 12 月 10 日）				
人类	-		- 82.3	91.2
#1 合奏 - nlnet	-		- 86.0	91.7
#2 合奏 - QANet	-		- 84.5	90.5
发表				
BiDAF+ELMo（单）	-	85.6	- 85.8	
RM阅读器（合奏）	81.2	87.9	82.3	88.5
我们的				
BERTBASE （单）	80.8	88.5	- 84.1	-
BERTLARGE （单人）	90.9 -	85.8	91.8 -	-
BERTLARGE （合奏）				-
BERTLARGE (Sgl.+TriviaQA)	84.2	91.1	85.1	91.8
BERTLARGE (Ens.+TriviaQA)	86.2	92.2	87.4	93.2

表 2:SQuAD 1.1 结果。 BERT 集合
是 7x 系统,使用不同的预训练检查点和微调种子。

系统	开发 EM	F1	测试 EM	F1
顶级排行榜系统（2018 年 12 月 10 日）				
人类	86.3	89.0	86.9	89.5
#1 单 - MIR-MRC (F-Net) -	-		74.8	78.0
#2 单 - nlnet -	-		74.2	77.1
发表				
unet（合奏）	-		- 71.4	74.9
SLQA+（单人）	-		71.4	74.4
我们的				
BERTLARGE （单人）	78.7	81.9	80.0	83.1

表 3:SQuAD 2.0 结果。我们排除以下条目
使用 BERT 作为其组件之一。

调整数据,我们只损失了 0.1-0.4 F1,仍然大大优于所有现有系统。
12

4.3 小队 v2.0

SQuAD 2.0 任务扩展了 SQuAD 1.1
通过允许可能性来定义问题
提供的段落中没有简短的答案,使问题更加现实。

我们使用一种简单的方法来扩展 SQuAD
此任务的 v1.1 BERT 模型。我们将没有答案的问题视为具有在
[CLS] 到 ken 处开始和结束的 swer span。开始和结束的概率空间

答案跨度位置扩展到包括
[CLS] 标记的位置。为了预测,我们
比较无答案跨度的分数： snull =
S·C + E·C 到最佳非空跨度的分数

12我们使用的 TriviaQA 数据由来自
TriviaQA-Wiki 由文档中的前 400 个标记组成，
至少包含所提供的可能答案之一。

系统	开发	测试
ESIM+手套	51.9	52.7
ESIM+ELMo	59.1	59.2
OpenAI GPT	-	78.0
伯特库	81.6	-
伯特大	86.6	86.3
人类（专家）†	-	85.0
人类（5 个注释）†	-	88.0

表 4:SWAG 开发和测试精度。 †人类表现是用 100 个样本测量的,如在
SWAG 论文。

s^ i,j = maxj≥iS·Ti + E·Tj 。我们预测一个非空
当 s^ i,j > snull + τ时回答 ，脱粒的地方
在开发集上选择旧的 τ 以最大化 F1。
我们没有为此模型使用 TriviaQA 数据。我们
以 5e-5 的学习率微调 2 个 epoch
批量大小为 48。

结果与之前的排行榜条目和最高发表的作品进行了比较（Sun et al., 2018;
Wang et al., 2018b)如表3 所示,不包括使用 BERT 作为其组件之一的系统。我们观察到 +5.1 F1 改进

以前最好的系统。

4.4 赃物

对抗世代的情况

(SWAG) 数据集包含 113k 个句子对完成示例,用于评估基础常识推理(Zellers et al., 2018)。给定一个句子,任务是在四个选项中选择最合理的延续。

在对 SWAG 数据集进行微调时,我们
构造四个输入序列,每个包含
给定句子的连接（句子
A)和可能的继续（句子B）。这
唯一引入的特定于任务的参数是一个向量,其与 [CLS] 令牌表示 C 的点积表示每个选择的分数

使用 softmax 层对其进行归一化。

我们用 3 个 epoch 微调模型
学习率为 2e-5,批量大小为 16。结果如表4 所示。BERTLARGE比作者的基线 ESIM+ELMo 系统高出 +27.1%,OpenAI GPT 高出 8.3%。

5 消融研究

在本节中,我们进行消融实验
BERT 的多个方面,以便更好地
了解它们的相对重要性。额外的

任务	开发集 MNLI-m (加速)	QNLI (加速)	MRPC (加速)	SST-2 (加速)	小队 (F1)
BERTBASE	84.4	88.4	86.7	92.7	88.5
无 NSP	83.9	84.9	86.5	92.6	87.9
LTR 和无 NSP	82.1	84.3	77.5	92.1	77.8
+ BiLSTM	82.1	84.1	75.7	91.6	84.9

表 5:使用
BERTBASE架构。 “No NSP”是在没有训练的情况下训练的
下一句预测任务。 “LTR & No NSP”是
训练为从左到右的 LM,没有下一句
预测,例如 OpenAI GPT。 “+ BiLSTM”在“LTR + No”之上添
加了一个随机初始化的 BiLSTM
NSP”模型在微调期间。

消融研究见附录C。

5.1 预训练任务的效果

我们通过使用与 BERTBASE 完全相同的预训练数据、微调方案和
超参数来评估两个预训练目标,证明了 BERT 的深度双向性的重
要性:

No NSP:经过训练的双向模型
使用“蒙面 LM”(传销)但没有
“下一句预测”(NSP)任务。
LTR & No NSP:仅左上下文模型
使用标准的从左到右 (LTR) 进行训练
LM,而不是传销。仅左约束
也适用于微调,因为去除
它引入了预训练/微调不匹配,
下游性能下降。此外,
该模型是在没有 NSP 任务的情况下进行预训练的。
这可以直接与 OpenAI GPT 相媲美,但是
使用我们更大的训练数据集、我们的输入表示和我
们的微调方案。

我们首先考察 NSP 带来的影响
任务。在表5 中,我们展示了去除 NSP
在 QNLI、MNLI、
和小队 1.1。接下来,我们评估影响
通过比较“No NSP”与“LTR & No NSP”来训练双向表示。
LTR
模型在所有方面的表现都比 MLM 模型差
任务,在 MRPC 和 SQuAD 上有大量下降。

对于 SQuAD,直观地清楚的是,LTR
模型在代币预测方面表现不佳,
因为令牌级隐藏状态没有右侧上下文。为了真诚地尝试加强LTR
制度,我们补充说

顶部随机初始化的 BiLSTM。这确实
显着提高了 SQuAD 的结果,但
结果仍然比预训练的双向模型差得多。 BiLSTM 很痛

GLUE 任务的性能。

我们承认也可以
训练单独的 LTR 和 RTL 模型并表示
每个标记作为两个模型的连接,就像 ELMo 一样。但是:(a) 这是两
倍
作为单一的双向模型很昂贵; (b) 这
对于像 QA 这样的任务是不直观的,因为 RTL
模型将无法确定答案
关于问题; (c) 这严格来说没有那么强大
比深度双向模型,因为它可以使用
每一层的左右上下文。

5.2 模型大小的影响

在本节中,我们探讨模型大小的影响
关于微调任务的准确性。我们训练了一个数字
具有不同层数的 BERT 模型,
隐藏单位和注意力头,否则
使用与前面描述的相同的超参数和训练过程。

所选 GLUE 任务的结果显示在
表6. 在此表中,我们报告了平均 Dev
从 5 次随机重新启动微调设置精度。
我们可以看到,更大的模型会导致所有四个数据集
的精确度都有所提高,甚至
对于 MRPC,它只有 3,600 个标记的训练示例,并且与

预训练任务。我们能够取得如此重大的成就也许也令人惊讶

相对于现有文献已经相当大的模型的改进。

例如,在
瓦斯瓦尼等人。 (2017)是 (L=6, H=1024, A=16)
编码器的参数为 100M,并且
我们在文献中发现的最大的变压器
是 (L=64, H=512, A=2) 有 235M 个参数
（Al-Rfou 等人， 2018 年）。相比之下， BERTBASE
包含 110M 个参数， BERTLARGE包含 340M 个参数。

人们早就知道,增加
模型大小将导致持续改进
在机器翻译等大规模任务上
和语言建模,这是演示
通过保留训练数据的 LM 困惑
如表6 所示。然而,我们认为
这是第一个令人信服地证明缩放到极端模型尺寸的工作

导致非常小规模的大改进
任务,前提是模型已经过充分的预训练。彼得斯等人。 （2018b）
提出

对下游任务影响的混合结果
将预训练的 bi-LM 大小从两个增加
到四层和Melamud 等人。 (2016)顺便提到,将隐藏维度大
小从 200 增加到 600 有所帮助,但增加

进一步提高到 1,000 并没有带来进一步的改进。这两项先前
的工作都使用了基于特征的方法 我们假设当

模型直接在下游微调
任务并且只使用非常少量的随机初始化的附加参数,任务特
定的模型可以受益于更大、更多

表达性的预训练表示,即使在
下游任务数据非常少。

5.3 使用 BERT 的基于特征的方法

到目前为止提供的所有 BERT 结果都使用了
微调方法,在预训练模型中添加一个简单的分类层,以及

所有参数都在下游任务上联合微调。然而,基于特征的方法,

从预训练模型中提取固定特征,具有一定的优势。首先,不

所有任务都可以很容易地用一个 Trans 前编码器架构来表
示,因此需要
要添加的特定于任务的模型架构。
其次,有主要的计算优势
预先计算一个昂贵的表示
训练数据一次,然后运行许多实验
在此表示之上还有更便宜的模型。

在本节中,我们将比较这两种方法
通过将 BERT 应用于命名为 CoNLL-2003
实体识别 (NER) 任务(Tjong Kim Sang
和 De Meulder， 2003 年）。在 BERT 的输入中,我们
使用保留大小写的 WordPiece 模型,我们
包括提供的最大文档上下文
由数据。遵循标准实践,我们将其作为标记任务进行模拟,但
不使用 CRF

超参数		开发集准确性	
#L	#H #A	LM (ppl)	MNLI-m MRPC SST-2
3	768 12	5.84	77.9 79.8 88.4
6	768 12	4.68	80.6 82.2 90.7
12	1024 16	3.54	81.9 84.8 91.3
24	1024 16	3.23	84.4 86.7 92.9
			85.7 86.9 93.3
			86.6 87.8 93.7

表 6:BERT 模型大小的消融。 #L =
层数； #H = 隐藏大小； #A = 注意头的数量。 “LM (ppl)”是被掩蔽的 LM 困惑
保留的训练数据。

系统	开发 F1	测试 F1
ELMo （彼得斯等人， 2018a）	95.7	92.2
CVT （克拉克等人， 2018）	-	92.6
CSE （Akbik 等人， 2018 年）	-	93.1
微调方法		
伯特大	96.6	92.8
伯特库	96.4	92.4
基于特征的方法（BERTBASE）		
嵌入	91.0	-
倒数第二个隐藏	95.6	-
最后隐藏	94.9	-
加权和最后四个隐藏	95.9	-
Concat 最后四个隐藏	96.1	-
加权和所有 12 层	95.5	-

表 7:CoNLL-2003 命名实体识别结果。使用 Dev 选择超参数

放。报告的开发和测试分数平均超过
使用这些超参数进行 5 次随机重启。

在输出层。我们使用的表示
第一个子令牌作为令牌级别的输入
NER标签集上的分类器。

为了消除微调方法,我们应用
通过从一层或多层中提取激活而不进行微调的基于特征的方法
BERT 的任何参数。这些上下文嵌入被用作之前随机初始化的两层 768 维 BiLSTM 的输入

分类层。

结果如表7所示。 BERTLARGE
与最先进的方法竞争。表现最好的方法将

来自预训练 Transformer 的前四个隐藏层的令牌表示,仅

0.3 F1后面微调了整个模型。这
证明 BERT 对于微调和基于特征的方法都是有效的。

六,结论

最近由于转移而得到的经验改进
用语言模型学习已经证明
丰富的、无监督的预训练是不可或缺的
许多语言理解系统的一部分。在
特别是,这些结果甚至可以使低资源
从深度单向架构中受益的任务。我们的主要贡献是将这些发现进一步推广到深度双向架构,允许相同的预训练模型成功处理广泛的 NLP 任务。

参考

Alan Akbik、Duncan Blythe 和 Roland Vollgraf。
2018. 序列的上下文字符串嵌入
标签。在第 27 届国际会议上
计算语言学会议,页数
1638–1649。

Rami Al-Rfou、Dokook Choe、Noah Constant、Mandy
郭和狮子琼斯。 2018. 具有更深自我关注的字符级语言建模。 arXiv

预印本 arXiv:1808.04444。

久保田安藤理惠和张彤。 2005. 一个框架
用于从多个任务中学习预测结构
和未标记的数据。机器学习杂志
研究,6（11 月）:1817–1853。

路易莎·本蒂沃利、贝尔纳多·马尼尼、伊多·达甘、
Hoa Trang Dang 和 Danilo Giampiccolo。 2009 年。
第五种 PASCAL 识别文本蕴涵
挑战。在 TAC 中。 NIST。

约翰·布利策、瑞恩·麦克唐纳和费尔南多·佩雷拉。
2006. 结构对应学习的领域适应。在 2006 年自然语言处理经验方法
会议论文集上,第 120-128 页。计算语言学协会。

塞缪尔·R·鲍曼、加博·安杰利、克里斯托弗·波茨、
和克里斯托弗 D. 曼宁。 2015. 用于学习自然语言推理的大型注释语
料库。
在 EMNLP 中。计算语言学协会。

彼得 F 布朗、彼得 V 德索萨、罗伯特 L 默瑟、
文森特 J 德拉彼得拉和珍妮弗 C 赖。 1992 年。
基于类的自然语言 n-gram 模型。
计算语言学,18（4）:467-479。

Daniel Cer、Mona Diab、Eneko Agirre、Inigo Lopez Gazpio
和 Lucia Specia。 2017. Semeval-2017
任务 1:语义文本相似性多语言和
跨语言重点评价。在诉讼中
第 11 届国际语义研讨会
评估 (SemEval-2017),第 1-14 页,加拿大温哥华版。计算语言学协
会。

西普里安·切尔巴、托马斯·米科洛夫、迈克·舒斯特、齐格、
Thorsten Brants、菲利普·科恩和托尼·罗宾的儿子。 2013. 衡量统
计语言建模进展的十亿字基准。 arXiv

预印本 arXiv:1312.3005。

Z. Chen、H. Zhang、X. Zhang 和 L. Zhao。 2018 年。
Quora问题对。

克里斯托弗克拉克和马特加德纳。 2018. 简单
和有效的多段阅读理解。在 ACL 中。

Kevin Clark、Minh-Thang Luong、Christopher D Man ning 和 Quoc
Le。 2018. 具有跨视图训练的半监督序列建模。在 2018 年自然语言
处理经验方法会议记录中,第 1914 页–

1925 年。

罗南·科洛伯特和杰森·韦斯顿。 2008.统一
自然语言处理架构:Deep
具有多任务学习的神经网络。在第25届国际会议论文集上

机器学习,第 160-167 页。 ACM。

Alexis Conneau、Douwe Kiela、Holger Schwenk、Lo¨ c
巴罗特和安托万博德斯。 2017.监督
学习通用句子表示
自然语言推理数据。在诉讼中
2017 年自然语言处理经验方法会议,第 670-680 页,丹麦哥本哈根。
计算协会

语言学。

Andrew M Dai 和 Quoc V Le。 2015. 半监督
序列学习。在神经信息处理系统的进展中,第 3079-3087 页。

J. Deng, W. Dong, R. Socher, L.-J. Li、K. Li 和 L. 飞飞。 2009.
ImageNet:大规模分层
图像数据库。在 CVPR09 中。

威廉 B 多兰和克里斯布洛克特。 2005. 自动构建句子释义语料库。

第三届国际研讨会论文集
关于释义（IWP2005）。

威廉·费杜斯、伊恩·古德费罗和安德鲁·M·戴。
2018. Maskgan:通过填写更好的文本生成
这 。 arXiv 预印本 arXiv:1801.07736。

丹·亨德利克斯和凯文·金佩尔。 2016.桥接
具有高斯误差线性单元的非线性和随机正则化器。 CoRR,abs/
1606.08415。

Felix Hill、Kyunghyun Cho 和 Anna Korhonen。 2016 年。
学习句子的分布式表示
来自未标记的数据。在 2016 年会议记录中
北美分会会议
计算语言学协会:人类
语言技术。计算语言学协会。

杰里米霍华德和塞巴斯蒂安鲁德。 2018.通用
用于文本分类的语言模型微调。在
访问控制列表。计算语言学协会。

胡明浩,彭宇兴,黄震,邱锡鹏,
福如卫,明周。 2018. 加强
用于机器阅读理解的助记符阅读器。在 IJCAI。

Yacine Jernite、Samuel R. Bowman 和 David Son 标签。 2017.基
于话语的快速无监督句子表示学习目标。心电图,

绝对/1705.00557。

Mandar Joshi、Eunsol Choi、Daniel S Weld 和 Luke
泽特尔莫耶。 2017. 琐事:遥远的大规模
用于阅读理解的监督挑战数据集。在 ACL 中。

瑞恩·基罗斯、朱玉坤、鲁斯兰·R·萨拉赫特迪诺夫、
理查德·泽梅尔、拉奎尔·乌尔塔松、安东尼奥·托拉尔巴、
和桑贾菲德勒。 2015. 跳过思维向量。在
神经信息处理系统的进展,
第 3294-3302 页。

Quoc Le 和 Tomas Mikolov。 2014. 句子和文档的分布式表示。在国
际机器学习会议上,页数

1188–1196。

Hector J Levesque、Ernest Davis 和 Leora Morgen 斯特恩。 2011.
winograd 模式挑战。在
Aaai春季研讨会:逻辑形式化
常识推理,第 46 卷,第 47 页。

Lajanugen Logeswaran 和 Honglak Lee。 2018.一个
学习句子表示的有效框架。在国际学习会议上

陈述。

布莱恩·麦肯、詹姆斯·布拉德伯里、熊才明和
理查德·索切尔。 2017. 在翻译中学习:上下文化词向量。在 NIPS 中。

Oren Melamud、Jacob Goldberger 和 Ido Dagan。
2016. context2vec:使用双向 LSTM 学习通用上下文嵌入。在
CoNLL 中。

Tomas Mikolov、Ilya Sutskever、Kai Chen、Greg S Cor rado 和 Jeff
Dean。 2013. 单词和短语的分布式表示及其组合性。神经信息处理
进展

系统 26,第 3111-3119 页。柯伦协会,
公司

Andriy Mnih 和 Geoffrey E Hinton。 2009.可扩展的分层分布式语言
模型。在
D. Koller、D. Schuurmans、Y. Bengio 和 L. Bot tou,编辑,神经信
息处理系统进展 21,第 1081-1088 页。 Curran As Associates,
Inc.

..
Ankur P Parikh、Oscar Tackstr ¨ om、Dipanjan Das 和 Jakob
Uszkoreit。 2016. 可分解的注意力
自然语言推理模型。在 EMNLP 中。

Jeffrey Pennington、Richard Socher 和 Christo pher D. Manning。
2014.手套:全球向量
词表示。在自然语言处理中的经验方法 (EMNLP),第 1532 页–

1543.

Matthew Peters、Waleed Ammar、Chandra Bhagavat ula 和
Russell Power。 2017. 具有双向语言模型的半监督序列标记。

在 ACL 中。

马修·彼得斯、马克·诺伊曼、莫希特·艾耶、马特
加德纳、克里斯托弗·克拉克、肯顿·李和卢克
泽特尔莫耶。 2018a。深度上下文化的词表示。在 NAACL 中。

马修·彼得斯、马克·诺伊曼、卢克·泽特尔莫耶、
还有易文头。 2018b。剖析上下文
词嵌入:架构和表示。
在 2018 年自然语言处理经验方法会议论文集上,页数

1499–1509。

亚历克·拉德福、卡提克·纳拉辛汉、蒂姆·萨利曼斯和
伊利亚·萨茨克维尔。 2018. 通过无监督学习提高语言理解能力。技
术报告,OpenAI。

Pranav Rajpurkar、张健、康斯坦丁·洛佩列夫和
珀西梁。 2016. 小队:100,000+ 个问题
机器对文本的理解。在诉讼中
2016 年自然语言处理经验方法会议,第 2383-2392 页。

Minjoon Seo、Aniruddha Kembhavi、Ali Farhadi 和
汉纳内·哈吉希尔兹。 2017. 双向关注
机器理解的流程。在 ICLR 中。

Richard Socher、Alex Perelygin、Jean Wu、Jason
Chuang、Christopher D Manning、Andrew Ng 和
克里斯托弗·波茨。 2013. 递归深度模型
用于情感树库上的语义组合性。在 2013 年会议论文集上

自然语言处理中的经验方法,
第 1631-1642 页。

孙福、李林洋、邱锡鹏、刘杨。
2018. U-net:机器阅读理解
带着无法回答的问题。 arXiv 预印本
arXiv:1810.06638。

威尔逊 L 泰勒。 1953. 完形填空程序:一个新的
测量可读性的工具。新闻公报,
30（4）:415-433。

Erik F Tjong Kim Sang 和 Fien De Meulder。
2003. conll-2003共享任务简介:
与语言无关的命名实体识别。在
控制。

约瑟夫·图里安、列夫·拉蒂诺夫和约书亚·本吉奥。 2010 年。
单词表示:一种简单而通用的方法
用于半监督学习。在诉讼中
第 48 届计算语言学协会年会,ACL 10,第 384-394 页。

Ashish Vaswani 诺姆·沙泽尔 Niki Parmar Jakob
Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz
凯撒和伊利亚·波洛苏欣。 2017. 关注就是一切
你需要。在神经信息处理系统的进展中,第 6000-6010 页。

帕斯卡·文森特、雨果·拉罗谢尔、约书亚·本吉奥和
皮埃尔-安托万·曼扎戈尔。 2008. 提取和
使用去噪自动编码器组成强大的特征。在第 25 届国际会议论文集中

机器学习会议,第 1096-1103 页。
ACM。

Alex Wang、Amanpreet Singh、Julian Michael、Fe lix Hill、
Omer Levy 和 Samuel Bowman。 2018a。
Glue:多任务基准和分析平台

用于自然语言理解。在 2018 年 EMNLP 研讨会论文集 BlackboxNLP:NLP 的分析和解释神经网络,第 353-355 页。

王伟、明彦、陈武。 2018b。用于阅读理解和问答的多粒度分层注意力融合网络。

在计算语言学协会第 56 届年会论文集（第 1 卷:长论文)中。计算语言学协会。

Alex Warstadt、Amanpreet Singh 和 Samuel R Bow 人。 2018. 神经网络可接受性判断。 arXiv 预印本 arXiv:1805.12471。

阿迪娜·威廉姆斯、尼基塔·南吉亚和塞缪尔·R·鲍曼。 2018. 通过推理进行句子理解的广泛覆盖挑战语料库。在 NAACL 中。

Yonghui Wu,Mike Schuster,Zhifeng Chen,Quoc V Le，Mohammad Norouzi,Wolfgang Macherey,Maxim Krikun，Yuan Cao,Qin Gao,Klaus Macherey 等。 2016 年。谷歌的神经机器翻译系统:弥合人类和机器翻译之间的差距。 arXiv 预印本 arXiv:1609.08144。

杰森·尤辛斯基、杰夫·克鲁恩、约书亚·本吉奥和霍德·利普森。 2014. 深度神经网络中特征的可迁移性如何?在神经信息处理系统的进展中,第 3320-3328 页。

Adams Wei Yu、David Dohan、Minh-Thang Luong、Rui Zhao、Kai Chen、Mohammad Norouzi 和 Quoc V Le。 2018. QANet:将局部卷积与全局自注意力相结合以进行阅读理解。在 ICLR 中。

Rowan Zellers、Yonatan Bisk、Roy Schwartz 和 Yejin Choi。 2018. Swag:用于基础常识推理的大规模对抗性数据集。在 2018 年自然语言处理经验方法会议 (EMNLP) 会议记录中。

Yukun Zhu、Ryan Kiros、Rich Zemel、Ruslan Salakhut dinov、Raquel Urtasun、Antonio Torralba 和 Sanja Fidler。 2015. 对齐书籍和电影:通过看电影和阅读书籍来实现类似故事的视觉解释。在 IEEE 计算机视觉国际会议论文集上,第 19-27 页。

“BERT:预训练
深度双向变压器
语言理解”

我们将附录分为三个部分：

- BERT 的其他实施细节见附录A；
- 附录B中提供了我们实验的更多细节;和
- 附加消融研究见附录C。

我们提出了额外的消融研究
BERT 包括：

–训练步数的影响;和

–不同掩蔽工艺的烧蚀
胁迫。

BERT 的附加细节

A.1 预训练任务的说明 我们在下面提供预训练任务的示例。

Masked LM 和 Masking Procedure假设未标记的句子是 my dog is hairy,并且在随机 masking 过程中我们选择了第 4 个 token（对应于hairy），我们的 masking 过程可以进一步说明为

- 80% 的时间:用 [MASK] 标记替换单词,例如， my dog is hairy → my dog is [MASK]
- 10% 的时间:用随机词替换该词,例如,我的狗有毛 → 我的

  狗是苹果
- 10% 的时间:保持单词不变,例如， my dog is hairy → my dog is hairy。这样做的目的是使表示偏向于实际观察到的单词。

该程序的优点是
Transformer 编码器不知道哪个字
它将被要求预测或被随机词替换,因此它被迫保留每个输入标记的分布上下文表示。此外,因为随机替换只发生在所有标记的 1.5%（即 15% 的 10%)上,这似乎不会损害模型的语言理解能力。在第 C.2 节中,我们评估了此过程的影响。

与标准语言模型训练相比,masked LM 仅对每批中 15% 的标记进行预测,这表明模型可能需要更多的预训练步骤

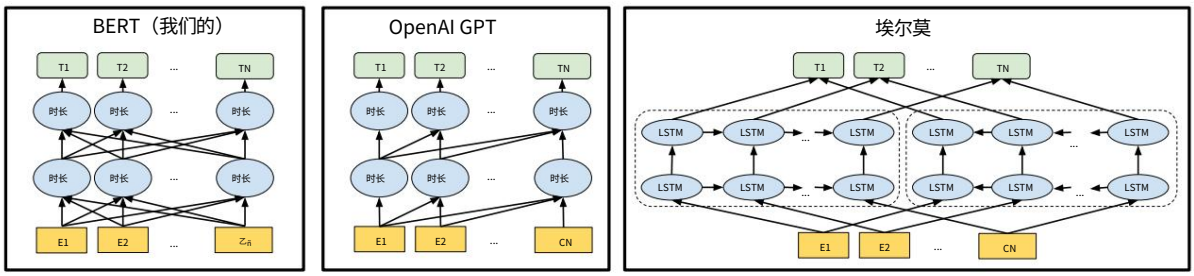

图 3:预训练模型架构的差异。 BERT 使用双向 Transformer。 OpenAI GPT
使用从左到右的 Transformer。 ELMo 使用独立训练的从左到右和从右到左 LSTM 的串联来为下游任务生成特征。三者中,只有BERT
表示是联合的
以所有层的左右上下文为条件。除了架构差异之外,BERT 和
OpenAI GPT 是微调方法,而 ELMo 是基于特征的方法。

收敛。在C.1节中,我们证明了
MLM 的收敛速度确实比从左到右的模型（预测每个
标记)稍微慢一些,但是
MLM 模型的经验改进
远远超过增加的培训成本。

下一句预测下一句

预测任务如下图所示
例子。

输入 = [CLS] 这个人去了 [MASK] 商店 [SEP]

他买了一加仑 [MASK] 牛奶 [SEP]

标签 = IsNext

输入 = [CLS] 人 [MASK] 到商店 [SEP]

企鹅 [MASK] 正在飞行 ##less 鸟类 [SEP]

标签 = NotNext

A.2 预训练程序

为了生成每个训练输入序列,我们从语料库中抽取两个文本跨度,
我们
被称为“句子”,即使它们通常比单个句子长得多（但可以

也更短）。第一句话收到A
嵌入,第二个接收 B 嵌入。 50% 的时间 B 是实际的下一句

跟随 A 并且 50% 的时间是随机的
句子,这是为“下一句预测”任务完成的。对它们进行采样,使得组
合长度≤512 个标记。 LM掩蔽是

在 WordPiece 标记化后应用,统一掩蔽率为 15%,对部分词块
没有特殊考虑。

我们训练批量大小为 256 个序列（256
1,000,000 * 512 个代币 = 128,000 个代币/批次）
步的序列,大约是 40

超过 33 亿个词的语料库。我们
使用 Adam,学习率为 1e-4， β1 = 0.9，
β2 = 0.999,L2 权重衰减 0.01,学习
前 10,000 步的预热速率和线性
学习率的衰减。我们在所有层上使用 0.1 的 dropout 概率。
我们使用凝胶激活(Hendrycks and Gimpel, 2016)而不
是
遵循 OpenAI GPT 的标准 relu。这
训练损失是平均掩蔽 LM 的总和
可能性和平均下一句预测
可能性。

BERTBASE的训练在 4 上进行
Pod 配置中的 Cloud TPU（16 个 TPU 芯片
总计）.13 BERTLARGE的训练进行了
在 16 个 Cloud TPU（总共 64 个 TPU 芯片)上。每
次预训练需要 4 天才能完成。

较长的序列非常昂贵,因为注意力是序列的二次方

长度。为了加快我们实验中的预训练，
我们用序列长度预训练模型
90% 的步数为 128。然后,我们训练其余的
512序列的10%的步骤来学习
位置嵌入。

A.3 微调程序

对于微调,大多数模型超参数是
与预训练相同,除了
批量大小、学习率和训练 epoch 的数量。辍学概率总是

保持在 0.1。最优超参数值
是特定于任务的,但我们发现了以下范围
在所有任务中运行良好的可能值：

·批量大小： 16、32

13https://cloudplatform.googleblog.com/2018/06/Cloud TPU-now-offers-preemptible-pricing-and-global availability.html

·学习率 (Adam)： 5e-5、3e-5、2e-5 ·时期数：2、3、4

我们还观察到，大数据集（例如，100k+ 标记的训练示例）对超参数选择的敏感度远低于小数据集。微调通常非常快，因此只需对上述参数进行详尽的搜索并选择在开发集上表现最佳的模型是合理的。

A.4 BERT、ELMo 和 OpenAI GPT 的比较

在这里，我们研究了最近流行的表示学习模型的差异，包括 ELMo、OpenAI GPT 和 BERT。图3 直观地显示了模型架构之间的比较。请注意，除了架构差异之外，BERT 和 OpenAI GPT 是微调方法，而 ELMo 是基于特征的方法。

与 BERT 最相似的现有预训练方法是 OpenAI GPT，它在大型文本语料库上训练从左到右的 Transformer LM。事实上，BERT 中的许多设计决策都是有意使其尽可能接近 GPT，以便可以最小限度地比较这两种方法。这项工作的核心论点是， 3.1节中介绍的双向性和两个预训练任务占了大部分经验改进，但我们确实注意到还有其他一些差异

BERT 和 GPT 的训练方式之间：

- GPT 在 BooksCorpus 上进行训练（8 亿字）；BERT 在 BooksCor pus（8 亿字）和 Wikipedia（25 亿字）上进行了训练。
- GPT 使用仅在微调时引入的句子分隔符（[SEP]）和分类器标记（[CLS]）； BERT 在预训练期间学习 [SEP]、[CLS] 和句子 A/B 嵌入。
- GPT 训练了 1M 步，批量大小为 32,000 字； BERT 接受了 1M 步训练，批量大小为 128,000 个单词。
- GPT 在所有微调实验中使用相同的 5e-5 学习率； BERT 选择在开发集上表现最好的特定于任务的微调学习率。

为了隔离这些差异的影响，我们在第5.1节中进行了消融实验，这表明大多数改进实际上来自两个预训练任务及其启用的双向性。

A.5 不同任务微调的说明

在图 4 中可以看到在不同任务上微调 BERT 的图示。我们的特定任务模型是通过将 BERT 与一个额外的输出层相结合而形成的，因此需要从头开始学习最少数量的参数。

其中，（a）和（b）是序列级任务，（c）和（d）是令牌级任务。图中，E 表示输入embedding， Ti表示token i的上下文表示，[CLS]是分类输出的特殊符号，[SEP]是分隔不连续token序列的特殊符号。

B 详细的实验设置

B.1 GLUE 基准实验的详细说明。

我们在表 1 中的 GLUE 结果是从https://gluebenchmark.com/从 leaderboard获得的和https://blog。 openai.com/language-unsupervised。

GLUE 基准包括以下数据集，这些数据集的描述最初在Wang 等人中总结。 （2018a）：

MNLI Multi-Genre Natural Language Inference 是一项大规模的众包蕴涵分类任务（Williams et al., 2018）。给定一对句子，目标是预测第二个句子相对于第一个句子是蕴涵、矛盾还是中性。

QQP Quora Question Pairs 是一个二元分类任务，其目标是确定 Quora 上提出的两个问题在语义上是否等效（Chen et al., 2018）。

QNLI问题自然语言推理是斯坦福问答数据集（Rajpurkar 等人，2016 年)的一个版本，它已被转换为二进制分类任务（Wang 等人， 2018a）。正例是包含正确答案的 (question, sentence) 对，负例是来自同一段落但不包含答案的 (question, sentence)。

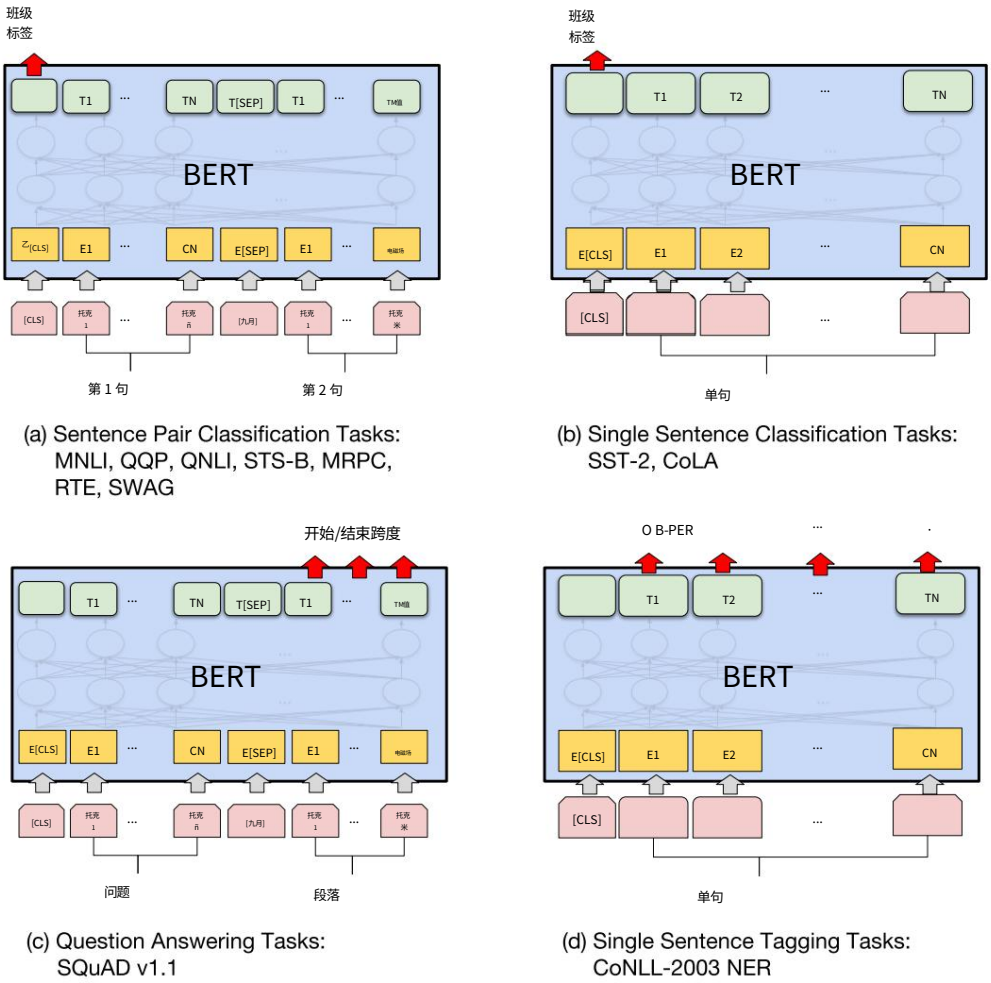

图 4:在不同任务上微调 BERT 的图示。

SST-2斯坦福情绪树库是一个
由从电影评论中提取的句子组成的二元单句分类任务

用人类对其情绪的注释（Socher
等人， 2013）。

CoLA语言可接受性语料库是
一个二元单句分类任务,其中
目标是预测一个英文句子是否
在语言上是否“可接受” （Warstadt
等人， 2018）。

STS-B语义文本相似度基准是从

新闻头条和其他来源（Cer 等人，
2017）。它们被注释为从 1 开始的分数
到 5 表示这两个句子的相似程度
语义方面的术语。

MRPC微软研究院释义语料库
由自动提取的句子对组成
来自在线新闻来源,带有人工注释

判断这对中的句子是否在语义上是等价的（Dolan and
Brockett, 2005）。

RTE识别文本蕴涵是类似于 MNLI 的二元蕴涵任务,但具有

训练数据少得多（Bentivogli et al., 2009）.14

WNLI Winograd NLI 是一个小型自然语言推理数据集
(Levesque et al., 2011)。
GLUE网页指出有问题
随着这个数据集的构建,每个 15
已提交给 GLUE 的训练有素的系统
表现比 65.1 基线准确度差
预测多数类别。因此,为了对 OpenAI GPT 公平,我们排除
了这个集合。对于我们的
GLUE投稿,我们一直都在预测ma

14注意,我们只报告单任务微调结果
在本文中。多任务微调方法可能会进一步提高性能。例如,我们

确实观察到使用 MNLI 进行多任务训练对 RTE 的显着改进。

15https://gluebenchmark.com/faq

多数派。

C 额外的消融研究

C.1 训练步数的影响

图5显示了从预先训练的检查点进行微调后的 MNLI 开发精度
为 k 步。这使我们能够回答以下问题
问题：

1.问：BERT真的需要这样吗
大量的预训练（128,000
字/批 * 1,000,000 步)来实现
微调精度高？
答：是的， BERTBASE几乎做到了
在 MNLI 上增加 1.0% 的准确度，当
与 500k 步相比，训练了 1M 步。

2.问：MLM预训练是否收敛
比 LTR 预训练慢，因为只有 15%
在每批中预测单词的数量，而不是
比每一个字？
答：传销模型确实收敛
比 LTR 模型稍慢。然而，就绝对准确度而言，MLM

模型开始优于 LTR 模型
几乎立即。

C.2 不同掩蔽的消融
程序

在第3.1 节中，我们提到 BERT 使用
掩蔽目标标记的混合策略
使用掩码语言模型进行预训练
（传销)目标。以下是消融
评估不同掩蔽效果的研究
策略。

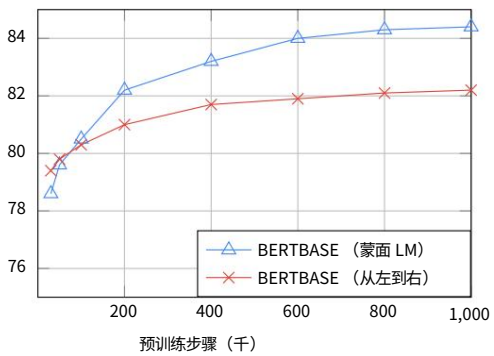

图 5：在训练步骤数上的消融。这
显示微调后的 MNLI 精度，开始
来自已预先训练的模型参数
k 步。 x 轴是 k 的值。

请注意，掩蔽策略的目的
是为了减少预训练之间的不匹配
和微调，因为 [MASK] 符号在微调阶段永远不会出现。我们报告了
MNLI 和 NER 的开发结果。对于 NER，
我们报告了微调和基于特征的方法，因为我们期望基于特征的方法作为模型的不匹配将被放大

将没有机会调整陈述。

掩蔽率			开发集结果	
面具相同的RND MNLI			NER	
			Fine-tune	Fine-tune 基于特征
80%	10%	10%	84.2	
100%	0%	0%	84.3	
80%	10%	10%	84.1	
80%	0%	20%	84.4	
0%	20%	80%	83.7	
0%	0%	100%	83.6	
			95.4	94.9
			94.9	94.0
			95.2	94.6
			95.2	94.7
			94.8	94.6
			94.9	94.6

表 8：不同掩蔽策略的消融。

结果如表8 所示。在表中，
MASK表示我们将目标令牌替换为
传销的 [MASK] 符号； SAME意味着
我们保持目标令牌不变； RND意味着
我们用另一个随机替换目标令牌
令牌。
表格左侧的数字表示使用的特定策略的概率

在 MLM 预训练期间（BERT 使用 80%、10%、10%）。论文的右半部分代表
开发集结果。对于基于特征的方法，
我们将 BERT 的最后 4 层连接为
功能，这被证明是最好的方法
在第5.3 节中。

从表中可以看出微调是
对不同的掩蔽策略具有惊人的鲁棒性。
然而，正如预期的那样，在将基于特征的方法应用于 NER 时，仅使用MASK策略是有问题的。有趣的是，仅使用
RND策略的表现比我们的差得多
策略也是如此。